

WebSphere[®] and DB2[®] Clustering and Scalability Study

Installation and Configuration Guide

On Windows 2000 Advanced Server

By Yongli An and Tsz Kin Tony Lau
IBM Toronto Lab, IBM Canada

© Copyright International Business Machines Corporation 2002. All rights reserved.

Contents

CONTENTS	I
1. INTRODUCTION	1
1.1 OVERVIEW	1
1.2 PURPOSE.....	1
1.2.1 Problem statement.....	1
1.2.2 Design statement.....	1
1.3 INTENDED AUDIENCE.....	1
2. CLUSTER TOPOLOGY	1
2.1 CLUSTERING WITHOUT SESSION AFFINITY.....	2
2.2 CLUSTERING USING SESSION AFFINITY BY SETTING STICKY TIME.....	2
2.3 CLUSTERING USING WEBSPHERE HTTP PLUG-IN SESSION AFFINITY	4
2.4 MACHINE SETUP AND SOFTWARE CONFIGURATION	5
3. SETTING UP THE DATABASES	6
3.1 INSTALLATION SUMMARY.....	6
3.2 INSTALL DB2.....	6
3.3 CREATE DATABASES	7
3.3.1 Creating the databases.....	7
3.3.2 Creating the tables	7
3.3.3 Database tuning	7
3.4 INSTALL DB2 CLIENT	8
3.5 ESTABLISH DATABASE CONNECTIONS	8
3.5.1 Enable remote connections	8
3.5.2 Catalog the node and databases	8
4. INSTALLING TRADE2 APPLICATION SERVER	9
4.1 INSTALLATION SUMMARY.....	9
4.2 INSTALL WEBSPHERE	9
4.2.1 Install the base code.....	9
4.2.2 Apply the fixpack	9
4.2.3 Start WebSphere adminserver.....	10
4.3 INSTALL TRADE2	10
4.4 CONFIGURE TRADE2 FOR PERSISTENT SESSION.....	10
4.4.1 Create connection to the session database	10
4.4.2 Enable persistent session	11
5. CLONING TRADE2	11
5.1 INSTALLATION SUMMARY.....	11
5.2 INSTALL WEBSPHERE	11
5.2.1 Install the base code.....	11
5.2.2 Apply the fixpack	12

5.2.3	<i>Start WebSphere administrative server</i>	12
5.2.4	<i>Update the host alias list</i>	12
5.3	INSTALL DATABASE DRIVER.....	12
5.4	COPY THE TRADE2 APPLICATION.....	12
5.5	CREATE TRADE2 MODEL AND CLONES	13
5.5.1	<i>Trade2 model</i>	13
5.5.2	<i>Trade2 clone</i>	13
5.6	CONFIGURE OSE REMOTE (ONLY FOR OSE REMOTE SETUP)	13
5.6.1	<i>Configure the transport type</i>	13
5.6.2	<i>Configure the plug-in for OSE remote</i>	13
6.	SETTING UP WEBSHERE EDGE SERVER	14
6.1	INSTALLATION SUMMARY.....	14
6.2	INSTALL WEBSHERE EDGE SERVER	14
6.2.1	<i>Installation</i>	14
6.2.2	<i>Configuration</i>	14
6.2.3	<i>Update the host alias list</i>	15
6.3	INSTALL THE LOOPBACK ADAPTER.....	15
6.3.1	<i>Install</i>	16
6.3.2	<i>Configure</i>	16
6.3.3	<i>Change the bind order as needed</i>	16
6.3.4	<i>Remove the extra route</i>	16
6.4	SPECIFY THE STICKY TIME.....	17
7.	TESTS FROM CLIENT MACHINES	17
8.	SUMMARY	17
	REFERENCES	18

1. Introduction

1.1 Overview

Performance and scalability are keys to your e-business systems. While your business is growing, you need more power on your e-business site to support more customers. This paper demonstrates the basic concepts and steps of setting up such an e-business site using WebSphere Performance Benchmark Sample (the Trade2 application) with DB2 EE and a cluster of WebSphere nodes (WebSphere Advanced Server 3.5.4) on Windows 2000. It walks through the major steps of setting up a small e-business system using DB2 and WebSphere, including

- How to scale up to a WebSphere cluster.
- What choices you have in term of balancing the load with a WebSphere cluster.
- Performance tuning tips.

1.2 Purpose

1.2.1 Problem statement

The current WebSphere documentation available from the IBM web site focuses on general installation and configuration information. This paper attempts to fill in some gaps with regard to setting up a real-world application in a cluster setting.

1.2.2 Design statement

Design and set up the popular multi-node topologies for installing WebSphere, and highlight and generalize the important steps for configuring the supporting software, where Trade2 is used as a sample application.

1.3 Intended audience

This paper is intended for anyone who is interested in setting up a small e-business system using DB2 and WebSphere. It's intended for any Software Engineers, Performance Analysts, Application Developers, or System Administrators who are responsible for or interested in deploying or tuning production e-business solutions using IBM WebSphere Advanced Server with DB2 Universal Database.

2. Cluster topology

To achieve the goals of demonstrating how to set up a WebSphere server and how to scale up to a cluster, we need multiple machines on which to install WebSphere Advance Server. The Trade2 application was cloned among these WebSphere Application Servers. Because there are several different methods of load balancing to distribute the traffic across these WebSphere nodes, the steps of how to set different load balancing methods are also included. In this paper, three load balancing methods are described and demonstrated in two different topologies. They are:

- Clustering without session affinity (the default).

- Clustering with session affinity by setting sticky time in WebSphere Edge Server.
- Clustering using WebSphere HTTP plug-in session affinity on each WebSphere node.

