

WebSphere®

DEVELOPER'S JOURNAL

The World's Leading Independent
WebSphere Developer Resource

WebSphereDevelopersJournal.com

MAY VOLUME 2 ISSUE 5

SAVE UP TO \$400

see page 47 for details



SUBSCRIBE TODAY

and get UP TO **3 FREE CDs!**
(WHILE SUPPLIES LAST)

FROM THE EDITOR

**Tech Bust Is a Shakeout
for Unqualified Executives**

BY JACK MARTIN • PAGE 4

TOOLS

**WebSphere Risk
Management, Part 2**

BY BASSEM W. JAMALEDINE • PAGE 8

WSDJ INTERVIEW

**WebSphere
Marketing Team, Part 3:
Exciting Plans Loom
on the WebSphere Horizon**

INTERVIEWED BY JACK MARTIN • PAGE 32

NEWS
PAGE 42

DISPLAY UNTIL JULY 31, 2003

\$8.99US \$9.99CAN

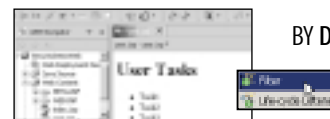


**SYS-CON
MEDIA**



BY MATT PUCCINI AND MORGAN SMYTH PAGE 16

**A WebSphere Approach to
Strategic Market Development**
Alignment of corporate goals and cultures is key



BY DEREK BILDFELL

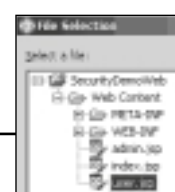
PAGE 24

WAS and Database Performance Tuning
Best practices provide a road map



BY MICHAEL S. PALLOS
PAGE 28

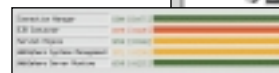
**Development Using Servlet 2.3
Filters and Event Listeners**
Features offer ample flexibility and control



BY RAJU MUKHERJEE ET AL.

PAGE 34

**Improving Performance of
J2EE Applications**
From a memory usage perspective



BY WILFRED C. JAMISON

PAGE 38

**Implementing WebSphere Security
Through LDAP**
Protocol offers many advantages



BY LOU MAUGET

PAGE 44

VERSATA

WWW.VERSATA.COM/BUSINESSLOGICDESIGNER

WILY TECHNOLOGY

WWW.WILYTECH.COM

Tech Bust Is a Shakeout for Unqualified Executives

BY JACK MARTIN

The *New York Times* recently reported that between December 2000 and January 2003 the New York City computer industry lost 21,000 jobs, which represents 41% of the positions that existed in December 2000. Since the end of 2000 the city's media and communications sector has been cut by 15%, telecommunications has lost 27%, 18% of Wall Street jobs have vanished, and 25% of advertising jobs have also gone the way of the 10-cent subway ride. Yikes!

Given the fantasy-based environment that created most of these jobs, it is unlikely that any of these positions – or the downstream employment they created for others – will ever return, a view that is supported as firms continue to lay off more people.

You may ask, where did these positions go? Well some of them went offshore, to countries with much lower labor costs, like India and China. I was concerned that the United States may give up the technology lead if this process continues, but it seems the majority of these jobs went back to where they came from – the netherworld of half-fulfilled dreams and promises. The primary product of many failed technology companies was smoke and mirrors, not technology.


I think that many of the people who jumped on the technology bandwagon have returned to their former positions in nontechnology-based employment. For example, the late '90s phenomenon of companies hiring people fresh out of school and giving them titles like "director of telecommunications" and "vice president of technology" just doesn't happen anymore. Companies have returned to the concept that someone who has a title that implies leadership is actually capable of leadership – and is not still wet behind the ears.



When these individuals lose their executive positions there is not a company out there that will hire them. They lack experience as leaders and have no demonstrable track record of quantifiable success. They are too young in most cases to have had the time or opportunity to prove themselves one way or the other. In life, if you truly reach maturity, you realize that some things take time and experience – and nothing can be done to rush the process or substitute for it.

While it is definitely a badge of honor for an individual to be the youngest vice president in a company, at the same time there may be a significant disadvantage for the individual's employer. In most cases a young executive lacks "seasoning" and cannot be depended upon to know what actions to take when things are not going the way the plan predicted they would.

If you are in the employer position, and don't believe me, *imagine* (if you actually did this, it would be weird) trying this experiment. Take the resume of one of your staff, rewrite it to change any identifying information, and send it over to your top three competitors. That individual is probably so unqualified that you would wait a very long time for any reply. This exercise may be of value when you have to trim your staff. By analyzing individuals in this manner you are likely to make a better decision and ultimately end up with the best people, the best technologists.

What the true technologist possesses that no down market can ever take away is the ability to see how things can potentially be better tomorrow – the ability to visualize new technology or use existing technology in a way that helps people do something they can't do now – or to do something faster, better, or cheaper. What the true technology executive can do is choose the people who have the greatest ability to start turning that vision of a better tomorrow into a solid business today. 

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

ADVISORY BOARD

Richard Arone, David Caddis, Mike Curwen, Devi Gupta, Lloyd Hagemo, Barbara Johnson, Shannon Lynd, James Martin, Doug Stephen, Victor Valle

EDITOR-IN-CHIEF JACK MARTIN
 EDITORIAL DIRECTOR JEREMY GEELAN
 EXECUTIVE EDITOR GAIL SCHULTZ
 CONTRIBUTING EDITOR PATTI MARTIN
 PRODUCT REVIEW EDITOR JAY JOHNSON
 SPECIAL PROJECTS EDITOR RICHARD ARONE
 MANAGING EDITOR JEAN CASSIDY
 EDITOR NANCY VALENTINE
 ASSOCIATE EDITORS JAMIE MATUSOW
 JENNIFER STILLEY

WRITERS IN THIS ISSUE

Derek Bildfell, Bassem Jamaledine, Wilfred C. Jamison, Jack Martin, Lou Mauge, Raju Mukherjee, Swaraj Pal, Michael S. Pallos, Matt Puccini, Deependra Shrestha

SUBSCRIPTIONS

For subscriptions and requests for bulk orders, please send your letters to Subscription Department. SUBSCRIBE@SYS-CON.COM

Cover Price: \$8.99/Issue Domestic: \$149/YR (12 Issues)
 Canada/Mexico: \$169/YR Overseas: \$179/YR
 (U.S. Banks or Money Orders)
 Back issues: \$15 U.S. \$20 All others

PRESIDENT AND CEO FUAT A. KIRCAALI
 SENIOR VP, SALES & MARKETING CARMEN GONZALEZ
 VP, INFORMATION SYSTEMS ROBERT DIAMOND
 VP, SALES & MARKETING MILES SILVERMAN
 PRESIDENT, SYS-CON EVENTS GRISHA DAVIDA
 PRODUCTION CONSULTANT JIM MORGAN
 FINANCIAL ANALYST JOAN LAROSE
 ACCOUNTS RECEIVABLE KERRI VON ACHEN
 ACCOUNTS PAYABLE BETTY WHITE
 ADVERTISING DIRECTOR ROBYN FORMA
 DIRECTOR OF SALES AND MARKETING MEGAN MUSSA
 ADVERTISING SALES MANAGER ALISA CATALANO
 ASSOCIATE SALES MANAGERS CARRIE GEBERT
 KRISTIN KUHNLE
 CONFERENCE MANAGER MICHAEL LYNCH
 ART DIRECTOR ALEX BOTERO
 ASSOCIATE ART DIRECTORS LOUIS F. CUFFARI
 RICHARD SILVERBERG
 TAMI BEATTY
 WEB DESIGNERS STEPHEN KILMURRAY
 CHRISTOPHER CROCE
 CONTENT EDITOR LIN GOETZ
 CIRCULATION SERVICE NIKI PANAGOPOULOS
 COORDINATORS SHELLA DICKERSON
 CUSTOMER RELATIONS MANAGER RACHEL MCCOURAN

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

© COPYRIGHT

2003 BY SYS-CON PUBLICATIONS, INC. ALL RIGHTS RESERVED. NO PART OF THIS PUBLICATION MAY BE REPRODUCED OR TRANSMITTED IN ANY FORM OR BY ANY MEANS, ELECTRONIC OR MECHANICAL, INCLUDING PHOTOCOPY OR ANY INFORMATION STORAGE AND RETRIEVAL SYSTEM, WITHOUT WRITTEN PERMISSION. FOR PROMOTIONAL REPRINTS, CONTACT REPRINT COORDINATOR: SYS-CON PUBLICATIONS, INC. RESERVES THE RIGHT TO REVISE, REPHRASE AND AUTHORIZE THE READERS TO USE THE ARTICLES SUBMITTED FOR PUBLICATION.

ALL BRAND AND PRODUCT NAMES USED ON THESE PAGES ARE TRADE NAMES, SERVICE MARKS OR TRADEMARKS OF THEIR RESPECTIVE COMPANIES. SYS-CON PUBLICATIONS, INC. IS NOT AFFILIATED WITH THE COMPANIES OR PRODUCTS COVERED IN WEBSphere DEVELOPER'S JOURNAL.

WEBSphere® IS A REGISTERED TRADEMARK OF IBM CORP. AND IS USED BY SYS-CON MEDIA UNDER LICENSE. WEBSphere® DEVELOPER'S JOURNAL IS PUBLISHED INDEPENDENTLY OF IBM CORP., WHICH IS NOT RESPONSIBLE IN ANY WAY FOR ITS CONTENT.

SYS-CON
 MEDIA

CANDLE

WWW.CANDLE.COM/WWW1/AXA1_WSDJ

MQSOFTWARE

WWW.MQSOFTWARE.COM

MQSOFTWARE

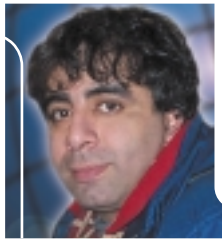
WWW.MQSOFTWARE.COM

Part 2: Recovering a failing WebSphere node

WebSphere Risk Management

BY BASSEM W. JAMALEDDINE

In part one of this series on risk management (*WSDJ*, Vol. 2, issue 4), I introduced the WASLED/WASMON application. In this article I will discuss how to initiate the takeover of a failing WebSphere server by a redundant server, and how to compile and deploy a Web application on this redundant server automatically.



ABOUT THE AUTHOR

Bassem W. Jamaledine is a Web systems engineer who has orchestrated the development of several projects at IBM's T.J. Watson Research Center, including IBM's Java-based network computer and the new generation of WebSphere Application Server technology. Bassem is the author of *IBM WebSphere Application Server Programming* (McGraw-Hill).

E-MAIL

bassem@tcnd.com

Recall the WebSphere region setup shown in Figure 1. The failing server, node1.tcnd.com, has been detected as malfunctioning by the WASMON supervisor and is to be replaced by node2.tcnd.com. WASMON will notify the roaming operator; for example, it will send an e-mail to his or her cell phone. The operator will ask WASMON to take a specific action by triggering a recovery script.

In the next section, I will show you some shell scripts that perform the compilation and deployment of Web applications without the intervention of any graphical tool. Such shell scripts can perform the installation of a Web application remotely.

Shell Scripts That Perform the Recovery

Assembling and deploying Web applications using graphical, visual, or studio tools such as AAT (Application Assembly Tool) and WSAD (WebSphere Studio Application Developer) do not promote the quick and automatic deployment of enterprise Web applications. What is needed is some shell script that can be executed with or without the attendance of an operator to compile, deploy, and register enter-

prise Web applications in WAS. The following sections discuss the importance of such scripts, and show examples.

UNATTENDED INSTALLATION OF A WEB APPLICATION UNDER WAS

In the literature, many books show how to develop or how to manage J2EE Web applications under WAS using a methodology in which a graphical interface of some sort intervenes in the process. Use of a graphical interface in the development or management process will sabotage an unattended installation of a Web application under WAS.

In the distribution code for my book, *IBM WebSphere Application Server Programming*, I introduce the Is/Was Quick Installation (IQI) method to demonstrate the compilation and deployment of an enterprise Web application under WAS. IQI is a method I developed and adapted to release the reader from dependency on a graphical interface, and to allow seamless transitions. Important characteristics of the use of IQI include:

- Clarity (what is being deployed is obvious to the eye; beans go into a directory structure and servlets go into another).

- Understandability.
- Portability (can be run on any Unix platform).
- Version-independent of WAS, less expensive, and eliminates any cascading licensing cost.
- A smart deployment that will show developers exactly what is happening behind the scenes of the J2EE Web application server.
- It promotes system management; the scripts can be used for risk management of a WebSphere region, as you will see next.
- It promotes quick hot deployment; Web applications are easily packaged into archives, as such archives can replace superceded code in a hot deployment. Refer to the IBM Infocenter for more information on hot deployment, and to Chapter 15 of *WebSphere Application Server Programming* for examples of the effects of WAS classloading and hot deployment.

Let's take a look at the IQI method. Listing 1 shows a shell script that can compile, deploy, and register the WASDG application on node1.tcnd.com remotely from any machine on the tcnd.com network. Therefore the application is deployed to a WAS server running on host node1.tcnd.com, such that this latter host accepts remote shell commands from other hosts attached to the same network. We will assume that the network is sitting behind a firewall to eliminate any discussion of security issues.

```
PATH="$PATH:/tools"
```

This adds the /tools directory that is NFS-exported and shared by all nodes on the tcnd.com network. The /tools directory contains important scripts that deploy a J2EE application, such as `refresh.sh`. The WebSphere computing environment is set using `setj.sh` scripts, such as `refresh.sh`, `setj`, etc.. The build process of the J2EE application takes place in the base development directory, pointed to by the `$BASE_DEV` environment variable.


```
cd $BASE_DEV
export BASE_DEV=$BASE_DEV
```

The J2EE tree is being created, compacted, and registered in WAS, then compiled and hot-deployed. Hot deployment requires a couple of seconds of sleep before it picks up, hence the sleep 8 command.

The lynx command in Listing 1 tests the deployment by requesting the DumpEnv servlet from the Web container, and e-mails the result to wasadmin@tcnd.com.

What you have just seen is IQI. It is an unattended compilation and deployment of a J2EE application based purely on basic scripts. One key element to the success of J2EE Web applications is the setting of the computing environment, which must be derived from the WAS environment. This is set with the setj.sh command.

The unattended compilation and deployment is not restricted solely to Web applications, as EJB modules can also be compiled and deployed locally or remotely on a network. Listing 2 shows the shell script to compile, deploy, and register the EJB module WasdgBeans.jar along with the WASDG application (which uses the beans from WasdgBeans.jar) remotely from any machine on the network into the WAS server running on host node1.tcnd.com.

For more details on IQI programming and deployment methodology, consult *IBM WebSphere Application Server Programming*. Readers who are interested in WebSphere and J2EE programming using nongraphical interfaces can refer to the Was:: Perl module and the Crow programming language (available from www.tcnd.com/p9).

TRIGGERS TO REPLACE A FAILING WAS NODE WITH ANOTHER

Consider Figure 1, in which node1.tcnd.com is being replaced by node2.tcnd.com. Such a replacement is simply done by having the HTTP server plug-in on hserve.tcnd.com redirect requests to node2.tcnd.com instead of node1.tcnd.com. This is a matter of string substitution in the configuration file of the WebSphere plug-in. The hredirect.sh script shown in Listing 3 can be executed remotely on any workstation on the network to

swap node1.tcnd.com with node2.tcnd.com at the HTTP server.

We will combine Listings 1 and 3 to create the shell script ledder_webapp.sh, which can initiate the rebuild of the WASDG application on node2.tcnd.com followed by the redirection of HTTP requests to WAS running on node2.tcnd.com. Listing 4 shows ledder_webapp.sh.

The script ledder_ejb.sh, shown in Listing 5, is the result of combining Listing 2 and Listing 3. It will be used as a trigger in WASMON. The script commits the following actions:

- **A compilation and deployment of the EJB module WasdgBeans.jar on the backup node node2.tcnd.com:** The EJB Web module is first compiled, then described and archived into a JAR file that is deposited in the EAR deployment tree of WasdgBeans.ear, then deployed and registered.

```
jall
j2ejb
jar -Mcvf ../DevSessDeploy/
WasdgBeans.ear/WasdgBeans.jar
-C ./ com -C ./ META-INF
....
\${SEAPPINSTALL} -install
Dist/WasdgBeans.ear -ejbde-
ploy true -interactive false
```

- **A deployment of the WASDG application followed by a restart of the WebSphere Application Server:** j2tree creates the J2EE structure; the ln command is used to create a symbolic link from the tree structure to the beans development tree; refreshear reinstalls the application and restarts WAS; svlbuild com-

pires and hot-deploys the application in the WAS runtime tree.

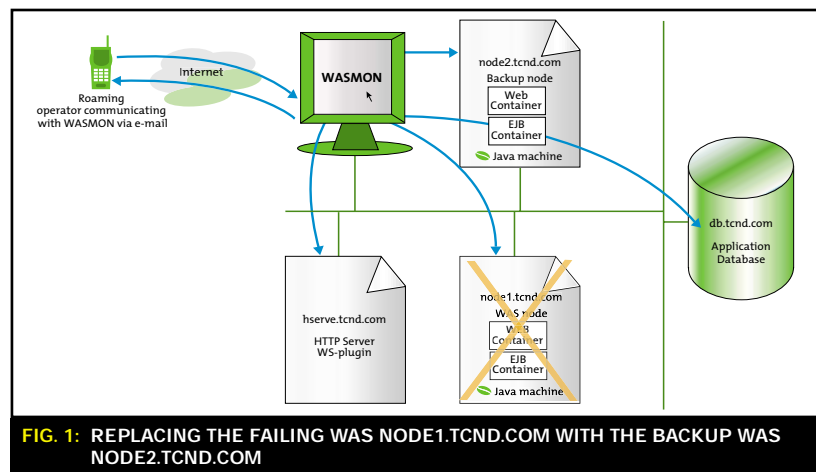
- **A redirection of the HTTP server protocol to the WAS backup node node2.tcnd.com:** Such a redirection is realized with a simple subscript and applied to the plug-in file pointed to by the environment variable \$PLUGIN_FILE. Such a plug-in file is on the same server as the HTTP server hserve.tcnd.com. (Also, for generality, it is assigned to the environment variable HTTP_SERVER.)

```
HTTP_SERVER=hserve.tcnd.com
....
rsh $HTTP_SERVER "chmod 750
/tmp/modplugin.sh"
rsh $HTTP_SERVER /tmp/mod
plugin.sh
```

- **A restart of the HTTP server daemons on the server machine \$HTTP_SERVER.**

Triggering the Recovery Scripts

The ledder_webapp.sh (Listing 4) and ledder_ejb.sh (Listing 5) scripts will be copied to the directory ./TriggersPool as wm_0100_ledder_webapp.sh and wm_0101_ledder_ejb.sh, respectively. The suffixes wm_0100 and wm_0101 are important because WASMON processes such suffixed files and maps them to be identified by the four-digit triggers 0100 and 0101 respectively. Such trigger numbers are then used in the WASMON console (first pane) (see Figure 2) or specified in the WASMON configuration file: wasmon.conf.



To trigger a script we need to associate a condition with the four-digit trigger. We do that by configuring the WASMON supervisor. For instance, consider the content of wasmon.conf as shown below.

```
SUPERVISOR:ON,15, 1,1,
0003,wasadmin@tcnd.com,SUPER
VISOR ALERT,,(1)
```

Whenever WASMON is started, and activated by clicking Listen in the first pane, the directive will trigger the script 0003 because the logical expression (1) will evaluate to true directly. Such a triggering is automatic and does not require the attendance of the operator. The logical expression (1) is used for testing, but we will use a more interesting logical expression to express and assert a malfunction of WAS.

There are two ways to trigger the recovery script in WASMON. The first is automatic, requires no operator, and is settable in wasmon.conf. The second is interrogative and is managed by the conversational aspect of WASMON. In the following section I will show you both ways to trigger the recovery script wm_0101_ledder_ejb.sh, mapped to trigger 0101.

AUTOMATIC TRIGGERING

Consider the following SUPERVISOR directive entered in wasmon.conf.

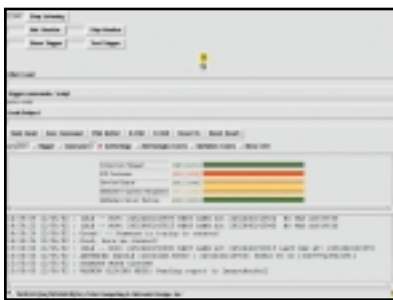


FIG. 2: THE WASMON CONSOLE

```
SUPERVISOR:ON,45, 1,1,
0101,wasadmin@tcnd.com,SUPER
VISOR ALERT,,(
(@BOOL_ACTIVEPORT_hserv.tcnd
.com_80) &&
(!(@BOOL_JNDILOOKUP_node1.tc
nd.com_900_DataAccessCompone
nt) &&
!(@BOOL_URL_node1.tcnd.com_9
080_/wasbook/dumpenv)) )
```

This directive will redirect WASMON to check every 45 seconds that “hserv.tcnd.com’s port 80 is active AND there is no error looking up the JNDI name DataAccessComponent on node1.tcnd.com AND there is no error requesting the DumpEnv servlet from the Web container on node1.tcnd.com.” Yet if hserv.tcnd.com has port 80 active and there is something wrong with WAS containment on node1.tcnd.com, then trigger 0101 is triggered, along with an e-mail to the system administrator.

