IBM DB2 Content Manager V8
Configuring High Availability
in a
Sun Cluster Environment

5/4/04

Content Management Performance
IBM Silicon Valley Lab

1. Special Notices

The information contained in this publication was derived under specific operating and environmental conditions. Furthermore it has not been submitted to any formal IBM test and is distributed on an "as is" basis without any warranty either expressed or implied. The use of this information or the implementation of any of these techniques is the responsibility of the user and depends on the user's ability to evaluate and integrate them into the user's operational environment. While each item might have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Users attempting to adapt these techniques to their own environments do so at their own risk.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM licensed program in this publication is not intended to state or imply that only IBM's program may be used. Any functionally equivalent program may be used instead. Equivalent product, program, or services that do not infringe any of the intellectual property rights of IBM may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

IBM may have patents or pending patent application covering the subject matter in this document. The furnishing of this document does not give you license to these patents.

Any information about non-IBM ("vendor") products in this document has been supplied by the vendor and IBM assumes no responsibility for its accuracy or completeness.

IBM, DB2, WebSphere, and Tivoli are trademarks or registered trademarks of IBM Corporation in the United States, other countries, or both.

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

UNIX is a trademark of The Open Group.

Microsoft and Windows are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

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

.

2. The team that wrote this paper

This white paper was produced by a team of specialists from Sun Microsystems and IBM Corporation. The project team and direct contributors were:

Annie Kuo	Sun Microsystems, Menlo Park CA
Jan Van Bruaene	Sun Microsystems, Menlo Park CA
Julie McDonald	Sun Microsystems
Xin Chen	IBM DB2 Ranger Team - IBM IM Solutions for Hardware Platform Alliances
Barry Beach	IBM DB2 Ranger Team - IBM IM Solutions for Hardware Platform Alliances
Cataldo Mega	IBM Content Manager Performance, SVL CA
Bruce Smith	IBM Content Manager Performance, SVL CA
Richard Heffel	IBM Content Manager Performance, SVL CA
David Bronner	IBM Content Manager Performance, SVL CA
Neil Pennington	IBM Content Manager System Test, SVL CA
Mike Janini	IBM Systems Group, Open Systems Validation Lab, Cottle
Dominick Nguyen	IBM Systems Group, Open Systems Validation Lab, Cottle
Alejandro Halili	IBM Systems Group, Open Systems Validation Lab, Cottle

2.1 Acknowledgements

We want to thank all our comrades at Sun Microsystems, in particular, Annie Kuo, Jan Van Bruaene, and Julie McDonald. Not only did we work with them on this project to produce this white paper, we enjoyed their company. Before the equipment necessary for this project became available at SVL, this team secured the necessary infrastructure at their Menlo Park facility so that both Sun and IBM could setup the environment in December 2003. Further thanks go out to the Sun Cluster team, especially Roger Autrand and Shishir Agrawal, with whom IBM has been working for nearly two years to ensure compliance with Sun Cluster specifications.

We want to especially thank Dean Underwood and Maureen O'Donnell for providing a loaner Shark model F20 and Thomas DeNatale for letting us borrow a McData 3216 Fibre Channel fabric switch. Mark Meier and Jeffry Larson aided this effort by loaning us fiber optic cabling to connect our SAN components. Lucian Williams, also from the IBM Systems Group at Cottle in San Jose, initially configured our storage device when it arrived. The Open Systems Validation Lab team at Cottle, that is responsible for the Sun platform, also aided in configuring Sun's Cluster V3.x software. Without the assistance of the IBM Systems Group, this white paper project would not have been completed at this time.

Many thanks also to Stewart Tate, Atul Gore, Mario Lichtsinnn and all the Content Manager Performance team for being available to us at any time to discuss details about specific HA configuration and their implications. Thank you to the Information Development and Legal department at SVL for reviewing and correcting this paper. A big thank you all first and second managers who helped organizing the required resources and supported us throughout the duration of this project.

Last but not least, many thanks to all the many other colleagues that helped make this paper a reality.

This edition applies to IBM DB2 Content Manager Version 8.2 and IBM WebSphere Server V5.0.2, or higher for use with Sun Solaris V8 or V9 operating systems and Sun Cluster Services V3.1.

3. Preface

IBM DB2 Content Manager for Multiplatform V8.2 is an Enterprise-class application that customers might want to install and set up in a high availability configuration. Installing this kind of application in a clustered environment is not an out-of-box solution. Proper design, installation, configuration, fail-over testing of DB2 Content Manager V8.2, and support for running DB2 Content Manager V8.2 in a production environment should be considered a complex endeavor potentially prone to error. Therefore, IBM advises the end-user/organization implementing this HA system to contract with an authorized consulting organization certified to provide support of high availability environments. Also be advised that High Availability support is not provided as part of IBM's Premium Support package

High availability (HA) environments are designed to eliminate single points of failure throughout the application stack by leveraging the high availability capabilities and services in IBM DB2 Universal Database (DB2 UDB), IBM WebSphere Application Server, and the clustering software support provided by the underlying operating systems. IBM DB2 Content Manager is built upon the strength of DB2 UDB and WebSphere Application Server and exploits many of their key features.

This publication is intended to help content management marketing and sales teams and content management solutions designers understand how to enable content management solutions to exploit high availability on the officially supported operating systems using IBM® DB2® Content Manager V8. It is not intended to be a generic high availability support statement of system-specific cluster services.

While the DB2 Content Manager high availability concepts discussed here are applicable to many different operating systems, the actual implementation of DB2 Content Manager high availability environment depends heavily on the individual hardware and software components used, thus rendering impossible a direct comparison of the different operational scenarios. Due to the numerous configurations on which customers can install DB2 Content Manager V8.2, it is not possible for IBM to test each of the potential configurations on each of the various hardware platforms. Given the considerable number of possible configurations of DB2 Content Manager in a high availability environment, IBM intends to publish several white papers as reference material for many of the high availability configurations. Each of these white papers will target a different operating system, for example, IBM AIX with HACMP, Sun's Solaris Operating Environment with SUN Cluster 3.1, and Microsoft Windows 2003 with Microsoft Cluster Server. The main objective of these white papers is to provide a description of a tested high availability implementation on certain operating systems with some detailed configuration procedures. These configurations might require a separate agreement with IBM Global Services (or another IBM certified vendor certified to provide high availability solutions in an IBM DB2 Content Manager context) for support. High availability support is not provided as part of Premium Support package.

The information provided in this white paper is intended merely to complement, not to substitute, the IBM DB2 Content Manager for Multiplatforms product documentation provided by IBM.
See the publications section of the IBM Programming Announcement for IBM DB2 Content Manager for Multiplatforms for more information about which publications are considered to be product documentation

3.1.1 How this white paper is organized

Many IBM DB2 Content Manager V8 customers have been asking for technical information on how to set up highly available Content Manager Production environments.

In this document we will provide detailed hands-on guidance for how to install and set up a Content Manager V8 high availability environment on a cluster of Sun multi-processor systems, using Sun Cluster Services V3.1.

This document provides information and tips to help you set up high availability for IBM DB2 Content Manager Version 8.2 (CM) in a clustered Sun environment using Sun Cluster Services V3.1. The Topics covered include:

- Planning for high availability
- Application design choices and tradeoffs
- In-depth configuration recommendations for the library server and resource manager DB2 databases
- In-depth configuration recommendations for the resource manager WebSphere® Application Servers, including how to use WebSphere horizontal and vertical cloning capabilities

The following table outlines the content of this paper.

Section	Description
3-4	An overview of IBM DB2 and WebSphere Application Server V5 high availability features.
5-10	Our recommended best practices for Content Manager high availability, including • An overview of the high availability configuration and application design choices. • Initial planning for a Content Manager high availability environment • Detailed installation and configuration steps of the Content Manager V8.2 servers and required applications.
11	Testing the Content Manager high availability environment and troubleshooting configuration problems
12	Additional installation and configuration steps for WebSphere NetDispatcher component.
13-16	References to other online resources

Table 3-1

Throughout this paper it is assumed that the reader has a good understanding of the business requirements for high availability in a Content Manager production environment. If available, platform-specific knowledge of all software components used for this solution will also help to better understand the technical details outlined.

Given the complexity of the Content Manager V8 product it self and the many pre required applications, it is almost impossible to document all variants of possible Content Manager V8 high availability scenarios. Therefore we will discuss a few alternative configurations and concentrate on the details of a representative configuration. We will also not try to fully describe how to set up the related applications, but rather focus on configuration aspects relevant to the setup of the chosen Content Manager high availability environment.

Product specific configuration information can be found in the relevant product documentation, for which we will provide references. It should be noted however, that the Content Manager co-requisite products, such as Windows 2003 Enterprise Server, DB2 V8.1, WebSphere V5, and Tivoli® Storage Manager, all come with their own high availability support, and with respective technical documentation of which the reader should have a good working knowledge. For the discussions in this paper, we assume that this documentation has been consulted, and that the relevant technical facts are known to an adequate level.

Most of the focus of this document is on describing the preferred, most practicable Content Manager high availability scenario on the Sun platform.

3.2 Introduction

Enabling Content Manager for high availability means eliminating single point of failure, (SPOF), in the entire Content Manager product stack. Beginning with the Content Manager eClient at the top of the stack and down to the Tivoli Storage Manager device manager component of the Content Manager V8 resource manager.

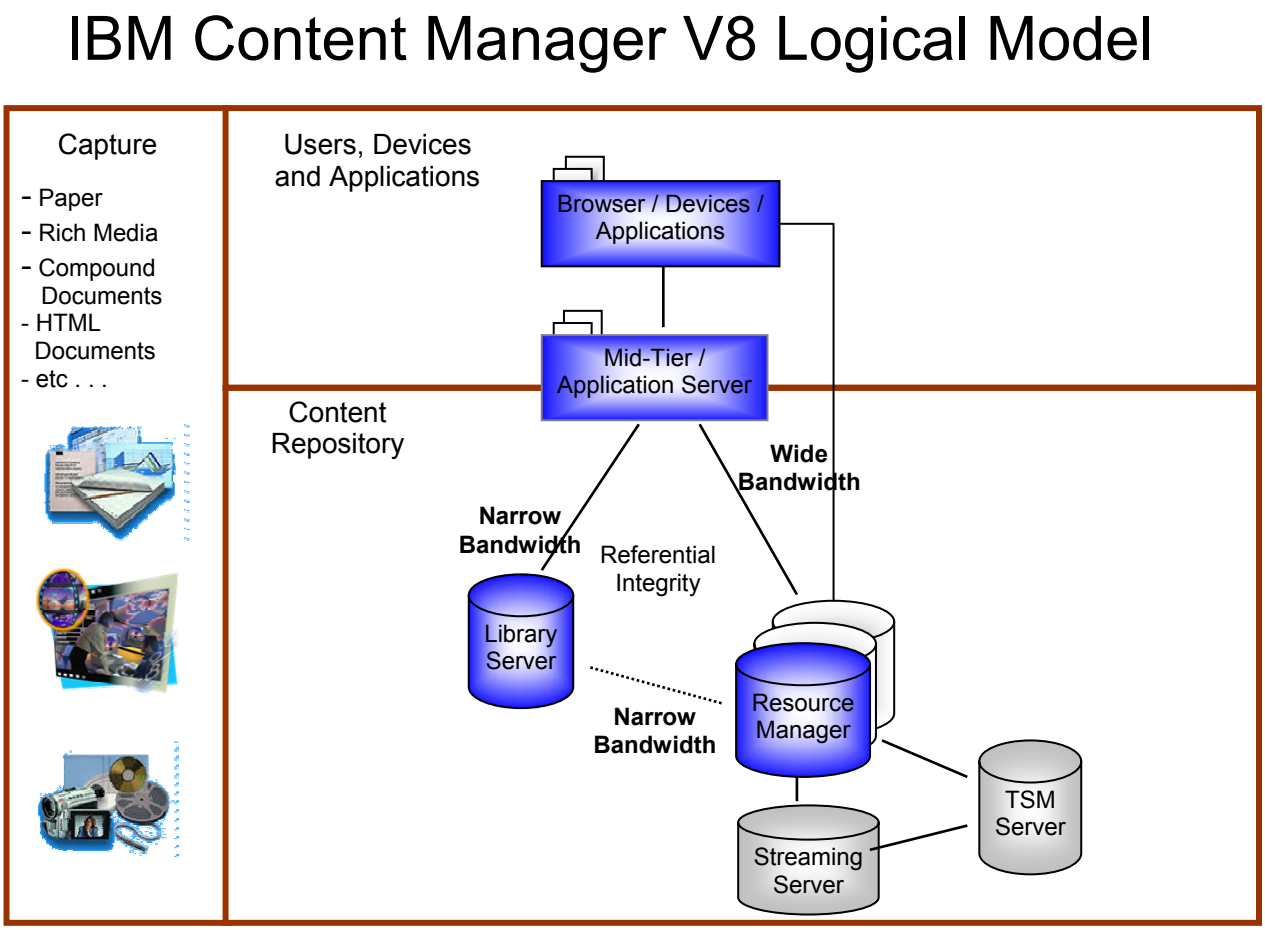


Figure-3-1. Content Manager logical system layout

As shown in
Figure-3-1, a Content Manager high availability production scenario must be built around the middle-tier and backend layers as these layers constitute the backbone of every Content Manager high availability solution.
High availability of these layers is achieved by means of hardware and software redundancy.

The entry point into the middle tier must be configured using a redundant pair of Web servers. The Web servers relay http requests from end users to a cluster of Web application servers hosting the Content Manager eClient and resource manager.

It is to be noted that the Content Manager resource manager can play two roles:

1. Backend server
2. Middle-tier server.

For some operations an end user would access the resource manager using the eClient; other operations would directly issue calls to the resource manager.

In all cases the eClient and resource manager a pair of redundant Web application clones must be configured.

The Web application clones then forward the client requests to a redundant pair of DB2 database instances hosting the Content Manager library server and resource manager databases. Finally, all production data is stored on a high availability disk storage subsystem.

3.3 Background and definitions of terms

Starting with version 8, IBM DB2 Content Manager has become a true DB2 and WebSphere application. As a consequence, Content Manager V8 is now able to fully exploit their respective high availability capabilities and services. This means that the high availability capabilities of Content Manager are basically a blend of the high availability features of its prerequisite applications.

In this sense, configuring a high availability environment for Content Manager means configuring and exploiting DB2, WebSphere, and the platform-specific high availability services in a smart way.

Before we start configuring a high availability environment for Content Manager, let's first define the terms we will use throughout the paper and then agree on what those mean.

1. High availability
2. Load balancing (load balancing high scalability)
3. Disaster recovery

Load balancing, high availability, and disaster recovery frequently overlap in the technologies they employ, but in fact have two distinct purposes.

High scalability and load balancing are the requirement to utilize available system resources in the most efficient way to the benefit of the customer business. The goal here is to maximize cost and efficiency.

High availability is the requirement that the data assets relevant to the business operations must be available whenever required by dependent business applications. The goal here is to eliminate or minimize business operations downtime.

Disaster recovery protects data against loss caused by catastrophic failure. A failure of this type would mean loss of data assets due to an unrecoverable condition. The goal here is to protect against the loss of vital business data.

For the purpose of this document, we are only interested in high availability**,** and to a certain extent on performance and scalability. Our goal is to describe how to configure a Content Manager production system in order minimize business operations downtime and to maximize resource system utilization.

3.4 DB2 UDB high availability support and features

High availability and disaster recovery are key requirements for critical database applications. IBM DB2 Universal Database™ (DB2 UDB) provides many features to meet these requirements.

Articles that summarizes these features and guide in understanding how DB2 technologies are widely available and can be used to design highly available and recoverable Content Manager library server and resource manager databases.

Note: In the context of DB2, the term *cluster* is used to refer to high availability clustering as supported by the Sun Cluster Software.

As mentioned before, a high availability DB2 cluster aims at maximizing database uptime. There are three parts of a high availability database solution:

1. The customer application
2. The DB2 client software
3. The DB2 database engine

Database applications are client/server based. The application depends on the integrity of the database software to produce uniform results. While this might seem fairly obvious, it is important to understand how this is accomplished when choosing and designing a solution.

Using the DB2 client software, a connection to a database must be established with the CONNECT statement before SQL transactions can be submitted. The DB2 client software relies on two directories to route database connection attempts:

1. The *node directory* lists the location of the server (IP address or hostname of the server) and the port the database server listens to for database connection attempts.
2. The *database directory* lists the database name, the database alias, and the corresponding entry in the node directory to use to establish the connection.

When an application issues the CONNECT statement, the database directory is searched to see if the database name or database alias exists. If the entry type is *indirect*, then the database is local and the connection is established using shared memory segments. If the entry type is *remote*, then the node name is used to point to the correct entry in the node directory.

The node directory information allows the client software to correctly route the connection attempt.

By understanding how clients establish database connections using the directory structure, you can plan for resource movement when planning the high availability solution.

Note: For a Content Manager production system, in order to achieve the overall high availability goal, all of the above aspects need to be considered.