2.1 Clustering without session affinity

In this topology (see Figure 1), WebSphere Edge Server is running on a separate machine to provide load balancing across the whole WebSphere cluster (up to 8 WebSphere nodes). No special settings are used in either the Edge Server (sticky time is not set by default) or the WebSphere node. Therefore, the Edge Server will balance the load across the cluster in a round-robin way without any session affinity turned on in any tier. This configuration is easy to set up but might not be able to make good use of the session cache in WebSphere Application Server.

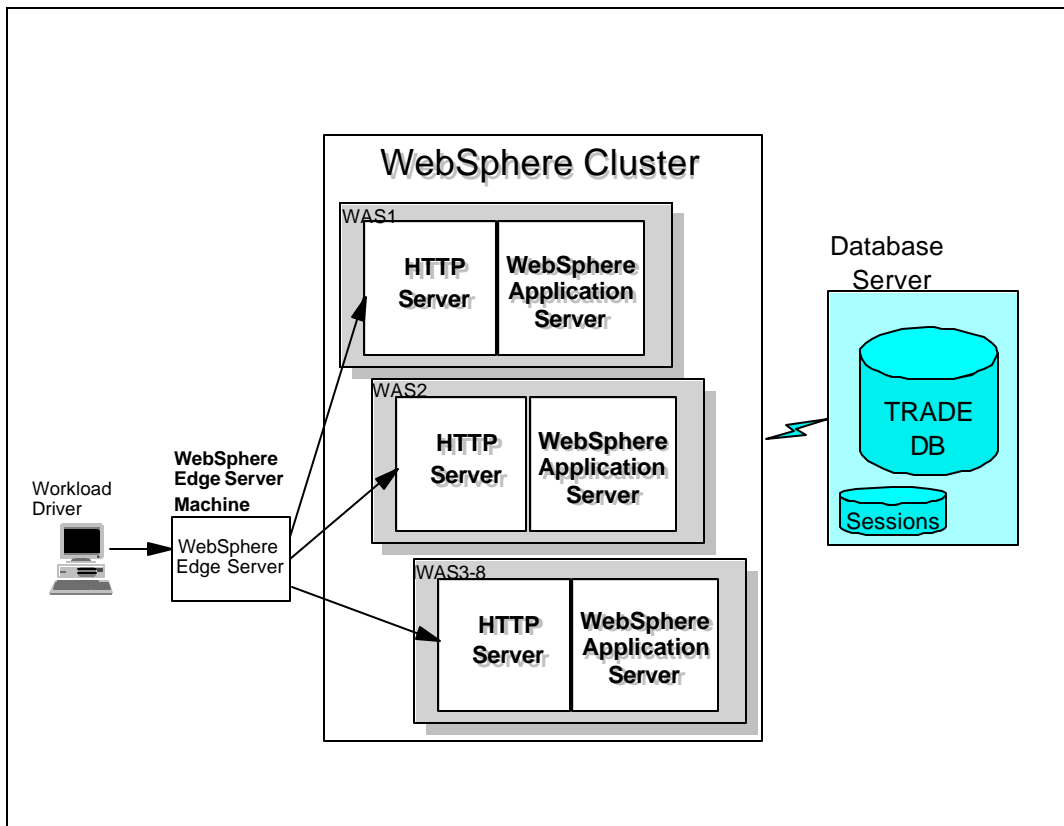


Figure 1: Clustering without session affinity

2.2 Clustering using session affinity by setting sticky time

In this topology (see Figure 2), WebSphere Edge Server is running on a separate machine to provide load balancing across the whole WebSphere cluster (up to 8 WebSphere nodes). We enabled the affinity feature by configuring a cluster's port to be “sticky,” thereby enabling subsequent client requests that came from the same IP address to be

directed to the same server. You can set sticky time by setting "port sticky time" to some number of seconds.

When sticky time is enabled, WebSphere Edge Server will balance the requests based on the IP address of the requests. In our test environment, all requests from each workload driver (Benchmark Factory in this study) used the IP address of the physical machine. The IP-based routing drove the need to have at least one client machine (IP address) per WebSphere server for testing the sticky time configuration.

The advantages of this configuration include:

- It is easy to configure.
- No coding is needed.
- There is no application server performance overhead.

Furthermore, by setting sticky time, the session cache hit ratio in WebSphere Application Server will be improved, as well as the performance in a big cluster. This configuration works best with most internet scenarios but should not be used when all clients come from a single client proxy with the same address, since they will be routed to the same Web server.