Recall that the script that maps to 0101 is the recovery script that will redirect the HTTP server to the back-up node2.tcnd.com. It will rebuild and deploy the EJB module and the Web application on node2.tcnd.com.

THE IN PAROLE CONVERSATIONAL ASPECT OF WASMON

The WASMON operator can enter in parole with WASMON. To enable in parole, we will simply specify the keyword PAROLE to precede the logical expression.

```
WASMON MAIL FILE:
/var/spool/mail/wasmon
SUPERVISOR:ON,15,1,1,
0003,wasadmin@tcnd.com,SUPER
VISOR ALERT,,PAROLE(
```

```
(@BOOL_ACTIVEPORT_node1.tcnd
.com_80) &&
(!(@BOOL_JNDILOOKUP_node1.tc
```

```
nd.com_900_DataAccessCompone
nt) &&
```

```
!(@BOOL_URL_node1.tcnd.com_9
080_/wasbook/dumpenv)) )
```

If the logical expression evaluates to true, then an e-mail is sent to the system administrator with a unique identifier. WASMON will wait for a reply that will tell it what to do next. This is typically an action where a script (identified by its four-digit trigger number) will be executed as WASMON enters standby mode.

Figure 3 shows the third pane of WASMON with the messages being logged from the time the failure occurred on node1.tcnd.com to the time node2.tcnd.com took over.

At 07:56:22 an e-mail is sent to the system administrator from WASMON with ticket number TK00 stating that the supervisor has detected some trouble.

```
Date: Mon, 10 Mar 2003
07:56:22 -0500
Message-Id:
<200211091256.gA9CuMq00651@n
ode1.tcnd.com>
To: wasadmin@tcnd.com
From: wasmon@supervise.
tcnd.com
Subject: SUPERVISOR ALERT
cc:
WASMON [TK00 SUPERVISOR]
07:56:22 03/10/03
```

Because the keyword PAROLE preceded the logical expression, WASMON also sent a second e-mail with the unique identifier to wasadmin@tcnd.com so that the operator enters in parole (see Label 1 in Figure 3).

```
Date: Mon, 10 Mar 2003
07:56:22 -0500
Message-Id:
<200211091256.gA9CuMv00657@n
ode1.tcnd.com>
To: wasadmin@tcnd.com
From: wasmon@supervise.
tcnd.com
Subject: SUPERVISOR ALERT
cc:
WASMON [TK00 SUPERVISOR]
07:56:22 03/10/03
Reply back with subject:
3dcd05f6899ed2_DDDD_R:sec:bf
```

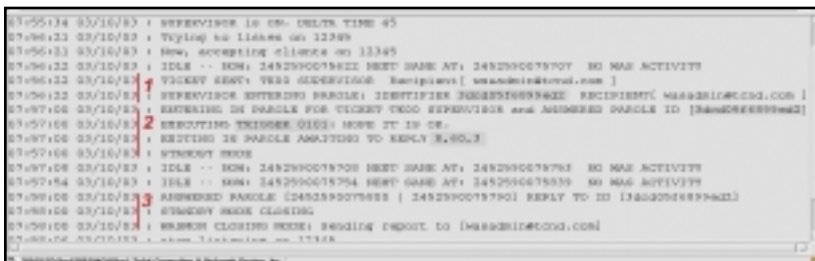


FIG. 3: THE WASMON CONSOLE MESSAGE LOG

PROLIFICS

WWW.PROLIFICS.COM

The roaming operator replies using the unique identifier followed by some attributes dictating what is to be done.

```
mail
wasmon@supervise.tcnd.com
Subject: 3dcd05f6899ed2_
0101_R:60:3
```

This is a string formed from the unique identifier followed by the trigger number 0101, followed by R, asking WASMON to reply back with a report after waiting 60 seconds in standby (so the script will finish execution). A final number, 3 in this case, specifies to switch the WASMON buffer on and set it to 3 lines. The buffer will hold data that flows from WAS (if any) and will be included in the report.

The e-mail is delivered to wasmon@supervise.tcnd.com at 07:56:42.

```
Date: Mon, 10 Mar 2003
07:56:42 -0500
From: Jamaledidine <wasad-
min@tcnd.com>
Message-Id:
<200211091256.gA9CugR00662@
ail.tcnd.com>
To: wasmon@supervise.
tcnd.com
Subject: 3dcd05f6899ed2_
0101_R:60:3
```

Seconds later, at 07:57:00, WASMON fetches the e-mail (see Label 2 in Figure 3), and enters standby mode for 60 seconds before replying back with a report (showing all diagnostic variables). In the meantime the trigger (script) number 0101 is executed and the WASMON internal buffer is enabled with 3 lines.

At 07:58:00, WASMON comes out of standby mode, and sends an e-mail to wasadmin@tcnd.com, closing the ticket with a report that shows the state of the diagnostic variables.

```
Date: Mon, 10 Mar 2003
07:58:00 -0500
Message-Id:
<200211091258.gA9Cw0700671@
odel.tcnd.com>
To: wasadmin@tcnd.com
From: wasmon@supervise.
tcnd.com
Subject: WASMON CLOSING
```

```
REPLY ID [3dcd05f6899ed2]
BOOL_ACTIVEPORT_hserv.tcnd.c
om_80: 1
BOOL_ACTIVEPORT_node1.tcnd.c
om_900: 0
BOOL_ACTIVEPORT_node1.tcnd.c
om_9090: 0
BOOL_ACTIVEPORT_db.tcnd.com_
50000: 1
BOOL_JNDILOOKUP_node2.tcnd.c
om_900_DataAccessComponent:
1
BOOL_JNDILOOKUP_node1.tcnd.c
om_900_DataAccessComponent:
0
...
BOOL_URL_hserv.tcnd.com/_was
book/dumpenv:1
```

Notice that in the report sent to wasadmin, the JNDI name lookup for the DataAccessComponent is successfully resolved on node2.tcnd.com, and the request to the DumpEnv servlet is also successfully served from WAS Web container at node2.tcnd.com. The recovery and takeover of node1.tcnd.com by node2.tcnd.com is successful.

The Writing of WASMON

It was a question about the importance of JNDI name lookup to J2EE Web applications that drove me to write an application to monitor WAS. The answer was obvious:

"It is important to make the JNDI repository highly available; its availability is as important as the availability of a hard drive of a running system. This is basically the same in principle, as the availability of either resource just specified implies the availability of a directory service."

I had this thought in my mind while writing the WASMON application. I was looking for a variable that could tell me about the state of the JNDI name lookup

```
@BOOL_JNDISERVER_node1.tcnd.
com_900
```

and another variable that could tell me if a particular name can be looked up in the EJB container

```
@BOOL_JNDISERVER_node1.tcnd.
com_900_DataAccessComponent
```


But why is it that a JNDI name that did exist at one time cannot be found at another? If you have a 5-year-old son, and one morning you cannot find your car keys, then you know the answer.

In addition, I had to think about answering concretely and satisfying the pedagogical curiosity of readers and students who are wondering about how memory, CPU, network services, and threads are being used by WAS. Other questions must also be answered, such as the changing state of the threads started by WAS, the state of files deployed within WAS that should never be changed, etc.

While WASMON focuses on WebSphere monitoring, the application also allows you to monitor J2EE Web applications. It also supplements the WebSphere development environment with handy commands, such as wglance to get a quick diagnostic report by glancing at a WebSphere region, and wcrawler to quickly crawl over WAS log files and retrieve sensitive information, such as the classpath of the WAS startup machine. WASMON is part of Project-9, which I initiated at TCND.

The next release of WASMON is expected to be configurable from an Internet browser (instead of manual editing), and to support a configuration to detect:

- A variation in the resources consumption of a process, for example a variation in the boundary of memory paging, etc.
- Variations at the thread level, to be made feasible by embedding MrThread (refer to www.tcnd.com/p9 for more information on MrThread) with WASMON.

Writing a monitoring application for a J2EE Web application server may not be that straightforward, in particular when the application is authored by a third-party company with no ties to the maker of the J2EE Web application server. Some of the initial goals of Project-9 included the concrete implementation of a benchmark standard for J2EE application servers; an easy-to-use J2EE programming language (see the Crow language at www.tcnd.com/crow); and a risk management standard for J2EE Web application servers (hence WASMON, discussed in this article). 

LISTING 1

```
#!/bin/ksh

NODE_HOSTNAME=node1.tcnd.com
NODE_PORT=9080

BASE_DEV=/BOOK/16/Code

cat > /tmp/buildwebapp.sh << EOF
#!/bin/sh

PATH="$PATH:/tools"
export PATH
. /tools/env/setj.sh w130

cd $BASE_DEV
export BASE_DEV=$BASE_DEV
./j2tree wasdg.ear webapp wasbook
refreshear
./svlbuild
sleep 8
lynx -dump http://$NODE_HOSTNAME:$NODE_PORT/wasbook/dumpenv |
mail -v wasadmin@tcnd.com
EOF

rcp /tmp/buildwebapp.sh $NODE_HOSTNAME:/tmp
rsh $NODE_HOSTNAME "chmod 750 /tmp/buildwebapp.sh"
rsh $NODE_HOSTNAME /tmp/buildwebapp.sh
```

LISTING 2

```
#!/bin/ksh

NODE_HOSTNAME=node1.tcnd.com

cat > /tmp/buildebjapp.sh << EOF
#!/bin/sh

PATH="$PATH:/tools"
export PATH
. /tools/env/setj.sh w130

# The development tree from where we will deploy the EAR:
wasdg.ear

BASE_DEV=/BOOK/18/Code
# The development tree for the EJB module: WasdgBeans.jar

BASE_DEVBEAN=/BOOK/18/DevSess
# The development tree from where we will deploy the EAR:
WasdgBeans.ear
DEPLOYDIR_DEVBEAN=/BOOK/18/DevSessDeploy
# Export BASE_DEVBEAN and BASE_DEV as shown in Chapter 18

export BASE_DEVBEAN BASE_DEV

# Build the EJB module WasdgBeans.jar
cd \${BASE_DEVBEAN}
jall
j2ejb
jar -Mcvf ../DevSessDeploy/WasdgBeans.ear/WasdgBeans.jar -C
./ com -C ./ META-INF

# Deploy the EJB module WasdgBeans.jar
cd \${DEPLOYDIR_DEVBEAN}
jar Mcvf Dist/WasdgBeans.ear -C ./WasdgBeans.ear .
\${SEAPPINSTALL} -install Dist/WasdgBeans.ear -ejbdeploy true
-interactive false

# Deploy the Web application
cd \${BASE_DEV}
./j2tree wasdg.ear webapp wasbook
ln -s \${BASE_DEVBEAN}/com/tcnd/wasdg/ejb
```

```
\${BASE_DEV}/wasdg.ear/webapp/WEB-
INF/classes/com/tcnd/wasdg/ejb
```

```
refreshear
```

```
./svlbuild
sleep 8
```

```
# Verify the JNDI name lookup
cd /tools
java WasJndi | grep DataAccess | mail -v
maxou@node1.tcnd.com
```

```
EOF
```

```
rcp /tmp/buildebjapp.sh $NODE_HOSTNAME:/tmp
rsh $NODE_HOSTNAME "chmod 750 /tmp/buildebjapp.sh"
rsh $NODE_HOSTNAME "/tmp/buildebjapp.sh > /dev/null 2>&1 "
```

LISTING 3

```
#!/bin/ksh

PLUGIN_FILE=/opt/WebSphere/AppServer/config/plugin-cfg.xml
HTTP_SERVER=node1.tcnd.com

INITIAL_NODE_HOSTNAME=node1.tcnd.com
INITIAL_NODE_PORT=9080

BACKUP_NODE_HOSTNAME=node2.tcnd.com
BACKUP_NODE_PORT=9080

cat > /tmp/modplugin.sh << EOF
#!/bin/sh

PATH="$PATH:/tools"
export PATH
. /tools/env/setj.sh w130

subs $PLUGIN_FILE "Transport Hostname=\"\$INITIAL_NODE_HOST
NAME\" Port=\"\$INITIAL_NODE_PORT\"
Protocol=\"http\" \"Transport
Hostname=\"\$BACKUP_NODE_HOSTNAME\"
Port=\"\$BACKUP_NODE_PORT\" Protocol=\"http\" "
grep -i "transport hostname" $PLUGIN_FILE | mail -v wasad
min@tcnd.com
$STOP_HTTPD
$START_HTTPD
EOF

rcp /tmp/modplugin.sh $HTTP_SERVER:/tmp
rsh $HTTP_SERVER "chmod 750 /tmp/modplugin.sh"
rsh $HTTP_SERVER /tmp/modplugin.sh
```

LISTING 4

```
#!/bin/ksh

PLUGIN_FILE=/opt/WebSphere/AppServer/config/plugin-cfg.xml
HTTP_SERVER=hserv.tcnd.com

INITIAL_NODE_HOSTNAME=node1.tcnd.com
INITIAL_NODE_PORT=9080
BACKUP_NODE_HOSTNAME=node2.tcnd.com
BACKUP_NODE_PORT=9080

# The development tree from where we will deploy the EAR
BASE_DEV=/BOOK/16/Code

#####
# REBUILD AND DEPLOY THE WEB APPLICATION ON THE NEW BACKUP
```

```

NODE #
#####

cat > /tmp/buildwebapp.sh << EOF
#!/bin/sh

PATH="$PATH:/tools"
export PATH
. /tools/env/setj.sh w130

cd $BASE_DEV
export BASE_DEV=$BASE_DEV

./j2tree wasdg.ear webapp wasbook
refreshear
./svlbuild
sleep 8
lynx -dump http://$BACKUP_NODE_HOSTNAME:$BACKUP_NODE_PORT/
wasbook/dumpenv | mail -v wasadmin@tcnd.com
EOF

rcp /tmp/buildwebapp.sh $BACKUP_NODE_HOSTNAME:/tmp
rsh $BACKUP_NODE_HOSTNAME "chmod 750 /tmp/buildwebapp.sh"
rsh $BACKUP_NODE_HOSTNAME /tmp/buildwebapp.sh

#####
#
# MODIFY THE PLUGIN FOR NEW REDIRECTION TO THE BACKUP NODE
#
#####

cat > /tmp/modplugin.sh << EOF
#!/bin/sh

PATH="$PATH:/tools"
export PATH
. /tools/env/setj.sh w130

subs $PLUGIN_FILE "Transport Hostname=\"$INITIAL_NODE_HOST
NAME\" Port=\"$INITIAL_NODE_PORT\" Protocol=\"http\""
"Transport Hostname=\"$BACKUP_NODE_HOSTNAME\"
Port=\"$BACKUP_NODE_PORT\" Protocol=\"http\""
grep -i "transport hostname" $PLUGIN_FILE | mail -v wasad
min@tcnd.com
$STOP_HTTPD
$START_HTTPD
EOF

rcp /tmp/modplugin.sh $HTTP_SERVER:/tmp
rsh $HTTP_SERVER "chmod 750 /tmp/modplugin.sh"
rsh $HTTP_SERVER /tmp/modplugin.sh

```

LISTING 5

```

#!/bin/ksh

PLUGIN_FILE=/opt/WebSphere/AppServer/config/plugin-cfg.xml
HTTP_SERVER=hserv.tcnd.com

INITIAL_NODE_HOSTNAME=node1.tcnd.com
INITIAL_NODE_PORT=9080
BACKUP_NODE_HOSTNAME=node2.tcnd.com
BACKUP_NODE_PORT=9080

#####
# REBUILD AND DEPLOY THE EARs ON THE NEW BACKUP NODE #
#####
cat > /tmp/buildebapp.sh << EOF
#!/bin/sh

PATH="$PATH:/tools"
export PATH

```

```

. /tools/env/setj.sh w130

# The development tree from where we will deploy the EAR:
wasdg.ear
BASE_DEV=/BOOK/18/Code
# The development tree for the EJB module: WasdgBeans.jar
BASE_DEVBEAN=/BOOK/18/DevSess
# The development tree from where we will deploy the EAR:
WasdgBeans.ear
DEPLOYDIR_DEVBEAN=/BOOK/18/DevSessDeploy
# Export BASE_DEVBEAN and BASE_DEV as shown in Chapter 18
export BASE_DEVBEAN BASE_DEV
# Build the EJB module WasdgBeans.jar
cd \BASE_DEVBEAN
jall
j2ejb
jar -Mcvf ../DevSessDeploy/WasdgBeans.ear/WasdgBeans.jar -C
./ com -C ./ META-INF

# Deploy the EJB module WasdgBeans.jar
cd \DEPLOYDIR_DEVBEAN

jar Mcvf Dist/WasdgBeans.ear -C ./WasdgBeans.ear .
\SEAPPINSTALL -install Dist/WasdgBeans.ear -ejbdeploy true
-interactive false

# Deploy the Web application
cd \BASE_DEV
./j2tree wasdg.ear webapp wasbook
ln -s \BASE_DEVBEAN/com/tcnd/wasdg/ejb
\BASE_DEV/wasdg.ear/webapp/WEB-
INF/classes/com/tcnd/wasdg/ejb
refreshear
./svlbuild
sleep 8

# Verify the JNDI name lookup for DataAccessComponent in the
EJB container cd /tools
java WasJndi | grep DataAccess | mail -v
maxou@node1.tcnd.com
EOF

rcp /tmp/buildebapp.sh $BACKUP_NODE_HOSTNAME:/tmp
rsh $BACKUP_NODE_HOSTNAME "chmod 750 /tmp/buildebapp.sh"
rsh $BACKUP_NODE_HOSTNAME "/tmp/buildebapp.sh > /dev/null
2>&1 "

#####
# MODIFY THE PLUGIN FOR NEW REDIRECTION TO THE BACKUP NODE #
#####
cat > /tmp/modplugin.sh << EOF
#!/bin/sh

PATH="$PATH:/tools"
export PATH
. /tools/env/setj.sh w130

subs $PLUGIN_FILE "Transport Hostname=\"$INITIAL_NODE_HOST
NAME\" Port=\"$INITIAL_NODE_PORT\" Protocol=\"http\""
"Transport Hostname=\"$BACKUP_NODE_HOSTNAME\" Port=\"$BACK-
UP_NODE_PORT\" Protocol=\"http\""
grep -i "transport hostname" $PLUGIN_FILE | mail -v wasad
min@tcnd.com
$STOP_HTTPD
$START_HTTPD
EOF

rcp /tmp/modplugin.sh $HTTP_SERVER:/tmp
rsh $HTTP_SERVER "chmod 750 /tmp/modplugin.sh"
rsh $HTTP_SERVER /tmp/modplugin.sh

```

SITETELLIGENCE

WWW.SITETELLIGENCE.COM



Maple Leafs Score on WebSphere's Midmarket "Power" Play

Next-generation Web content management system built using Power Servlets

— BY MATT PUCCINI AND MORGAN SMYTH —

Can you build a custom dynamic Web content management solution and a high-performance dynamic professional sports Web site from scratch on J2EE in a short period of time? Oh yes, it must have Fortune 500 sophistication and be delivered within a midmarket budget. Impossible? Well, that's exactly what Braegen Group, Inc. (www.braegen.com), a Toronto-based e-business solution provider, recently accomplished for Maple Leaf Sports & Entertainment Ltd. (MLSE), owners of the Toronto Maple Leafs hockey team. Braegen's breakthrough Web content management software, called Oriel, is among the first software products in the market to take advantage of "Power Servlets," the servlet middleware technology from Espressiv, Inc. (www.espressiv.com).



ABOUT THE AUTHOR

Matt Puccini is the president of Espressiv, Inc., the company behind Power Servlet technology.

E-MAIL

matt@espressiv.com

Formed in 1927, the Toronto Maple Leafs is one of the NHL's oldest and most decorated hockey teams. Leaf fans are fiercely loyal, hoping for a chance to celebrate the return of Lord Stanley's Cup to Toronto this year. And with their anticipation comes the need to know all the latest stats, standings, and lineup details. For Leaf fans, www.mapleleafs.com is the source for all of the up-to-the-minute action and post-game analysis.

MLSE faced a dilemma. Administering the site was challenging because content revisions happened constantly – and only the site administrator could make changes. As a result, with numerous departments within the organization creating contributions to the site, the site administrator was often backlogged with requests. MLSE wanted a Web content management solution that would be easy for staff members to use and that would allow them to dynamically and directly publish content in real time.

MLSE turned to Braegen to create an integrated Web content management solution and redesign the entire Web site. They wanted a Web site that would be easy to maintain, and would allow internal departments with little technical knowledge the ability to update content. MLSE also wanted a simple and productive system to manage the massive amounts of constantly changing content on the site. Added to the requirements was the fact that the NHL season was under way; Leaf fans could not be left without access to real-time Leafs stats and stories.

To meet these requirements, Braegen designed Oriel, a dynamic Web content management system. It featured an in-line interface that presents users with the actual Web-page template with content management functions (i.e., replace an image, update a cover story, or add a new page). This graphical interface allows contributors with little training or technical skills to become productive. In addition, Oriel needed to be flexible enough to be extended with minimal effort as new features are required.

Power Servlet Technology Achieves New Levels of Price-Performance

The challenge was to deliver a sophisticated Web content management system and improve the performance and stability of the Web site. Braegen first selected a low-cost but scalable platform suitable for MLSE that included WebSphere Application Server Express, DB2, and Linux. Then, after exploring conventional architectural approaches, Braegen's architects chose to use Power Servlet middleware technology

to simplify and streamline application development and provide a higher level of software agility. "The simplicity, productivity, and flexibility of Power Servlets allowed us to build Oriel rapidly but without sacrificing any of the sophistication that MLSE requested," said Morgan Smyth, president and CEO of the Braegen Group Inc. "The reaction to Oriel from MLSE and other prospects has been so tremendous that we are now offering it as a commercial product. And by choosing Power Servlets to develop with, we are able to customize Oriel for new customers at a fraction of the cost."