The availability of the client software depends on what type of connection client applications will use to connect to the database. DB2 provides runtime clients for different native connections. For Java™ applications, DB2 V8 supports type 2, type 3 (deprecated), and type 4 JDBC drivers. From a DB2 standpoint the Content Manager client applications are:

- Content Manager eClient
- Content Manager Java Connector, and when used in the context of WebSphere the JDBC Connection Pool
- Resource manager Web application

All of the clients must use the type 2 JDBC driver; therefore the DB2 runtime client to choose is:
DB2.ADCL DB2 Application Development Client for SUNOS

Note: An important aspect for application high availability implementation is the ability to automatically reconnect after the failover of the database. The proper catalog of the high availability database, for example, logical hostname catalog in *node directory* and *database directory* pair, will provide the client software the ability to connect to the intended database.

With Content Manager applications automatically reconnecting is accomplished by the WebSphere Application Server, which makes it transparent to the end users. Because of the inherent loss of connection during database failover, the user application must be able to withstand a single point of failure. This means it has to be designed in a way to recover from a connection lost, for example by implementing a connect retry loop.

The Content Manager resource manager and the Content Manager V8 Java Connector both use JDBC Connection Pools with self monitoring facilities that are capable to recover from lost connections.

In Content Manager the following background processes are all designed as explained above and can recover from stale connections:

- Library server monitor
- Resource manager Migrator, Replicator, Purger and Stager

The high availability of this application is supported through the high availability implementation of WebSphere Application Server. See WebSphere high availability support and features on page 17 for additional information

The availability of the database engine itself can be supported in many different implementations. In this project, we implemented cluster software supported high availability failover.

3.5 High availability

The choice of a high availability solution is largely dependent upon the customer's business requirements. There are two types of high availability to choose from:

1. Continuous availability
2. Failover availability known as High Availability

3.5.1 Continuous availability

Continuous availability demands that the database engine be available for processing SQL transactions 100 percent of the time. This type of availability is typically only implemented in the most critical of applications. To achieve this goal, total redundancy is required. That means there must be two systems that are fully independent of one another, both in hardware and software.

Advantages:	**Disadvantages:**
100% uptime	Requires duplicate hardware Requires additional application coding

In this paper, we will not consider Content Manager in a continuous availability environment.

3.5.2 Failover availability

Failover availability is differentiated from continuous availability by the fact that for some period of time, however small, the database engine is not available for transaction processing.

The essential elements for this type of solution are:

1. Primary and secondary systems
2. Failure detection
3. Data source movement

The two systems have copies of the database data, and when a failure is detected, a failover takes place. In the failover process, the data source is moved from the primary to the secondary system.

There are two types of failover availability:

1. synchronous
2. asynchronous

Synchronous availability guarantees that the data sources on primary and secondary systems are identical, and complete continuity is maintained after a failover.

Asynchronous availability does not guarantee that primary and secondary system databases are completely in sync.

The method of moving database changes from the primary to the secondary system varies, but the process produces a window of time during which data has not migrated from one system to the other. The amount of data may be very small and the window very short, but it must be taken into consideration when you are deciding on a solution.

Let’s discuss the options that will provide either synchronous or asynchronous failover availability.

3.5.3 Specialized high availability software option

The synchronous method involves tight integration of the database software with specialized high availability software to produce a high availability cluster. High availability software support varies by operating system platform. The commonly available high availability solutions are:

platform	vendor	product
z/OS®	IBM	IBM Sysplex Distributor & z/OS Workload Manager
Solaris	Sun	Sun Cluster V3.x
Solaris	Veritas	Veritas Cluster Server
AIX®	IBM	High Availability Cluster Multi-Processing (HACMP)
AIX	IBM	IBM eServer™ Cluster 1600
AIX	Veritas	Veritas Cluster Server
Windows®	Microsoft	Microsoft Cluster Server
Windows	Veritas	Veritas Cluster Server
Linux	IBM	IBM Cluster 1350
Linux	Veritas	Veritas Cluster Server
HP-UX	HP	MC/ServiceGuard
Tru64	HP	TruCluster

Table 3-2

All these clustering solutions essentially work the same way. If there is a failure, the database server can move from one system to a backup system. To accomplish this task, the high availability software moves all the necessary resources to the secondary system. These resources include the disk resources of the physical database, the network resources, and the database server resources.

In the high availability cluster solution, a single copy of the physical database is stored on a shared storage system. In the DB2 environment only one system can own the storage array at a time. When a failure is detected, ownership of the storage is moved from the primary system to the secondary system. The network resources are moved as well. Finally, the database server resources are started on the secondary system and the database is made available. The detection of a failure is performed by a heartbeat connection between the servers. This heartbeat is a function of the high availability software and is aware of both hardware and software failures.

Since there is only a single copy of the database, it is always in sync. The time for the failover and restart of the database engine depends on several factors:

1. The time needed to detect a failure.
2. The length of time necessary to move database resource dependencies (storage array, networking resources, and so forth).
3. The time required for the DB2 engine to perform crash recovery.

The entire failover might take just a few seconds or it could take longer if a large workload needs to be processed from the log files. One advantage of this type of availability solution is that it does not require that any changes be made to the application or to the client configuration directories.

The high availability software provides a virtual IP address resource for database connections. The IP address will fail over when a failure is detected, and the same connect statement can be used by the application that it used before. When a failover takes place, all applications are disconnected, and the client returns a communication error condition to the application. Once the database server is running on the secondary system, the application can simply reissue the connect statement and continue to process work as before. This is known as a *hot standby* configuration. However, the secondary system does not have to remain idle while waiting for a failover.

The systems can also be configured in a *mutual takeover* configuration where both servers are actively hosting different databases. Each system is prepared to take over the workload of its partner in the event of a failure.

3.5.4 Specialized high availability software option tradeoffs

The advantages of this configuration are:

- Database is always in sync.
- No changes to application or client needed.
- No user interaction needed to detect and initialize failover.
- Performance is not hindered by any extra workload.

The disadvantages are:

- Extra software needed to create and configure solution
- Data is not duplicated providing less redundancy
- External storage required that must meet some high availability standards
- Distance between servers limited due to hardware requirements

3.5.5 Data replication option

DB2 UDB includes integrated replication abilities. The implementation of replication in DB2 UDB consists of two pieces:

1. Capture
2. Apply

The replication administrator designates replication sources from tables to be replicated, and then creates replication subscriptions on a target database, the secondary system, using the replication sources from the previous step as its source.

The Capture process monitors the transaction logs for all changes to the replication source tables, placing any changes made to these tables into staging tables.

The Apply program reads the staging tables and moves the changes to the subscription target on a timed interval.

Data replication is an asynchronous process. For a period of time either while the changed data has not yet been placed in the staging tables or while the Apply program has not replicated the changes to the target system, the two databases are out of sync.

In this document we will not consider DB2 Data Replication services, but will focus on operating system-assisted high availability cluster services provided by Sun Cluster Services V3.1.

3.6 WebSphere high availability support and features

Like IBM DB2 Universal Database, IBM WebSphere can be used to enable applications for high availability. IBM WebSphereV5 allows the setup of a clustered, multi-node Web application configuration with high availability capabilities as explained in detail: IBM WebSphere V5.0 Performance, Scalability, and High Availability: WebSphere Handbook Series . Unlike DB2, WebSphere cluster services are independent of the operating system and do not directly rely on platform-specific high availability cluster services.

In this study we will show how to setup a Content Manager Resource Manager Web application, on a multi node cluster using IBM WebSphere Application Server V5 with its horizontal and vertical cloning capabilities. For more details see the reference section in the appendix of this document and the product specific documentation available on the IBM web site.

The possible alternatives to the IBM WebSphere cluster services are the previously mentioned cluster solutions, which we will not discuss here.

3.6.1 Overview of the WebSphere Cluster Services

The following chapters will highlight the WebSphere V5 high availability services. We will briefly introduce the WebSphere Application Server V5 cluster capabilities and explain how they will be used by the Content Manager resource manager in order to be able for failover.

This introduction is not meant to replace the WebSphere Application Server V5 documentation, and we assume the reader has a working knowledge of the WebSphere Application Server V5 and the WebSphere Network Deployment V5 manager.

The following set of slides are from the presentation *HA Considerations in a WAS V5.0 Deployment*, by Tom Alcott (alcott@us.ibm.com) dated October 06, 2003: HA Considerations in WAS V5 .

We will use some of his foils to explain how the high availability services in WebSphere can be used to enable the Content Manager V8 middle-tier layers for high availability.

According the WebSphere Application Server V5 documentation, the WebSphere Application Server golden standard for high availability consists of the following layers:

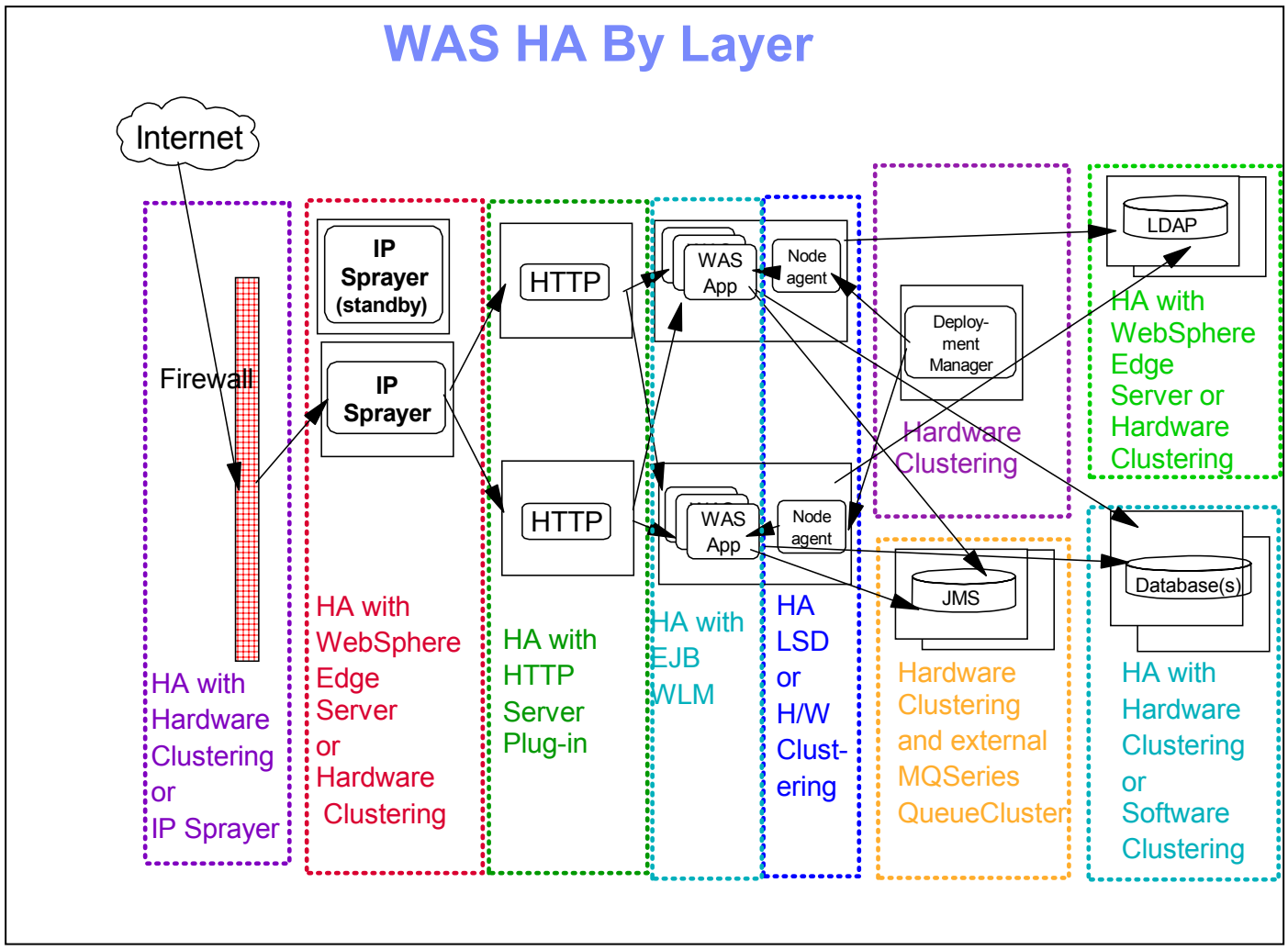


Figure 3-2. A typical WebSphere topology for a production system

Let us consider the topology shown in Figure 3-2, starting at the left.

The firewall is the first line of defense in an enterprise and usually the entry point into the DMZ. The firewall is usually constructed from hardware-assisted clustered equipment.

Within the DMZ is a pair of redundant IP sprayers. In the case of WebSphere, we would have a pair of redundant WebSphere Application Server NetDispatcher services. This WebSphere Application Server component sprays end user IP requests to the next layer, the HTTP servers. NetDispatcher can be tuned to use different schemes to distribute requests and to provide HTTP failover and load balancing services. In case the underlying HTTP servers dies, NetDispatcher automatically reroutes all incoming requests to the next available Web server.

In order to be self-redundant, a pair of NetDispatcher nodes must be configured. NetDispatcher has built in heartbeat and failover capabilities, such that it does not require any external support for this service.

The next layer, the Web server layer, is made redundant by replication. That is, a typical high availability scenario would use two HTTP servers, both serviced by the NetDispatcher node(s).

The next layer is the WebSphere Application layer. Within this layer you can use the enhanced WebSphere Application Server V5 cluster capabilities. With WebSphere Application Server V5, applications can now deploy their Web application into a cluster of application servers. The cluster can be configured to use application servers on multiple nodes, and on different platforms. The final layer is the database layer, which we have already discussed. The WebSphere golden standard would resemble Figure 3-3:

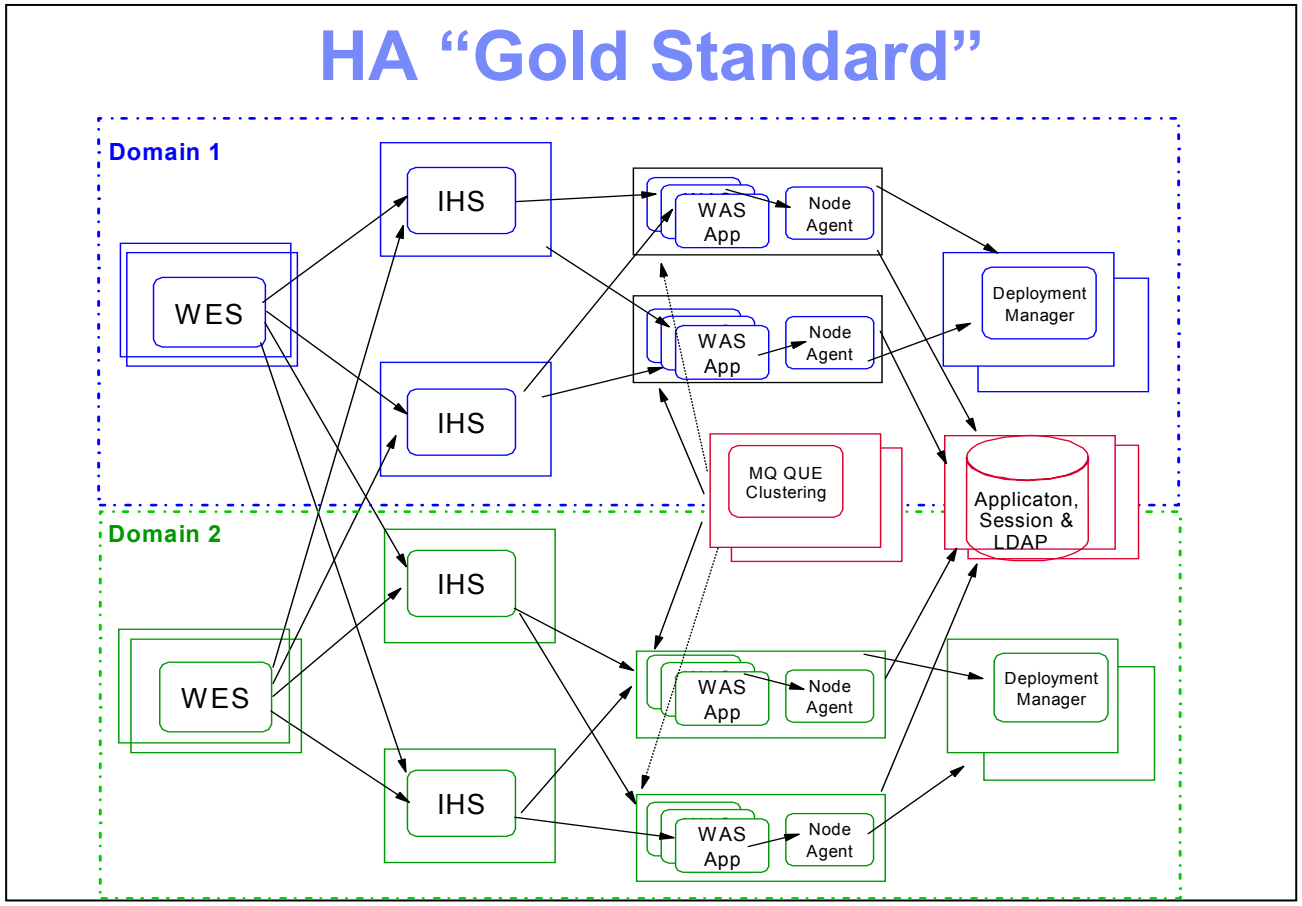


Figure 3-3. WebSphere Application Server golden standard topology

A simpler version of the above configuration would look more like the topology shown in Figure 3-4:

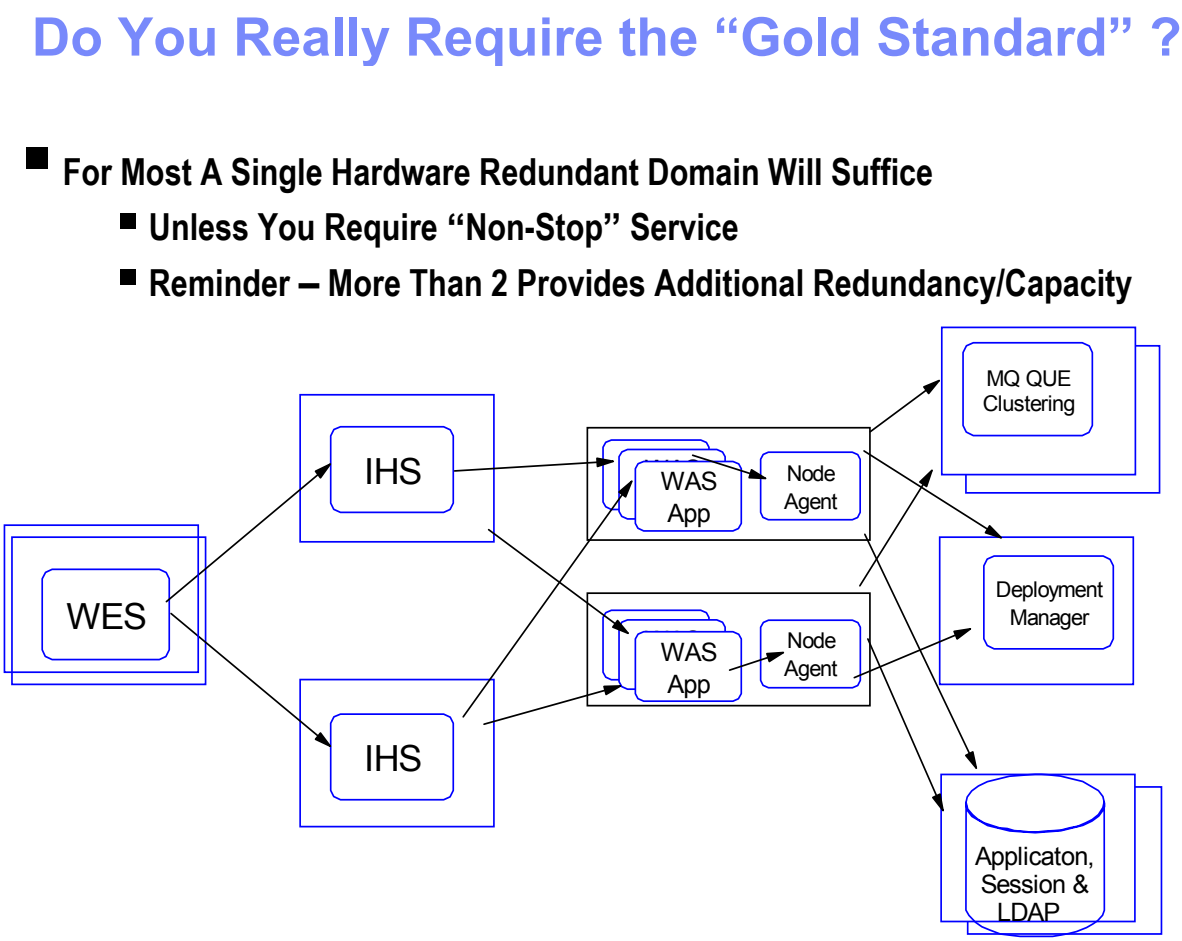


Figure 3-4. Simplified WebSphere Application Server topology

The determining factor is usually the business need for high availability during production and the costs associated with it.

3.6.2 Conclusions

Our Content Manager high availability test scenario will not adopt the WebSphere Application Server golden standard for cross-domain redundancy. Instead we will use the simplified WebSphere Application Server topology, where the pair of Web servers (IHS) and WebSphere Application Server (WebSphere Application Server App) are collocated on the same system, similar to the configuration illustrated in Figure 3-4.

4. Enabling IBM DB2 Content Manager for high availability

Now that we have defined what high availability means, and how DB2 and WebSphere Application Server V5 provide these services, let us examine how to design a Content Manager V8 high availability clustered environment.

The goal for this chapter is to discuss possible Content Manager V8 high availability configuration variants, stating from the more complex and leading to the minimal ones. Based on the[1] 80/20 rule, we then will pick the most promising Content Manager HA configuration, and explain how it can be installed, configured and tuned.

4.1 A complex Content Manager high availability configuration

Considering the software components required for a basic Content Manager system, you could sketch a possible Content Manager high availability configuration in the following way:

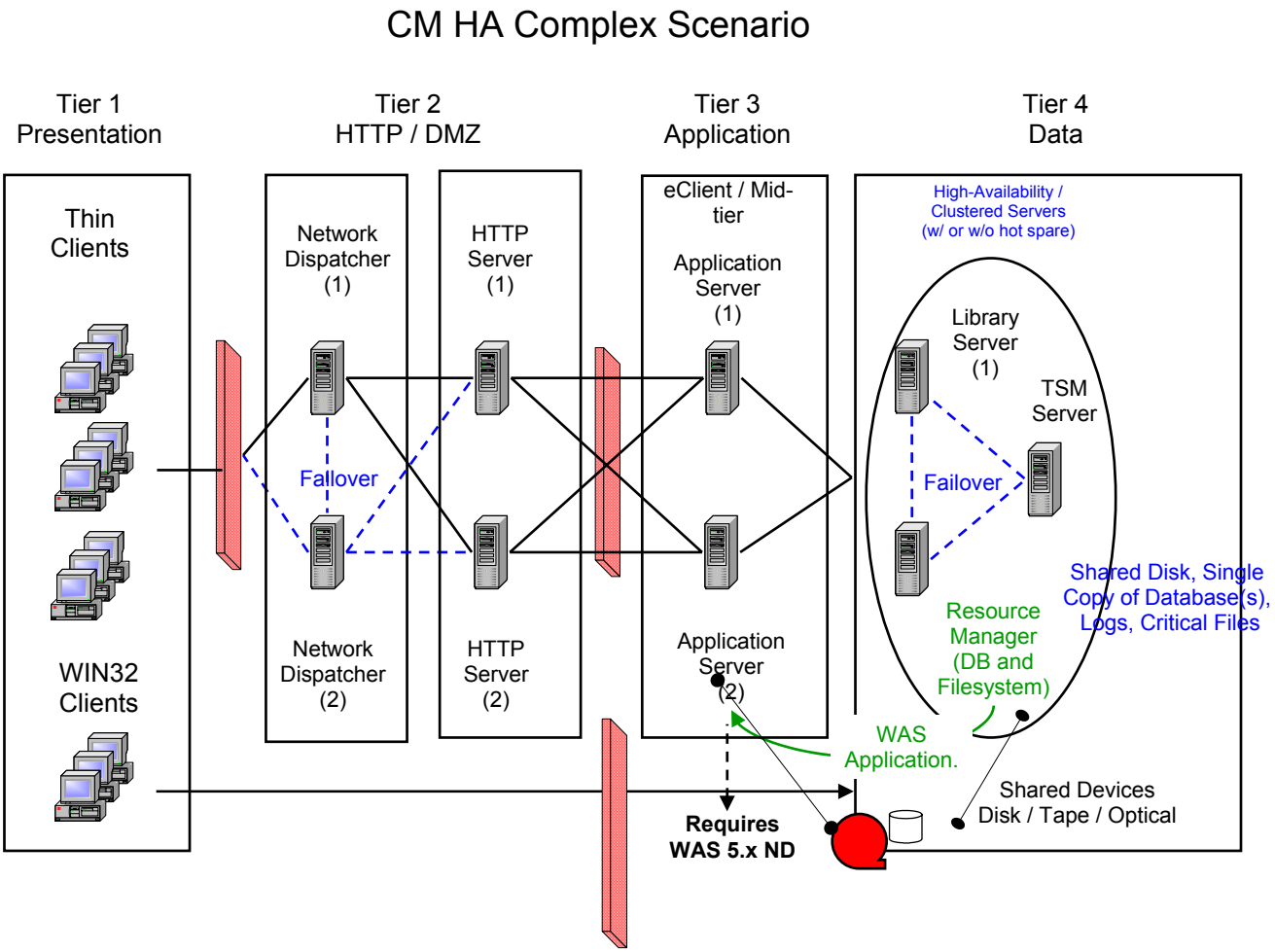


Figure 4-1. Content Manager V8 high availability preferred configuration

[1] The 80/20 rule says that in 80 % of the cases there is one HA configuration that satisfies the customer's needs, for the remainder 20% a special HA configuration must be designed.

Let us consider the configuration in Figure 4-1, starting at the left.

In the first tier an end user submits http requests across the Internet. These requests would pass through a firewall and reach into the DMZ of our Content Manager configuration.

Tier 2, the DMZ, is composed of 3 sub-layers:

1. NetDispatcher IP sprayers
2. HTTP servers
3. Web Application layer

Each layer uses its own pair of physical nodes, configured in a redundant way.

Next, the tier 3 layer hosts the cluster of application servers with its shared filesystem resources and the staging area.

Tier 4 represents the data layer, the back end of the Content Manager high availability production environment. This layer is assigned the task to provide high availability database services.

This topology requires a cluster of 8x physical nodes, set up and configured into a Content Manager high availability production system.

Furthermore, it requires the use of a mixture of system-provided cluster services (like HACMP, Sun Cluster, or Windows Cluster Software) and the WebSphere failover technology (WS HA) for a full functional high availability environment.

We will call this more complex scenario the *ideal Content Manager high availability configuration*. If cost is not a primary factor, then this configuration is preferred over the others.

For our tests, we will use a simplified form of the ideal Content Manager high availability configuration. We decided to co-locate the HTTP server and the WebSphere Application Server on the same physical system, reducing the deployed topology to six physical nodes for the Content Manager servers, not including the Content Manager System administration console. See

Content Manager high availability test system topology on page 28.

4.2 Content Manager V8 high availability alternative scenarios

Some of the intermediate logical layers can be collapsed onto the same pair of physical nodes. In general, the number of components running on a single system and the overall number of systems depends strongly on the specific high availability requirements and the budget available. In the extreme, it is possible to set up a Content Manager high availability cluster using just two physical nodes, but this is not recommended.

4.2.1 Content Manager HA simplified configuration

A simpler Content Manager V8 high availability variant, requiring fewer physical nodes, is shown in Figure 4-2.

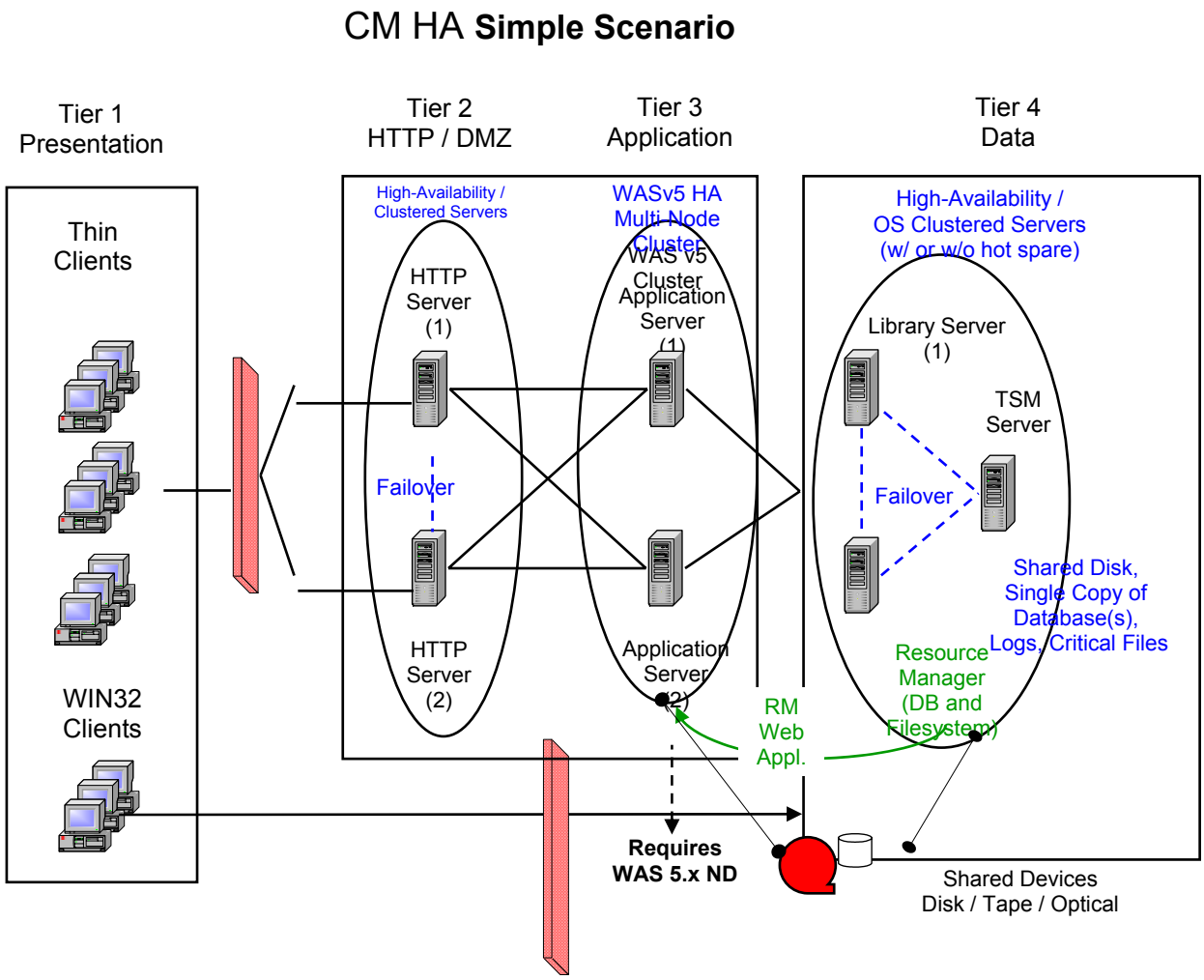


Figure 4-2. Content Manager high availability simplified configuration

In this configuration, the NetDispatcher IP sprayer layer is completely omitted. Instead, its function is provided by combining the capabilities of the WebSphere Application Server HTTP plug-in and the system-assisted cluster manager failover services.

What happens is the following:

1. The two HTTP servers are both enhanced by the WebSphere Application Server HTTP server plug-in. This plug-in provides request dispatch functionality which distributes IP requests from end users to the Web application servers across the WebSphere Application Server cluster on the two application server nodes.
2. Under normal operations, one HTTP server services requests targeted to the Content Manager resource manager application servers, and the WebSphere Application Server HTTP plug-in distributes these requests to the four resource manager application clones across nodes. In case of HTTP server process failure, the cluster manager would restart the HTTP service on the same node, and if the node itself becomes unavailable, then the HTTP server would be restarted on the surviving node by the cluster manager.

For this configuration, the HTTP services must be set up for high availability using the platform-specific cluster software.

Web application failover is achieved with the help of a WebSphere Application Server V5 cluster, which can be configured to host multiple cluster members (previously called clones) on the same or across nodes in the cell. By doing so, a cluster not only provides application failover but also provides the benefit of horizontal and vertical scalability.

Technically, at runtime every Web application clone (in our case the Content Manager resource manager Web application), will run in its own Web application server (and therefore its own JVM process). If one clone fails, all incoming requests are directed to the remaining clones. Application failover is again accomplished by virtue of the IP request dispatching capabilities of the HTTP server plug-in provided by WebSphere.

Heavy production conditions call for the use of different nodes for the Web servers and the application servers. And in an active-active configuration, the use of a redundant pair of IP sprayers must be considered. This way, both high availability and load balancing can be achieved at the middle-tier layer.

From a functional point of view, both configurations can be considered to be equivalent. Therefore, for the purpose of this document, the simplified configuration should be sufficient to demonstrate the Content Manager high availability capabilities.

4.2.2 Content Manager HA minimal configuration

In this last configuration, we further simplified the system topology by collapsing the HTTP server and the WebSphere Application Server layer onto one node.