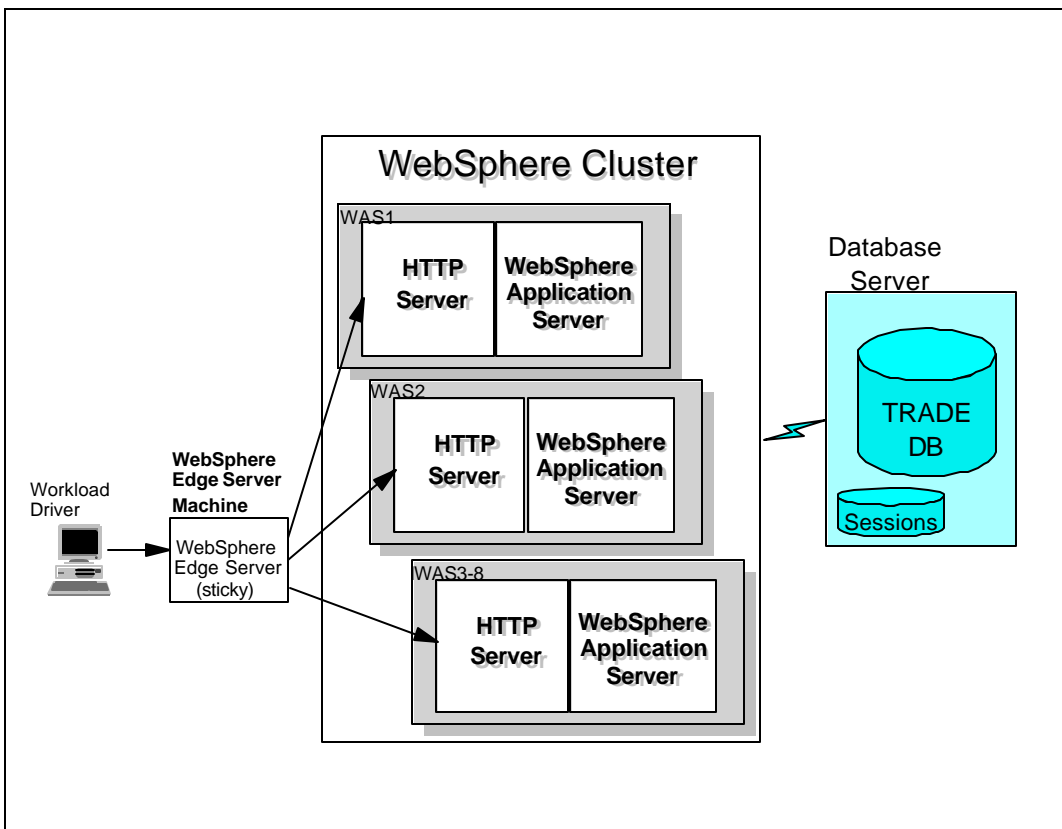


Figure 2: Clustering using session affinity by setting sticky time in WebSphere Edge Server

For information about how this configuration scales, see the related paper, “A Scalability Study for WebSphere Application Server and DB2 Universal Database” [7].

2.3 Clustering using WebSphere HTTP plug-in session affinity

In this topology (see Figure 3), WebSphere Edge Server is running on a separate machine to provide initial load balancing or dispatching across the whole WebSphere cluster (up to 8 WebSphere nodes). No special settings are used in the Edge Server (sticky time is not set by default). Session affinity is achieved with the WebSphere HTTP plug-in configured for OSE remote. Remote OSE is a capability provided with WebSphere 3.5 Advanced Edition. This load balancing function uses the proprietary Open Servlet Engine (OSE) transport to route requests from the HTTP Server plug-in to application servers on remote machines.

This OSE remote configuration provides additional flexibility for load distribution, and also offers some distinct advantages in secured configurations where the Web server may be separated from the rest of the WebSphere system by a firewall. This configuration will not have problems even if all the clients connect through the same client-side proxy. The disadvantage is that the configuration can incur some performance overhead on the Web server.

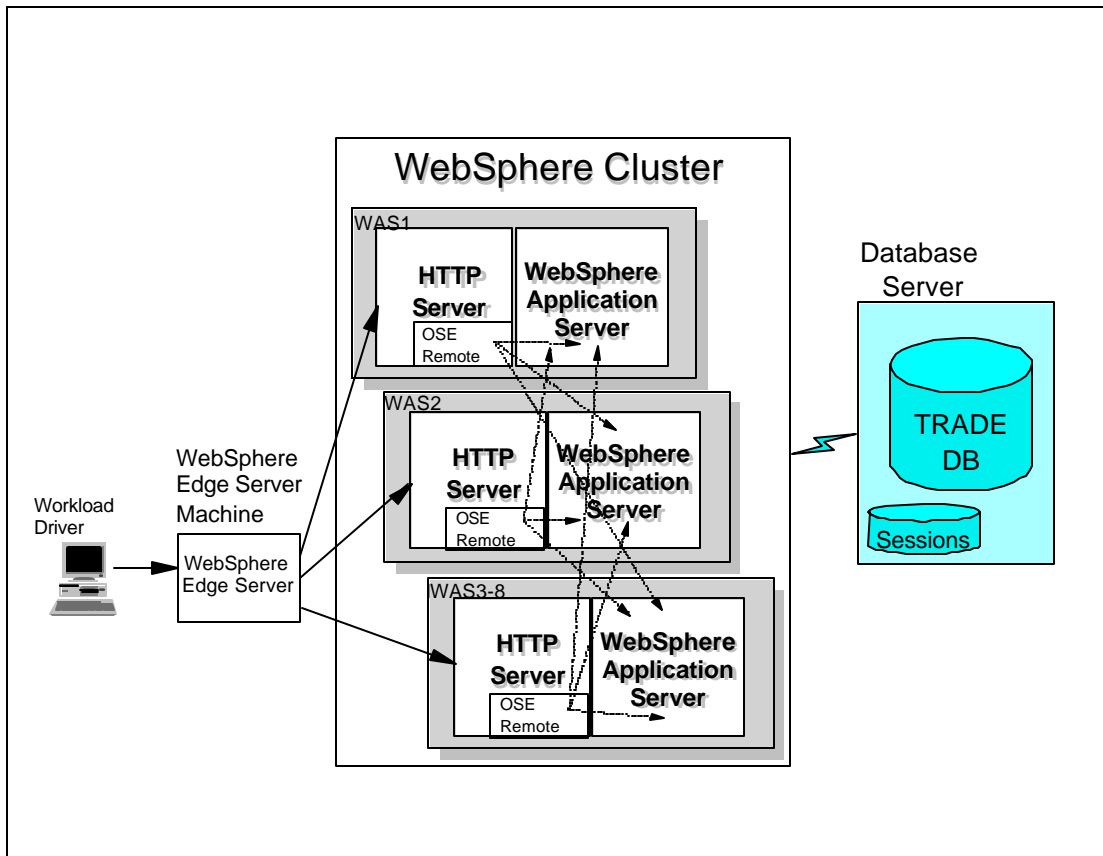


Figure 3: Clustering using HTTP plug-in session affinity (OSE remote) on each WebSphere node