"The Power Servlet is an innovation whose time has come," said Matt Puccini, president of Espressiv. "The complexity of the middle tier has slowed J2EE adoption by millions of skilled programmers and inflated project schedules and costs. Power Servlets introduce a stateful, structured middleware solution that sets new levels of simplicity, productivity, quality, security, agility, and maintainability for J2EE control programming. We forecast tremendous growth in Power Servlet deployments in 2003, driven by the need for next-generation software products like Oriel."

IBM has been working closely with Espressiv to integrate Power Servlets into WebSphere Studio tooling as a "Perspective" and is introducing the technology to partners and customers at numerous WebSphere Road Shows. "We are pleased that Espressiv has embraced IBM WebSphere software as a strategic platform for Power Servlets," said IBM's Scott Hebner, director of WebSphere marketing. "Power Servlets work with WebSphere to offer our customers an easy-to-use programming environment that streamlines Web application development."

Where Did Power Servlets Come From?

Bruce Cichowlas, the inventor of the Power Servlet, is no stranger to technology innovation. After graduating from MIT at 19, Bruce joined the original Lotus Notes development team and from there went on to invent the sophisticated technologies behind the Kurzweil piano keyboard. "Power Servlets came to me two years ago when I tried to design a Java-based music Web site for my teenagers to build," said Cichowlas, Espressiv cofounder and vice president of R&D. "The complexity I encountered drove me to come up with a simpler approach so that my kids could build it themselves." The result was the Power Servlet, an entirely new middleware for Web applications.

"J2EE is an incredibly open, robust, and scalable platform. But it is also incredibly complex from an application developer's perspective," said Bill Hughes, another Espressiv cofounder and vice president of product development. "The reason is simple: the Web was never designed for building and maintaining enterprise applications. What J2EE needs is a new kind of middleware that removes the burden of state management complexity from developers and architects."

What's the Difference Between a Servlet and a Power Servlet?

Servlets were designed as a page-centric processing model for Web applications. For example, when a user fills out a form and submits that information, a servlet is responsible for processing the information on that page, storing any information from that page that is required by another page (this information defines the application "state"), and displaying the next page to the browser. Servlets give developers a very fragmented, rigid, and fragile model for pulling together application components. Developers must piece these hundreds of small programs together themselves, or use a framework, which is often equally rigid (though maybe faster). Certain methodologies, such as M-V-C, provide a model to guide developers to best practices that make it easier for one developer on a team to build applications using similar structures, yet many developers feel that a richer solution is required to simplify the servlet/controller tier, particularly as users are demanding more sophisticated application functionality.

Power Servlets are stateful, modular application "controllers" that can handle multiple page flows (such as a wizard or master-detail) from a single program. They automatically manage state, providing a simple, intuitive way to integrate the components of a Web application: Web pages, beans, EJBs, Web services, databases, and legacy assets.

The result is that control coding is reduced by up to 90%, making it easier to design, build, modify, and maintain applications. This means that any developer with traditional structured programming skills can write well-architected applications on the J2EE platform.



FIG. 1: ORIEL'S GRAPHICAL USER INTERFACE PROVIDES EASY IN-LINE FUNCTIONS THAT MAP DIRECTLY TO THE WEB SITE



ABOUT THE AUTHOR

Morgan Smyth is the founder and president of Braegen Group Inc., the company that developed Oriel.

E-MAIL

msmyth@braegen.com

Power Servlet programs are written using a Java IDE or code editor with a development tool called Presto for Power Servlets, and then converted at runtime (similar to JSPs) into a set of instructions that tell the Power Servlets which J2EE resources to invoke. Because programmers write these instructions in Java, they can employ ordinary programming constructs to dramatically simplify the complexity of J2EE.

To better understand the power of Power Servlets, let's look at a familiar example: the Java Pet Store.

Power Servlets In Action: The Java Pet Store

Using Power Servlets, the now-famous Pet Store application can be decomposed into two kinds of Power Servlet classes: an event controller and several event-handler controllers. In this example, we will look at the shopping cart controller.

EVENT CONTROLLER

The Java Pet Store application is typically implemented with an event controller responsible for translating end user actions (clicking a link or a button) into a program event. This event controller can be implemented as a single Power Servlet method, called "main()" in Listing 1. This main method reflects the top-level logic for the entire application.

Very simply, this Power Servlet implements the following logic:

- Initialize the database connection and query file (used by the Data Set class later).
- Set the first application action as the login() method in the Power Servlet called "Security".
- Start an event control loop that involves the Power Servlet method representing the next user-requested action (e.g., "doCart"). Note that this action, because it is a Power Servlet class, can implement user work flows that span multiple page displays, such as login() or doCart().
- If the requested action is "logout", then exit the application.

One breakthrough feature of Power Servlets is that user interactions are represented as method calls that return control to the calling program. This is revolutionary for developers, so let's look at a simple example.

In Listing 1, if the `esp.invoke()` call fails (for example, if the program attempts to execute a Power Servlet method that doesn't exist), the `main()` method will execute the statement

```
esp.showMessage("Error! invalid action: " +
    next);
```

This statement does two important things: first, it forwards the page `ShowMessage.jsp` (a template message page) to the JSP engine, substituting the string argument as the message on that page. Then it waits until the user has submitted another request for the PetStore Power Servlet. When that request is received, the program executes the next logical statement (in this case it checks to see whether the next action is "logout").

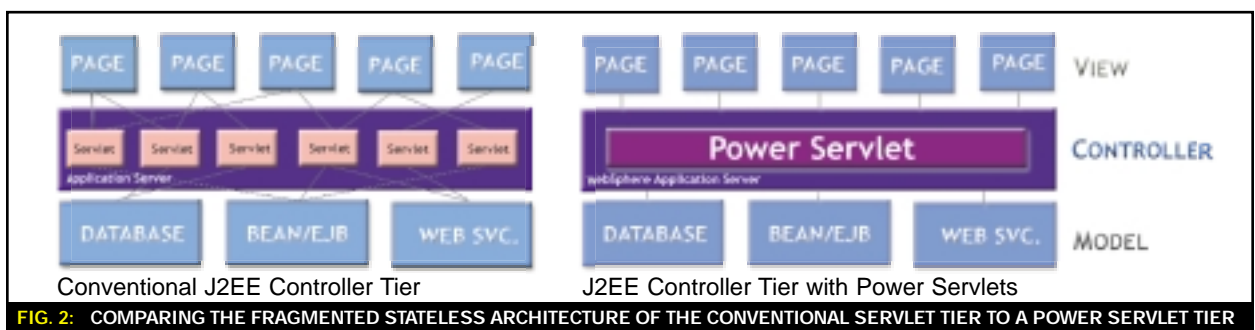
To developers who know J2EE and the Web, this is magic. To developers just learning Web development, it's very natural. Why is it magic to the experienced Web developer? Because the Web in general – and J2EE in particular – does not let a program display a user page and then continue execution of that same program. That is a fundamental limitation of the current servlet tier from the standpoint of application architectures and developer productivity. It breaks the structured programming model.

However, Power Servlets fix that problem by implementing the magic of "freeze-drying," a patented method for halting and resuming the execution of programs that preserves the J2EE scalability model. This means that your application no longer needs to be designed around the limitations of the servlet tier. Rather, it can be written the way you were taught in Programming 101: modular, top-down, coherent, and object-oriented.

SHOPPING CART CONTROLLER

Unlike the event controller `main()`, the Power Servlet method `doCart()` is an event handler controller. It handles user requests to "Add to Cart" or "View Cart". The logic is straightforward:

- First, `doCart()` creates a `FormPage` object, which allows the Power Servlet to treat the JSP page for the cart like a `JavaBean`. This means that there is much less logic in the JSP page – it can focus on simply formatting the user controls and data, and taking user input.
- The "while" loop keeps users processing the cart, unless they indicate that they are finished and want to do something else.
- The page is loaded with data from the `Cart` object.
- The page is displayed to the user.
- The user actions are captured via the `getButtonUsed()` and `getButtonUsedIndex()` methods of `FormPage`. The Power Servlet decides what action to take based on the button selected – remove a cart item, update cart information, show detail on a cart item.
- Unless the user requests to perform another action, the `Cart.jsp` page is redisplayed with its particular view updated based on changes to the `Cart` object.
- `doCart()` is called from `main()`, so when it completes, it returns control to `main()`.



REPORTINGENGINES

WWW.REPORTINGENGINES.COM/DOWNLOAD/WSDJ.JSP

Listing 3 shows the fragments of Cart.jsp that define the FormPage controls (text fields, arrays, and buttons using the FormPage tag library provided with Presto).

Developing Applications from the User's Perspective

Another important characteristic of Power Servlets is that they allow developers to program from the user's perspective. For example, the doCart() or doCheckout() routines correspond precisely to the entire use case that the user describes for those functions. In ordinary servlet programming, the doCheckout() use case would have to be broken into several fragments: loadAndShowCartForReview(), handleCartReviewAction(), loadAndShowShippingPage(), handleShippingPageAction(), loadAndShowBillingPage(), handleBillingPageAction(), loadAndShowConfirmationPage(), handleOrderSubmission(), showOrderSubmissionApprovalPage(), showOrderSubmissionConfirmation(). These fragments cannot be abstracted into a higher-level user function such as doCheckout().

The other user actions – whether they call a single page, require that page to loop (in order to implement validation, previous, and next buttons), or call multiple pages in sequence (such as “checkout” or “register”) – can be implemented as modular Power Servlet methods as well. These methods can be implemented and tested by separate developers with no knowledge of one another, and then be immediately integrated and run.

Presto for Power Servlets and WebSphere Studio

Presto for Power Servlets is the development tool that is used in conjunction with WebSphere Studio and other popular Java IDEs/editors to create Power Servlets. Presto for Power Servlets includes a runtime dashboard, a powerful debugger, and foundation classes. The foundation classes are optional libraries that further simplify Web application development. They include:

- **FormPages:** These expose a bean interface to your JSP page from your Power Servlet program. This enables a VB-like coding model for Web pages, cleanly separating Web page design, application logic, and data binding.
- **DataSets:** These provide a generic data access object to manage SQL interactions. This eliminates the necessity of writing Java classes to interact with your database, accelerating the application development process.

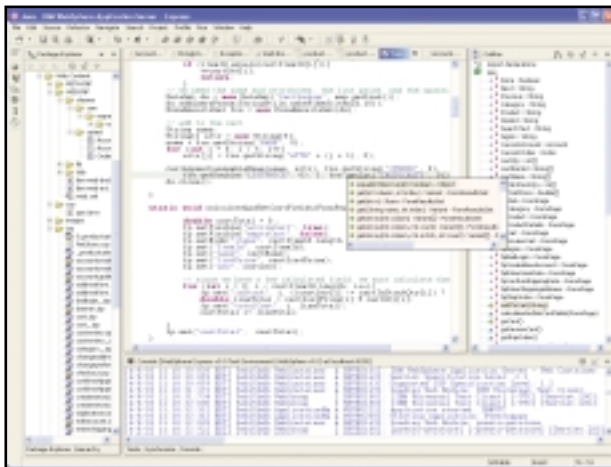


FIG. 3: POWER SERVLET DEVELOPMENT WITH WEBSHERE STUDIO

Presto for Power Servlets has an auto-compiler that generates XML-based instructions that tell the Power Servlet what J2EE resources to call – beans, EJBs, JSPs, Web services, or any other Java classes. It does not, however, need to generate Java source to be compiled and run on the application server. Instead, the Power Servlet uses these instructions to run Java classes and libraries that run on the application server.

“POWER STRUTS”

Struts is a popular MVC framework, yet most developers need to add three important capabilities: workflow, security, and debugging – all features that are inherent to Power Servlets. For example, to add workflow to Struts, a developer can simply call a Power Servlet as a Struts action. The Power Servlet implements an entire workflow process (e.g., master-detail, wizard, or checkout), requiring no changes to the Struts architecture.

Power Servlets also simplify the Struts configuration file, which was never designed to implement workflow. Power Servlets are workflows that are self-contained, coherent, and more secure. These flows can be tested, debugged, and analyzed centrally. JSPs implemented using Struts tag libraries don't need to be changed, although by using Power Servlets they can be simplified. Power Servlets can be incorporated immediately into new or exiting Struts implementations to get these benefits.

Integrating Power Servlets into a Project


While Power Servlets are revolutionary, they are not disruptive to your existing developer resources, business systems, or applications. Power Servlets provide for an incremental migration process.

There are two ways to migrate to Power Servlets:

1. **Extend existing functionality.** Presto for Power Servlets (the development tool for creating Power Servlets) programs can be written to extend an existing J2EE Web application. For example, if a company wants to add a personalization interface to their e-commerce site, this interface could be written in Power Servlets instead of writing it using raw servlets and JSPs.
2. **Replace application logic.** Presto for Power Servlets programs can leave the components of a Web application – Web pages, beans, EJBs, database access – intact and simply replace the content-centric servlet programming that is hard to write, maintain, and debug.

Espressiv offers a free evaluation license for Presto for Power Servlets that can be downloaded from www.espressiv.com.

Leafs Success, Oriel Future Opportunity

Coinciding with the pre-playoffs push, within the first three weeks of the new mapleleaves.com site going live, fan traffic increased by more than 60%. Through the new design, Braegen has been able to improve the site's download performance and stability significantly. “We think these statistics verify our design and approach with Oriel,” says Smyth. “In addition to the rapid development, MLSE also realized the added benefits of increased performance, stability, and scalability.” No doubt; and Oriel has made the management of mapleleaves.com. one of Oriel's greatest fans. 

LISTING 1: EVENT CONTROLLER, POWER SERVLET METHOD

```
import esp.*;           // Essential Power Servlet classes

class PetStore {
    /*
     * nextAction specifies the name of next the Power Servlet
     * method to execute. It is set based on the link or
     * button the user selects on the Web page.
     */
    static String nextAction = "";
    static void setNextAction (String s) { nextAction = s; }
    static String getNextAction () { return (nextAction); }

    /*
     * This method runs when the user enters the URL
     * http://localhost:8080/pps/csp/PetStore.java
     */

    public static void main() {
        // Initialize SQL DB connection and query file
        Setup.initDB("/Data/PetstoreDB");
        Setup.loadQueries("/queries/Petstore.queries");

        // This is the main event controller for Pet Store
        setNextAction("Security.login");
        boolean done = false;
        while (!done) {
            String next = getNextAction();
            /*
             * esp.invoke(next) runs the Power Servlet method
             * with the name specified by the String "next".
             * Each Power Servlet method calls
             * setNextAction() to tell the event controller
             * what to do next.
             */
            if (! esp.invoke(next))
                esp.showMessage("Error! invalid action: " + next);
            done = next.equals("logout");
            esp.showMessage("Thanks for visiting Pet Store!");
        }
    }
}
```

LISTING 2: SHOPPING CART CONTROLLER (USING POWER SERVLETS)

```
public static void doCart() {

    // FormPage provides a bean interface to
    // the form controls defined as tags in Cart.jsp
    FormPage fp = new FormPage("/jsp/Cart.jsp");

    boolean done = false;
    while (! done) {
        // load the form with data from the Cart
        // object, which is implemented as a collection.
        // FormPage methods such as fromObject() know to
        // map an array (of CartItems in this case)
        // to form fields defined in a repeater.
        fp.fromObject(cart.asArray());

        // display the form to the user
        fp.show();

        // Cart.jsp form has a table of cart entries,
        // with a "remove" button (deleted "and Update")
        // for each cart entry. The form also has an
        // "update" button and a "checkout" button
        // to invoke the doCheckOut() Power Servlet
        // workflow.getButtonUsed() returns the button name,
        // which determines the application flow.
        String button = fp.getButtonUsed();

        // getButtonUsedIndex() returns the cart row
        // selected by the user in the repeater table.
```

```
int i = fp.getButtonUsedIndex();
if (button.equals("remove"))
    // delete only the selected row
    cart.delete(i);
else if (button.equals("update")) {
    // modify the cart from the user input
    // table in Cart.jsp
    cart.setQty(fp.get("qty"));
    cart.setItemId(fp.get("itemId"));
    cart.setName(fp.get("name"));
    cart.setInStockQty(fp.get("inStockQty"));
    cart.setListPrice(fp.get("listPrice"));
}
else if (button.equals("itemName"))
    // displays the item detail page and handles
    // any changes made to that page
    doCartItemDetail(cart, i);
else {
    // if no form-specific action requested
    // then the next action will be the
    // name of the Power Servlet method to handle
    // the next user action (e.g., "logout")
    setNextAction(button);
    done = true;
}
}
```

LISTING 3: CART.JSP

```
<!-- Cart.jsp: JSP code fragments that define FormPage
controls that can be manipulated in a
Power Servlet -->

<!-- Specify the FormPage tag library -->
<%@taglib uri="http://espressiv.com/presto/frm" prefix="f" %>
...
<f:form>
<table border="0">
    <tr>
        <td><jsp:include page="mylist.jsp" flush="true" /></td>
        <td align="left">
            <font size="5" color="black">
                Shopping Cart:
            </font></td>
        </tr>
    </table>
    <table>
        <tr>
            <td>
                <!-- The repeater defines an array of cart items. The
                size of the repeater array is determined by
                the number of array elements specified by
                the array argument to fp.fromObject() -->
                <f:repeater name="items">
                    <tr>
                        <td><!-- f:button and f:link tags define "buttons" returned
                        to the Power Servlet -->
                        <td><f:button name="remove"
                            image="../images/button_remove.gif"
                            repeater="items"/></td>
                        <td><f:text name="itemId" repeater="items"/> </td>
                        <td><f:link name="name" repeater="items"/></td>
                        <td><f:text name="inStock" repeater="items"/></td>
                        <td><f:text name="listPrice" type="double" format="$"
                            repeater="items"/></td>
                        <td><f:text name="qty" edit="true" type="int"
                            repeater="items"/></td>
                        <td><f:text name="totalCost" format="$" type="double"
                            repeater="items"/></td>
                    </tr>
                </f:repeater>
                <tr>
                    <td><font size="3" color="white">Total:</font></td>
                    <td><f:text name="cartTotal" type="double" format="$"/>
                    </td>
                </tr>
            </table>
            <f:button name="update"
                image="/images/cart-update.gif"/>
        </table>
        <f:button image="/images/button_checkout.gif"
            name="checkout"/>
    </tr></table>
```


JAVAONE

WWW.JAVA.SUN.COM/JAVAONE/SF

JAVAONE

WWW.JAVA.SUN.COM/JAVAONE/SF



A WebSphere Approach to Strategic Market Development

Alignment of corporate goals and cultures is key

— BY DEREK BILDPELL —



ABOUT THE AUTHOR

Derek Bildfell is the program director of IBM's WebSphere Market Development Strategy. He leads the market development strategy and management for the WebSphere Application Servers and Tools. These products are at the core of the infrastructure for IBM's e-business software platform. Most recently, Derek was the business and marketing lead on the Rational Software acquisition. Previously, Derek was the Program Director for the WebSphere Application Servers, as well as SecureWay Software Brand.

E-MAIL

dbildfel@ca.ibm.com

Successful relationships in market development are based more on the alignment of the organizations' goals and cultures than on any individual relationship involved. It is exactly this alignment that has made the Rational relationship and the ultimate acquisition of that company such a successful venture for both companies.

Conversely, a lack of alignment is usually the largest factor that negatively impacts partnerships and joint ventures.

But before we get to any analysis of the alignment of potential market development business partnerships, we follow a thoughtful and thorough process to determine where we want to invest our limited resources, including time, engineering resources, and marketing investments. The question on the minds of most potential partners is, "How does WebSphere select partners for strategic relationships?"

Within the WebSphere application servers and tools space, we take a very simple approach to our process. We are in the business of delivering the infrastructure and tools that allow our customers to build an on demand e-business. In order to further our success in this mission, we look at three key areas where external business partnerships can help.

1. Business relationships that accelerate our strategy
2. Business relationships that help IBM penetrate new markets
3. Business relationships that improve our competitiveness and solution capabilities

You can think of this as three approaches to growing our share of the pie. Accelerating the strategy usually increases the overall size of the Java application server opportunity marketplace. Penetrating new markets grows the size of the

opportunity in which WebSphere can effectively compete. And improving our competitiveness serves to allow WebSphere to win a larger share of customer engagements.

To illustrate our approach, I'll look at some recent examples in each of these areas.

Business Relationships That Accelerate Our Strategy

WebSphere, like any other IBM business, has a very well-defined strategy. Within this strategy, IBM identifies some major initiatives that are critical to our success. Two examples of tactics that can be impacted by intelligent market development strategies are increasing the number of developers and applications that target WebSphere as a deployment platform, and increasing the adoption rate of open standards-based applications and application development.

Starting in 2001 we looked at the community of business partners and began implementing initiatives that would accelerate the achievement of these objectives. Everything we do inside WebSphere must be integrated and consistent to achieve the maximum effectiveness of our resources. For example, the work on Eclipse, making it open source and driving partners to it as a point of integration for all roles in the application development life cycle, maximizes the number of developers building applications that both leverage leading open standards and run on WebSphere application servers.