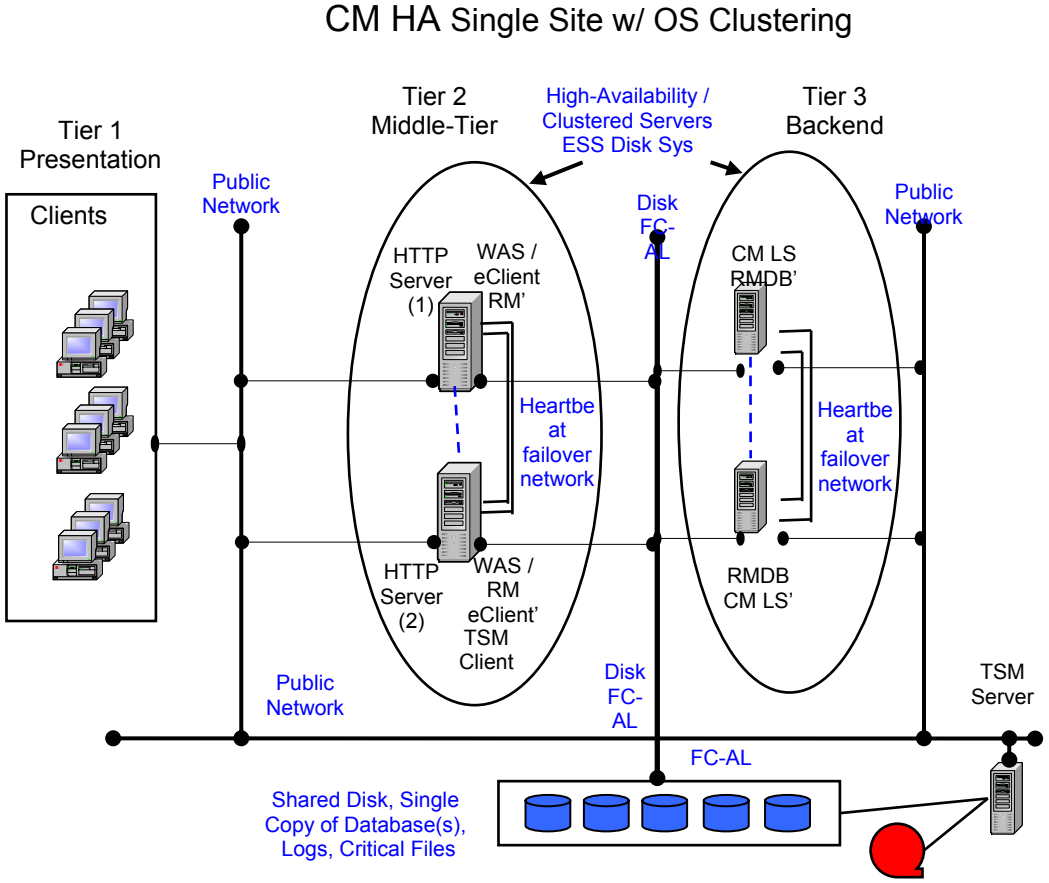


Figure 4-3. Content Manager high availability minimal configuration

In concept and at a logical level nothing has changed, and everything will work as described before. What has changed is the physical system layout, which now is reduced to four physical nodes. The weakness of this approach is the absence of firewalls and the benefits of the DMZ layer. Some customer might even choose to use the WebSphere built-in HTTP server instead of the external standalone HTTP server.

It is possible to further collapse the data and application layer to two physical nodes.

Note: This discussion does not include administrative aspects. For example, as we will see, you need to consider:

- Directory server
- DNS server
- Content Manager and WebSphere Application Server administration consoles

In our discussion, we only consider the cluster nodes necessary to provide the actual Content Manager high availability services.

4.2.3 Content Manager HA components at runtime

Figure 4-4 outlines the relevant Content Manager high availability software components needed at runtime for the chosen test configuration. In order to complete the picture we also included the Content Manager eClient Web application and the Information Integrator for Content connector layer. On the far left side is the storage area, which uses Fibre Channel links.

Note: In the diagram the solid black line means that both resource manager application nodes have concurrent access to the data at any given time. The database table and log storage area instead use a failover filesystem that is available only to the current active node. This is depicted by the solid and dashed lines.

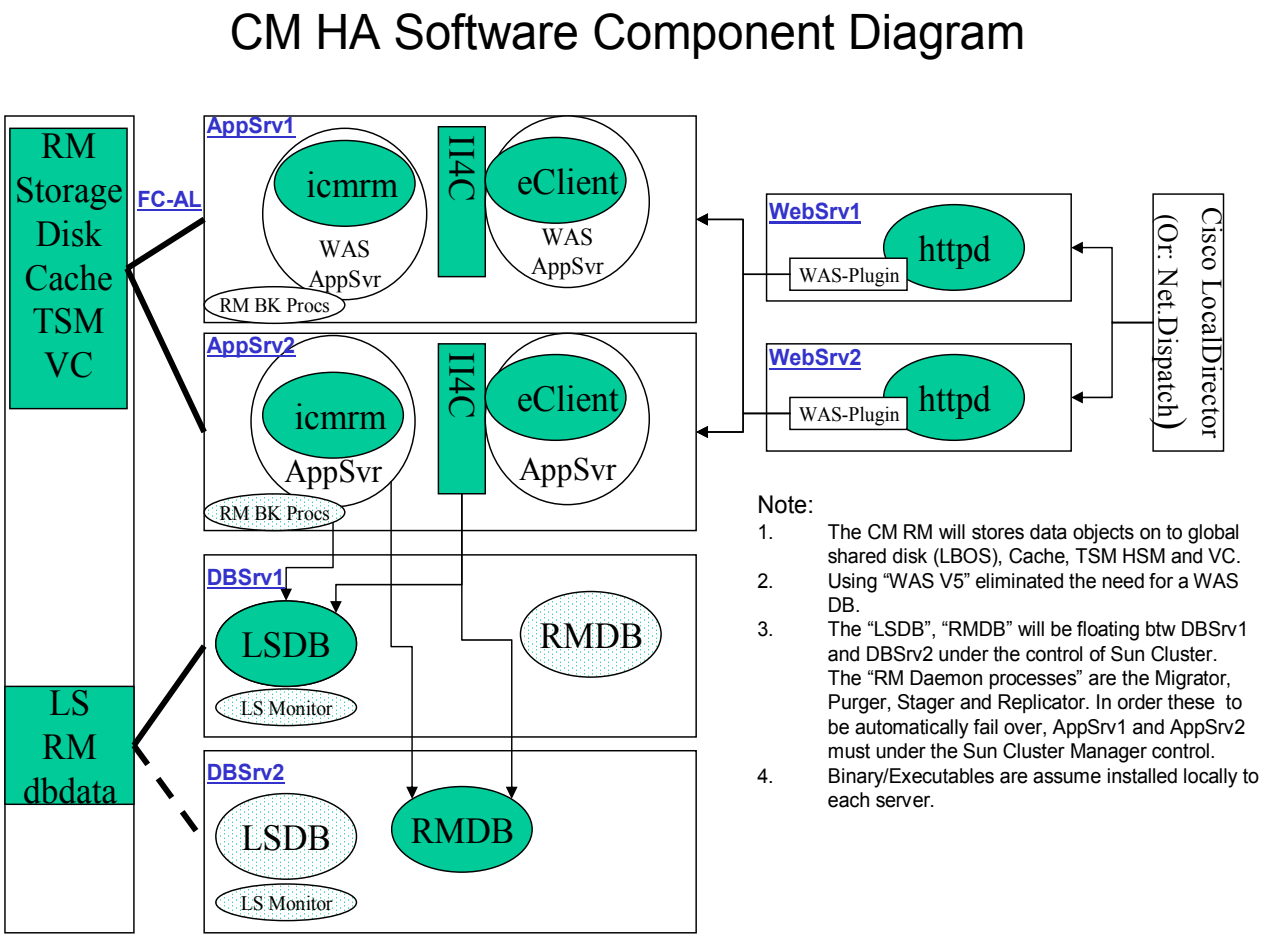


Figure 4-4. Content Manager Software components required at runtime

The IP sprayer appears to the right in Figure 4-4. This component can be a network appliance, like the Cisco Load Director, or a pair of workstations running the IBM NetDispatcher.

Recommendation: Under normal production conditions we do not recommend running all the Content Manager software components on the same node. Experience suggests that splitting the Web servers from the application servers, and application servers from the database layer, is a more practical design.

5. Planning for high availability in a CM environment

Before the actual setup can be initiated, some planning needs to be done. Most importantly, all the resources and configuration data needed during the installation must be defined and made available. By doing so, we will ensure a smooth and uninterrupted setup procedure.

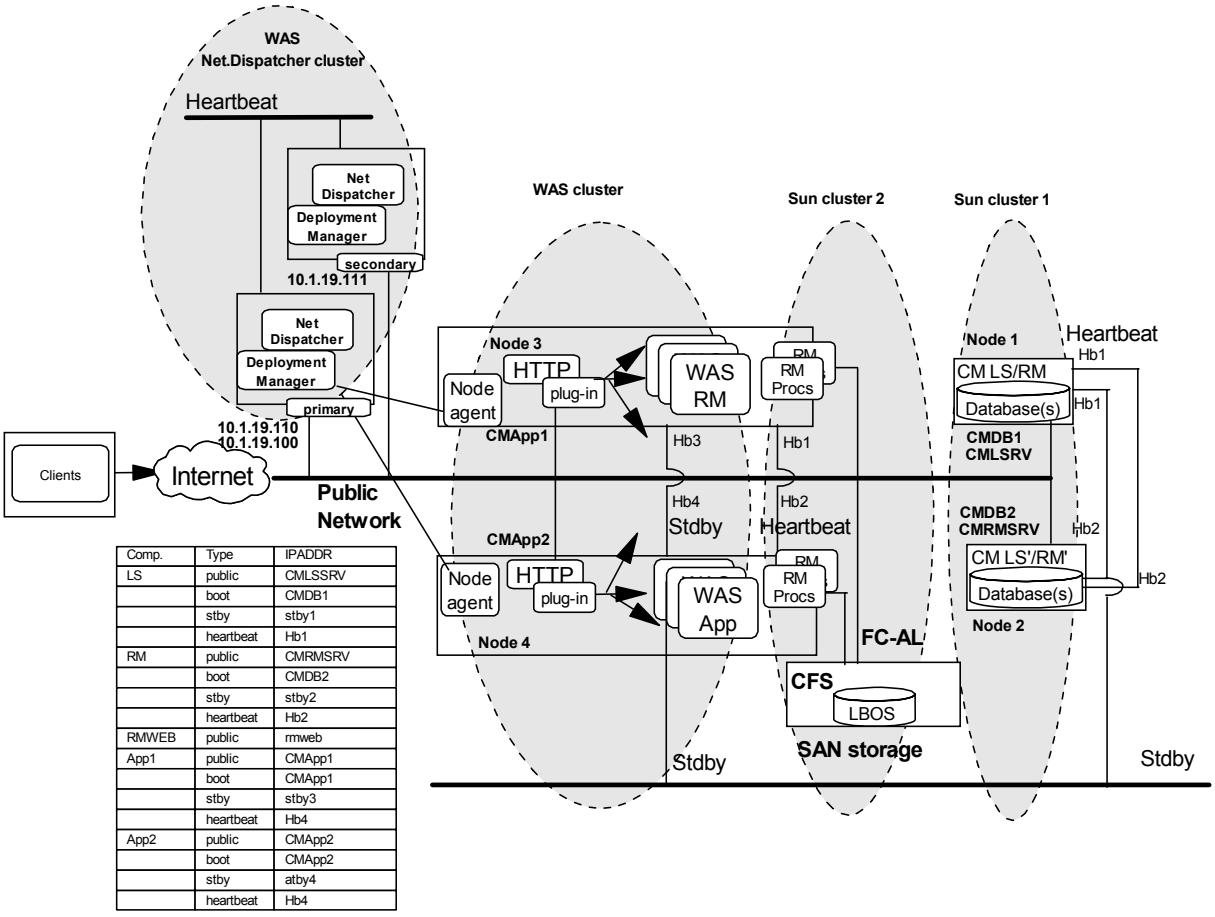
Let's start by summarizing what we want to do and create a laundry list of all resources needed.
In the previous section, various possible Content Manager high availability scenarios were discussed. We decided to use a simplified Content Manager high availability system topology for our tests. The platform on which to deploy this topology is a set of Sun multiprocessor systems running Solaris 9. The high availability environment will be set up using the following components:

- IBM DB2 Content Manager V8.2
- IBM DB2 Universal Database V8.1
- IBM WebSphere Application Server V5
- Sun Cluster Manager V3.1

All the physical data will be stored on fiber channel attached IBM ESS Model 800 storage subsystem. By physical data we mean, all database related data, the Content Manager resource manager object data, staging area as well as all files that need to be globally shared.

5.1 Content Manager high availability test system topology

Figure 5-1 illustrates the selected Content Manager high availability test environment topology.



Comp.	Type	IPADDR
LS	public	CMLSSRV
	boot	CMDB1
	stby	stby1
	heartbeat	Hb1
RM	public	CMRMSRV
	boot	CMDB2
	stby	stby2
	heartbeat	Hb2
RMWEB	public	rmweb
App1	public	CMApp1
	boot	CMApp1
	stby	stby3
	heartbeat	Hb4
App2	public	CMApp2
	boot	CMApp2
	stby	atby4
	heartbeat	Hb4

Figure 5-1 Content Manager high availability simplified topology

5.1.1 Content Manager high availability cluster minimal system

As an intermediate step, we will use this further simplified version of the system topology to test the basic high availability functions before we then introduce the last layer, the WebSphere NetDispatcher service.

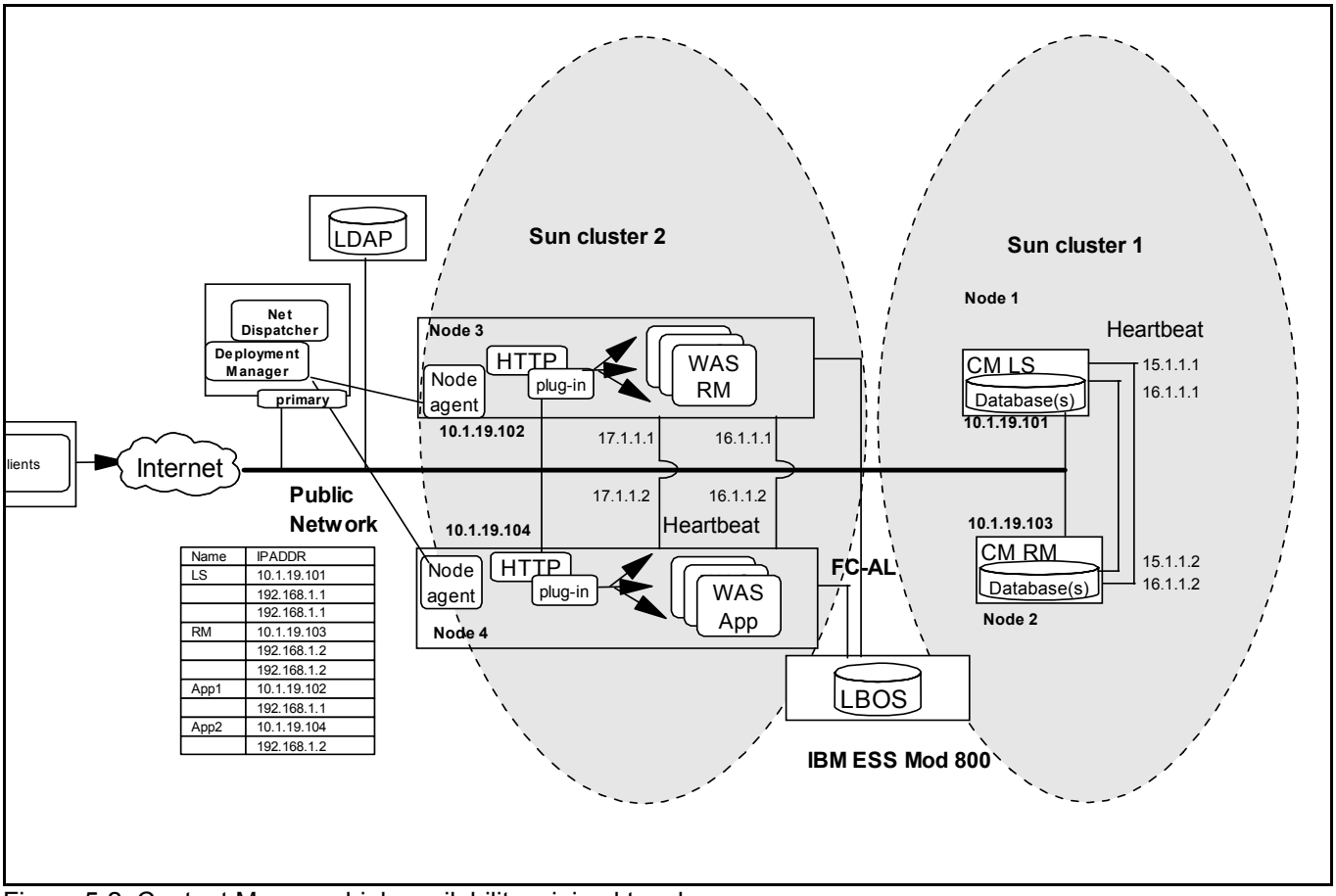


Figure 5-2. Content Manager high availability minimal topology

To conclude our planning, we need a cluster of workstations made out of three pairs of nodes:

- Two systems for the database layer
- Two systems for the application layer
- Two systems for the IP sprayer

Note: All the cluster scenarios presented need to be provisioned with directory and DNS services. Throughout this document, we will assume that both directory and DNS services are already available and don't need to be considered. Also, system administration consoles for both WebSphere Application Server and Content Manager are needed, although they can be on the same system.
The system with the administration consoles is in addition to the preceding list.