2.4 Machine setup and software configuration

The following table (Table 1) provides the hardware and software configuration on each of the machines that were used for this demonstration. For simplicity, we used the same machines with common hardware configurations. The benefit of doing so is that we can easily change the function of a particular machine to explore multiple configurations. The only exception is that the machine running the DB2 server is a 8x700 MHz instead of a 4x700 MHz. Readers are encouraged to look at the respective product manuals for the minimum requirements of running the software products.

Table 1 : Information summary for hardware and software setup

Tier	Setup information
Tier 4: Database server (1 machine, D1)	Intel Xeon 8X700MHz Windows 2000 Advanced Server DB2 UDB v7.2 FP 3
Tier 3: WebSphere Application Server, web server, and Trade2 application (8 machines, WAS1 – WAS8)	Intel Xeon 4X700MHz Windows 2000 Advanced Server IBM HTTP Server 1.3.12.3 WebSphere Advance Server 3.5.4 JDK (default) DB2 UDB v7.2 FP3 (client only)
Tier 2: WebSphere Edge Server (1 machine, ND)	Intel Xeon 4X700MHz Windows 2000 Advanced Server WebSphere Edge Server 2.0 (Network Dispatcher v3.6)
Tier 1: Workload driver (N machines, Client1 – ClientN)	Intel Xeon 4X700MHz Windows 2000 Advanced Server Workload Generator

Figure 4 provides an overview of how the software components interact to form the testing environment. For the clients, you can use any workload generator, or you can just use a web browser. IBM WebSphere Edge Server routes the requests from the clients to the WebSphere Application Servers in the WebSphere cluster. WebSphere services the requests and interacts with the DB2 database that contains the application data. You can use as many or as few clients as you want. However, when demonstrating clustering with session affinity by setting sticky time in WebSphere Edge Server (Figure 2), you have to use as many clients as the number of WebSphere Application Servers in order to see request arrivals on every WebSphere Application Server.

In the example provided in this paper, we used up to 8 WebSphere Application Servers in the WebSphere cluster. This paper describes only the procedures for setting up the first and second nodes. The procedure for setting up the other nodes will be the same as that for setting up the second node.

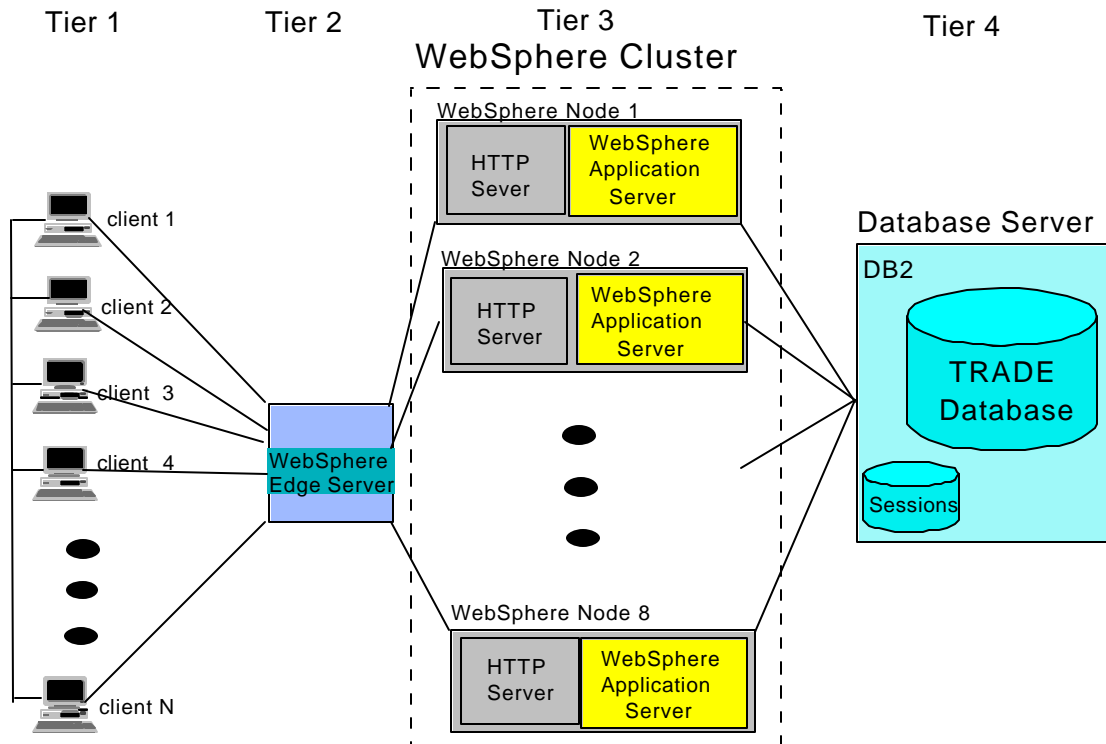


Figure 4: Software configuration and system diagram

3. Setting up the databases

In this section, we describe the steps involved in setting up the application database, WebSphere administrative repository, and the database for persistent session data.