From a strategic partner perspective, it was evident that Rational Software (www.ibm.com/rational), with over 700,000 enterprise application developers, would be key to growing the developer population in the professional development community. We worked with Rational Software through Eclipse adoption, as well as a series of OEM deals, to ensure that the Rational developer community would have a first-class ability to build and deploy WebSphere applications. IBM includes Rational's Clearcase technology with WebSphere Studio to provide fundamental basic software configuration and management. Rational leverages



Studio to provide the IDE and other technologies required to support the Java developers using their XDE offerings.

After years of a mutually successful relationship with Rational, we began to seriously consider the next steps in the relationship. At the same time, IBM needed to be even more aggressive in providing enterprise-scale solutions for developing software for the on-demand era. As we analyzed the overall market of application development tools and services businesses, we settled on a strategy to pursue the acquisition of Rational, more in order to further IBM's wider goal of providing the best software development capability.

As companies move to the on-demand era, they are finding that competency in flexible software development is critical. This flexibility will come from a well-defined process (the Rational Unified Process), development tools and techniques that are integrated from requirements definition through to deployment, reusable component architecture, and an open architecture for both building and deploying solutions. So while Rational gives IBM exactly what we needed as a software development capability, they also dramatically increase the number of application developers and tools that support the IBM WebSphere platform.

A second type of strategic marketing relationship initiative that we pursue is more community based than a specific one-to-one business relationship. As we look at our strategy for WebSphere, it's apparent that we need to accelerate the industry adoption of the compelling open standards that WebSphere and IBM are implementing in our products. To that end, we considered the challenges to Web services adoption in late 2001. While the standards were being received with great enthusiasm by the market, the road map for how to start was difficult.

Enter the strategic market development team. We worked with many parts of IBM, including the Web services marketing team, the jStart services team, and our partner technical support and enablement organizations to build a program-focused initiative to support WebSphere partners who were aggressively adopting Web services. This initiative, called Web services on WebSphere (WoW) (www.ibm.com/webSphere/wow), actively seeks out the leading services and software providers in Web services, and helps them to enable their products and services to support WebSphere. This community – 25 advisory council members and over 80 registered partners – works together to provide a more complete solution, whether by working with the other members of the WoW community or by tailoring their solutions to be consistent with IBM's Web services strategies and offerings.

Community-targeted initiatives such as WoW, work we've done in the past on security, and of course the Eclipse-based WebSphere Studio partner community, help to provide the best solutions for our customers as well as improve the strategic position of WebSphere and its partners in the market.

Business Relationships That Help IBM Penetrate New Markets

The second major thrust is to develop relationships that help IBM penetrate new markets. In a growth industry like ours, we perform exceedingly well by winning in our core market as it grows. Our longer-term strategic interest is more in increasing the size of the market that we can compete in. Obviously, in the core Java and open-standards worlds, WebSphere has become a formidable competitor. But what of the lower-end markets? Unique industry niches? Or emerging product arenas?

Emerging technology and market opportunities are generally addressed by the core market development teams, with support from our research divisions as well as Rod Smith's emerging technologies team. For unique industry niche opportunities, IBM has a robust industry solutions unit structure. The industry solutions units are responsible for building unique solutions consisting of hardware, software, and services for their specific markets. For example, the finance industry solution unit will look at the requirements for Web-based trading solutions, and will pull from all divisions of IBM the solution components they need, and then work with industry partners to find unique business development opportunities.

The new focus of the WebSphere marketing team is mostly the small and medium business segments. In this space, we had major success last year with the WebSphere - Express launch. WebSphere - Express is a great entry-level solution that includes the tools and runtime required for smaller, less technically complex applications. With this launch we targeted a completely different business partner community through a Web-based recruitment and enablement model, attracting over 600 partners worldwide. For example, a new company in Boston, Espressiv, has a Power Servlet product called Presto that is extremely well-suited to the procedural developer, such as someone with deep Microsoft Visual Basic skills. Presto allows non-Java programmers to build applications for WebSphere using a procedural approach. We worked with Espressiv early to allow Presto to run well with WebSphere - Express, since this is an excellent solution for many midmarket companies. A second company in the WebSphere - Express initiative, Braegen Systems, used Espressiv to build the next generation of the Toronto Maple Leafs Web site (www.mapleleafs.com) 6 (see accompanying article on p.16). This new approach to market development made WebSphere - Express one of our most successful product launches.

Another new market that we have successfully addressed is Macromedia ColdFusion applications. IBM has had a long and prosperous relationship with Macromedia, the leader in Web-based development. For example, IBM has worked with Dreamweaver for years, making sure that it works well with our WebSphere Studio application develop-

ment tools. We also acquired the LikeMinds collaborative filtering technology from Macromedia in 2001, which we integrated into Commerce Suite and our Portal solutions.

Macromedia ColdFusion was perfect for our strategic market development team to pursue. When Macromedia acquired Allaire in 2001, we both saw the opportunity to have a ColdFusion for Java/WebSphere. There are over 200,000 servers running ColdFusion applications, and many of these companies were interested in adopting WebSphere as their corporate standard since they were buying more robust packaged applications that run on WebSphere.

We worked with Macromedia to develop and test the ColdFusion for WebSphere offering, ensuring that we took advantage of the performance and scaling capabilities of WebSphere. In September of 2002 we announced that IBM would be selling Macromedia ColdFusion for WebSphere. This product allows current ColdFusion-based applications to run on the corporate standard WebSphere Application Servers, while dramatically improving the scaling and performance attributes of the underlying application server.

Similar new market development opportunities exist in many niche migration markets. As customers consolidate and standardize their e-business infrastructures, earlier investments need to be maintained while newer, more strategic, or cost-effective solutions are implemented. Another great example of this is our work with Prolifics and their offering, XMLink, which helps customers migrate and interoperate Tuxedo applications toward WebSphere. There are thousands of applications that are tied to the proprietary Tuxedo transaction processing software. We have worked with Prolifics to provide cost-effective solutions to help these customers improve the ROI of these investments by creating XML links to the Tuxedo application and building out additional value applications on WebSphere. More recently, Prolifics has provided services and solutions to help customers migrate from their expensive Tuxedo infrastructure toward the open standards-based WebSphere infrastructure (www.prolifics.com/websphere).

Prior to our work with Prolifics, and Macromedia, these market segments were prohibited from taking advantage of IBM's enormous investments in WebSphere. Our strate-

gic approach to market development is changing this, increasing IBM's market opportunity while improving our customers' ROI with a very nondisruptive approach.

By allocating the market development opportunities to the teams best able to make strategic and long-term business commitments, IBM has been very successful with our new market initiatives, from WebSphere - Express to relationships like the one with Macromedia, to our industry solutions, such as the eGovernment Portal solution.

Relationships That Improve Our Competitiveness and Solution Capabilities

The third major thrust of the strategic market development team has been to establish business relationships that improve our competitiveness and solution capabilities. In this area, we work with established partners to round out our WebSphere offerings, or to identify a partner with capabilities that help our sales organizations. These relationships are the most tactical; usually we identify a partner and provide strategic guidance to the business development team working with that partner.

There are too many examples to list all of the partners that have delivered solutions for WebSphere that improve our competitive position. (They can be found at www-3.ibm.com/software/info1/websphere/partners/index.jsp?tab=home&aka=bp/wspartnerlist.) Let me give you an interesting example from the early days. In 2000, we determined that improving our capabilities in Java application performance and problem determination would help our deployments. At that time, most delays in production deployments of WebSphere were related to application problems. The trouble was, there weren't many solutions to really address the complexity of the Java application architecture and runtimes. We started working with a company called Wily Technology, and their product, Introscope. With a small amount of support, we were able to help optimize Wily's products to provide best-of-breed support for the WebSphere Application Server. Wily now has Introscope PowerPack for WebSphere Application Server and customized PowerPack offerings that help customers with very specific challenges with WebSphere MQ and WebSphere CICS Transaction Gateway.

Most relationships share elements of each of the strategic drivers mentioned above but can usually be classified as predominantly fulfilling one of these drivers. Once a business partner, or community of partners, is identified as being strategically important to our success, then we would look at the alignment of these partners and build marketing plans with the most aligned partners to achieve the goals of the relationship. We look at alignment on many planes, such as:

- Architectural alignment
- Cultural alignment
- Offering overlaps or competition
- Industry or customer alignment
- Geographic alignment
- Relationship history

There isn't a specific scorecard to use to assess the strength of the alignment, as it is usually a relative assessment and differs based on what we're trying to achieve. For example, in trying to penetrate new markets, we expect very low geographic or customer alignment, and might not be too concerned with cultural alignment. Conversely, when looking to improve our customer solution capabilities, we look for high industry, geographic, and cultural alignment.

IBM PartnerWorld

IBM PartnerWorld is a comprehensive marketing and enablement program for all business partners across IBM – whether you're a software solution provider, developer/ISV, reseller, or influencer. Through this program IBM provides a vast set of benefits to business partners based on the different needs and qualifications of the partners.

Through IBM PartnerWorld we can provide the following resources:

- Selling
- Marketing
- Information on our products and technologies
- Training and certification
- Business discounts and awards

There are three membership levels:

- **Member:** The most basic level of membership, requiring only simple online registration
- **Advanced:** Recognizes and rewards business partners who have made significant commitments to IBM technologies
- **Premier:** Long-term strategic relationships for business partners who are investing in IBM technologies

By becoming an IBM partner you can access many of the WebSphere partner-enablement programs (www.ibm.com/websphere/partners). To learn more about IBM's partner programs, please visit www.ibm.com/partnerworld.

Conclusion


Our partners are finding IBM's approach to market development refreshing and effective. "IBM's strategic market development team proved their agility by working effectively with our early-stage company to bring Power Servlet technology to market," says Matt Puccini, president of Espressiv, Inc. "In today's risk-averse environment, it is difficult to drive adoption of new innovations such as the Power Servlet without cooperation from mainstream companies. IBM is by far our most aggressive and effective partner at delivering the marketing resources we need to successfully launch our business. It's no surprise that the vast majority of our new business opportunities can be traced directly to our relationship with IBM."

So whether you have a new solution that takes WebSphere into a market we never considered, or you're competing in IBM's space and can make our joint solutions more valuable to our customers, we have programs and offerings to help you. The question on everyone's mind is likely "What's the first step?" First, you need to assure yourself that you understand what IBM PartnerWorld has to offer. IBM PartnerWorld (www.ibm.com/partnerworld) is the most comprehensive business development program in the industry, supporting individual developers, software companies, consultants, integrators, services providers, resellers, and value added distributors.

If you are more interested in a tighter relationship with IBM WebSphere, we have provided some additional resources to consider. Start with the WebSphere Innovation Connection Online (www.ibm.com/websphere/partners). Here you'll find all the resources you need to integrate and communicate with the various parts of the WebSphere organization.

Over the past 2 years, we've introduced more than 700 new independent software vendors to working with WebSphere partners. We have taken the lessons learned from our experience and built a streamlined road map that helps potential partners get connected and productive with WebSphere as quickly as possible. From the WebSphere Innovation Connection Online site, click on either "Partner with IBM for WebSphere Software" (in the left sidebar) or "Easy Steps to Partnering with WebSphere Software" (on the right-hand side).

This site is structured to guide the partner through the likely steps in three areas: getting started, getting solutions enabled, and helping with go-to-market activities. Getting started is easy; simply join PartnerWorld for Developers, then follow the easy steps that will guide you to the best resources for your particular interests, such as building WebSphere Portal Portlets, or integrating with WebSphere Studio application development solutions.

The next step is usually to get enabled. In this phase, most partners need technical help and education. IBM offers the broadest array of online, seminar-, and fee-based education and support to partners, which is well described in this road map. Once you are enabled, you'll want to get your product to market as soon as possible. We'll list your solutions in IBM's solutions database, and provide additional support statements as appropriate. IBM has over 9,000 partners, many of whom are looking for more solutions built on WebSphere that they can sell to their customers. We also provide useful information such as advice for a successful partnership, and answers to relevant FAQs. If you can't find what you need through this excellent road map approach, we also have a direct contact that can help, at aimisv@us.ibm.com. 

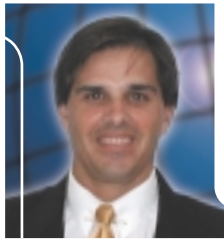
PERFORMANT

WWW.PERFORMANT.COM/WEBSHERE1

Best practices

WebSphere Application Server and Database Performance Tuning

BY MICHAEL S. PALLOS

**ABOUT THE AUTHOR**

Michael S. Pallos, MBA, is a senior solution architect for Candle Corp. (www.candle.com) and has 18 years of experience in the IT industry. He is a consultant to some of Candle's largest corporate customers, a featured speaker at industry conferences, and a doctoral student at Walden University. Candle, based in El Segundo, California, provides solutions to manage and optimize business-critical infrastructures, systems, service levels, and other information technology assets.

E-MAIL

michael_pallos@candle.com

Optimization of the production runtime environment boosts the performance of WebSphere Application Server applications, allowing organizations to harness the full potential of their hardware and software investments. Performance tuning of the network and database interfaces are two of the most important elements of the optimization process. This two-part series explores best practices for performance tuning as it relates to the persistence layer of WebSphere Application Server and a database management system (DBMS).

Organizations running WebSphere Application Server AE are most likely using three databases: the WebSphere Application Server configuration repository, the session persistence repository, and the application data repository. If, however, an enterprise runs the single edition version of WebSphere Application Server, it is running only two database images: one for session persistence and another for application data. The configuration repository, when running the single edition version, is contained in a single XML configuration file. Figure 1 depicts the WebSphere Application Server AE architecture with repositories. Multiple repositories increase the number of database connections and require organizations to optimize each connection independently.

The interface between WebSphere

Application Server and the DBMS(s) is critical, since this connection can either act as a bottleneck or facilitate high throughput.

Best Practices

An exploration of best practices in the following areas will provide a road map for the vital task of optimizing the storage and retrieval of persistence data:

1. Database connection pooling
2. Prepared statement cache
3. Session persistence
4. Enterprise JavaBeans
5. Java database connectivity
6. Application monitoring

We explore the first three areas in this article and will review the balance in Part 2 of the series.

Connection Pooling

Whenever an application uses a

database resource, a connection must be established, maintained, and then released when the operation is complete. These processes consume time and IT resources. The complexity of accessing data from Web applications often imposes a strain on the system. Web applications are more taxing on the system for several reasons. Specifically, Web users:

- Connect to and disconnect from the database more often than users of non-Web applications
- Normally have shorter interaction times, making the database connection the longest component in the transaction
- Participate in unpredictable usage patterns, placing more demands on the database connection

WebSphere Application Server provides connection pooling to address some of the challenges of database access. Connection pooling is the process of creating a predefined number of database connections to a single data source. This process allows multiple users to share connections without requiring each user to incur the overhead of connecting and disconnecting from the database. Connection pooling can speed up application processing significantly.

When a user makes a request to a data source, WebSphere Application Server examines the connection pool for an existing resource connection. If one exists, it is provided to the user. The system then processes the database request. Once processing is complete, the connection resource is released from the user and placed back into the connection pool.

IBM used Sun Microsystems' JDBC 2.0 Option Pack API to incorporate connection pooling. The system administrator sets the connection pool thresholds, which are easily configured using the WebSphere software administrator tools.

According to IBM's *DB2 UDB/WebSphere Performance Tuning Guide*, best practices for connection pooling that will allow an application to achieve optimum performance include the following:

- **Use the same method to both obtain and close the connection.** This approach allows the connection resource to be released efficiently to the connection pool.
- **Minimize the number of Java Naming and Directory Interface (JNDI) lookups, an expensive process in application performance.** The JNDI expense is incurred by the out-of-process network call required for each JNDI lookup. To limit JNDI lookups, the developer should create a separate method to handle these calls. Once created, the separate method can be called from the servlets `init()` method or from an EJB's `ejbActive()` method.
- **Do not declare connections as static objects.** If a connection is declared as static, then it is possible to have the same connection used on different threads at the same time. This creates a problem for the connection pool and for the database.
- **Do not close connections in the finalize method.** As discussed earlier, connections should be opened and closed using the same method. There is, however, a school of thought that promotes closing everything in the finalize method, ensuring one location for complete closure. The finalize method is not called until the object is garbage-collected, since that is when all finalized methods are called. Closing connections in the finalize method can lead to a delay in releasing the connection and should be avoided.

- **If you open a connection, close the connection.** Closing a connection is not absolutely required, since WebSphere Application Server will implicitly close the connection after the connection has timed out. (It has a default value of 30 minutes.) An explicit close, however, expedites the process and follows good application development practices. More specifically, according to the *DB2 UDB/WebSphere Performance Tuning Guide*, "[I]t is very important that

ResultSet, Statement, PreparedStatement, and Connection objects get closed properly in an application. If connections are not closed properly, users may experience long waits for connections to time out, and delay return of the connection to the free pool."

- **Do not manage data access in container-managed persistence (CMP) beans.** Developers should create CMP beans with the understanding that the container is going to handle the persistence. If one wishes to assume responsibility for persistence, then bean-managed persistence (BMP) should be used. Incorporating BMP processes in CMP beans slows performance.

Prepared Statement Cache

WebSphere Application Server applications may access a database using JDBC via a SQL callable statement, SQL statement call, or SQL prepared statement call. A callable statement removes the need for the SQL compilation process entirely by making a stored procedure call. A statement call is a class that can execute an arbitrary string that is passed to it. The SQL statement is compiled prior to execution, which is a slow process. Applications that repeatedly execute the same SQL statement can decrease processing time by using a prepared statement. A prepared statement redefines a

statement call by separating the compilation process and adding substitution variables. This approach allows the application to prepare the statement once (via compilation) and reuse it at execution multiple times by leveraging different variables.

As a best practice, use prepared statements instead of a SQL statement call for applications that repeatedly execute the same SQL statements.

Session Persistence

Many applications, based on their size and activity level, find the use of memory local session cache on a local application server acceptable for session processing. As the application grows, however, so does its complexity. Such growth may require the incorporation of fault tolerance and redundancy, or the establishment of server clusters. As an application grows, the system administrator may also wish to achieve a higher level of control over the environment. Any or all of these requirements may render in-memory use on a local machine inadequate, requiring the use of session persistence to a database.

In order for data to be persisted to a database, the data must first be serialized. Serialization is the process of writing information to the database or disk, or flattening it out to be trans-

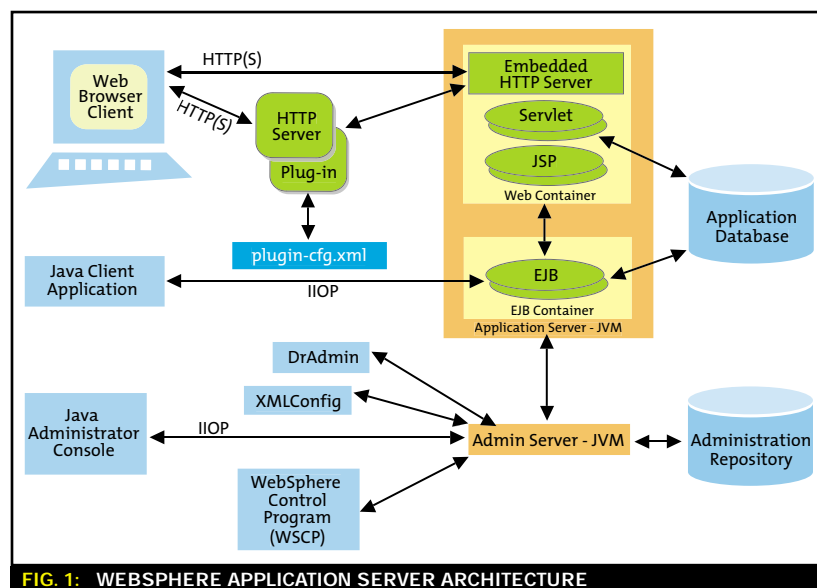


FIG. 1: WEBSHERE APPLICATION SERVER ARCHITECTURE

ferred over the wire. To serialize data, you must implement the Java interface `java.io.Serializable`.

Persistent session management does not impact the API. Applications that may require session persistence, therefore, should include `java.io.Serializable` in the code. When the application then scales using session persistence, the code will not require modification.

Once session persistence has been implemented, the session manager will keep the most recent 1,000 sessions in cache memory (this number is configurable) and persist the rest. This process allows the session manager to reduce the number of database accesses required for processing, thereby reducing memory requirements. The decision flow for the session manager is captured in Figure 2.

Following are several best practices to consider when planning for session persistence.

- **Enable session persistence.** Consider making Java objects held by `HttpSession` serializable, even if the application currently operates with local session management. This approach will equip the application should the Web site grow to a size that requires persistence session management. The practice of preparing for growth makes the transition from local to persistent session manage-

ment transparent to the application.

- **Reduce session size.** Reducing session size becomes particularly important when leveraging persistence sessions. The larger the session, the longer the write to the database, which, in turn, requires more disk i/o. According to *Database System Concepts* (McGraw-Hill), “disk access takes tens of milliseconds, whereas memory access takes a tenth of a microsecond.”

When developing sessions, place easily retrieved information in the application and not in the session. Also, remove stale or old data from the session. According to the *DB2 UDB/WebSphere Performance Tuning Guide*, “[T]he best performance will be realized with session objects that are less than 2K. Once the objects start to exceed 4–5K in size, a significant decrease in performance can be expected.”