5.1.2 Content Manager high availability hardware components

Our hardware component list is as follows:

Hardware:	Quantity	Comment
Content Manager library server and resource manager database cluster: Nodes 1, 2: CLDB1 CLDB2	2	Two Sun Fire 4800 systems, each with: • 12 CPUs • 16 GB memory • 2 domains **Note:** CLDB1 and CLAPP1 are installed on domains on one SF4800; CLDB2 and CLAPP2 are installed on domains on the other SF4800. The SVL Content Manager high availability configuration utilized 4 Sun 15K Domains. Two domains were used for the database cluster and two for the resource manager WebSphere Application Server cluster. All domains were identical and were configured as follows: • 2 CPUs • 4 GB memory
Content Manager WebSphere Application Server cluster: Nodes 3,4: CLAPP1, CLAPP2	2	
Network load distributor: IBM WebSphere Application Server NetDispatcher Nodes: ND1, ND2	2	Sun Ultra 5s (333 MHz Iii)
Content Manager and WebSphere Application Server administration console	1	IBM xSeries Server Model 345
Storage subsystem:	2	Two Sun 3510 systems, each with: • 3 LUNs • 2 RAID controllers • 73 GB disk space (12 disks) **Note:** One Sun 3510 is used for the Content Manager library server and resource manager database cluster; the other is used for the Content Manager WebSphere Application Server cluster.

Table 5-1

Note: The systems have three network adapters: one for the public network and two for the private network. The private network is used for the heartbeat service, and is established with a point-to-point network using Cat5e Ethernet patch cables.

5.1.3 Content Manager high availability software components

All software products needed for this setup are listed below.

Software:	Comment
Operating system	Solaris 9
Domain and LADP services	Not used in this scenario
Cluster software	Sun Cluster Services V3.1 Update
Database	IBM DB2 Universal Database V8.1 with fix pack 4
Web application server	IBM WebSphere Application Server V5 with fix pack 2
Web application server cluster services	IBM WebSphere Application Server Network Deployment V5 FP2
IP sprayer	IBM WebSphere Edge Services Network Dispatcher V5.1
C compiler	Forte Developer 6 Update 2 C/C++
Content management system	IBM DB2 Content Manger V8.2 with fix pack 3

Table 5-2

5.2 Content Manager high availability planning data

In the following tables we collected all relevant configuration parameters and data needed during the subsequent installation steps. Most of the data is implicitly or explicitly requested by the respective product installation scripts.

Note: In this paper we will assume that the size of the machines is chosen based on the projected load their role in the HA design. For example, the library server and resource manager data base nodes should be sized to be capable to sustain the expected transaction load of both CM components in case of failure. Similarly, the resource manager web application nodes must be sized to be big enough to sustain the http request load and to host both WebSphere clones, which means two Java Virtual Machines and the HTT P server.

5.2.1 Required administration and database instance user IDs

Let's start with the user IDs required for the selected Content Manager high availability configuration. An "X" in the node column indicates that the user ID must be configured on that node.

Users	ID	Nodes				
		LS	RM	App1	App2	Comment
db2inst1	101	X	X			CM LS instance
db2fenc1	102	X	X			CM LS fenced instance
db2inst2	103	X	X			CM RM instance
db2fenc2	104	X	X			CM RM fenced instance
db2clnt	105			X	X	db2 instance on App Server
icmadmin	106	X	X			CM LS admin useri d
rmadmin	107	X	X	X	X	CM RM admin user id
icmconct	108	X	X	X	X	CM connect proxy user id

Table 5-3

5.2.2 Cluster resources, names and network addresses

Next, we need to choose and define all network addresses, hosts, domains and virtual resource names.

Sample `/etc/hosts` file entries:

```
# Physical Host IP Address
10.1.19.103     clapp1
10.1.19.102     cldb1
10.1.19.104     cldb2
10.1.19.105     clapp2
# VIP
10.1.19.201          lsdbsrv    # VIP: HA-IP for Library Server DB Server
10.1.19.202          rmdbsrv    # VIP: HA-IP for Resource Manager DB Server
10.1.19.100          rmweb      # VIP: HA-IP for Resource Manager Web Interface
```

Table: Physical nodes vs. functional areas

Physical Nodes	VD2	ND1	ClApp1	ClApp2	Cldb1	Cldb2	Comments
Node1: cldb1					X	X	Library server database active on cldb1 or cldb2
Node2: cldb2					X	X	Resource manager database active on cldb2 or cldb1
Node3: clapp1							
Node4: clapp2				X			
Node5: ND1		X					
Node6: ND2		X					

Table 5-4

Table: Physical nodes vs. VIP: logical host

VIP: Logical Host	IP	ND1	ND2	Clapp1	Clapp2	Cldb1	Cldb2	Comments
VIP1: lsdbsrv	10.1.19.201					X	X	Library server virtual address. Floating between library server and res manager servers controlled by SC3.1
VIP2: rmdbsrv	10.1.19.202					X	X	Resource manager virtual address. Floating between library server and res manager servers controlled by SC3.1
VIP3: rmweb	10.1.19.100	X	X	X	X			Resource manager cluster alias for access. Floating btw ND1/ND2 controlled by NetDispatcher built-in heartbeat

Table 5-5

Table: Network interfaces and IP addresses

VIP Resources nodes	IP Address	CIdb1	CIdb2	CMApp1	CMApp2	Comments
VIP1: lsdbsrv Library server database server	VIP1=lsdbsrv 10.1.19.201	X	X			Library server virtual address. Floating btw li server and resource manager servers contro by SC3.1
VIP2: rmdbsrv Resource manager database server	VIP2=rmdbsrv 10.1.19.202	X	X			Resource manager virtual address. Floating btw l server and resource manager servers controlled b SC3.1
VIP3: rmweb NetDispatcher high availability	VIP3=RMWEB 10.1.19.100			X*	X*	Resource manager cluster alias VIP=10.1.19 Floating btw ND1/ND2 controlled by NetDispatcher built-in heartbeat
Node1: CIdb1	10.1.19.102	X				Library server primary node
qfe1	172.16.1.1					Heartbeat
qfe0	172.16.0.129					Heartbeat
clprivnet0	172.16.193.1					Heartbeat (logical private IP floating btw qfe0/qfe
Node2: CIdb2	10.1.19.104		X			Resource manager database primary node
qfe1	172.16.1.2					Heartbeat
qfe0	172.16.0.130					Heartbeat
clprivnet0	172.16.193.2					Heartbeat (logical private IP floating btw qfe0/qfe
Node3: CIapp1	10.1.19.103			X		Resource manager Web application primary node1
qfe1	172.16.1.1					Heartbeat
qfe0	172.16.0.129					Heartbeat
clprivnet0	172.16.193.1					Heartbeat (logical private IP floating btw qfe0/qfe
Node4: CIApp2	10.1.19.105				X	Resource manager Web application node2
qfe1	172.16.1.2					Heartbeat
qfe0	172.16.0.130					Heartbeat
clprivnet0	172.16.193.2					Heartbeat (logical private IP floating btw qfe0/qfe
NetDispatcher high availability	VIP=RMWEB 10.1.19.100					Floating btw ND1/ND2 controlled by NetDispatcher build-in heartbeat
Node5: ND1	10.1.19.110					WebSphere Application Server NetDispatcher no
HB5-6	10.1.19.110					Heartbeat
Node6: ND2	10.1.19.111					WebSphere Application Server NetDispatcher no
HB6-5	10.1.19.111					Heartbeat

Table 5-6

5.2.3 Disk and filesystem resources

The next table lists all relevant file and filesystem resources grouped into a local, global and failover category. Notice that we decided to globally share the following resources:

1. The Content Manager library server stored procedure code (under the instance home directory; already on global filesystem):
```
db2v8w32@ranger2:/export/home/db2v8w32/sqllib/function$ ls -l | more
total 14400
-r-xr-xr-x   1 db2v8w32 other    4426380 Apr 11  2003 ICMNLSSP
-r-xr-xr-x   1 db2v8w32 other      60312 Apr 11  2003 ICMNLSUF
-r-xr-xr-x   1 db2v8w32 other    2835536 Apr 11  2003 ICMNWFSP
```
2. The Content Manager database load modules, created at runtime located on ~db2fenc1/<LSDBNAME>/dll
3. The Content Manager resource manager ICMRMKeyStore file containing the secret key used to decrypt the Content Manager resource manager and library server DB2 user password.
4. The `/opt/IBMicm` directory structure on the RM web application nodes is needed in order to enable the resource manager background processes for high availability.

type of filesystem local	Filesystem location	Nodes				
		LS	RM	App1	App2	Comment
C++ Compiler	/opt/	X				
Database Server	/opt/IBM/db2/V8.1	X	X			DB2 server binaries
	/opt/IBMicm	X	X			
DB2 CAE	/opt/IBM/db2/V8.1			X	X	DB2 client runtime DB2 Appl. Dev. Client
Cluster Filesystems		**LS**	**RM**	**App1**	**App2**	
	/var/db2	X	X			DB runtime
	/var/lum	X	X			License Manager
	~db2inst1	X	X			Library server database instance ID
	~db2fenc1	X	X			Library server database fenced instance ID
	~db2fenc1/<LSDBNAME>/dll	X	X			Library server load database modules
	~db2inst2	X	X			Resource manager instance ID
	~db2fenc2					Resource manager fenced instance
	/global/rmdata			X	X	Resource manager LBOS storage area
	/global/rmstage			X	X	Resource manager LBOS staging area
Failover Filesystems		**LS**	**RM**	**App1**	**App2**	
	/db/lsdb	X	**X**			Library server database table spaces
	/db/lslog	X	**X**			Library server database logs
	/db/rmdb	X	**X**			Resource manager database table spaces
	/db/rmlog	X	**X**			Resource manager database logs

Table 5-7

Note: For the convenience of installation, the storage device for the library server (`/db/lsdb` and `/db/lslog`) and resource manager (`/db/rmdb` and `/db/rmlog`) database can also be globally mounted at this time. For improved I/O performance in production environment, these file systems or devices can be configuration with HAStoragePlus as failover file system. See *DB2 Universal Database and High Availability on Sun Cluster 3.X* (http://www-306.ibm.com/software/data/pubs/papers/#suncluster).

5.2.4 Content manager library server and resource manager databases and cluster aliases

The Content Manager library server database and the Content Manager resource manager databases are given a cluster-managed virtual IP address (VIP). In case of a node or network failure, the MSCS cluster managers will re-assign the respective VIP to the remaining node, ensuring service availability.

Component	Component Alias	LS	RM	App1	App2	Comment
Library server database	CMLS	X	X			Virtual IP address (VIP)
Resource manager database	CMRM	X	X			Virtual IP address (VIP)
RM WAS Cluster						
RMWEB	CM RM Cluster Alias			X	X	VIP=10.1.19.100

Table 5-8

At this point, all relevant information has been collected and we are ready to begin the installation and setup.

5.3 Summary of the installation steps

What needs to be done:	Step	Comment
First step on library server node:		
Install and set up the cluster hardware	1	Make sure that the cluster hardware is functioning properly. External disks: database and LOBS as well as the network.
Set up NIS and DNS	2	Configure and define IP, NIS and DNS resources.
Configure cluster environment	3	Sun cluster nodes, disks, storage, network
Cluster 1: Database cluster 2 nodes	4	For library server and resource manager resource group (Node1, Node2)
Cluster 2: Web application cluster 2 nodes	5	For resource manager Web application cluster (Node1, Node2)
Set up IP sprayers	6	For WebSphere Application Server NetDispatcher (Node5, Node6). Build-in failover ability.
Test installed software components:	7	Disk subsystem, Sun Cluster V3.1, and Network

Table 5-9

Now that planning is done, the designed Content Manager high availability configuration can be implemented. The Content Manager resource manager in our implementation is split between the database node and multiple WebSphere Application Server V5 application server nodes. The database server will host the Content Manager resource manager database schema. The WebSphere Application Server nodes will be

configured to host a cluster of Content Manager resource manager clones across all available nodes. This configuration not only achieves Content Manager resource manager intra-node and inter-node failover, but horizontal and vertical scalability as well.

The drawback of this configuration choice is the default Content Manager Installation process requires modification to accommodate this implementation.

5.4 Installing the software components

The following table summarizes the installation steps for Content Manager server software components:

What needs to be done	LS	RMDB	App1	App2	Comment
Set up user and group IDs.					Cluster wide user and group IDs for Content Manger
Install Forte C++ v6.	X	X			Not required with Content Manager V8.3
Install DB2 UDB.	X	X			Use DB2 V8.1 and FP4.
Create two database instances.	X	-			Use the db2icrt command with `db2inst1` / `db2fenc1`, and `db2inst2`/`db2fenc2` IDs.
Test the DB2 installation.	X	-			Use db2sampl to do so
Install Java JDK 1.3.1.x	X	X	X	X	If it is not already installed, install it.
Install and set up the Content Manager library server.	X				First prepare the library server instance environment, then install. Update the log path to include `db/lslog` and `db/rmlog`.
Validate the library server with the system administration client application.	X				Library server only at this point.
Install and set up the Content Manager resource manager.	X	X			Create the resource manager database and initial data on global filesystem.
Enable the library server and resource manager databases for high availability.	X	X			Follow the steps described in Chapter 5.12 on how to enable the library server and resource manager databases for HA.
Enable the library server monitor process for high availability.	X	X			Use `/etc/inittab` to respawn `cmlsproc` on both potential LSDBSRV hosts
Install WebSphere Application Server V5.			X	X	WebSphere V5 with fix pack2 or V5.1
Deploy the resource manager application.			X		On the primary node only
Install WebSphere Application Server Network Deployment.			X		WebSphere Application Server Network Deployment. V5 with fix pack 2 or WebSphere Application Server V5.1
Deploy `icmrm` in WebSphere Application Server cluster environment.			X		Use WebSphere Application Server Network Deployment Manager to deploy or clone the `icmrm` application to the WebSphere Application Server application cluster.
Enable the resource manager background processes for high availability.			X	X	Use Sun Cluster agent builder to create and configure the high availability agent to support the failover of the resource manager Migrator, Purger, Replicator, and Stager background processes.
Configure NetDispatcher.			X	X	Configure NetDispatcher to support IP spray between the redundant HTTP servers.

Table 5-10

For the Content Manager library server and resource manager server components, we decided to use two independent database instances, which will run under the db2inst1 and db2inst2 instance IDs respectively. In addition, we will have to create two more user IDs that the DB2 engine can use to fence out stored procedure code that must run in a different process. We plan to use db2fenc1, and db2fenc2 for this purpose.

Background information: The Content Manager library server is written as a set DB2 C stored -procedures. At runtime this code is fenced out by the DB2 engine and it executes in its own child process. For every Content Manager client connection that is set up with the Content Manager library server, separate fenced child process will be created.

Next the first cluster needs to be set up. This cluster consist of a pair of two nodes, hosting the aforementioned databases. The cluster will be configured as a having a pair of active – active nodes, designed to run one of the database instances each. In case of a node failure, the surviving node will take over the entire load.

The second and third column in the above table represent the library server and resource manager database nodes on which the respective installation steps must be executed or repeated. For the Content Manager high availability environment setup, the single node installation must be mirrored.

5.5 Sun cluster configuration for setting up Content Manager V8

5.5.1 Introduction

In this section we will explain all the necessary installation and configuration steps to install and configure Content Manager V8.2 for high availability on a cluster of Sun servers using Sun Cluster Services.

The Content Manager high availability environments presented in this white paper represent the findings of two separate implementation efforts. The initial Content Management high availability environment was implemented in Sun Microsystems' Menlo Park Customer Benchmark facility. Initial configuration and proof-of-concept work was undertaken at the Menlo Park facility using Sun mid-range servers and storage. The second phase was implemented at IBM Silicon Valley Labs and utilized Sun Cluster nodes implemented on Sun 15K domains and IBM Shark storage.

Note: As summarized in Summary of the installation steps, we will build two 2-node Sun Clusters (database cluster and Web application cluster) for the Content Manager high availability configuration. In this section, we will document the necessary steps to set up the library server and resource manager database cluster. For the Web application cluster, the device names and the mount points will be different in the sections of Configuring the Disk I/O Subsystem and Configuring the global devices. See Installing the CM V8 RM Web Application for specific installation steps for the Web application cluster.

5.5.2 Install the Sun Cluster V3.1 software

The steps outlined below are required to successfully install Sun Cluster:

Configure operating system boot disk partitions according to Sun Cluster requirements.