3.1 Installation summary

Step no.	WAS1 – WAS8	D1
1		Install DB2
2		Create WAS database Create Session database Create Trade database
3	Install DB2 Client	
4	Catalog node Catalog databases	

3.2 Install DB2

You can use product documentation to install DB2 [1] on the database server D1. Use the “Typical Installation” option.

After you install and reboot, stop and disable the Warehouse logger and Warehouse server services, which are not needed for this configuration.

3.3 Create databases

3.3.1 Creating the databases

To create the required databases on the database server, run the following commands:

```
db2 -v create db WAS
db2 -v create db SESSION
db2 -v create db TRADE
```

These commands create databases named WAS, SESSION and TRADE.

The WAS database is used as the WebSphere administrative repository. It is used to store the configuration information of the entire system.

The SESSION database is used to store the persistent HTTP session data. In WebSphere, the only way to share a session between multiple application servers is to make the session persistent in a database.

The TRADE database is the application database. All the users and trading information is stored in this database.

3.3.2 Creating the tables

To create the Trade tables on the database server, run the following script:

```
<Trade Install>\database\DB2\createTable.bat
```

The createTable.bat script creates all tables required by the Trade2 application. It makes a connection to the TRADE database, and creates the tables through DB2 command line processor (CLP).

3.3.3 Database tuning

Database tuning is very important in a large cluster. The databases in our lab (having 1,000,000 users and 5000 quotes to support 12,800 users with 7-second think time) were tuned as following:

```
rem Change the default bufferpool setting for TRADE database
db2 -v connect to trade
db2 -v "select * from sysibm.sysbufferpools"
db2 -v alter bufferpool IBMDEFAULTBP size -1
db2 -v "select * from sysibm.sysbufferpools"
db2 -v terminate

rem Update TRADE database configuration parameters
db2 -v update db cfg for TRADE using applheapsz 256
db2 -v update db cfg for trade using LOGBUFSZ 256
db2 -v update db cfg for trade using BUFFPAGE 400000
db2 -v update db cfg for trade using MAXLOCKS 10
db2 -v update db cfg for trade using MAXAPPLS 256 AVG_APPLS 128
db2 -v update db cfg for trade using LOGFILSIZ 5000 LOGPRIMARY 3 LOGSECOND 10
```

```

rem Change the default bufferpool setting for SESSION database
db2 -v connect to session
db2 -v "select * from sysibm.sysbufferpools"
db2 -v alter bufferpool IBMDEFAULTBP size -1
db2 -v "select * from sysibm.sysbufferpools"
db2 -v terminate

rem Update SESSION database configuration parameters
db2 -v update db cfg for session using applheapsz 256
db2 -v update db cfg for session using LOGBUFSZ 256
db2 -v update db cfg for session using BUFFPAGE 100000
db2 -v update db cfg for session using MAXLOCKS 10
db2 -v update db cfg for session using MAXAPPLS 256 AVG_APPLS 128
db2 -v update db cfg for session using LOGFILSIZ 5000 LOGPRIMARY 3 LOGSECOND
10

rem Change the default bufferpool setting for WAS database
db2 -v connect to was
db2 -v "select * from sysibm.sysbufferpools"
db2 -v alter bufferpool IBMDEFAULTBP size -1
db2 -v "select * from sysibm.sysbufferpools"
db2 -v terminate

rem Update was tuning parameters
db2 -v update db cfg for was using LOGBUFSZ 256
db2 -v update db cfg for was using BUFFPAGE 10000
db2 -v update db cfg for was using MAXLOCKS 10
db2 -v update db cfg for was using MAXAPPLS 128 AVG_APPLS 64

db2 -v terminate

```

3.4 Install DB2 client

You can use product documentation to install DB2 [1] on WebSphere nodes (WAS1, WAS2, ..., and WAS8). Use the “Typical Installation” option.

3.5 Establish database connections

After the databases are created, establish remote connections from the WebSphere servers to the DB2 server.

3.5.1 Enable remote connections

On the DB2 server, issue the following commands:

```

db2set db2comm=tcPIP
db2 update dbm cfg using svcname db2cDB2

```

The first command sets the DB2 environment variable `db2comm` to indicate that TCP/IP will be used as the protocol for remote communication.

The second line updates the service port to `db2cDB2`. This is the default port available from a Typical Installation.

3.5.2 Catalog the node and databases

On *each* of the WebSphere servers, issue the following commands:

```

db2 catalog tcPIP node D1 remote D1 server db2cDB2

```

```
db2 catalog db was at node D1
db2 catalog db session at node D1
db2 catalog db trade at node D1
```

In the commands above, a TCP/IP node and some remote databases are cataloged.

A TCPIP node named D1 is cataloged in the first statement. The host name of the database server is D1 and the service port is the default, db2cDB2.

The next three statements then catalog each of the databases at the TCP/IP node that were created in the first statement.

4. Installing Trade2 application server

In this section, we describe the steps involved in installing the Trade2 application server on the first WebSphere Server node (WAS1).

4.1 Installation summary

Step no.	WAS1
1	Install WebSphere Apply fix pack
2	Install Trade2
3	Configure for persistent session Enable persistent session

4.2 Install WebSphere

4.2.1 Install the base code

Read the installation guide [2] for complete installation instructions. Ensure that the following options are checked:

- IBM HTTP Server
- IBM JDK 1.2.2
- Configure default server and web application

This will ensure that the Web server, JDK, and sample servers are being installed and configured automatically.

4.2.2 Apply the fixpack

After the installation, the fixpack must be applied.