- **Release HttpSession when finished.** WebSphere Application Server will release the session once the session has expired, but this can take up to 30 minutes when using the default configuration parameter. Explicitly release the `HttpSession`, forcing an immediate release of memory and garbage collection.

- **Choose persistence options.** IBM offers extensive guidelines for choosing persistence options in the *DB2 UDB/WebSphere Performance Tuning Guide*.

- **Avoid creating HttpSession by JSP by default.** Java Server Pages (JSP), according to the J2EE specification, create `HttpSession` objects by default. In the event you are not going to use the `HttpSession` object, processing requirements would be reduced by not creating the object. To prevent the default `HttpSession` object from being created, add

```
<%@page session="false" %>
```

to the JSP.

- **Tune the cache size.** The default setting provides for 1,000 session objects to be cached. Reducing the number of session objects in turn reduces the amount of required memory for session cache.

- **Add additional application server clones.** To implement vertical scaling, WebSphere Application Server allows the administrator to create multiple instances, or clones, of the application server. This process spreads the requirement for memory across multiple JVMs, thereby reducing the burden of a particular instance. You can also achieve horizontal scaling by adding additional physical hardware resources (servers) to the WebSphere Application Server configuration and implementing WebSphere Application Server clustering. Attempt vertical scaling first to save on purchasing additional hardware, followed by horizontal scaling, with the exception of standard baseline redundancy, which typically supports two physical hardware servers for failover.

- **Tune multirow persistence management.** Multirow session support allows session information obtained from multiple JSPs and servlets to be stored in the session database using multiple rows. For example, Table 1 represents the data collected from session AB12345, depicting information (Session Object data) that a user retrieves.

Referencing Table 1, suppose that the computer needs only a small piece of data, such as first.name, “Michael.” Using multirow persistence, the user can retrieve only the row in the table that houses the first.name variable, leaving the much larger `CandleDemo.String` in the session database until the servlet requests it. However, if the system were leveraging single-row

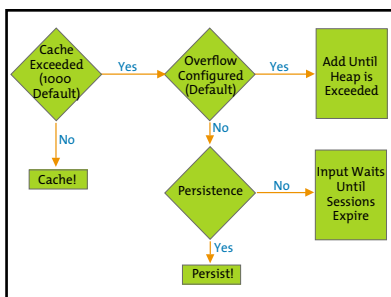


FIG. 2: IN-MEMORY CACHE OVERFLOW ALGORITHM

Source: *DB2 UDB/WebSphere Performance Tuning Guide*

Session ID	Web Application	Property	Small Value	Big Value
AB12345	CandleDemo	CandleDemo.First.Name	"Michael"	
AB12345	CandleDemo	CandleDemo.last.Name	"Pallos"	
AB12345	CandleDemo	CandleDemo.String		A huge string...

TABLE 1: SIMPLIFIED MULTIROW SESSION REPRESENTATION

–continued on page 37

International Web Services Conference & Expo

WEST

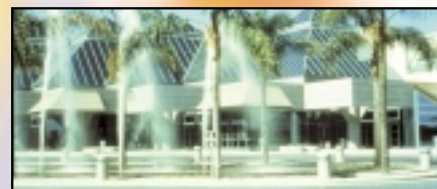
Web Services Edge 2003

web services
conference & expo

EDGE
conference & expo

SEPT. 30 - OCT. 2, 2003

Santa Clara, CA



**EXTENDING THE ENTERPRISE
WITH WEB SERVICES THROUGH JAVA,
.NET, WEBSphere, MAC OS X
AND XML TECHNOLOGIES**



XML

WebSphere

Microsoft
.net



Mac OS X



LINUX **EDGE**
conference & expo

web services **EDGE**
conference & expo

BOSTON

February 24-27, 2004

Featured technologies and topics will include:

- Focus on WebSphere
- Focus on Java
- Focus on .NET
- Focus on Mac OS X
- Focus on XML

For more information visit
www.sys-con.com

or call

201 802-3069

Contact information: U.S. Events: 201 802-3069 or e-mail grisha@sys-con.com



Over 100 participating companies will display and demonstrate over 300 developer products and solutions.

Over 2,000 Systems Integrators, System Architects, Developers, and Project Managers will attend the conference expo.

Over 60 of the latest sessions on training, certifications, seminars, case studies, and panel discussions will deliver REAL World benefits, the industry pulse and proven strategies.

WebServices
JOURNAL

JAVA
DEVELOPER'S JOURNAL

WebSphere
DEVELOPER'S JOURNAL

XML
JOURNAL

.NET
DEVELOPER'S JOURNAL

SAMS

WebLogic
DEVELOPER'S JOURNAL

wireless
BUSINESS TECHNOLOGY

LINUX
BUSINESS TECHNOLOGY

CF Advisor

ColdFusion
DEVELOPER'S JOURNAL

PowerBuilder
DEVELOPER'S JOURNAL

Java is a registered trademark of Sun Microsystems, .NET is a registered trademark of Microsoft, Mac OS X is a registered trademark of Apple Computer, Inc., WebSphere is a registered trademark of IBM. All other product names herein are the properties of their respective companies.

**WEB SERVICES EDGE WEST 2003
CALL FOR PAPERS NOW OPEN**

Submit your papers online at:

www.sys-con.com/webservices2003west

PRODUCED BY
SYS-CON
EVENTS



Stefan Van Overtveldt
Program director for
WebSphere Technical
Marketing

Scott Hebner
Director of marketing for
WebSphere infrastructure
software

Joe Anthony
Program director for
WebSphere Extensions
marketing

Derek Bildfell
Program director for
business development

Bernie Spang
Program director for
WebSphere Studio marketing

Aimee Munsell
Program director for
WebSphere Application
Server

Exciting Plans Loom on the WebSphere Horizon

Part 3 of a conversation with the WebSphere marketing team

Eclipse and IBM's open platform are helping to shape an exciting future for WebSphere, according to the WebSphere marketing team, headed up by Scott Hebner, director of marketing for WebSphere infrastructure software.

The WebSphere marketing team includes Joe Anthony, program director for WebSphere Extensions marketing; Derek Bildfell, program director for business development; Aimee Munsell, program director for WebSphere Application Server; Bernie Spang, program director for WebSphere Studio marketing; and Stefan Van Overtveldt, program director for WebSphere Technical Marketing.

WebSphere Developer's Journal editor-in-chief Jack Martin recently had the opportunity to talk with Hebner and his team in a wide-ranging and exclusive discussion. This third installment focuses on tools for rapid application development, the broader development community, the need to address the full life cycle of an application, and how Eclipse has significantly shortened time-to-market for new tools.

WSDJ: Bernie, what are the three most important things that you see coming out in Studio over the course of the next 12 months? Do you have anything on the horizon that's just going to melt the developers' minds when they see it, that will make them say, "This is so cool! I can't stand it! I have to buy another copy of Studio!"

Bernie: Well, what will melt people's minds are the tools that we are going to be delivering in the next 12 months that are focused on rapid development – easy development that doesn't require you to be a Java programmer for Web applications, form-based development, as an example. The transition that we just went through to create the open platform and to bring our Java development tools over was a significant step. Now we are focused on tools for the platform that further accelerate development. The second thing is building on what we talked about earlier ("Solutions, Not Technology, Drive WebSphere Products," *WSDJ*, Vol. 2, issue 4) with grid computing – and the Web services that we mentioned. What you see in Studio right now is just the start, just scratching the surface of a service-oriented approach to development of applications and building the software as a set of services that can be loosely coupled and quickly integrated. It's leveraging our service-oriented infrastructure that's going to blow developers away and make them more productive.

WSDJ: Do you have the business rules going into this? Is that where you are going with this?

Bernie: That is another thing that will make a significant dif-

ference that will be tied in with this. What you are seeing today in Studio with Web services is focused on workflow – the choreography of services into an application and the evolution of that. You see the beginning of bringing business rules into WebSphere Application Server Enterprise. Adding business rules support means the ability to build the intelligence into the service to operate based on business rules that can be managed without having to redevelop the application – that can be managed by the businesspeople. We change the behavior of the business application because the business has changed and the rule has changed. You are going to see an increase in that capability.

WSDJ: And the programmatical skills that they are going to need are going to be significantly reduced – is that what you are saying?

Bernie: Yes. Because what we are focused on now is not the basic programming environment. We have that. Now, in a sense, we are focusing on delivering developer solutions. That says I need to build a loosely coupled system that ties it together, that's modifiable by the businessperson by adjusting a rule. We are not going to give them programming tools to do that; we are going to give them the development environment that guides them through doing that to generate as much of the code as possible. We're simplifying the interface to the developer so that they can do it rapidly and productively and – we often forget the other, most important, thing – with high quality. The best practices for how to implement this are built into the development model.

Derek: The thing that I would want to add is that as we look at the broader community and the broader IBM portfolio – at the tools that we intend to acquire with Rational, for example – we're looking at how we can establish a development model where our customers really tie the business user and developer together, by using Holosofx for example, for business modeling. We also think about the key development communities, such as ColdFusion developers, UML developers, and Visual Basic developers. What do we have to do both with the product portfolio and through creative relationships with selected partners and communities of partners? You see the results of this focus in our deals such as Macromedia ColdFusion for WebSphere and the recent acquisition of Rational Software.

WSDJ: What is Holosofx?

Stefan: Holosofx is a company that we acquired last year – a very small company, but they are a leader in three key areas. One is business process modeling. So it's a tool that, for example, anyone who is proficient in PowerPoint can use at that level.

It's for someone who needs to understand how business should operate. What you can do is take the business model that you develop with Holosofx and plug that into Rational's application modeling environment and have an IT architect model from an application perspective – design how that

business process should work throughout your IT infrastructure. Then you can take that application model, which is greatly enhanced, and you can plug it right into WebSphere Studio Application Developer, for example. You have all of your class libraries, all of your application's structure. It is predefined for you.


At the back end of this, you use some of the other Holosofx tools like the monitoring function to see how one of the applications in production is performing against the predefined business process. This is what we mean when we talk about the notion of how business processes can be modified and put into production very rapidly. This is what we have enabled by getting those two companies, in addition to our WebSphere Studio portfolio. The fact that all of those tools are based on the same Eclipse foundation enables an immediate integration between those capabilities.

Derek: That's one dimension. Another important dimension of the complement that Rational provides as a strategic partner or as an IBM division is the life-cycle management from requirements definition and maintenance through to the testing and building on WebSphere Studio capabilities. That life cycle – addressing the full life cycle of an application from the idea of a business need, to capturing the requirements through to the testing, deployment, and ultimately, retirement of the application – is another dimension in which you are going to see WebSphere Studio grow, complemented by the Rational offering.

Bernie: There is one other exciting thing that I just want to add to what you said that's going to knock developers' socks off – your comment about research and that in every cubicle there's a different researcher with a different thing. One of the other very important values of the Eclipse platform to IBM is these researchers – who used to develop stand-alone tools, used to create their own code base from the ground up, their own user interface, their own this and that. Even if we saw something that we liked and needed and that the customer wanted, it would take us 6 months, 18 months, sometimes 2 years, to turn what the research guys did into a product. Now, they have plug-ins to Eclipse that we can put on alphaWorks tomorrow, that WebSphere Studio developers can download and plug in, and it's an integrated part of their development.

WSDJ: So is it a corporate edict now that everything at Watson has to be Eclipse compliant?

Bernie: The good news is, it's not a corporate edict. It didn't need to be. The base is cool. They like it; they want it; they don't have to re-create what's there.

When I started two years ago, every couple of weeks someone would come to my office and show me the great new tool that we should be putting into WebSphere Studio and I'd say, "What base?" This would take us years to incorporate in with everything else we are doing. Now everything that I'm seeing is, "Let me show you this cool new Eclipse plug-in." 

"What you see in Studio right now is just the start, just scratching the surface of a service-oriented approach"

Features offer ample flexibility and control

Development Using Servlet 2.3 Filters and Event Listeners

BY RAJU MUKHERJEE, SWARAJ PAL, AND DEEPENDRA SHRESTHA

With every release of the Servlet API new features are added that offer new functionality and provide developers with more flexibility when designing new Web applications. With a changing user base and business requirements, most developers would prefer an easy, cost-effective solution to shape their applications rather than rearchitecting existing applications.



ABOUT THE AUTHOR

Raju Mukherjee is a senior architect at Noospherics Technologies, Inc., with wide experience in industry and government. He has worked extensively with J2EE, WebSphere, and databases. He has led various WebSphere-based projects for many enterprises. Raju contributes articles on the WebSphere family of products to various magazines.

E-MAIL

raj_mukherjee@
yahoo.com

The application event listener and servlet filter features are included in the Servlet 2.3 API for these kinds of tasks and much more. In this article we will look at using these new features from the WebSphere Studio Application Developer v5 IDE. Event listeners are Java classes modeled as JavaBeans, which listen to Servlet-Context and HttpSession objects life-cycle events (create and destroy) and changes to attribute events (binding, modifying, and unbinding). On the other hand, filters are objects that intercept requests for static and dynamic resources from the client (static pages, JSPs, servlets). Both components are packaged within the WAR (Web archive) module and configured in web.xml and managed by the Web container at runtime.

It is possible to have multiple filters or listeners for each event type, and you can configure the order of invocation in web.xml. One important aspect to note is that the container doesn't synchronize notifications to attribute listener classes, so it is the responsibility of the develop-

er to maintain thread safety of these classes for state maintenance. The best aspect of this new feature is that you can enhance and extend your application using these features – in most cases without affecting the existing application – or use these components to bind an out-of-the-box Web application with your application. For example, you could write and configure a filter that would wrap the servlet response so that it could be sent to wireless devices very easily. In the next couple of sections we will walk through the development process for both listener and filter.

Using WebSphere Studio for Development

Open the J2EE perspective. Create a J2EE 1.3 Enterprise Application Project. Switch to the J2EE navigator view within the perspective. Expand the Web module folder under the project and expand the Java Source folder. Create your package structure, under which you will create the filters and listeners (create a separate package for clarity). Right-click on the package and Select

New → Select either Filter or Life-cycle Listener (see Figures 1 and 2), which will bring up the appropriate wizard. The wizard will create the classes with all the methods. Open the class in the editor view and start coding.

Filters

You might still be wondering why you need filters and how you can use them. Well, as the name suggests, you can use filters to intercept requests or responses to or from the controller, make a custom setting for header information, wrap/modify the request/response, or redirect requests by performing a check on the parameter passed. You can also format the input/output stream, interact with external resources (e.g., authentication can be redirected to some other components), reuse existing code, temporarily block requests from your business partner without even changing your existing code, or prevent malicious request attacks.

In this sense, the filter could well act as a part of the Controller servlet or control the flow of the application by itself. Possible usage includes user authentication, auditing, logging, image conversion, data compression, encryption/decryption, XML transformation, response data formatting for different devices, etc. Later we will create an authentication filter to understand how to work with filters and to learn how they can be used to redirect requests after performing checks. Remember that filters are multithreaded.

Filter Life Cycle

The Filter class implements javax.servlet.Filter. The Web container calls the init() method to initialize the filter, puts the filter into service, and sets the configuration object, FilterConfig. The Web container creates the javax.servlet.FilterConfig object and passes it to the Filter class while calling the init() method. The FilterConfig interface contains the methods to retrieve the filter's init parameters, defined in web.xml, and also can get the current servlet context. The Web

container calls the `destroy()` method to take the filter out of service. The Filter class's `doFilter` method (ServletRequest request, ServletResponse response, FilterChain chain) performs the actual processing of the intercepted request.

The FilterChain object holds the filter information that needs to be processed. To clarify, filters can be chained just like the servlets. So one filter can call another filter before the request and response is passed to the called servlet. Filter-chaining definitions are also defined in `web.xml`. You can also define the order in which each filter in the chain will be executed.

To summarize, the Web container calls the `init()` method once in the life-time of the filter to initialize the filter. Subsequent requests are handled by the `doFilter()` method, which can be called any number of times. Within the `doFilter()` method you could first toy with the request and response object, then pass the control to the next filter in

the chain or dispatch the request and response to another resource. The control for the next filter or the actual requested resource is passed by calling the `chain.doFilter(request, response)` method. When all the filters in the chain finish processing the call and send the response back, the first filter in the chain receives control of the response object. You can perform some additional processing with the response object before allowing the response to go back to the client.

Redirect Request with AuthenticationFilter

Let's build an AuthenticationFilter that will intercept requests for the Controller servlet and authenticate the user. Follow the steps mentioned in the previous section. While using the wizard to create the filter you will have the option to define the init parameter, along with the URL and servlet mapping for the filter. If you don't mention any of these, the wizard by default creates a URL mapping with the name of the filter itself. We will define our mapping later by editing `web.xml`, so accept the defaults for now. Note that if you want to use more than one filter in the chain, you need to manually edit the `web.xml` file.

WebSphere Studio thus creates the skeleton of the Filter class in a snap. The only thing we will do here for this example is insert our code into the `doFilter()` method (see Listing 1). As you can see in the listing, you will check user authentication in the usual manner. To get to the session object you should cast the ServletRequest to `HttpServletRequest`. If the user is unauthenticated, we don't need to pass the request to the controller servlet; instead we would forward the request to our reporting.jsp (failure.jsp) by means of the RequestDispatcher object. If authentication is successful, then we allow the response to go to the Controller by calling `chain.doFilter()`. Since there are no more filters in this chain, the control is passed to the servlet as a post method. Essentially, `chain.doFilter()` will call the Controller's `doPost()` method. Before sending the request to the Controller, you can set request attributes based on the authenticated user credentials, which will be used by the Controller and its helper

classes to process the request. As an example, we have set a "login" attribute to the request, which we check in the Controller, which sends the response back to "success.jsp" (see Listing 2). The response will go back to the filter, where you can again modify the response object, and then will eventually be forwarded to "success.jsp".

Wrap Request/Response

With the introduction of ServletRequestWrapper, ServletResponseWrapper, HttpServletResponseWrapper, and HttpServletResponseWrapper in the Servlet 2.3 API, you can now create your own wrapper class to extend these classes.

```
public class
myRequestWrapper extends
HttpServletResponseWrapper {
    public myRequestWrapper (
        ServletRequest request){}
}
```

Wrap the request and response objects with your own Wrapper class and use those wrapped objects in the Filter class. By using this technique (Wrapper pattern), you will gain more control over the request and response object since you can override the methods of the superclass.

```
public void
doFilter(ServletRequest
req, ServletResponse resp,
FilterChain chain) throws
ServletException,
IOException {
    ....
    myRequestWrapper rw = new
    myRequestWrapper( req);
    chain.doFilter(rw, resp);
    ....
}
```

Using this you can change the header information, set/modify request parameters, and even get access to the encoding part of request and response. For example, this feature will really come in handy when you want to add support for your business partner to share data. You can use a response wrapper that formats your data to a format your partner's application understands.

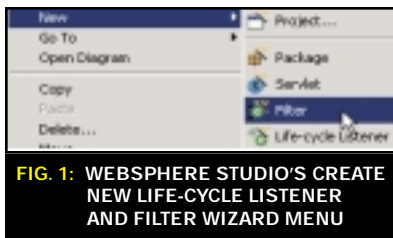


FIG. 1: WEBSHERE STUDIO'S CREATE NEW LIFE-CYCLE LISTENER AND FILTER WIZARD MENU



FIG. 2: CREATE LISTENER AND FILTER WIZARD IN WEBSHERE STUDIO



ABOUT THE AUTHOR

Swaraj Pal is a senior WebSphere consultant at Noospherics Technologies, Inc. He has seven years of IT experience and has worked with WebSphere for several years. Swaraj writes articles on the WebSphere suite of products for various magazines and white papers on WebSphere performance and tuning using the Mercury suite of products.

E-MAIL

swarajpal@yahoo.com

Listeners

As mentioned earlier, listeners listen to the Web container's life-cycle events and hence are managed by the container. There can be multiple listeners for the same type of event; these will be invoked in order. There are four interfaces:

1. **javax.Servlet.ServletContextListener**: Listens for context events such as create and destroy
2. **javax.Servlet.ServletContextAttributeListener**: Listens for context attribute changes
3. **javax.Servlet.http.HttpSessionListener**: Listens for session create, invalidate, and timeout events
4. **javax.Servlet.http.HttpSessionAttributeListener**: Listens to Session attribute changes

Follow the steps mentioned earlier to create the skeleton of your desired listener class within WebSphere Studio. While creating the new listeners, the wizard allows you to choose from the four interfaces you want to implement in your class. We will take a look at two listeners, ContextListener and SessionListener.

The method signature of the ContextListener class will look like the following code snippet. The WebSphere Studio wizard will automatically create all the methods available in both interfaces.

```
public class ContextListener
implements ServletContext
Listener,
ServletContextAttribute
Listener
{
    /** javax.servlet.Ser
vletContextListener methods
    public void context
Destroyed(ServletCon
textEvent arg0) {}
    public void context
Initialized(ServletContext
Event arg0) {}
    /** javax.servlet.ServletCon
```

```
textAttributeListener meth
ods
    public void attributeAdded
(ServletContextAttribute
Event arg0) {}
    public void attributeRe
placed(ServletContextAttri
buteEvent arg0) {}
    public void attributeRe
moved(ServletContextAttri
buteEvent arg0) {}
}
```

Similarly for HttpSessionListener, implement the javax.Servlet.http.HttpSessionListener and the javax.Servlet.http.HttpSessionAttributeListener. Look at the available methods per interface.

```
public class HttpSession
implements HttpSession
Listener, HttpSessionAttri
buteListener
{
    /** javax.servlet.http.
HttpSessionListener methods
    public void sessionCreated
(HttpSessionEvent arg0) {}
    public void sessionDe
stroyed(HttpSessionEvent
arg0) {}

    /** javax.servlet.http.Http
SessionAttributeListener
methods
    public void attributeAdded
(HttpSessionBindingEvent
arg0) {}
    public void attributeRe
placed(HttpSessionBinding
Event arg0) {}
    public void attributeRe
moved(HttpSessionBinding
Event arg0) {}
}
```

In both examples notice the respective event object being passed by the container to the methods.