1. Install Solaris operating system.
2. Install Solaris operating system.
3. Install operating system and hardware patches.
4. Install Sun Cluster 3.1 software.
5. Install Sun Cluster 3.1 patches.
6. Configure Sun Cluster 3.1.

Detailed instructions for these steps are provided in the *Sun Cluster 3.1 Software Installation Guide* at http://docs.sun.com.

It is important to configure the operating system boot disk partitions to meet Sun Cluster requirements. Sun recommends a minimum 750 MB swap partition for Sun Cluster and the Solaris operating system. A partition labeled `/globaldevices` is also required to exist on the boot disk partition layout. The `/globaldevices` partition is used by the scinstall utility during cluster install and is later assigned a new mount point and serves as a cluster file system mount for managing cluster nodes. Sun recommends the `/globaldevices` partition be at least 512 MB.

The operating system can be installed after the boot disk is successfully partitioned to accommodate the Sun Cluster software. The Content Manager high availability environments discussed in this white paper were implemented with both Solaris 8 and Solaris 9 operating environments. Patch requirements will vary among operating system releases. An acceptable methodology would be to install the latest recommended patch cluster available for your operating environment and the latest patches and drivers for your installed hardware.

Prior to installing the Sun Cluster it is important to understand how the cluster heartbeat networks will be configured. Configure the device names of the interface adapters for heartbeat service and determine if they

will be used with a switch or point-to-point cabling. The Content Manager high availability environments implemented for this study utilized point-to-point cabling for the cluster heartbeats.

With this information available the Sun Cluster software can be installed. The Sun Cluster software for the Content Manager high availbility environment was installed using the scinstall utility. The Sun Cluster software can be installed from the CD-ROM. The scinstall utility is located in:

```
/cdrom/suncluster_3_1/SunCluster_3.1/Sol_version/Tools
```

The Sun Cluster software can also be preinstalled. If the software is preinstalled the scinstall utility will be in the `/usr/cluster/bin` directory.

The scinstall utility is menu-driven and allows for initial configuration of the cluster. Below are the menu options required to initially configure the cluster. The initial configuration allows you to specify the cluster name, the nodes participating, heartbeat interfaces, and configuration.

```
*** Main Menu ***
Please select from one of the following (*) options:
* 1) Install a cluster or cluster node
2) Configure a cluster to be JumpStarted from this install server
3) Add support for new data services to this cluster node
* 4) Print release information for this cluster node
* ?) Help with menu options
* q) Quit
Option: 1
```

Select 1 to configure the cluster.

```
*** Install Menu ***
Please select from any one of the following options:
1) Install all nodes of a new cluster
2) Install just this machine as the first node of a new cluster
3) Add this machine as a node in an existing cluster
?) Help with menu options
q) Return to the Main Menu
Option: 1
...
*** Installing all Nodes of a New Cluster ***
...
Do you want to continue (yes/no) [yes]? Y
```

Selecting 1 again here allows all nodes to configured into the cluster during initial installation.

The following sub-menus allow cluster name and participating nodes to be specified.

```
>>> Cluster Name <<<
...
What is the name of the cluster you want to establish? <Clustername>

>>> Cluster Nodes <<<
...
Node name: node2
Node name (Ctrl-D to finish): Control-D
This is the complete list of nodes:
...
Is it correct (yes/no) [yes]?

>>> Authenticating Requests to Add Nodes <<<
...
Do you need to use DES authentication (yes/no) [no]?
```

The option to enable DES encryption is also available here. The Content Manager high availability configurations in this white paper set this option to *no*.

The menus below allow the network heartbeat and interfaces to be configured.

```
>>> Network Address for the Cluster Transport <<<
...
Is it okay to accept the default network address (yes/no) [yes]?
Is it okay to accept the default netmask (yes/no) [yes]?
>>> Point-to-Point Cables <<<
...
Does this two-node cluster use transport junctions (yes/no) [no]?

>>> Cluster Transport Adapters and Cables <<<
Select the first cluster transport adapter for "node":
1) adapter
2) adapter
...
N) Other
Option: N
```

Run the preceding command for each adapter.

```
Is it okay to use autodiscovery for the other nodes (yes/no) [yes]?
```

A *yes* response to the above command will allow the interfaces on the other nodes in the cluster to be configured at this time as well.

```
>>> Software Patch Installation <<<
...
Do you want scinstall to install patches for you (yes/no) [yes]? N
```

The scinstall utility will also install patches if specified.

```
>>> Global Devices File System <<<
...
The default is to use /globaldevices.
...
Is it okay to use this default (yes/no) [yes]?
```

The `/globaldevices` partition on the boot disk needs to be available at this point

```
Is it okay to begin the installation (yes/no) [yes]? y

Interrupt the installation for sccheck errors (yes/no) [no]?
```

The scinstall now executes the installation and configuration on all nodes. Install logs will be placed in `/var/cluster/logs/install`. At this point the initial cluster software installation is complete. The cluster will reboot after the initial configuration. When the cluster comes back up after reboot the quorum devices can then be configured using the scsetup utility.

5.6 Configuring the disk I/O subsystem

5.6.1 Proof-of-concept setup at both the Sun Microsystems and IBM labs

Sun Microsystems and IBM engaged in a joint proof-of-concept at both vendors' labs using Sun Cluster Version 3.1 with IBM DB2 Content Manager.

Different server class systems and different storage devices were used at each site.

5.6.2 Sun Microsystems' Menlo Park setup

At Sun's Menlo Park campus, four domains were used from a Sun Sunfire 4800 in conjunction with 3 LUNs each from two Sun 3510 storage devices. This configuration was not configured using any multi-path solution.

5.6.3 IBM's Silicon Valley Lab configuration

IBM's configuration used four domains on a Sun F15K with an IBM ESS storage server, and Sun's multi-path STMS software, commonly referred to as MPxIO.

5.6.4 Shared storage requirements

In order to implement a failover cluster, access to shared storage is required. Not very long ago, most high availability systems implemented shared storage through multi-hosted SCSI adapters communicating with an external RAID array. However, with advances in technology, most modern systems are implemented using a SAN (storage area network) topology. In other words, two or more hosts communicate across Fibre Channel (FC) host bus adapters (HBAs) to external Fibre Channel storage RAID devices using a Fibre Channel fabric switch.

End users who implement high availability systems almost always design their systems to localize failures. The failover of an application from one node to another is a very expensive operation in terms of time and expense. Therefore, most high availability systems are built on top of layers of redundancy.

The five most common types of layers of redundancy from a storage perspective encompass the following classes:

- External storage - RAID arrays versus independent disks (JBOD, or "just a bunch of disks")
- Redundant SAN networks
- Multiple storage HBAs on each server
- Multi-path software for failover, load-balancing, and concurrent microcode upgrade
- Volume manager for virtualization of physical LUNs and software raid capability

The choice of a RAID-enabled instead of using independent disks means that in the event that one disk is lost, the system continues to function. Next, RAID systems are designed with dual mirrored cache so that write updates are protected from being lost.

SAN designers build networks with two or more fabric switches and place two or more HBAs into the server. Designing SAN network infrastructures is much akin to building multi-homed networks.

Finally, most designers will integrate multipathing software to make use of the multiple adapters in the server. In the event that one path is disabled (if someone accidentally pulls out a cable, for example), the remaining paths to the storage device are able to handle the transactions. Some multipathing software environments only implement failover scenarios where one path is used only until a failure is encountered and then one of the remaining paths is brought into service. Other multipathing software packages make better use of the number of host bus adapters in the host and permit load balancing across all paths. Various vendors implement different multipathing packages such as Sun Microsystems' Sun StorEdge Traffic Manager Software (commonly referred to as MPxIO), IBM Subsystem Device Driver (commonly referred to as SDD) , and Veritas' Volume Manager software with dynamic multi-path capabilities (commonly referred to as VxVM/dmp or VxDMP).

Sun StorEdge Traffic Manager software (STMS) is a multipathing, load-balancing, and failover application. STMS is also commonly referred to as MPxIO, and it is bundled with the SAN Foundation Suite (SFS).

Although IBM's Subsystem Device Driver (SDD) does support Sun Solaris, this SDD multipathing software is not supported in a Sun Cluster V3.x environment. Instead, IBM recommends the choice of the Sun MPXIO host type when configuring the Shark with the ESS Specialist and use of Sun's STMS software.

Veritas' DMP might be supported with V3.1.1 or higher in a Sun Cluster environment; see the documentation.

A volume manager also provides a level of virtualization and redundancy. Sun offers Solaris Volume Manager (Solaris 9) and Solstice Disk Suite as part of the Solaris operating system. Veritas offers Veritas Volume Manager for Solaris. Volume managers can add an additional layer of protection by implementing various RAID levels at the software level. Volume managers are most commonly used to provide RAID 1 level mirroring, but can also be used to create RAID 5 levels volumes over multiple storage arrays.

Both Sun and IBM provide storage subsystems.

Given that setup of shared storage is fundamental to installing any cluster environment, we will proceed with providing explicit instructions to set up IBM's ESS in a Sun Cluster V3.x system.

IBM manufactures RAID storage devices. IBM's high-end storage (TotalStorage® Enterprise Storage Server) supports the Sun Solaris operating system and has been certified by Sun Microsystems to support Sun Cluster V3.x. Furthermore, IBM's ESS, also known as the Shark, fully implements Sun's Open MPxIO specification for multipathing.

Refer to Sun's documentation and Web site for more information about Sun StorEdge Traffic Manager Software (STMS)**:**
http://www.sun.com/products-n-solutions/hardware/docs/pdf/817-3671-11.pdf .

More information regarding Sun's SAN Solutions V4.4 is available at:
http://www.sun.com/products-n-solutions/hardware/docs/Network_Storage_Solutions/SAN
http://www.sun.com/storage/san/open_san/
http://www.sun.com/storage/software/storage_mgmt/traffic_manager/index.xml
http://www.sun.com/products-n-solutions/hardware/docs/pdf/817-3674-10.pdf
http://www.sun.com/products-n-solutions/hardware/docs/pdf/816-1420-11.pdf

http://www.sun.com/products-n-solutions/hardware/docs/pdf/817-0385-11.pdf

We will discuss how best to setup IBM's ESS (the Shark) in a Sun Cluster environment.

Because readers of this document will have varying levels of familiarity with Sun's STMS software and IBM's Shark, we will briefly describe how to set up a Shark in a Sun Cluster environment.

5.6.5 Brief description of the setup of Shark for a Sun cluster environment

00. We assume the Solaris host already has the Operating System loaded with appropriate patch levels. Furthermore, we also assume that the user is implementing only Sun Fibre Channel Host Bus Adapters in their system per Sun's SFS requirements.

01. Only Sun HBAs support Sun's Open MPXIO. Check Sun's SFS documentation to determine restrictions.

02. Modify /kernel/drv/sd.conf file from default settings on Sun host to include support for more LUNs behind each target. See Figures 20 and 21 for indications of the original defaults and the version after modifications to .add support for more LUNs.

03. Install ESSutl for Solaris:
http://www-1.ibm.com/support/dlsearch.wss?rs=540&tc=ST52G7&dc=D417

04. Install the ESS command-line interface (ESS CLI) on Solaris:
http://publibfp.boulder.ibm.com/epubs/pdf/f2bc1i03.pdf

05. Use the ESS command-line interface to access the Shark.

```
esscli -u user1 -p pass1 -s 9.30.130.15 -b 9.30.130.19 List Server
```

06. Create a default answer file for the ESSCLI command `def`. The following example uses 9.30.130.15 and 9.30.130.19 for the IP addresses and 18878 for the serial number of the IBM Shark:

```
!VERSION 1.0
-u user1 -p pass1
-d "ess=2105.18878"
#-fmt "server,model,mfg,wwn,codeec,cache,nvs,racks"
-s 9.30.130.15
-b 9.30.130.19
```

07. Test ESSCLI with a `def` answer file:

```
esscli –a def  List Server
```

08. Download and install Sun's SAN Foundation Software (SFS), SAN v4.4 software and associated patches from: http://www.sun.com/storage/software/storage_mgmt/traffic_manager/index.xml

09. Modify the `/kernel/drv/scsi_vhci.conf` file. See Figures 22 and 23.

a. Do not enable MPXIO

```
mpxio-disable="yes";
```

Initially we recommend not enabling MPXIO until IBM LUNs are seen by the OS using the format command.

b. Specific configuration identifying the Shark array must be provided. For a Shark model F20, use the following command:

```
device-type-scsi-options-list = "IBM     2105F20", "symmetric-option";
 symmetric-option = 0x1000000;
#
```

For a Shark model 800, use the following command:

```
# symmetric-option = 0x1000000;
 device-type-scsi-options-list =
 "IBM     2105800", "symmetric-option";
symmetric-option = 0x1000000;#
```

There are five blank spaces between "IBM" and "2105". Initially, we recommend not appending this information to the `/kernel/drv/scsi_vhci.conf` file until IBM LUNs are recognized by Solaris.

10. Find the WWPN of the Sun HBAs in the Solaris host.

Run the IBM developed-Korn shell script found in Figure 07.

The WWPNs for the two HBAs are:

210000e08b0838ed and 210100e08b2838ed

11. Choose the proper Sun host type on the Shark.

a. Shark offers three types of Sun host types for the Solaris environment. Choose SunMPXIO when defining the hosts being attached to the Shark.

b. Find the WWPN(s) of the Sun FC HBA , as described in step 10.

c. The IP address of the Sun host

Define the host type configuration for node1_hba1 using the ESS Specialist

Nick Name = node1_hba1

Host Type = Sun MPXIO
Host Attachment = Fibre Channel attached
Hostname/IP Address = *your IP address*
World-Wide Port-Name = 16 digit hexadecimal number from step 10
= 210000e08b0838ed
Fibre-Channel Ports = All installed ports

12. Assign TWO Shark LUNs to only ONE FC HBA in the Solaris host using the ESS Specialist. We recommend using different size LUNs to facilitate the identification of each logical unit. Assigning two LUNs assists in ensuring the `/kernel/drv/sd.conf` file is correct.
13. Build the Solaris device tree
    a. Use the `cfgadm` command to display all devices.
    ```
    cfgadm –al
    ```
    To display Shark devices, use the following command:
    ```
    cfgadm –al | grep 5005
    ```
    To configure devices on controller c2, run the command:
    ```
    cfgadm –c configure c2
    ```
    b. The command to build the hardware tree was `devfsadm` in earlier versions.
    c. The combination of three commands in the following sequence might need to be invoked:
    ```
    drvconfig ; disks ; devlinks
    ```
14. Use the Solaris `format` command to recognize IBM LUNs.
    See Figure 24.
15. List the HostConnection associated with the Solaris host node1_hba1:
    ```
    esscli –a def List HostConnection –d “host=node1_hba1”
    ```
16. List the LUNs associated with node1_hba1:
    ```
    esscli -a def list VolumeAccess -d "host=node1_hba1"  -fmt vol,lun,sz,init
    ```
17. Enable MPXIO by modifying the `/kernel/drv/scsi_vhci.conf` file
    ```
    mpxio-disable="no";
    ```
18. Enter specific information regarding the ESS model in `/kernel/drv/scsi_vhci.conf`.
    There are five blank spaces between “IBM” and “2105”.
19. Run the IBM ESSutl commands and the Solaris luxadm command:
    ```
    /opt/IBMessutl/bin/lsess
    /opt/IBMessutl/bin/ls2105
    /opt/IBMessutl/bin/lssdd
    luxadm probe
    ```

See Appendix 11.4.1 through 11.4.3 for sample output from these commands.

20. Define the second HBA in Solaris host node1 using the ESS Specialist or using the ESSCLI create HostConnection command. We recommend using the ESS Specialist for this task. Enter the following command on one line:
    ```
    esscli –a def   Create HostConnection –d “host=node1_hba2  init=210100e08b2838ed
    profile=sunmpxio port=ALL”
    ```
21. Confirm that node1_hba2 has been setup on the Shark:
    ```
    esscli -a def  List HostConnection | grep node1_hba2
    ```
22. List the LUNs associated with node1_hba1:
    ```
    esscli -a def list VolumeAccess -d "host=node1_hba1"
    ```
23. Assign the LUNs defined in step 12 to a second FC HBA using the ESS Specialist or using the ESS command-line interface (ESSCLI).
    ```
    esscli -a def create VolumeAccess -d "host=node1_hba2 volume=1000,1001"
    ```
24. Confirm that the LUNs previously assigned to node1_hba1 are now also assigned to both node1_hba1 and node1_hba2:
    ```
    esscli -a def list VolumeAccess  -fmt host,vol,lun,sz,init | grep -i node*
    ```
25. Reboot the Solaris host to make the scsi_vhci.conf changes active:
    ```
    sync; sync; sync; reboot -- -r
    ```
26. List the volumes on the Shark:
    ```
    esscli -a def List Volume
    ```

27. Show the access to the volumes from the various nodes and associated HBAs:

```
esscli -a def list VolumeAccess -fmt server,vol,target,lun,sz,init,host,ports
```

28. Use the IBM-developed Korn shell script `mpxiopaths.ksh`, shown in Figure 37, to verify multiple paths are online.

29. Perform the same sequence of operations on the second Solaris host, node2. Ensure that node2_hba1 and node2_hba2 have access to the same two Shark LUNs (7.0 GB & 15.0 GB) that host node1 does.