1. Stop the service “IBM HTTP Server”.
2. Unzip the package to a directory.
3. Run install.bat.
4. Follow the on-screen instructions. When asked whether to upgrade the HTTP Server, choose **yes**.

4.2.3 Start WebSphere adminserver

WebSphere Administrative Server (adminserver) can be started from a command line by running a batch file in the <WebSphere root>/AppServer/bin/debug directory.

```
adminserver.bat
```

If it shows “A WebSphere Administration server open for e-business,” the server is started successfully.

The script will set the required classpath and launch the adminserver class, com.ibm.ejs.sm.server.AdminServer. If it is the first time the adminserver has been started, the adminserver will register its existence to the WAS database at the same time.

4.3 Install Trade2

To install the Trade2 application server, edit the setup.bat file to customize the setup parameters to match the system configuration *before* executing this batch file:

```
WAS_SERVER (Enter WebSphere server hostname - this is case sensitive)
WAS_HOME (WebSphere installation directory)
WAS_UID (WebSphere Administrator Login ID)
WAS_PWD (WebSphere Administrator password)
TRADE_DB (The default, "trade" is correct for DB2.)
DB_UID (DB2 Administrator ID)
DB_PWD (DB2 Administrator password)
JDBC_DRIVER_LOC (JDBC jar/zip file location, i.e. for DB2, db2java.zip)
```

Then run the setup.bat batch file.

The script will validate the parameters and import the Trade2 web application through the WebSphere XML interface. All the configuration information will be stored in the administrative repository database.

4.4 Configure Trade2 for persistent session

To specify that Trade2 use a persistent session, provide information to WebSphere server on the database driver and the session database. Do this by defining a JDBC database driver and data source from the WebSphere administrative client, respectively.

You can configure the Trade2 application server through the WebSphere administrative client, which you can start from a command line:

```
adminclient.bat
```

The file is under the <WebSphere root>\AppServer\bin directory.

4.4.1 Create connection to the session database

To create the JDBC driver and data source:

1. Right click on WebSphere Administrative Domain and select Create->JDBC Driver.
2. Right click on the JDBC driver and select Install.

3. Right click on WebSphere Administrative Domain and select Create->DataSource.

4.4.2 Enable persistent session

To enable a persistent session:

1. Locate the session manager, Trade2 Server->Servlet Engine->Session Manager
2. At the Enable tab, check off Enable Persistent Sessions.
3. At the Persistence tab, enter the session data source, user name, and password.

All these settings will be in effect after the Trade2 application server is restarted.

5. Cloning Trade2

Cloning is a mechanism provided by the WebSphere as a clustering tool. It allows the creation of multiple application servers across WebSphere nodes. In general, the WebSphere administrator creates a model from an application server, and from this model he or she can create any number of clones.

To clone Trade2, create a model from the Trade2 application server on WAS1 and then make a Trade2 clone on each of the WebSphere nodes. The following sections use the second node, WAS2, as an example. You can do the same for the other WebSphere nodes.

5.1 Installation summary

Step no.	WAS1	WAS2
1		Install WebSphere
2	Install database drivers to WAS2	
3		Copy Trade2 code to WAS2
4	Create Trade2 model Make Trade2 clones	
5*	Configure OSE remote	Configure OSE remote

*This step is for OSE Remote setup only (when clustering using WebSphere HTTP plug-in session affinity on each WebSphere node, as shown in Figure 3).

5.2 Install WebSphere

This step is for all the other WebSphere Server nodes other than the first node. Refer to the installation guide [2] for complete installation instructions.

5.2.1 Install the base code

Make sure that the following options are checked:

- IBM HTTP Server
- IBM JDK 1.2.2

Make sure that the following option is *unchecked*:

- Configure default server and web application

5.2.2 Apply the fixpack

After the installation, apply the fixpack:

1. Stop the service “IBM HTTP Server”.
2. Unzip the package to a directory.
3. Run install.bat.

Follow the on-screen instructions. When asked whether to upgrade the HTTP Server, choose **yes**.

5.2.3 Start WebSphere administrative server

The WebSphere administrative server can be started from a command line by running a batch file in the <WebSphere root>/AppServer/bin/debug directory.

```
adminserver.bat
```

If it shows “A WebSphere Administration server open for e-business,” the server is started successfully.

The script will set the required classpath and launch the adminserver class, com.ibm.ejs.sm.server.AdminServer. If it is the first time the adminserver is started, the adminserver will register its existence to the WAS database at the same time.

5.2.4 Update the host alias list

After a new WebSphere node is installed, the host alias list on WAS1 must be updated. These aliases are needed so that the virtual host will accept requests from any specified machines. To update the list, follow these steps on WAS1:

1. Start WebSphere administrative client.
2. Locate default host.
3. At the general tab, add WAS2 to the host alias list.

All the administrative servers must be restarted for this setting to take effect.

5.3 Install database driver

There are three JDBC drivers being used by the Trade2 application server:

- Admin DB Driver
- TradeDBDriver
- SessionDBDriver.

All of these drivers must be installed on the newly added WebSphere node.

To install the drivers, right click on each of the database drivers, select Install, and choose the new node at the Node Parent dialog box.

5.4 Copy the Trade2 application

Copy the following directories from WAS1 to WAS2:

```
<WebSphere root>\Appserver\hosts\default_host\trade2_app
<WebSphere root>\Appserver\deployedEJBs
```

The trade2_app directory contains all the Trade2 web pages, and the deployedEJBs directory has the Trade2 Enterprise JavaBeans.