ServletContextEvent and HttpSessionEvent get you access to the current servlet context object and the HttpSession object, respectively. ServletContextAttributeEvent and HttpSessionBindingEvent contain the name and value of the object added, replaced, or removed from the context or session. You can also get the current session object from HttpSessionBindingEvent.

Filter, Listener Definitions in web.xml

A filter can be configured to act on a single resource or a group of resources. You can specify the servlet name under the "filter-mapping" node, or you can specify a URL pattern. Within the URL pattern you can be very innovative by using wildcard characters like /*; /*.jsp, etc., just like with servlet mapping. Refer to the bold lines in Listing 3, which indicate the filter's order of execution by the container. Notice that for both filters the servlet name is defined the same way, to tell the container that both filters act on Controller.

On the other hand, the listeners are defined one after another, and the container will invoke them in the order in which they are defined. Depending on the specific requirement for listening to the life-cycle events, you should order your listeners. So context listeners will be listed before the session listener. Listeners are defined before the servlet definition starts.

Conclusion

As we have seen thus far, application event listeners and filters give developers ample flexibility and control over Web application functionality. Although this discussion lays out the initial findings, you can do wonders by fully exploring the possibilities. Get creative and harness the power of these features. 



ABOUT THE AUTHOR

Deependra Shrestha is a Sun Certified Java programmer and IBM Certified WebSphere developer with extensive experience in analysis, design, development, and administration of WebSphere and J2EE applications. He is a senior programmer for Client Network Services, Inc. Deependra has been involved in a number of WebSphere and J2EE projects for various large organizations.

E-MAIL

deependra202@
yahoo.com

LISTING 1: AUTHENTICATIONFILTER.JAVA

```
public void doFilter(ServletRequest req, ServletResponse resp,
FilterChain chain) throws ServletException, IOException {

    String nextPage;
    RequestDispatcher rd = null;
    //check USERID and PASSWORD
    if (req.getParameter("userid") != null) {
        if (!(req.getParameter("password").equals("pass
```

```
word"))
        && (req.getParameter("userid").equals("user")))) {
            ArrayList actionreport = new
            ArrayList();
            actionreport.add("Logon failed ...");
            (((HttpServletRequest) req).getSession())
            .setAttribute(
                "actionreport", actionreport);
        }
```



```

        nextPage = "failure.jsp";
        System.out.println("Response got from the
        Filter");
        //forward request directly to next page instead of
        //Controller Servlet
        rd = req.getRequestDispatcher(nextPage);
        rd.forward(req, resp);

    } else {
        req.setAttribute("login", "loginsuc
        cess");
        //forward request to the Controller Servlet
        //control to next filter or resource
        chain.doFilter(req, resp);
        //Include code to execute on the way back to the
        //client
        System.out.println("Response got from the
        servlet");
    }
} else {
    rd = req.getRequestDispatcher("Welcome.jsp");
    rd.forward(req, resp);
}
}
}

```

LISTING 2: CONTROLLER.JAVA

```

protected final void doPost(HttpServletRequest request,
        HttpServletResponse response) {

    // beginning codes
    //-----User is authenticated
    if (((String)request.getAttribute("login")).equals("login
    success")){
        ArrayList actionreport = new ArrayList();
        actionreport.add("Correct Password");
        session.setAttribute("actionreport",actionreport);
        nextPage="success.jsp";
    }
    if(dispatch){
        RequestDispatcher rd =
        getServletContext().getRequestDispatcher(nextPage);
    } else { rd.forward(request, response);
    }
}

```

```

session.invalidate();
    }
    // ending codes
}

```

LISTING 3: WEB.XML (FILTER CHAIN AND ORDERING)

```

<filter> First Filter in the chain for Controller
<filter-name>Authentication</filter-name>
    <display-name>Authentication</display-name>
    <filter-class>
com.servlet.filter.Authenticationfilter </filter-class>
</filter>
<filter-mapping>
    <filter-name>Authentication</filter-name>
    <servlet-name>Controller</servlet-name>

< -- <url-pattern>/Controller</url-pattern> -- >
</filter-mapping>
<filter> Second Filter in the chain for Controller
<filter-name>HTTPWrapperfilter</filter-name>
    <display-name> HTTPWrapperfilter </display-name>
    <filter-class>
com.servlet.filter. HTTPWrapperfilter </filter-class>
</filter>
<filter-mapping>
    <filter-name> HTTPWrapperfilter</filter-name>
    <servlet-name>Controller</servlet-name>
</filter-mapping>
<listener>
    <listener-class>
com.servlet.eventlistener.ContextListner</listener-class>
</listener>
<listener>
    <listener-class>
com.servlet.eventlistener.SessionListner</listener-class>
</listener>

```

PERFORMANCE TUNING

—continued from page 30

persistence, it would ask for first.name and would also get the huge.string since the data is part of the AB12345 session object. When retrieving only selected variables located in the session object, multirow persistence saves time on data retrieval and serialization overhead. Even if multirow session management is incorporated, data contained in the session objects should be kept small, targeting 2K in size and not exceeding 4–5K.

• Tune the session timeout interval.


The WebSphere Application Server default setting for a session time-out is 30 minutes. Depending on the type of Web site being tuned, this number may be too high. At the same time, you must avoid setting the number too low, as it could frustrate users. Ensure that users have ample time to complete online forms. Users who invest a significant amount of time in completing

an online document and then discover their session has timed out when they select Submit may not return to the site or recommend it to others.

Conclusion

The WebSphere Application Server provides an exceptional framework for running distributed Internet applications. Organizations can achieve service-level requirements by optimizing WebSphere software configuration. Best practices serve as a definitive road map for this critical process. Incorporating best practices for connection pooling, prepared statement cache, and session persistence creates an environment in which organizations are empowered to reduce costs associated with hardware and software, while simultaneously enhancing the user experience with faster Internet processing. In the second installment of this series, I will explore strategies for leveraging Enterprise JavaBeans, JDBC, and application monitoring.

References

- Alur, N., Lau, A., Lindquest, S., and Varghese, M. (2002). "WebSphere Application Server and DB2 UDB Performance." *DB2 UDB/ WebSphere Performance Tuning Guide*. IBM Redbooks.
- Erickson, D., Lauzon, S., and Modjeski, M. (2001, August). *WebSphere Connection Pooling: www-3.ibm.com/software/web-servers/appserv/whitepapers/connection_pool.pdf*
- Silberschatz, A., Korth, H., and Sudarshan, S. (2002). *Database System Concepts (4th ed.)*. McGraw-Hill Higher Education.
- *Java 2 Platform, Standard Edition, v1.3.1 API Specifications*: <http://java.sun.com/j2se/1.3/docs/api/>
- *JDBC Data Access API, JDBC 2.0 Optional Package API*: <http://java.sun.com/products/jdbc/index.html>
- *JDBC Data Access API*: <http://indus.try.java.sun.com/products/jdbc/drivers> 

From a memory usage perspective

Improving Performance of J2EE Applications

BY WILFRED C. JAMISON



ABOUT THE AUTHOR

Wilfred C. Jamison, PhD, is a performance analyst and software engineer for IBM WebSphere Application Server and a member of the WebSphere Performance team. Wilfred focuses on Java memory management issues and is currently leading the effort on WebSphere performance on the Linux platform. His interests include security, Web services, and language processing.

E-MAIL

wjamison@us.ibm.com

In the Java programming model, one of the best-known performance issues is what we refer to as the garbage collection bottleneck. Garbage collection (GC) reduces the degree of concurrency in the JVM, where at worst only the garbage collector thread executes while all other threads are suspended, thus producing no work for the application until the collection is over. Consequently, the longer the GC takes, the longer the threads are suspended and therefore the less throughput and the slower response time will be. Garbage collection also introduces scalability problems in multithreaded applications and SMP systems.

Advances in garbage collection research focus on increasing the degree of concurrency during garbage collection. Some of them, such as parallel collectors, concurrent garbage collection, and incremental garbage collection, are already introduced in the newer JVMs. Despite these new approaches, garbage collection remains a fundamental challenge.

The focus of this article is to give the readers a closer look at the cost of memory usage and how it affects performance. Then I present some ways in which developers and software engineers can improve performance of J2EE applications by reducing the cost of garbage collection.

How Expensive Is an Object?

The cost of an object is directly attributed to the amount of memory overhead and the management cost associated with it. One of the primary reasons why performance can be a problem is the way memory is allocated and col-

lected dynamically in Java. All nonstatic objects, which comprise the majority of objects in a Java application, are created from the runtime heap. The heap itself is protected whenever memory is allocated or collected. Furthermore, the JVM specifications provide semantics in which threads have a local copy of memory used in the heap. The copies then have to be synchronized with the master copies. Thus, as more and more memory is used and collected, more overhead is incurred along the way.

The most fundamental cost of an object is incurred during its creation, which is dependent on the following:

- **Class size:** The more methods and attributes are defined, and the more complex the methods are, the bigger the resulting bytecode will be.
- **Class structure:** The deeper the inheritance relationships are, and the more object compositions there are, the more complicated the class structure will be.
- **Class paths:** The more entries there

are in the classpath, the longer the search time will be.

- **Available memory:** The less memory that is available, the more time is spent on managing and allocating the amount of memory required.

These costs are distributed in different stages of object creation.

- **Class loading:** If the class of the object being created has not been loaded in memory yet or if it has been garbage-collected, then class loading is included in the object creation procedures. The cost includes traversing through the classpath and searching through an archive or directory listing. Thus, a considerable file system search overhead is involved. It is wise to place the paths of the most commonly used classes early in the classpath. The actual class loading is performed in a synchronized fashion. The cost of class loading is a function of the size of the class. Furthermore, if the class is a subclass or if it implements an interface, then the superclasses or interfaces will have to be loaded as well, if they are not loaded already. Classes and interfaces are loaded into the method area of the JVM. If not enough space is available to load a class or interface, garbage collection is triggered.
- **Object memory allocation and initialization:** The next stage is the actual creation of the object whereby memory is allocated for the object. Typically, memory for the attributes of the object must be allocated, including some system-level storage, e.g., monitor, waiters list, etc. If there is not enough memory, garbage collection starts. Note that the total memory required by the object would be the sum of the memory requirements of each of the superclasses and interfaces in its inheritance tree. Any initialization of attributes is also performed. Note also that the initialization may involve creating new objects, in which case another round of object creation is performed.

- **Constructor chaining:** At this stage, the constructors of the object are executed. Each and every constructor in the hierarchy chain is executed, from the topmost down. Note also that within the constructor, new objects may be created.

The Cost of Garbage Collection

Note that object creation is the primary reason why garbage collections occur. Garbage collection can take place in any of the stages of object creation. It can also occur multiple times in a single object creation event. The performance of garbage collection varies depending on the current size of the heap and the constraints specified, such as the minimum percentage of available memory, etc.. The performance cost is composed of the following:

- **Marking time:** The length of time the garbage collector spends traversing through all of the reachable objects, starting from the root set and marking each object as being “used.”
- **Sweeping time:** The length of time the garbage collector spends traversing through the entire heap to determine which objects can be reclaimed by checking for the “used” mark; it also includes the time spent in executing finalizer methods of objects to be reclaimed.
- **Compaction time:** The length of time the garbage collection spends in moving all used memory to one side so that available memory is contiguous; it also includes updating all references to objects that were moved.
- **Expansion/Shrinkage time:** The length of time the garbage collection spends in expanding or shrinking the heap depending on the conditions met.
- **Copying objects:** For garbage collections based on generational algorithms, objects are copied back and forth in the young generation space and their ages are monitored for tenuring.
- **Execution of finalizers:** The management of objects with finalizers and their actual execution are costs that should be considered seriously as well.

Given this, a smaller heap may be faster to collect. However, if the applica-

tion is aggressive with creating objects, garbage collection can occur more frequently. Bigger heaps have the advantage of fewer garbage collections, however, a garbage collection can take longer. Searching for available memory may also take longer depending on how memory management is implemented. In severe cases, frequent paging may also occur.

Since garbage collection prevents application threads from doing useful work, there are two main concerns: (1) the duration of a garbage collection and (2) the frequency of garbage collection. The duration of a garbage collection depends primarily on the size of the heap and the amount of live memory (memory that is currently being referenced and used by an application), which we will refer to as the memory footprint or working set of the application. The frequency of garbage collection is also influenced by the size of the heap and the rate at which the application creates objects, which may also be dependent on the number of active worker threads. Thus, the goal is to reduce the overhead of garbage collection by keeping its duration as short as possible and its frequency rate very low.

The Heap Size Challenge

Choosing the values for *ms* (initial heap size) and *mx* (maximum heap size) is one of the major challenges, and it gets harder as your application becomes more complex. As we have already noted above, the heap size is critical to both the duration and frequency of garbage collection. The goal is to find a middle ground in which we can balance both duration and frequency.

Here are some simple guidelines for setting your heap size.

- **Set your performance requirements:** Quantify your expected throughput and response time.
- **Estimate your *mx* and *ms* based on your knowledge of the application's memory usage pattern.**
- **Stress test your application with the expected average workload.**
- **If the results do not meet your performance requirements, analyze the application's garbage collection statistics.** Use the `-verbosegc` output from the JVM to examine how long your garbage collections take and how often they occur.

- **If the duration/frequency of garbage collections is unacceptable, fix the problem by adjusting your heap settings.**

Typically, a garbage collection that takes more than a second on the average is indicative of either a high memory footprint or too big a heap. Also, an acceptable rate of GC occurrence is between 3% and 5% of the time.

Most JVMs also offer a way to control the heap size dynamically by specifying minimum and maximum percentages of the heap that must be kept free, as well as the minimum and maximum amount of memory that can be added to or removed from the heap.

What Application Developers Can Do

Setting and controlling heap sizes are things that can be done outside of the application itself. Experience shows, however, that the majority of the performance improvements can be achieved by improving the way the application is written. In the extreme case, this may involve redesigning the architecture of the application. The less extreme case involves finding better ways to implement certain parts of the application. In this section, we shall assume the latter, that is, a careful analysis of the design has been made and that it is acceptable. Below are some best practices that I refer to as maxims from which developers can learn to minimize the use of memory. The goal of these maxims is to minimize live memory at any point in time.

MAXIM 1: SIMPLIFY YOUR CLASS STRUCTURE

Memory usage is directly affected by the complexity of the class structures in the application. Not only do complicated class structures consume more memory, but they also take longer to get loaded.

Simplicity of class structures also makes understanding program logic much easier, and thus makes debugging easier. Take note also that interfaces are cheaper than using superclasses or abstract classes. Some things to remember when designing a class are:

- Keep your constructors short and simple.
- Avoid too much specialization.
- Prefer composition to inheritance.

MAXIM 2: DO NOT CREATE OBJECTS PREMATURELY

The idea is to avoid the cost of creating an object that may be left unused anyway. This is possible if later in the computation some conditions must be met in order to use the created object. The following snippets provide a very simple illustration. The method stores a String object in a Vector only if the length of the String is between 1 and 32. In the case of longer Strings (or the rare case of a 0-length String) the Vector object is created but never used.

Premature creation can also happen when a class field attribute has been initialized, either in a constructor or during object initialization, but then either the field does not get used at all or it is used much later. By creating objects prematurely, the chance of garbage collection occurring sooner becomes higher.

```
private Vector foo (String
obj)
{
    Vector v = new
    Vector ();

    if ( obj != null &&
        obj.length() > 0
        &&
            obj.length() <=
            32) {

v.addElement(obj)
        return v;
    }
    else
        return null;
}
```

The recommendation is to always move object creation as close as possible to the location where the reference to the object is going to be used. We can rewrite the foregoing example as:

```
private Vector foo (String
obj)
{
    if ( obj != null &&
        obj.length() > 0
        &&
            obj.length() <=
            32) {
```

```
        Vector v =
            new Vector
            ();

v.addElement(obj)
        return v;
    }
    else
        return null;
}
```

Also, you can follow a lazy initialization approach wherein a null reference is checked at the point where the object is actually needed. If the reference is null, then the object is created. Checking for a null reference is a very fast operation.

Premature creation may not be as rampant during the early part of development. Most programmers know the right place to create objects. However, it is when the code is updated and maintained that these things become unnoticeable or aren't checked.

MAXIM 3: BE CAREFUL WITH HIDDEN OBJECTS

This maxim points out the fact that some methods create temporary objects that we may not be aware of. Some of the Java core library classes, for example, do produce a considerable number of these temporary objects, e.g., String, BigInteger, and Date. One common source of hidden objects is the + operator used to concatenate String objects. In his white paper, "WebSphere Application Server Development Best Practices for Performance and Scalability" (www-3.ibm.com/software/webservers/appserv/ws_best_practices.pdf), Harvey Gunther showed the high cost of this operator compared with the preferred method, which is to use a StringBuffer object and then call the append method. His measurements revealed that the total throughput when using + can be reduced by more than half using the preferred method.

Another popular method call – which almost all objects use – is the toString() method. It is recommended that before using this method, you investigate how expensive the operation is going to be. If there is a cheaper way to do it – either by overriding the method or by using some other way – then use that method instead.

MAXIM 4: RELEASE OBJECTS AS SOON AS THEY ARE NO LONGER NEEDED

This maxim addresses the issue of memory leaks; it is also a mirror image of the second maxim, in which memory that was used but no longer needed still exists. It is when objects of this kind accumulate over time that the memory leak phenomenon happens. By setting a reference to null, the referent object becomes eligible for garbage collection. As soon as you know that the object is no longer needed, the null assignment should be done immediately. In other words, release the object as close as possible to the location where it was last used. The reasoning remains that holding on to too many objects causes more frequent and possibly longer garbage collections. Some of the common practices to avoid memory leaks also include the following:

- **Set nonstack variables to null as soon as their intended use is over.**
- **Remove or clear objects from a collection when they are no longer needed.** Understand the semantics of the methods available in the collection. For example, simply getting an object from a Hashtable does not release the reference to the object by the Hashtable. Instead, the clear() or remove() methods must be called.
- **Avoid objects that assume different states.** When an object is in a given state, a set of fields is used to represent the state; when its state changes, a different set of fields is used. The danger comes when the values of the fields in the old state are left hanging after they are no longer needed. If you decide to represent many different states with one object, make sure to nullify fields that are no longer needed in the current state of the object.

MAXIM 5: REUSE OBJECTS WHEN IT HELPS

Object reuse remains one of the elusive object-oriented practices today. Part of the reason is that reusing objects can break away from the rules of object-oriented design. However, there is an obvious and direct benefit of reusing objects.

- **Time:** By reusing objects we avoid the actual creation of new objects and thereby save the cost of such operations. The cost varies depend-

ing on how large the class definition is and how specialized it is.

- **Space:** By reusing objects, we are actually reusing the same memory space that has been allocated already. This results in a smaller memory footprint and consequently faster garbage collection.

The main objective of reusing objects is to minimize the total cost incurred by creating objects by reducing the number of objects actually created. We present several ways.

- **Use of static classes:** The easiest way to avoid creating objects is by creating static classes. Declaring a static class should be done with proper care, however, because this would imply that attributes are shared by all threads accessing the class, which might not be the appropriate semantics for the intended purpose of the class. On the other hand, it would help greatly if we could identify existing classes that can otherwise be declared static. For example, most utility classes do not need to have a unique object instance to do its job.
- **Use of Init and Cleanup methods:** Most objects store information and references to other objects as part of their attributes. One of the reasons why some objects cannot be used easily is because they were created for a specific attribute value. A simple example is the `java.lang.String` class. A `String` object is immutable; that is, it cannot be set to a completely different string value without manipulating its current value. Thus, the only way to operate on a new string is to create another `String` object. We can achieve object reusability by introducing init and cleanup methods. The init method serves to reinitialize the values of the object's attributes. Optionally, the cleanup method is called to do any necessary cleaning up of the object before it can be reused. Ideally, each constructor of the class would have an equivalent init method. In this way, if an object ceases to serve its purpose, it can be reinitialized as needed.
- **Use of an Object Pool:** Pooling objects is another way to reuse objects that is especially useful if the reusable objects are used in a larger scope within the application. A global pool is managed

whereby threads can request object instances as well as return them to the pool. To be able to reuse the objects, an object would typically have methods similar to the init and cleanup methods described previously.

One design scheme is to declare an interface, say `Poolable`, with a definition as shown in Listing 1.

A class whose object instances need to be pooled should then implement this interface. The implementation allocates a pool and precreates objects of that class. The implementation can be made more sophisticated by automatically resizing (shrinking or expanding) the pool depending on the demand for object instances.

Using an object pool dictates some programming methodology on the part of the developer. Instead of creating an object, you use a static call to the get method of the class and then initialize the object. Also, to actually benefit from reuse, the object must be returned explicitly to the pool using the return method.