The brief sequence of tasks above describes how to quickly set up an IBM Shark as shared storage in preparation for installing Sun Cluster V3.x. Now, we will provide a more in-depth description of how to set up Shark for a Sun Cluster environment.

In this scenario, we will assume that there are two nodes: node1 and node2. Furthermore, each node has two Fibre Channel host bus adapters. We will refer to these nodes with their HBAs as:

node1: node1_hba1, node1_hba2
node2: node2_hba1, node2_hab2

Both node1 and node2 will form the basis of Sun Cluster #1.

Next, we will create two LUNs:

- 7.0 GB
- 15.0 GB

These two LUNs are assigned to node1_hba1. And because we want to localize any failures experienced on node1, the same two LUNs are also assigned to node1_hba2. Sun's MPxIO software (STMS) will ensure that if a path goes offline and there remains at least one surviving path, then the failover and load-balancing features of MPxIO will permit access to the storage.

We also know that node2 must be able to access the shared storage (that is, the same two disks) in the event of a nodal failover situation governed by Sun Cluster software. Therefore, node2_hba1 must be defined to also have access to the same two LUNs that node1 has. And, because we wish to localize any failures on this node, we will also permit access to the same two LUNs through node2_hba2.

Then the next step is to set up the environment.

The Shark can be accessed using a browser-based Java GUI. Type one of the two valid IP addresses of the Shark into the Address line of the browser and press Enter. You might see a Security Alert dialog box stating:

> You are about to view pages over a secure connection.
> Any information you exchange with this site cannot be viewed by anyone else on the Web.

Click **OK** to continue. The first security alert might be followed by another:

> Information you exchange with this site cannot be viewed or changed by others. However, there is a problem with the site's security certification

Click **Yes** to proceed.

You should now see on your screen the same information as shown in Figure 01, although the IP addresses assigned to your Shark will probably be different.

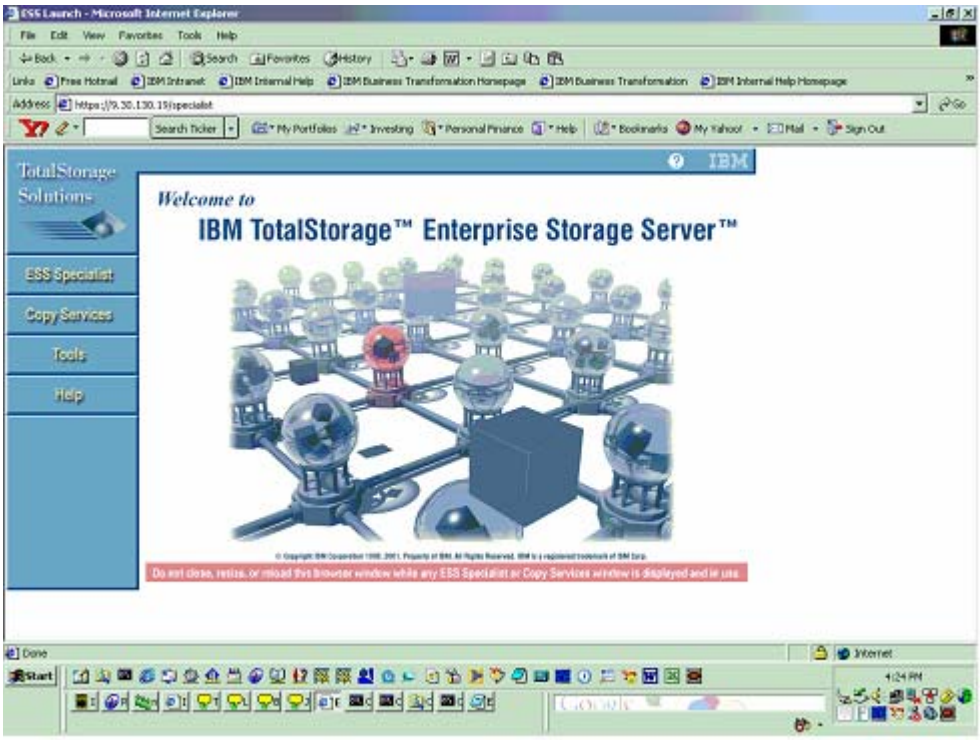


Figure 5-3 ESS Specialist splash screen

Click the **ESS Specialist** button at the left side of the browser. You will be asked to enter a network password. Enter the user name and password that the IBM engineer configured for your Shark.

You might see a warning about a mismatch in the hostname of the security certificate and the name of the server.

> Warning - HTTPS
> Hostname Mismatch
> The hostname in the server security certificate does not match the name of the server.

Click **Yes** to continue. You might be prompted for your user name and password. Simply enter user1 for the User name and pass1 for the password. Click **Yes** to proceed.

You should see a small dialog box saying "Initializing Specialist, please wait". Finally, you will see a window similar to the one shown in Figure 02.

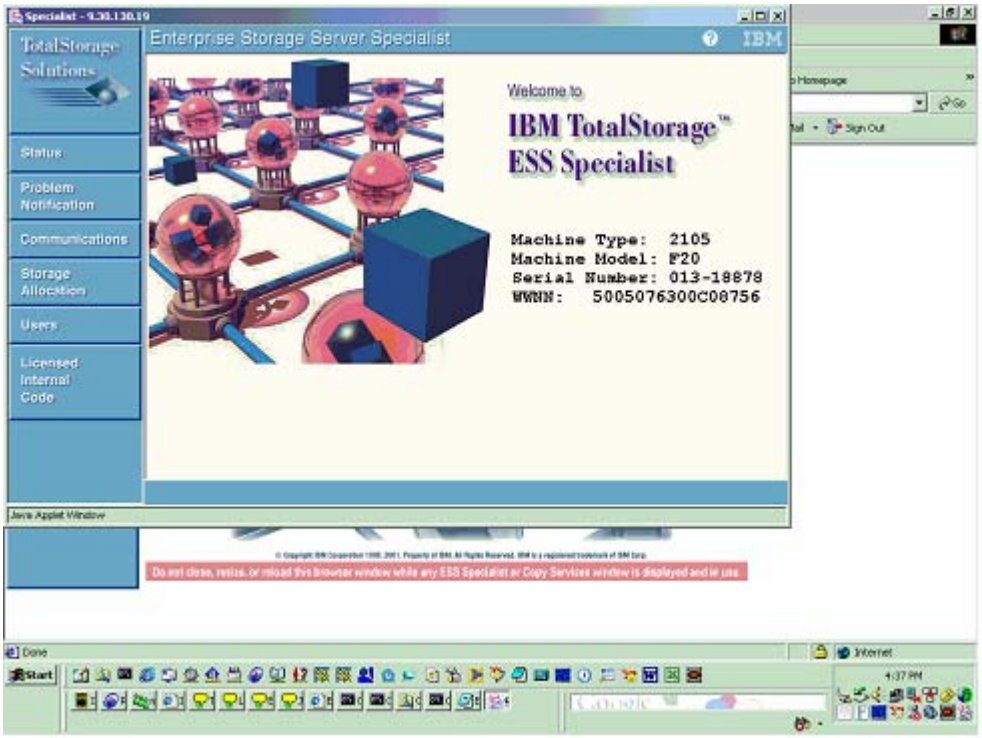


Figure 5-4 ESS Specialist main panel

If communication with the Shark is successful, you will see the following information, with values that reflect your system.)

- Machine Type
- Machine Model
- Serial Number
- WWNN

Pay attention to the first four digits of the Shark's WWNN, specifically 5005. When we prepared the Shark for a Sun Cluster V3.x environment, we used "5005" as a parameter to the `grep` command in step 13. Specifically, we issued the command:

```
cfgadm –al | grep 5005
```

Click **Status** on the left side of the window. You should see a display similar to the one in Figure 03.

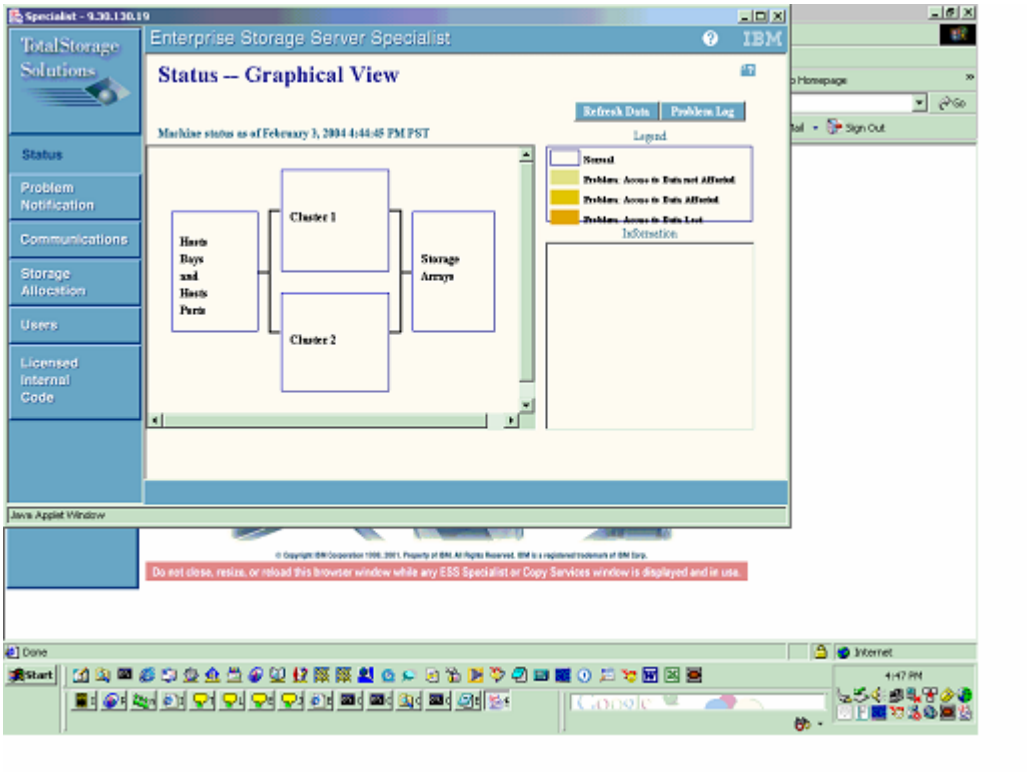


Figure 5-5 ESS status information panel

Click **Refresh Data**, which is located at the upper right. A status message will appear that indicates the system is retrieving the current configuration data and you will be asked to wait.

Click **Storage Allocation** on the left side. You will see a window line the one in Figure 04.

As we are not dealing with a mainframe, click **Open Systems Storage**. You should see a window similar to the one in Figure 05.

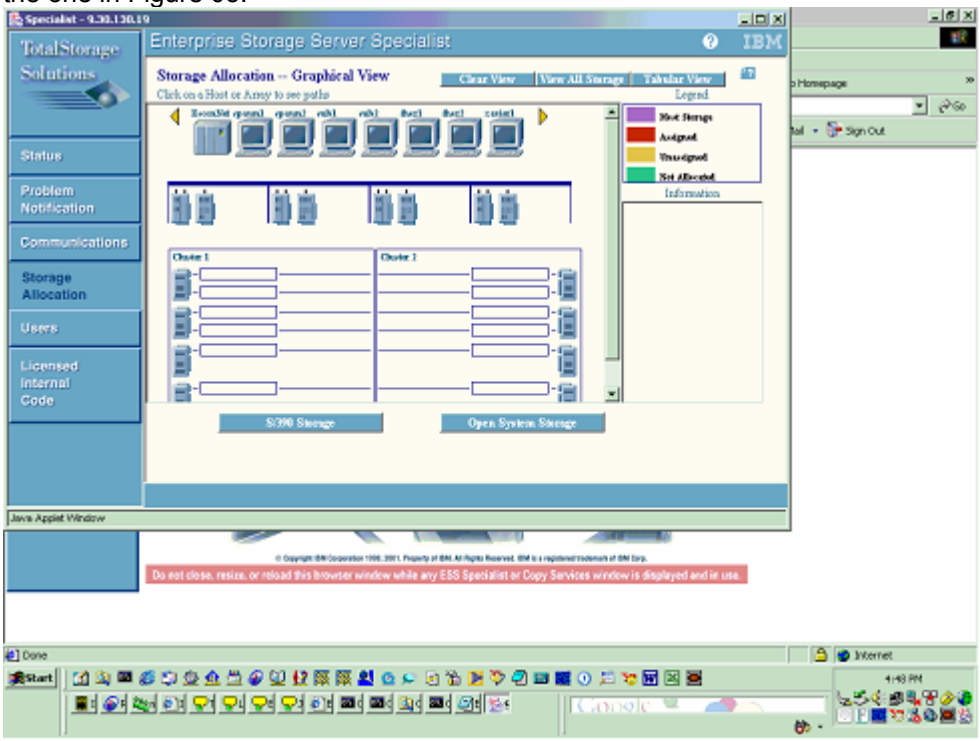


Figure 5-6 Storage Allocation panel

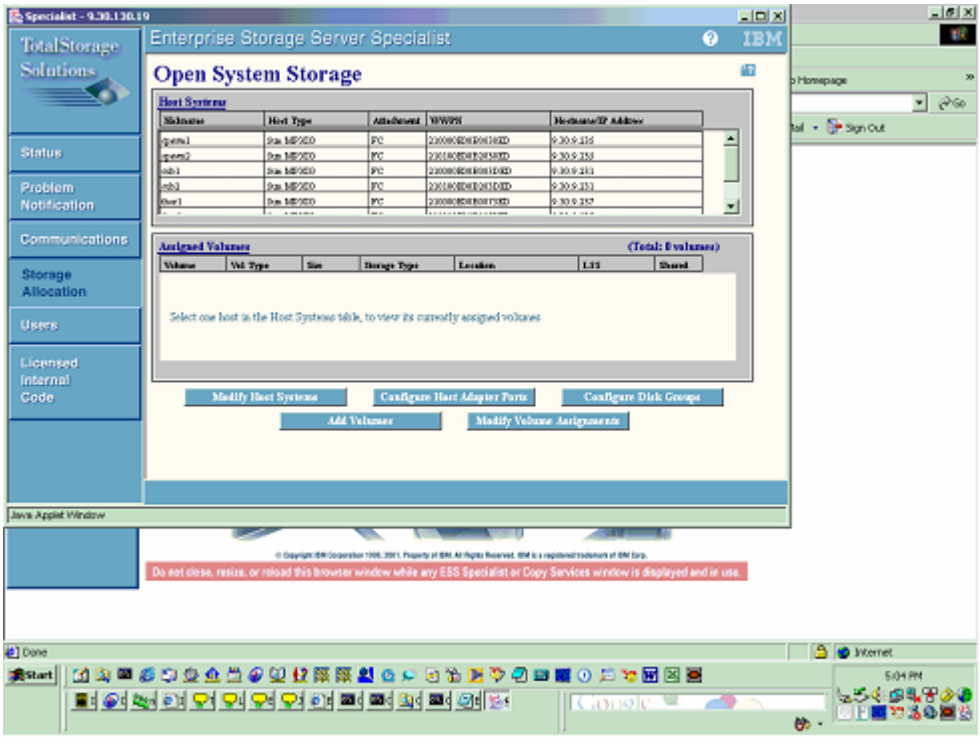


Figure 5-7 Open System Storage panel

Our first operation on the ESS using the ESS Specialist will occur shortly. Click **Modify Host Systems**. You should see a window similar to the one shown in Figure 06.

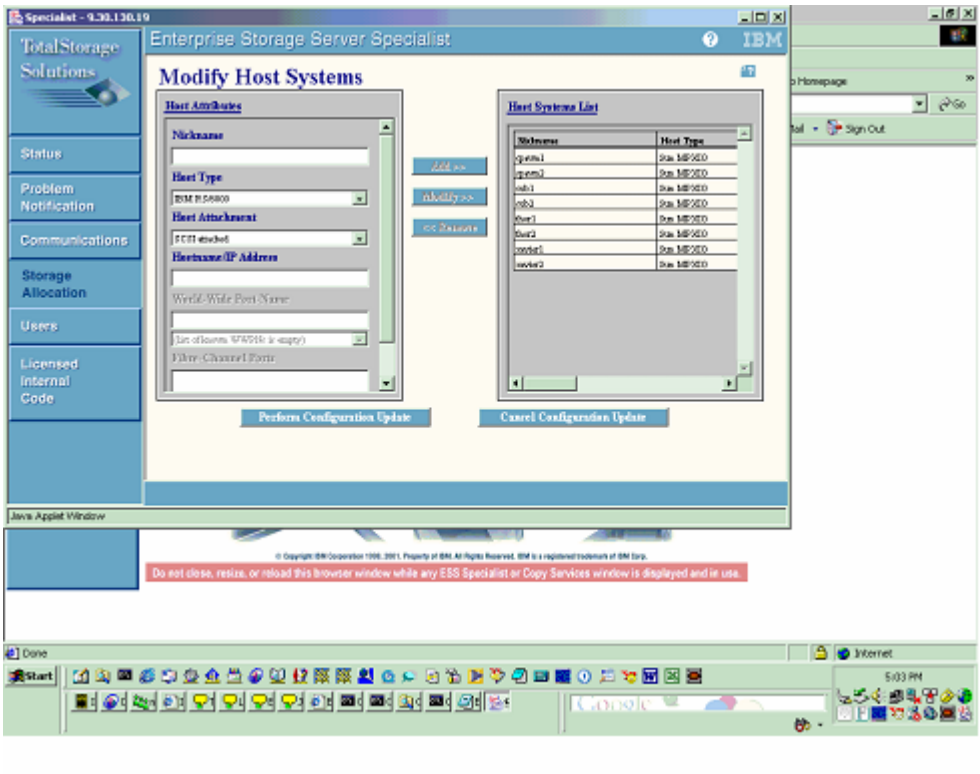


Figure 5-8 ESS Specialist Modify Host Systems panel

Now, we are going to define our first Solaris host, node1, so that the Shark is aware of it. Specifically, we are going to define the first HBA in node1 to the ESS. We label this particular HBA instance as node1_hba1. In order to do this, we need to determine the WWPN of the HBA_1 in the Solaris host.