5.5 Create Trade2 model and clones

5.5.1 Trade2 model

To create Trade2 model, follow the steps below on WAS1:

1. Select Trade2 Server and right click then select Create->Model.
2. In the clone properties dialog box, make sure the following options are checked:
 - Make Trade2 Server a Clone
 - Recursively Model all instances under Trade2 Serverthen click OK.

By following these steps, a Trade2 model is created that uses the Trade2 server as a template, and the original Trade2 server is turned into a clone of the model.

5.5.2 Trade2 clone

To create a Trade2 clone, follow the steps below on WAS1:

1. Select the Trade2 Model and right click then select Create->Clone.
2. In the Clone Parent dialog box, select the node WAS2 and click Create

5.6 Configure OSE remote (only for OSE remote setup)

Note: This step is for OSE remote setup only. If you choose to use sticky time in WebSphere Edge Server, you can skip this step.

5.6.1 Configure the transport type

By default, WebSphere runs on local pipe. To change the transport type to OSE remote, do the following on WAS1:

1. Locate the ServletEngine under Trade2 Model.
2. At the Advanced tab, click Settings.
3. In the Edit Servlet Engine Transport box, choose INET Sockets for Transport type, and click OK.
4. Click Apply to confirm the change.
5. Restart Trade2 Model.

5.6.2 Configure the plug-in for OSE remote

To generate the plug-in for OSE remote, run the script OSERemoteConfig.bat under <WebSphere root>\Appserver\bin on each WebSphere node:

```
OSERemoteConfig.bat -serverRoot <WebSphere root>\AppServer
                    -adminNodeName WAS2
```

Then restart the HTTP server.

When the script has finished running, the following temporary files are created for the HTTP server plug-in and reside in the subdirectory <WebSphere root>\temp:

queues.properties
rules.properties
vhosts.properties

Restart the HTTP server to pick up the changes in these properties files.

6. Setting up WebSphere Edge Server

6.1 Installation summary

Step no.	WebSphere Edge Server (ND)	WAS1
1	Install WebSphere Edge Server, and configure Network Dispatcher	
2		Update the host alias list
3*		Install loopback adapter
4**	Specify the sticky time	

* This step must be repeated for WAS2-WAS8.

** This step is for Edge Server sticky time session affinity only (when clustering using session affinity by setting sticky time in WebSphere Edge Server, as shown in Figure 2).

6.2 Install WebSphere Edge Server

6.2.1 Installation

In our case, we used a separate WebSphere Edge Server machine. See the installation guide [4] for the installation instructions. For our purposes, we selected “your choice of components” option and make sure to check “Network Dispatcher” during the installation.

6.2.2 Configuration

Network Dispatcher (ND) provides a graphical user interface to configure the server. To start the configuration wizard, bring up the ND admin client. Right click on the Dispatcher node and select Start Configuration Wizard. With the configuration wizard, you see the following panels:

1. Introduction to the wizard
2. What is going to happen
3. Preparing for the setup
4. Choosing a host to configure
5. Defining a cluster
6. Adding a port
7. Adding a server
8. Starting an advisor
9. Server machine setup

The wizard will guide you step by step through the process of creating a basic configuration for the Dispatcher component.

Steps 1 to 3 describe some background information about the wizard and the prerequisites for setting up the network dispatcher.

In step 4, you are asked to give the host name of the network dispatcher. In our case, the host name is ND.

In step 5, a cluster is required to be defined. This is the address that will be provided to the clients who wish to access the site. In our example, the cluster address is NDcluster.

In step 6, the ports that are going to be used for the cluster have to be defined. In our case, we will specify port 80 only because we will use HTTP clients in our study.

In step 7, the servers are added to the cluster. WAS1 can be added to the cluster at this point. WAS2-8 can all be added to it afterwards.

In step 8, the wizard asks if the Advisor function should be started. The advisor gives the network dispatcher more information about the ability of the load-balanced server machines to respond to requests. For this example, the assumption is that all the servers have the same ability, and therefore the Advisor is not needed.

Finally in step 9, the wizard guides you through the process of setting up the server machines, i.e. WAS1-WAS8.

6.2.3 Update the host alias list

After the network dispatcher (ND) is set up, the host alias list must be updated. These aliases are needed so that the virtual host will accept requests from any specified machines. To update the list, follow these steps on WAS1:

1. Start WebSphere administrative client.
2. Locate default_host.
3. At the general tab, add ND to the Host Alias list.

The setting will be in effect after all the administrative servers are restarted.

6.3 Install the loopback adapter

The Dispatcher creates the illusion of having just one server by grouping systems together into a cluster that behaves as a single, virtual server. All the servers that will have their loads balanced by this ND machine should have the cluster address alias on the loopback adapter. Before the ND starts to forward TCP/IP connection requests to the WebSphere Servers, it is necessary to add an alias to the cluster address on the loopback interface. The loopback IP address is usually 127.0.0.1 and is never forwarded as a destination on the network media. On Windows 2000, the loopback adapter is a network adapter driver that can be installed from the Windows 2000 CD. The following are the steps for installing the loopback adapter on every WebSphere machine.

6.3.1 Install

1. Click Start, click Settings, then click Control Panel.
2. Double-click Add/Remove Hardware. This launches the Add/Remove Hardware Wizard.
3. Click Next, select Add/Troubleshoot a Device, then click Next.
4. The screen blinks off/on, then presents the Choose a Hardware Device panel.
5. If the MS loopback adapter is in the list, it is already installed-- click Cancel to exit.
6. If the MS loopback adapter is *not* in the list-- select Add a New Device and click Next.
7. To select the hardware from a list, from the Find New Hardware panel, click No and then click Next.
8. Select Network Adapters and click Next.
9. On the Select Network Adapter panel, select Microsoft in the Manufacturers list, then select Microsoft loopback adapter.
10. Click Next, then click Next again to install the default settings (or select Have Disk, then insert CD and install from there).
11. Click Finish to complete installation.