MAXIM 6: AVOID COPYING OF OBJECTS

Another obvious – yet commonly ignored – source of performance problems is the practice of copying objects, especially deep copying from one component/layer to another, e.g., objects in a `Result Set` are copied by another component that uses them. Sometimes copying objects across components is the right way to go, depending on the semantics of the operations being done on these objects. Developers and designers, however, should carefully analyze whether these objects can be shared. If these objects are used in a read-only fashion, there is no reason to copy them. Be careful, however, that those objects are not refreshed or modified while they are still being used by other components.

A careful inspection of the design is warranted and possibly requires redesigning the whole data interaction between components. Nevertheless, the main point is that for a medium-to-large set of objects, the savings in memory consumption can be very significant.

MAXIM 7: AVOID THE USE OF FINALIZERS

We have heard many times that using finalizers is not a good idea. Every class

has a finalizer method that can be overridden by the programmer. Typically, the finalizer contains code for cleaning up and releasing resources held by an instance of the class. When an object is to be garbage collected, the finalizer must be executed first. The semantics of Java do not guarantee when the finalizers are actually executed; it is possible that they are executed in the next garbage collection occurrence. In any case, the cost of executing the finalizers may be significant, especially when programmers misuse them to do more than cleaning up. This can potentially increase the duration of garbage collection. Finalizers are also prone to deadlock situations.

Conclusion

We have seen the cost of managing objects in the Java programming environment and learned that it can be very significant. Sometimes, application developers overlook the impact of creating too many objects and/or designing complex class structures on runtime performance. Most of the time they focus on pathlength and synchronization reduction. This article emphasizes the importance of looking at memory usage overhead and the effect of garbage collection on the overall performance of a J2EE application. Although advances in research and development are taking place, the garbage collection bottleneck remains a big challenge. Today JVMs are providing more and more methods and ways to tune garbage collection performance. It is our job to discover and use these methods. Also, application developers should learn good programming practices that alleviate the consumption of memory. 🌐

LISTING 1

```
public interface Poolable {
    /* initializes the pool with preset
    instances, it throws an exception
    if initialization fails */
    public static synchronized void
    init() throws Exception;
    /* to get one object from the pool ,
    it throws an exception
    if no object can be given*/
    public static synchronized Object
    get () throws Exception;
    /* to return an object into the pool
    */
    public static synchronized void
    return(Object o);
    public static void cleanup();
}
```

H&W'S DIAGNOSYS MANAGES PERFORMANCE AND AVAILABILITY

(Boise, ID) – H&W, an industry leader in enterprise software tools, has announced H&W DiagnoSys, a comprehensive solution for managing the performance and availability of J2EE applications critical to organizations' business operations. DiagnoSys analyzes the performance and availability of enterprise J2EE applications and alerts companies to potential issues before users experience a problem.

DiagnoSys is one of the few J2EE application management solutions in the industry that is both complete and easy to use, according to H&W. As a single, unified solution, DiagnoSys provides users with

information about potential problems and their probable root causes throughout all phases of the application life cycle, from development to QA and on to production.

www.hwcs.com

MQSOFTWARE OFFERS Q PASA! BUSINESS DASHBOARD CAPABILITY

(Minneapolis, MN) – MQSoftware, Inc., a provider of EAI and middleware technology solutions, has announced a new real-time business dashboard capability available in Q Pasa! version 3. The Q Pasa! Business Dashboard gives business management and IT staff the ability to see real-time performance and availability information from IT busi-



ness systems in a dashboard view. The Q Pasa! Business Dashboard also includes integrated alerting, allowing users to be notified of business issues via the dashboard, e-mail, or pager.

MQSoftware's Q Pasa! Business Dashboard is significantly easier to create than most other dashboards, according to the company, because it has a built-in capability to build a view by simply dragging and dropping performance and availability data into a dashboard. Building a dashboard view requires no coding and can be totally customized along user roles for particular system metrics or have an industry vertical focus like financial or insurance. Using the dashboard, performance and availability metrics can be tracked continuously so that users

can have real-time visibility into areas that need the most attention. www.mqsoftware.com

CA ANNOUNCES AVAILABILITY OF CHANGE MANAGEMENT SOLUTION

(Islandia, NY) – Computer Associates International, Inc., has announced the general availability of AllFusion Harvest Change Manager (AllFusion Harvest CM), its powerful software configuration management solution that helps ensure consistent, rapid delivery of business applications built with WebSphere Studio version 5.0.

Using AllFusion Harvest CM, developers can perform critical change management tasks directly from the WebSphere Studio environment – making it easy to enforce effective change management control.

The new release also integrates into the open-source community's Eclipse 2.0 platform. This interoperability makes AllFusion Harvest CM an ideal change management solution for today's multiplatform enterprise IT environments. www.ca.com

INTROSCOPE POWERPACK OFFERS VISIBILITY INTO WEBSphere

(New Orleans) – Wily Technology, the leader in Enterprise Java Application Management, has announced Introscope PowerPack for IBM WebSphere 5.0 for Distributed Systems, the newest addition to its family of products for maintaining the health and availability of WebSphere applications, including Introscope PowerPacks for WebSphere MQ and CICS Transaction Gateway.

Wily's Introscope PowerPack for IBM WebSphere 5.0 combines Introscope's ability to monitor the performance of production J2EE components such as EJBs, JSPs, and servlets with measurements specific to WebSphere 5.0. Developed in

IBM Unveils New Technologies and Resources at developerWorks Live!

(New Orleans) – At its annual developer conference, IBM announced new technologies, resources, and support programs to help developers build business solutions based on its open infrastructure software.

- **Small to medium size businesses:** IBM introduced the ISV Advantage Initiative. This new component will offer go-to-market agreements to developers who specialize in serving SMB customers. ISV Advantage is designed to use IBM's sales and marketing reach and growing portfolio of IBM Express middleware and IBM eServer systems to help developers penetrate niche market segments with enterprise-proven support.

IBM also introduced Integrated Platform Express, a new Linux offering specifically priced and designed for SMBs.

- **Web services:** IBM Global Services debuted an Enterprise Modernization Initiative that will give developers new tools to enable legacy applications to be transformed into Web services. IBM also released a new Speed-Start Web Services program, available on developerWorks, to help speed adoption of Web services in the enterprise.

In addition, IBM announced the first WebSphere Studio development tool for BEA's WebLogic. With the free plug-in available on IBM's developerWorks

site, developers can use WebSphere Studio to test and deploy J2EE applications for BEA WebLogic in addition to WebSphere.

- **Autonomic Computing:** IBM simplified the system design process by introducing the industry's first blueprint to assist customers in the creation of autonomic computing systems. The company also plans to deliver the first in a series of open technologies based on this blueprint, to help make IT systems more self-managing. IBM is making the blueprint available free of charge and without royalty, and is working with third-party partners, customers, and open standards committees to help drive the architecture's continued evolution.

- **Developer resources:** For its growing, two million-strong developer community, IBM introduced access to open standards-based middleware, tools, online resources, and programs for emerging technologies – including grid and autonomic computing. New resources include the new developerWorks Toolbox subscription (www.ibm.com/developerWorks/toolbox), which now includes the industry's first customizable CD option for developers to self-select tools and middleware from IBM's extensive software collection.




cooperation with the WebSphere Development Team, Introscope PowerPack for IBM WebSphere 5.0 is preconfigured to offer instant component-level monitoring of critical WebSphere resources, including JDBC connection pools, HTTP sessions and thread pools, as well as EJB container and pool, ORB, and JTA.



www.wilytech.com

SPI DYNAMICS OFFERS APPLICATION SECURITY PRODUCT FOR WEBSPHERE

(Atlanta) – SPI Dynamics, a pioneer in Web application security assessment solutions, has announced WebInspect for IBM's WebSphere Studio, the first and only security testing plug-in for developers utilizing the WebSphere platform. WebInspect for WebSphere Studio will enable application developers to automatically discover security vulnerabilities as they build applications, remediate those vulnerabilities, and deliver secure code for final quality assurance testing.

Industry analysts have estimated that over 70% of today's security breaches occur at the application level. Many are due to the exploitation of security defects that are injected into the  code during the development process. WebInspect for WebSphere Studio allows the application developer to view the application through the eyes of a security expert, providing a strong framework for early discovery and remediation of security vulnerabilities.

www.spidynamics.com

MKS PREVIEW'S PRODUCTION DEPLOYMENT SOLUTION FOR ENTERPRISE DEVELOPMENT

(New Orleans) – MKS Inc., a provider of tools to automate, secure, manage, and control the software development process, previewed a new solution at IBM developerWorks Live! aimed at solving critical production deployment problems faced by today's enterprise development shops.

The MKS Build and Deployment solution is targeted at development organizations within Global 1000 companies responsible for managing changes to software applications running core business operations. Built on the award-winning MKS Integrity Solution, the soft-

ware deployment process can be secured to prevent unauthorized changes from being introduced, while providing rollback and audit capabilities to minimize risk of failures.




MKS demonstrated how this and other significant business challenges are solved with the MKS Integrity Solution, in combination with Openmake from Catalyst Systems, in their presentation. MKS's Build and Deployment solution, a part of the MKS Integrity Solution for enterprise software configuration management, will be generally available in May 2003.

www.mks.com/partners/ibm
www.catsyscorp.com

IBM, OPENDMAND UNVEIL SOLUTION FOR LOAD AND STRESS TESTING

(New Orleans) – IBM and OpenDemand Systems officially unveiled OpenLoad v3.5, the industry's first easy-to-use, browser-based, rapid performance optimization solution for load and stress testing of dynamic Web sites, at IBM's developerWorks Live! conference. The feature-rich product offers ease of use, superior analysis, and affordability, allowing application development and testing teams to easily perform fast and productive optimization of dynamic Web sites without the need to develop complex scripts or perform complicated performance analysis.

OpenLoad v3.5 is powered by IBM WebSphere Internet infrastructure software, IBM DB2 Universal Database, and IBM eServer xSeries system.

OpenLoad v3.5 is the first testing solution to provide script-free Rapid Application Recording technology, which  enables users to effortlessly build data-driven user scenarios in seconds, and adds full support for pages with complex JavaScript, Java applets, Flash, DHTML, NTLM authentication, and more.

www.opendemand.com

DIOGENES' I MERCURY EXTENDS THE POWER OF WEBSPHERE

(New Orleans) – At IBM's developerWorks Live! conference Diogenes showcased its iMercury tool, which helps developers create, deploy, and manage widely distributed, service-oriented applications using the IBM WebSphere product line.

iMercury can be used with


WebSphere Application Server to create, deploy, and manage widely distributed agile applications using a service-oriented architecture that shortens IT cycles and reduces maintenance costs.

iMercury can be used with WebSphere MQ Integrator to derive even more value from MQ Integrator by creating distributed "integration service grids" that provide a more agile, scalable, easily maintainable, service-oriented solution for application integration.

 www.diogenesinc.com

CGS, IBM TEAM TO OFFER 'E-LAB' FOR DEVELOPERS


(San Francisco) – IBM has announced that Computer Generated Solutions (CGS), a worldwide IT services company based in New York City, will team with IBM Learning Services to launch the first online "virtual" IBM WebSphere Innovation Center.

The site, www.cgselearning.com, is a "knowledge portal" that offers programmers online "e-tutorials" with access to "e-labs" that  provide hands-on training using remote hardware and software running WebSphere, as well as WebSphere technical information.

SEGUE'S SILKPLAN PRO UPGRADES SOFTWARE QUALITY MANAGEMENT

(Lexington, MA) – Upgrading its market-leading software quality management application, Segue Software has announced the release of SilkPlan Pro 6.0, which includes new features that allow enterprises to introduce more order, structure, and visibility to the application testing process. Segue Software has made it  easier for companies in regulated industries like health care, telecommunications, financial services, and pharmaceuticals to manage complex or large-scale manual and automated testing projects.

SilkPlan Pro 6.0 empowers both quality assurance testers and application developers by streamlining and automating the application testing process through robust new features, such as a flexible integration framework, improved reporting, a script object library, unattended test execution, and additional enterprise-wide security.

www.segue.com 

Implementing WebSphere Security Through LDAP

Protocol offers many advantages

— BY LOU MAUGET —



ABOUT THE AUTHOR

Lou Mauget is a senior consultant with CrossLogic Corporation, where his assignments include J2EE mentoring and designing instructional material. He has coauthored three computer-related books. His prior career at IBM included work as a consultant, programmer, and developer.

E-MAIL

Lmauget@crosslogic.com

Lightweight Directory Authentication Protocol (LDAP) is often promoted as a means to leverage an organizational directory as a principal registry for WebSphere authentication and authorization. Advantages include the capability to configure single sign-on across application servers, enabling additional organizational applications, centralized user administration, multimastered replication across authentication sites, and flexible, extensible data formats – not to mention that LDAP is a vendor-neutral protocol and API backed by IETF. This begs the question of how to implement WebSphere security through LDAP.

This two-part series presents a simplified example of how to configure WebSphere Application Server version 5.0 to use IBM Directory Server v5.1 as its user registry for J2EE application user authentication and role-based authorization. This registry enables the ability to configure single sign-on capability across applications residing on multiple WebSphere Application Servers and Lotus Domino servers. By the end of the second part of this article, you will understand how to create an LDAP user registry from scratch, and will have gained an understanding of how to incorporate an existing LDAP user directory as the user registry for securing WebSphere J2EE applications.

Required Software Components

WebSphere Studio Application Developer, version 5.0 (hereafter called WebSphere Studio), will be used to create and host the example using the embedded WebSphere Test Environment v5 as the target application server.

IBM Directory Server v5.1 will host the LDAP directory service. This directory server, available as a free download from IBM (www-3.ibm.com/software/network/directory/server/download), includes a copy of IBM DB2 UCB v8, used to contain the underlying directory information.

The example was written for a Microsoft Windows NT/2000/XP platform, but the principles used apply to any supported platform. The directory server and the application server are often installed on separate machines or even separate platforms, if available.

Create Directory Information

You are starting from scratch, so you first need to design a directory information tree (DIT) and then create an import file that contains entries that conform to it.

DIT DESIGN

The J2EE application users will authenticate themselves through WebSphere Lightweight Third Party Authentication (LTPA). A security token granted by LTPA can propagate to other WebSphere and Domino servers that participate in single sign-on. The user registry can be an LDAP directory or a custom implementation. LDAP has the advantage of being an IETF (Internet Engineering Task Force) standard protocol. Its implementations can be distributed as multimaster replicated directories. These may support additional clients such as organizational white pages applications or service locators.

The example involves LTPA configured to use a scratch-built DIT for its user registry. An LDAP directory stores



information in nodes. In this directory, each user has a node that stores information unique to her or him. Each group has a node that maintains a list of unique members. You will be able to set optional attribute values at will to track an organization's personnel information for use by other applications.

WebSphere LPTA uses LDAP to map authorization roles to users and groups. Therefore the DIT needs to contain a set of user entries. In addition it needs a set of groups such that each group entry refers to a subset of users that belong to that group.

The literature generally agrees that a shallow directory hierarchy is less sensitive to organizational changes than a deep hierarchy. The sample organization name is Rogers60, located in the United States. The J2EE application will be constrained by user and admin roles, but it is easy to add more application roles later as the need arises. The example subtree of the LDAP hierarchy is defined by a suffix so that the users and groups fall under that suffix (see Figure 1).

Each person node under the `ou=people` node contains the object classes `person` and `ePerson`. These object classes include optional attributes that can fulfill most anticipated personnel directory requirements. The `ePerson` object class defines the `uid` and `userPassword` attributes that WebSphere will use for authenticating a request having a supplied user ID and password.

Each `ou=groups` node contains the `groupOfUniqueNames` object class, which specifies a multivalued attribute named `uniqueMember`. A group entry will use the value list of this attribute to reference the distinguished name (DN) of each user in a given group. WebSphere will use this information to check group membership for a role mapped to a DIT group.

LDAP Interchange Format File

LDAP uses an import and export file format, called LDAP Interchange Format (LDIF), that is composed of name-value pairs that define and populate nodes in the DIT. The first line for a node entry must define its DN. The definitions of the node's object classes and attributes follow the DN. Each line starts with a nonblank, unless it continues the previous line. The definition of an entry ends with a blank line. A `#` begins a comment line. The LDIF format is easy to understand, as the example will illustrate.

Create an LDIF file to initially populate the directory for the example. A DIT resembles a file directory tree, so begin by creating higher-level nodes that contain lower-level nodes, and then create the contained nodes. The DIT root will be the directory suffix `o=rogers60,c=us`, which will be defined later using the IBM Directory Server Configuration

Tool. Thus the LDIF file begins at the `o=rogers60` node. Next, it defines the people and groups nodes. Finally, it populates the people and groups nodes with data nodes.

Use an editor such as VI or Notepad to create the text file named `rogers60.ldif`, shown in Listing 1. Double-check your work and then save the file in a work directory. We will import the LDIF file into the directory server later.

Notice that the final user node has a `uid` attribute set to "was". This defines the user account to be used by WebSphere when security is enabled. A sampling of optional attributes, such as `telephoneNumber` from objectClass `ePerson`, illustrates how this directory could be extended as a personnel directory. Option attribute values can be added at any time.

Notice that the `userPassword` attribute is not encrypted. The default access control defined for the `userPassword` attribute makes it invisible to queries by anonymous LDAP clients. The directory server stores it using `imask` encryption, meaning that the value is encrypted on the disk but sent in clear text on the network. WebSphere will search the directory as an authenticated privileged client, so the password will be received in clear text. An SSL connection will prevent snooping or undetected alteration of the data on the network.

IBM Directory Server v5.1

Insert the IBM Directory Server v5.1 CD-ROM into the CD drive in the target machine. Choose all options, including the GSKit, DB2, and WebSphere - Express. If you already have DB2 7.2 or DB2 8, the directory server will offer to use it. GSKit is needed for securing the directory with SSL. The WebSphere - Express Server hosts a Web-based directory administration Web application.

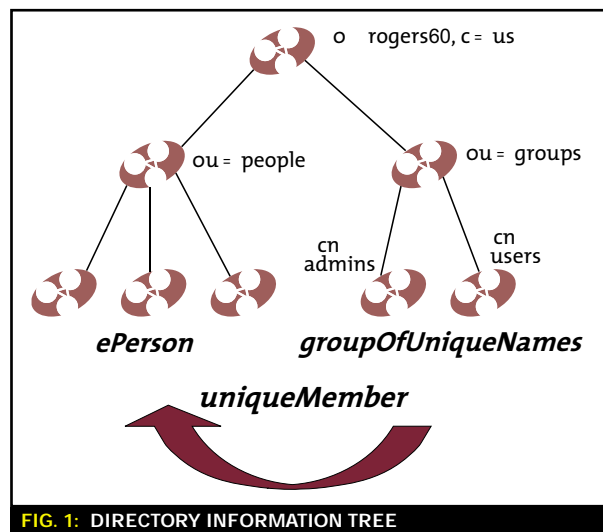


FIG. 1: DIRECTORY INFORMATION TREE

DEFINE THE ADMINISTRATOR DISTINGUISHED NAME

After installation there is a forced reboot, followed by automatic invocation of the IBM Directory Server Configuration Tool. The administrative distinguished name is used to bind to the directory with administrative privileges. Set this DN by carrying out the following steps:

1. Click the Administrator DN tree node.



FIG. 2: WEBSHERE STUDIO SHORTCUT PROPERTIES



FIG. 3: ENTERPRISE APPLICATION PROJECT

2. Ensure that the administrator DN is cn=root. This is a DN that has administrative privileges. Its user can see and update anything in the directory.
3. Set the password to secret.
4. Set the confirmation password to secret.
5. Click OK.

CONFIGURE A DATABASE

Use the following steps to define the DB2 database container for the LDAP information:

1. Click the Configure Database tree node.
2. Name the database ldapdb2.
3. Supply user ID db2admin or your existing DB2 administrative user ID.
4. Enter the DB2 user password.
5. Create the database.

Create the LDAP Directory

The directory server has not yet been started. Use the IBM Directory Server Configuration Tool to set directory suffixes and import the LDIF file.

SET DIRECTORY SUFFIXES

The directory root DN o=rogers60,c=us refers to the relative tree root node in the DIT. Carry out the following steps to set the suffixes for the DIT:

1. Click the Manage Suffixes tree node.
2. Set the Suffix DN field to o=rogers60,c=us.
3. Press the Add button to add the suffix.
4. Press OK.

IMPORT THE LDIF FILE

The IBM Directory Server Configuration Tool can be used to import the LDIF file. An alternative would be to use the ldapadd command found in the bin directory of the server, but the server would need to be running. Instead, carry out the following steps to use the configuration tool to import the LDIF file to populate your directory:

1. Click the Import LDIF data tree node.
2. Use the Browse button to set the Path and LDIF file name field to point to rogers60.ldif.
3. Ensure that Standard import is selected.
4. Press the Import button to carry out the operation.
5. The task messages should show each DN imported with no error.

START THE IBM DIRECTORY SERVER

The IBM Directory Server installs itself as a Windows service. On AIX or Linux it would install as a daemon. You can start the WebSphere - Express application server that accompanies the directory server and then use the served administration Web page to create a console page for the server. Then you would start the server from the console page. For brevity, use the following steps to start the server on Windows:

1. Open the Windows Control Panel from the Start menu.
2. Open Administrative Tools.
3. Open Services.
4. Find the row labeled IBM Directory Server v5.1.
5. Right-click it and choose Start.