6.3.2 Configure

1. From the Control Panel, Double-click Network and Dial-up Connections.
2. Select the connection with Device Name "Microsoft loopback adapter" and right-click on it. Select Properties from the dropdown.
3. Select Internet Protocol (TCP/IP), then click Properties.
4. Click Use the following IP address. Fill in *IP address* with the cluster address, and *Subnet mask* with the default subnet mask (255.0.0.0).

6.3.3 Change the bind order as needed

1. From the Control Panel, Double-click Network and Dial-up Connections.
2. Select Advanced from the dropdown.
3. Select Advanced Settings from the pull down.
4. Ensure the loopback adapter is last.
5. Add a DNS entry as needed.

6.3.4 Remove the extra route

On Windows 2000, a default route may have been created as a result of adding the loopback adapter. You must delete this extra route.

View the active routes by using the following command:

```
route print
```

The extra route is characterized as a row with a repeated cluster address under the Gateway Address column. We need one of these routes and will need to delete the other route, which is extraneous. The route to be deleted is the one whose network address begins with the first digit of the cluster address, followed by three zeroes.

The extra route can be deleted by the following command

```
route delete <network_address> <cluster_address>
```

The network address and cluster address can be copied from the extra route pointed out from the routing table. This command must be issued after every reboot. You can put this command into the AUTOEXEC.BAT file so that it can be done automatically.

6.4 Specify the sticky time

Note: This step is for Edge Server sticky time session affinity only. If you choose to use OSE remote in section 5.6, you don't need to do this step.

Sticky time is a feature that Network Dispatcher used to maintain session affinity. To specify the sticky time, bring up the network dispatcher administrative client on the ND machine and do the following:

1. Locate the Dispatcher node.
2. Connect to the ND host.
3. Locate Port:80 node.
4. At the Configuration Settings tab, update the Sticky time.
5. Click Update Configuration to confirm the change.

By using sticky time, an affinity record is created for each client based on the IP address of the client, and each affinity record lives for the specified period of time. In other words, the subsequent requests go to the same Web server, as long as the client responds within the defined sticky time. If a subsequent connection is not received within the sticky time, the record is purged, and a connection that is received after that time will have a new server selected for it.

7. Tests from client machines

After you complete all the previous steps, you have a WebSphere cluster with the Trade2 application deployed. The system is ready for doing transactions from a client machine. You can send some Trade2 requests by pointing your browser to the cluster address: <http://<Ndcluster>/trade/servlets/TradeSenarioServlet>.

You can also use some other workload drivers [5, 6] to simulate a more realistic workload to stress a part of the cluster or the whole cluster.

8. Summary

This paper describes several installation and configuration options for setting up a small e-business system using DB2 and WebSphere. It also demonstrates how to scale up to a WebSphere cluster, and what choices you have in terms of load balancing a WebSphere cluster. With these simple steps, you can start to enjoy the IBM solution for building an infrastructure that will support demanding e-business workloads.

References

- [1] DB2 Product Technical Library at <http://www-4.ibm.com/software/data/db2/library/>
- [2] WebSphere Application Server Version 3.5 and 4.0 Standard and Advanced Editions and Version 4.0 Enterprise Edition at <http://www-4.ibm.com/software/webservers/appserv/library.html>
- [3] Installation of WebSphere Performance Benchmark Sample at http://www-4.ibm.com/software/webservers/appserv/wpbs_readme.html
- [4] WebSphere Edge Server at <http://www-4.ibm.com/software/webservers/edgeserver/library.html>
- [5] LoadRunner at <http://www-svca.mercuryinteractive.com/products/loadrunner/>
- [6] Benchmark Factory from Quest Software at http://www.quest.com/benchmark_factory/
- [7] Yongli An, Tony Lau, and Peter Shum. “A Scalability Study for WebSphere Application Server and DB2 Universal Database” at <http://www7b.boulder.ibm.com/dmdd/library/techarticle/0202an/0202an.pdf>

About the authors

Yongli An, DB2 UDB Performance Engineer, IBM Certified Solutions Expert – DB2 UDB 7.1 Database Administration for UNIX, Windows and OS/2®. Yongli is experienced in TPC-C and WebSphere benchmarking. His current focus is DB2 performance for WebSphere Advanced Server and e-business applications.

Tony Lau is a DB2 Performance Engineer. He earned his Bachelor Degree of Applied Science in computer engineering from the University of Waterloo. His current focus is DB2 performance for WebSphere Advanced Server and e-business applications .

Notices, Trademarks, Service Marks and Disclaimers

This document contains proprietary information of IBM. It is provided under a license agreement and is protected by copyright law. The information contained in this publication does not include any product warranties, and any statements provided in this document should not be interpreted as such.

The following terms are trademarks or registered trademarks of the IBM Corporation in the United States and/or other countries: IBM, DB2, DB2 UDB, DB2 Universal Database, WebSphere, OS/2.

Intel and Intel-based trademarks and logos are trademarks or registered trademarks of Intel Corporation.

Windows and Windows-based trademarks and logos are trademarks or registered trademarks of Microsoft Corp.

Unix and Unix-based trademarks and logos are trademarks or registered trademarks of The Open Group.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Other company, product or service names may be the trademarks or service marks of others.

The furnishing of this document does not imply giving license to any IBM patents.

References in this document to IBM products, Programs, or Services do not imply that IBM intends to make these available in all countries in which IBM operates.