Starting will take a minute or two.

SUBSCRIBE TODAY TO MULTIPLE MAGAZINES AND SAVE UP TO \$400 AND RECEIVE UP TO 3 FREE CDs!

RECEIVE
YOUR DIGITAL
EDITION
ACCESS CODE
INSTANTLY
WITH YOUR PAID
SUBSCRIPTIONS



TO ORDER

- Choose the Multi-Pack you want to order by checking next to it below.
- Check the number of years you want to order.
- Indicate your location by checking either U.S., Canada/Mexico or International.
- Then choose which magazines you want to include with your Multi-Pack order.

Pick a 3-Pack, a 6-Pack or a 9-Pack

<input type="checkbox"/> 3-Pack	<input type="checkbox"/> 1YR <input type="checkbox"/> 2YR	<input type="checkbox"/> U.S. <input type="checkbox"/> Can/Mex <input type="checkbox"/> Intl.
<input type="checkbox"/> 6-Pack	<input type="checkbox"/> 1YR <input type="checkbox"/> 2YR	<input type="checkbox"/> U.S. <input type="checkbox"/> Can/Mex <input type="checkbox"/> Intl.
<input type="checkbox"/> 9-Pack	<input type="checkbox"/> 1YR <input type="checkbox"/> 2YR	<input type="checkbox"/> U.S. <input type="checkbox"/> Can/Mex <input type="checkbox"/> Intl.

Linux Business & Technology

U.S. - Two Years (24) Cover: \$143	You Pay: \$79.99 /	Save: \$63 + FREE \$198 CD
U.S. - One Year (12) Cover: \$72	You Pay: \$39.99 /	Save: \$32
Can/Mex - Two Years (24) \$168	You Pay: \$119.99 /	Save: \$48 + FREE \$198 CD
Can/Mex - One Year (12) \$84	You Pay: \$79.99 /	Save: \$4
Intl - Two Years (24) \$216	You Pay: \$176 /	Save: \$40 + FREE \$198 CD
Intl - One Year (12) \$108	You Pay: \$99.99 /	Save: \$8

Java Developer's Journal

U.S. - Two Years (24) Cover: \$144	You Pay: \$89 /	Save: \$55 + FREE \$198 CD
U.S. - One Year (12) Cover: \$72	You Pay: \$49.99 /	Save: \$22
Can/Mex - Two Years (24) \$168	You Pay: \$119.99 /	Save: \$48 + FREE \$198 CD
Can/Mex - One Year (12) \$84	You Pay: \$79.99 /	Save: \$4
Intl - Two Years (24) \$216	You Pay: \$176 /	Save: \$40 + FREE \$198 CD
Intl - One Year (12) \$108	You Pay: \$99.99 /	Save: \$8

Web Services Journal

U.S. - Two Years (24) Cover: \$168	You Pay: \$99.99 /	Save: \$68 + FREE \$198 CD
U.S. - One Year (12) Cover: \$84	You Pay: \$69.99 /	Save: \$14
Can/Mex - Two Years (24) \$192	You Pay: \$129 /	Save: \$63 + FREE \$198 CD
Can/Mex - One Year (12) \$96	You Pay: \$89.99 /	Save: \$6
Intl - Two Years (24) \$216	You Pay: \$170 /	Save: \$46 + FREE \$198 CD
Intl - One Year (12) \$108	You Pay: \$99.99 /	Save: \$8

.NET Developer's Journal

U.S. - Two Years (24) Cover: \$168	You Pay: \$99.99 /	Save: \$68 + FREE \$198 CD
U.S. - One Year (12) Cover: \$84	You Pay: \$69.99 /	Save: \$14
Can/Mex - Two Years (24) \$192	You Pay: \$129 /	Save: \$63 + FREE \$198 CD
Can/Mex - One Year (12) \$96	You Pay: \$89.99 /	Save: \$6
Intl - Two Years (24) \$216	You Pay: \$170 /	Save: \$46 + FREE \$198 CD
Intl - One Year (12) \$108	You Pay: \$99.99 /	Save: \$8

XML-Journal

U.S. - Two Years (24) Cover: \$168	You Pay: \$99.99 /	Save: \$68 + FREE \$198 CD
U.S. - One Year (12) Cover: \$84	You Pay: \$69.99 /	Save: \$14
Can/Mex - Two Years (24) \$192	You Pay: \$129 /	Save: \$63 + FREE \$198 CD
Can/Mex - One Year (12) \$96	You Pay: \$89.99 /	Save: \$6
Intl - Two Years (24) \$216	You Pay: \$170 /	Save: \$46 + FREE \$198 CD
Intl - One Year (12) \$108	You Pay: \$99.99 /	Save: \$8

WebLogic Developer's Journal

U.S. - Two Years (24) Cover: \$360	You Pay: \$169.99 /	Save: \$190 + FREE \$198 CD
U.S. - One Year (12) Cover: \$180	You Pay: \$149 /	Save: \$31
Can/Mex - Two Years (24) \$360	You Pay: \$179.99 /	Save: \$180 + FREE \$198 CD
Can/Mex - One Year (12) \$180	You Pay: \$169 /	Save: \$11
Intl - Two Years (24) \$360	You Pay: \$189.99 /	Save: \$170 + FREE \$198 CD
Intl - One Year (12) \$180	You Pay: \$179 /	Save: \$1

ColdFusion Developer's Journal

U.S. - Two Years (24) Cover: \$216	You Pay: \$129 /	Save: \$87 + FREE \$198 CD
U.S. - One Year (12) Cover: \$108	You Pay: \$89.99 /	Save: \$18
Can/Mex - Two Years (24) \$240	You Pay: \$159.99 /	Save: \$80 + FREE \$198 CD
Can/Mex - One Year (12) \$120	You Pay: \$99.99 /	Save: \$20
Intl - Two Years (24) \$264	You Pay: \$189 /	Save: \$75 + FREE \$198 CD
Intl - One Year (12) \$132	You Pay: \$129.99 /	Save: \$2

Wireless Business & Technology

U.S. - Two Years (24) Cover: \$144	You Pay: \$89 /	Save: \$55 + FREE \$198 CD
U.S. - One Year (12) Cover: \$72	You Pay: \$49.99 /	Save: \$22
Can/Mex - Two Years (24) \$192	You Pay: \$139 /	Save: \$53 + FREE \$198 CD
Can/Mex - One Year (12) \$96	You Pay: \$79.99 /	Save: \$16
Intl - Two Years (24) \$216	You Pay: \$170 /	Save: \$46 + FREE \$198 CD
Intl - One Year (12) \$108	You Pay: \$99.99 /	Save: \$8

WebSphere Developer's Journal

U.S. - Two Years (24) Cover: \$360	You Pay: \$169.99 /	Save: \$190 + FREE \$198 CD
U.S. - One Year (12) Cover: \$180	You Pay: \$149 /	Save: \$31
Can/Mex - Two Years (24) \$360	You Pay: \$179.99 /	Save: \$180 + FREE \$198 CD
Can/Mex - One Year (12) \$180	You Pay: \$169 /	Save: \$11
Intl - Two Years (24) \$360	You Pay: \$189.99 /	Save: \$170 + FREE \$198 CD
Intl - One Year (12) \$180	You Pay: \$179 /	Save: \$1

PowerBuilder Developer's Journal

U.S. - Two Years (24) Cover: \$360	You Pay: \$169.99 /	Save: \$190 + FREE \$198 CD
U.S. - One Year (12) Cover: \$180	You Pay: \$149 /	Save: \$31
Can/Mex - Two Years (24) \$360	You Pay: \$179.99 /	Save: \$180 + FREE \$198 CD
Can/Mex - One Year (12) \$180	You Pay: \$169 /	Save: \$11
Intl - Two Years (24) \$360	You Pay: \$189.99 /	Save: \$170 + FREE \$198 CD
Intl - One Year (12) \$180	You Pay: \$179 /	Save: \$1

3-Pack

Pick any 3 of our
magazines and save
up to **\$275⁰⁰**
Pay only \$175 for a
1 year subscription
plus a **FREE CD**

- 2 Year - \$299.00
- Canada/Mexico - \$245.00
- International - \$315.00

6-Pack

Pick any 6 of our
magazines and save
up to **\$350⁰⁰**
Pay only \$395 for a
1 year subscription
plus **2 FREE CDs**

- 2 Year - \$669.00
- Canada/Mexico - \$555.00
- International - \$710.00

9-Pack

Pick 9 of our
magazines and save
up to **\$400⁰⁰**
Pay only \$495 for a
1 year subscription
plus **3 FREE CDs**

- 2 Year - \$839.00
- Canada/Mexico - \$695.00
- International - \$890.00

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

Subscribe Online Today www.sys-con.com/2001/sub.cfm

**SYS-CON
MEDIA**

Create a Sample Web Application

You need a sample application to lock down through roles. In the following sections you will configure a workspace for WebSphere Studio, create a simple JSP Model 1 application as a vehicle to demonstrate role-based authorization, and finally, configure an application server.

WEBSPPHERE STUDIO WORKSPACE

You will need a new workspace to contain the application and the server configuration. Carry out the following

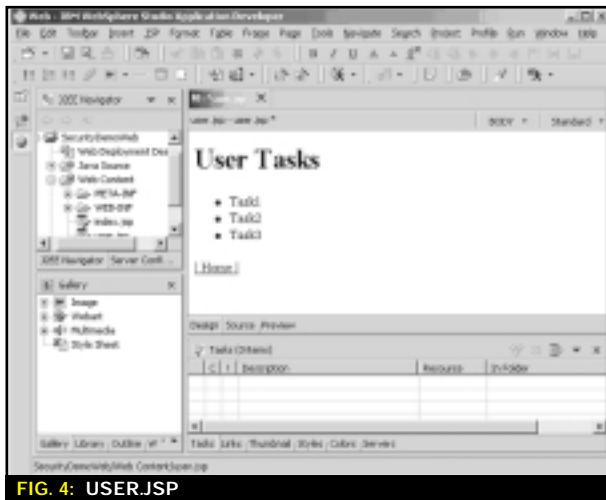


FIG. 4: USER.JSP



FIG. 5: CREATE ADMIN.JSP



FIG. 6: HYPERLINK TARGET

steps on your development machine:

1. Create a directory named c:\workspace-LDAP-auth.
2. Drag and drop a shortcut to WebSphere Studio to the desktop using the right mouse button.
3. Choose Create Shortcuts Here from the pop-up menu.
4. Right-click the new shortcut and choose Rename to rename it to WSAD LDAP Auth.
5. Right-click the new shortcut and choose Properties from the pop-up menu.
6. Append a blank followed by -data C:\workspace-LDAP-auth to the Target field (see Figure 2). Don't forget the leading dash (-).
7. Press OK to create the shortcut.
8. Use the shortcut to start WebSphere Studio. Some time elapses as the new workspace is created.
9. WebSphere Studio opens in the J2EE perspective. Close the Welcome tab (read it first, if you wish).

CREATE AN ENTERPRISE APPLICATION PROJECT

Next, create an enterprise application project that contains a Web application by following these steps:

1. Invoke main menu File -> New -> Enterprise Application Project. The Enterprise Application Project Creation wizard appears.
2. Keep the default Create J2EE 1.3 Enterprise Application Project option. Press Next->.
3. In the Enterprise application name field enter Security-Demo.
4. Clear the options labeled Application Client Module and EJB Module.
5. The dialog should resemble that shown in Figure 3.
6. Press Finish to create the two projects. They appear in the J2EE Hierarchy view.

IMPLEMENT THE WEB PROJECT

The Web project will define a boilerplate JSP Model 1 implementation that acts as a secured "Hello World" demonstration. It will have a start page that invites the user to navigate to either an application page or an administration page. WebSphere will authenticate the user through basic authentication, although you could easily configure forms-based authentication later.

Roles determine if the user is an administrator who is allowed to navigate to the administration page. Once you see how it is done, you can carry the technique forward to actual applications.

Next, you will create the three JSP files that form the entire application.

User Page

Carry out these steps to create a page that represents the application user interface to tasks:

1. Open the Web Perspective to project SecurityDemoWeb in the J2EE Navigator view.
2. Right-click the Web Content folder. Choose New JSP File. The New JSP File wizard opens.
3. Name the file index.jsp and press Finish to create the page.
4. Use main menu Insert -> Paragraph -> Heading 1 to insert User Tasks.
5. Choose main menu Insert -> List -> Bulleted List.
6. Enter the follow bullet items: Task 1, Task2, Task3.
7. Click beneath the bullet list. Enter the following text: -> Home ->.

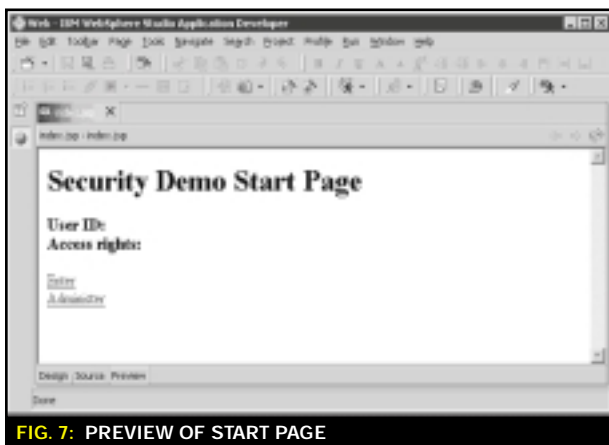


FIG. 7: PREVIEW OF START PAGE

6. Open admin.jsp from the J2EE Navigator view.
7. Edit the heading to read Administrative Tasks.
8. Save the file and close it.

Start Page

The start page will link to user.jsp and admin.jsp. Use the following steps to create the first page of the Web application:

1. Create a JSP file named index.jsp under the Web Content folder, as you did for user.jsp.
2. Use main menu Insert -> Paragraph -> Heading 1 to insert Security Demo Start Page.
3. Click beneath the text and insert the text "Enter the application".
4. Select the text. Choose main menu Insert -> Link.
5. Click the Browse button. Choose File from the pop-up menu. A File Selection dialog will appear.
6. Select user.jsp, as shown in Figure 6.
7. Press OK. Press OK again to create the link.
8. Use the same procedure to create a link to admin.jsp for the Administer text.

Keep the JSP open for the following section.

Start Page Scriptlets

The start page should identify the logged-in user and his or her role. Moreover, the admin link on the page should only be visible to callers in the admin role. You need to add two scriptlets to implement this behavior. Follow these steps:

1. Click the Source tab.
2. Display the user name and rights by entering the following HTML element beneath the H1 element.

8. Double-click that text to select it. Choose Insert -> Link from the main menu. The Insert Link dialog will open.
9. Enter index.jsp in the URL field of the dialog. Press OK. The user.jsp file should resemble Figure 4.
10. Save the file and close it.

Administration Page

Carry out the following steps to clone user.jsp to create a representation of an administration page:

1. Select user.jsp in the J2EE navigator.
2. Right-click user.jsp and choose Copy.
3. Select the Web Content node, right-click, and choose Paste. A Name Conflict dialog will appear.
4. Name the new page admin.jsp, as shown in Figure 5.
5. Press OK to create the new JSP file.

WebSphere
DEVELOPER'S JOURNAL

**Coming
Next Month...**

STANDARDS:
Web Services Invocation Framework
 BY BORIS LUBLINSKY

TOOLS:
WebSphere/J2EE Programming with Crows
 BY BASSEM W. JAMALEDDINE

SECURITY TUTORIAL:
Implementing WebSphere Security Through LDAP, Part 2
 BY LOU MAUGET

TIVOLI:
Tivoli Access Manager for WebSphere Application Server
 BY EDWARD MCCARTHY

PERFORMANCE TUNING:
WebSphere Application Server and Database Performance Tuning, Part 2
 BY MICHAEL S. PALLOS

WebSphere
DEVELOPER'S JOURNAL

**Advertiser
Index...**

COMPANY	URL	PHONE	PG
CANDLE	WWW.CANDLE.COM/WWW1/AXA1_WSDJ	800-898-4426	5
JAVAONE	WWW.JAVA.SUN.COM/JAVAONE/SF	650-372-7054	22,23
MACROMEDIA	WWW.MACROMEDIA.COM/GO/WSDJ/	415-252-2000	51
MQSOFTWARE	WWW.MQSOFTWARE.COM	952-345-8720	6,7
PERFORMANT	WWW.PERFORMANT.COM/WEBSPPHERE1	866-773-6268 x0	27
PROLIFICS	WWW.PROLIFICS.COM	800-675-5419	11
REPORTINGENGINES	WWW.REPORTINGENGINES.COM/DOWNLOAD/WSDJ1.JSP	888-884-8665	19
SITETELLIGENCE	WWW.SITETELLIGENCE.COM	201-445-0121	15
SYS-CON PUBLICATIONS	WWW.SYS-CON.COM/2001/SUB.CFM	888-303-5282	47
TRILOG GROUP	WWW.FLOWBUILDER.COM	800-818-2199	52
VERSATA	WWW.VERSATA.COM/BUSINESSLOGICDESIGNER	800-984-7638	2
WEB SERVICES EDGE WEST 2003	WWW.SYS-CON.COM/WEBSERVICES2003WEST	201-802-3069	31
WILY TECHNOLOGY	WWW.WILYTECH.COM	888-GET-WILY	3

```
<H3>User ID:
  <%= request.getUserPrincipal().getName()
    %><br />
  Access rights:
  <%= (request.isUserInRole("admin") ?
    "administrator": "user")%>
</H3>
```

3. Locate the Administer hyperlink. Surround it with scriptlet tags so it appears as follows:

```
<% if (request.isUserInRole("admin")) { %>
  <A href="admin.jsp">Administer</A>
<% } %>
```


4. Press the Preview tag. The page should resemble Figure 7.
5. Save the JSP file.

6. Click the Preview tab and check that all links operate correctly.

The start page name, index.jsp, is one of the standard names on the welcome file list of the Web deployment descriptor. When you run the application on a server, that page will appear automatically.

Summary

This article has shown you how to obtain the infrastructure needed to implement an example of LDAP authentication through WebSphere. IBM Directory Server v5.1 contains the DIT to be used for authenticating users through WebSphere. We created a small application that will use admin and user roles to enable access to its pages. The start page is customized by Java scriptlets that display user information and conditionally enable an administrative link.

In the second article in this series, I will discuss how to define roles and associated constraints in the application deployment descriptor. The WebSphere v5.0 Test Environment will host the application after roles are associated with resource constraints and are mapped to groups in the LDAP directory. I will demonstrate how WebSphere can authenticate users using basic authentication against a mapped LDAP group. Application resources will be accessible on a per-role basis. Finally, I will discuss how to secure the transport channels between WebSphere, the LDAP directory, and the browser through SSL to prevent password snooping. 

LISTING 1

```
#-----
# White pages for rogers60.com
# -Louis E. Mauget
#-----
dn: o=rogers60,c=us
o: rogers60
objectclass: top
objectclass: organization

#-----
# People subtree
#-----
dn: ou=people,o=rogers60,c=us
ou: people
objectclass: organizationalUnit

objectclass: top

#-----
# Group subtree
#-----
dn: ou=groups,o=rogers60,c=us
ou: groups
objectclass: organizationalUnit
objectclass: top

#-----
# Kinds of groups
#-----
dn: cn=users,ou=groups,o=rogers60,c=us
cn: users
objectclass: groupOfUniqueNames
objectclass: top
uniqueMember: cn=Bolliver
Schagnasty,ou=people,o=rogers60,c=us
uniqueMember: cn=Lou Mauget,ou=people,o=rogers60,c=us
uniqueMember: cn=Elwood Suggins,ou=people,o=rogers60,c=us
uniqueMember: cn=was,ou=people,o=rogers60,c=us

dn: cn=admins,ou=groups,o=rogers60,c=us
cn: admins
objectclass: groupOfUniqueNames
objectclass: top
uniqueMember: cn=Lou Mauget,ou=people,o=rogers60,c=us
uniqueMember: cn=was,ou=people,o=rogers60,c=us

#-----
# Persons
#-----

dn: cn=Bolliver Schagnasty,ou=people,o=rogers60,c=us

cn: Bolliver Schagnasty

objectclass: top
objectclass: person
objectclass: ePerson
sn: Schagnasty

uid: bschagnasty

userPassword: password
telephoneNumber: 1-999-555-4321
mail: bschagnasty@isp.net
title: Sales manager
postalCode: 509

dn: cn=Lou Mauget,ou=people,o=rogers60,c=us
cn: Lou Mauget
objectclass: top
objectclass: person
objectclass: ePerson
sn: Mauget
uid: lmauget
userPassword: password
telephoneNumber: 1-999-555-1234
mail: lmauget@isp.net
title: IS administrator
postalCode: 501

dn: cn=Elwood Suggins,ou=people,o=rogers60,c=us
cn: Elwood Suggins
objectclass: top
objectclass: person
objectclass: ePerson
uid: esuggins
sn: Suggins
userPassword: password
telephoneNumber: 1-999-555-3421
mail: esuggins@isp.net
title: Sales Rep
postalCode: 509

#-----
# WebSphere user ID
#-----
dn: cn=was,ou=people,o=rogers60,c=us
cn: was
objectclass: top
objectclass: person
objectclass: ePerson
sn: was
uid: was
userPassword: secret
```

MACROMEDIA

WWW.MACROMEDIA.COM/GO/WSDJ/

TRILOG GROUP

WWW.FLOWBUILDER.COM