While you would normally `grep` the file `/var/adm/messages` or alternatively grep the output of the `dmesg` command for EMULEX lpfc HBAs, JNI fca adapters, and QLOGIC qlc/qla cards to identify the FC HBA's WWPN, these techniques do not work for Sun HBAs. However, there are few methods to determine the WWPN of Sun Host Bus Adapters:

- Plug only this fiber optic cable into a Fibre Channel switch then execute the IBM-developed Korn shell script found in Figure 09. This `sun_hbas_wwpn.ksh` script uses the combination of output from the `luxadm -e port` command as input to the `luxadm - e dump_map` command. The output of the Korn shell script displays the WWPNs of Sun HBAs only.

- Plug only this fiber optic cable into a Fibre Channel switch (such as a Brocade, CNT, McData). By logging into the fabric switch, you can discover the WWPN of the Sun FC HBA by examining the port into which the cable is plugged.

- Navigate the Open Boot Prom device tree.

We recommend the first option.

```
#!/bin/ksh
################################################################################
# Program       :      sun_hbas_wwpn.ksh
# Purpose       :      Display WWPN of SUN HBAs
# Author        :      Atul Gore.
# Create Date   :      March 23, 2004.
# Copyright © International Business Machines; IBM
# Algorithm by Dominick Nguyen
# Modification History  :
# Date          Author          Reason
#
################################################################################
echo "After connecting the SUN HBAs to the fabric switch, use this shell script"
echo "Script to determine WWPN of SUN HBAs :"

HBA_PORTS=`luxadm -e port | grep -i connect | wc -l`
echo "No. of paths found to HBA Ports : $HBA_PORTS \n"
for i in `luxadm -e port | grep -i connect | awk '{ print $1 }'`
do
#    echo "Map to port $i is : \n`luxadm -e dump_map $i `"
     PORT_WWN=`luxadm -e dump_map $i | grep -v Port | grep -i "host bus adapter" | awk '{print $4}'`
     echo "Port WWN is : $PORT_WWN"
     echo ""
done
################################################################################
#  End: sun_hbas_wwpn.ksh
################################################################################
```

Figure 07: Korn Shell script to determine WWPN of Sun's FC HBAs

You will need the WWPN information about the Sun HBAs to plug into the ESS Specialist.

Fill in the information into the "Modify Host System" panel as shown in Figure 08.

- Nick Name = node1_hba1
- Host Type = Sun MPXIO
- Host Attachment = Fibre Channel attached
- Hostname/IP Address = *your IP address*
- World-Wide Port-Name = 16 digit hexadecimal number from step 10
- = 210000e08b0838ed
- Fibre-Channel Ports = All installed ports

Then click **Add**, then **Perform Configuration Update**.

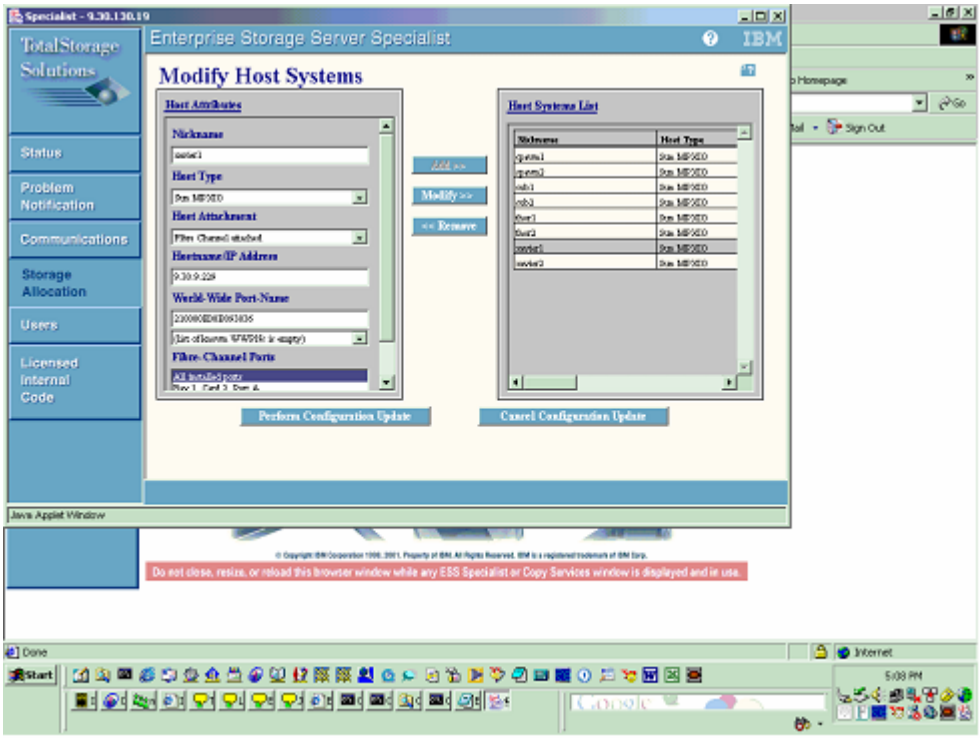


Figure 5-9 Defining a host with Host Type Sun MPXIO to Shark

At this point, you have instantiated your first HBA found in Solaris host node1. The result is that node1_hba1 has been defined to the Shark. There are several ways to confirm the definition has been generated. Run the following ESSCLI command.

```
esscli -a def list HostConnection | grep node1_hba1
```

Click **Storage Allocation**, then **Open Systems Storage.** Find node1_hba1 in the Host Systems sub-panel.

Now that we have defined the first HBA in the Solaris host to the Shark, we can either use the ESS Specialist or the command-line interface to define the second HBA. Enter the following command on one line:

```
esscli –a def   Create HostConnection –d “host=node1_hba2 init=210100e08b2838ed profile=sunmpxio port=ALL”
```

5.6.6 Configuring disk groups

On Shark, as on other RAID systems, you have to allocate a RAID level with specific physical disks. To do so on a Shark, start the ESS Specialist, which is a Web-based configuration tool, click **Storage Allocation > Open Systems Storage**. You should see a window similar to the one in Figure 05.

Now, we need to define ranks. Click **Configure Disk Groups**. You should see a window similar to the one in Figure 09.

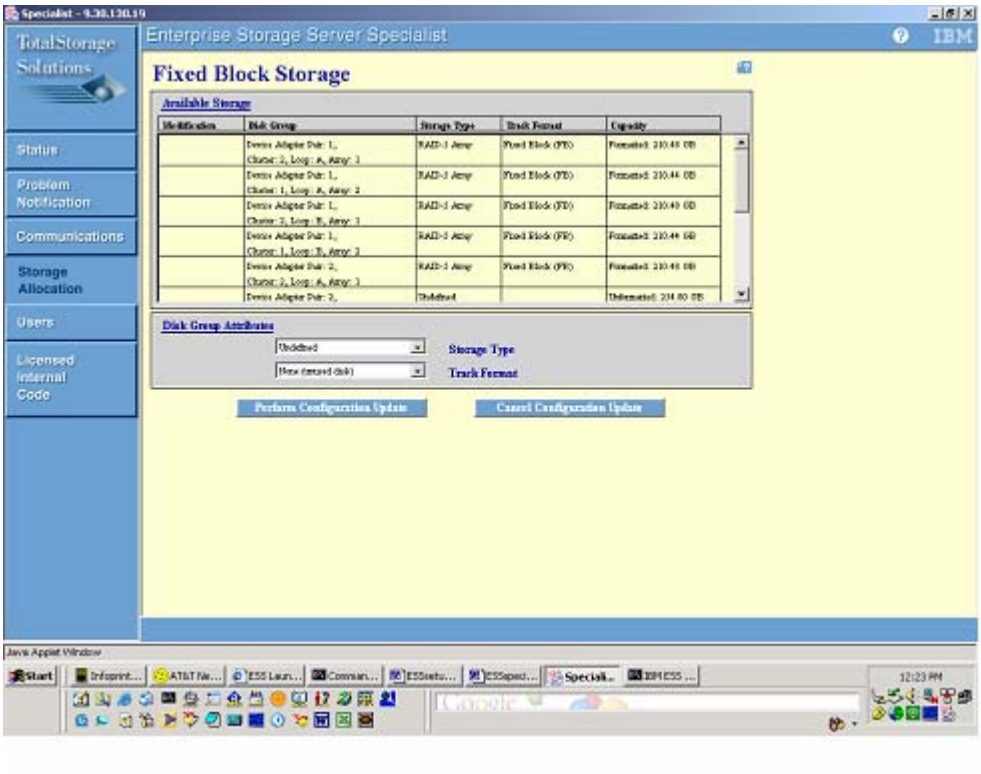


Figure 5-10 Configure Disk Groups panel

Notice in this picture that the first five rows indicate that the ranks have already been formatted as RAID 5 arrays. However, the sixth row shows that that rank has 254.80 GB of unformatted capacity and is still undefined.

If you highlight the first row, you will see that the attributes in the Disk Group Attributes sub-pane will change to reflect the characteristics of that particular rank. Now highlight row number six. As this rank is currently unused, the attribute storage type is undefined and the track format attribute is set to none (unused disk).

Let's modify this undefined rank in row six. While our choices are to leave it undefined, make it appear as a set of JBOD device that does not have any RAID protection, we will change it to a RAID-5 array. Change the attribute storage type to RAID-5 array and you will immediately notice the track format attribute changes to fixed block automatically. Click **Perform Configuration Update** to commit this change. The Shark GUI will show the message:

> Warning 1401: Time-intensive Action.
> The requested configuration update will take approximately 50 to 75
> seconds to complete. Would you like to continue and perform the update?

Click **Yes** to continue.

A Java applet dialog box will be displayed with a message similar to:

```
Performing Configuration Update
Defining RAID Array (ssa02, Gray, 2)
% Percentage Bar
    Estimated Remaining Time: xx seconds
```

When the task is complete, another message will be displayed, informing you that task is done. Click **OK** to continue. You should be returned to a screen similar to that found in Figure 05.

5.6.7 Adding volumes to hosts

We need to allocate a number of LUNs of a given capacity to our Solaris host. Assuming that the ESS Specialist displays a screen similar to that found in Figure 05 (where we left off from the previous section), we will use the ESS Specialist by clicking **Add Volumes**. You should see a screen similar to Figure 10.

The process of adding LUNs to a particular host is a multi-step process.

Find the host's icon at the top of the display. Click on the host (identified by name) to select it. You will see the icon representation of the host change color and a line is drawn to one or more FC HBAs in the Shark (as is shown in Figure 11).

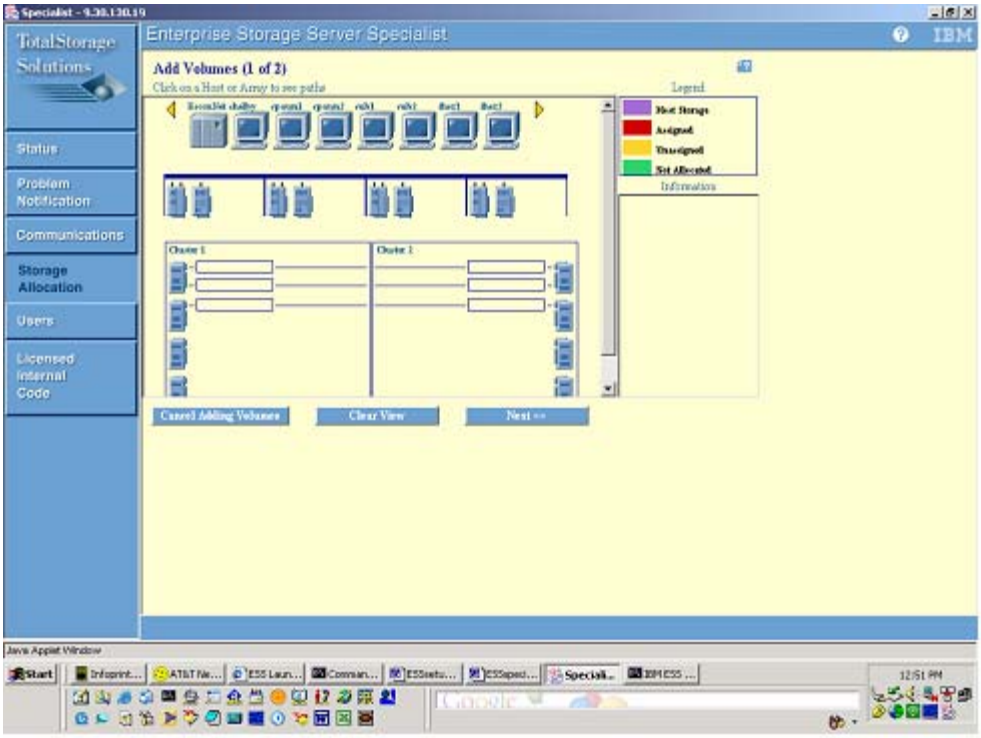


Figure 5-11 Adding Volumes panel

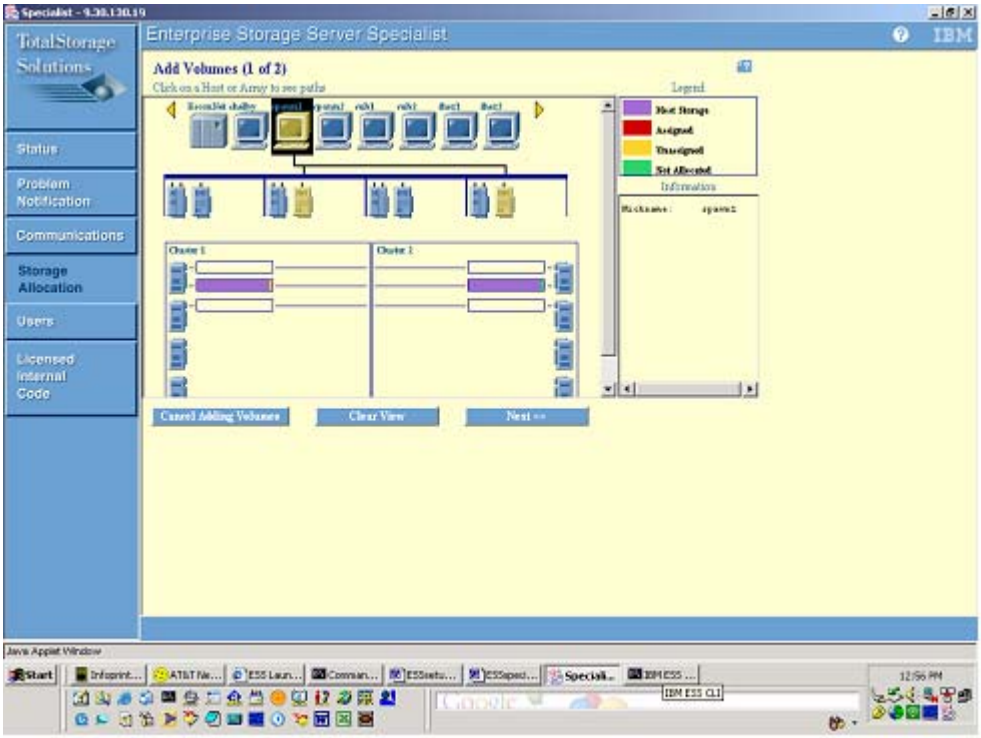


Figure 5-12 Selection of spawn1 highlights the host and the path to FC HBAs

If you click on the Next button at this point, you will get an Error Message dialog box that says:

> Error 1532: Cannot continue.
> The minimum number of items have not been selected.
> Please select at least a host and host adapter port.

If you see this error message, click **OK**. We need to provide more information.

Since we just defined a RAID-5 rank on Cluster 1, we will choose the Shark Fibre Channel HBA on the left side. (The left side corresponds to Cluster 1). The box in third row of cluster one should be entirely green. This rank is the one we just defined. The color green indicates the storage is not allocated.

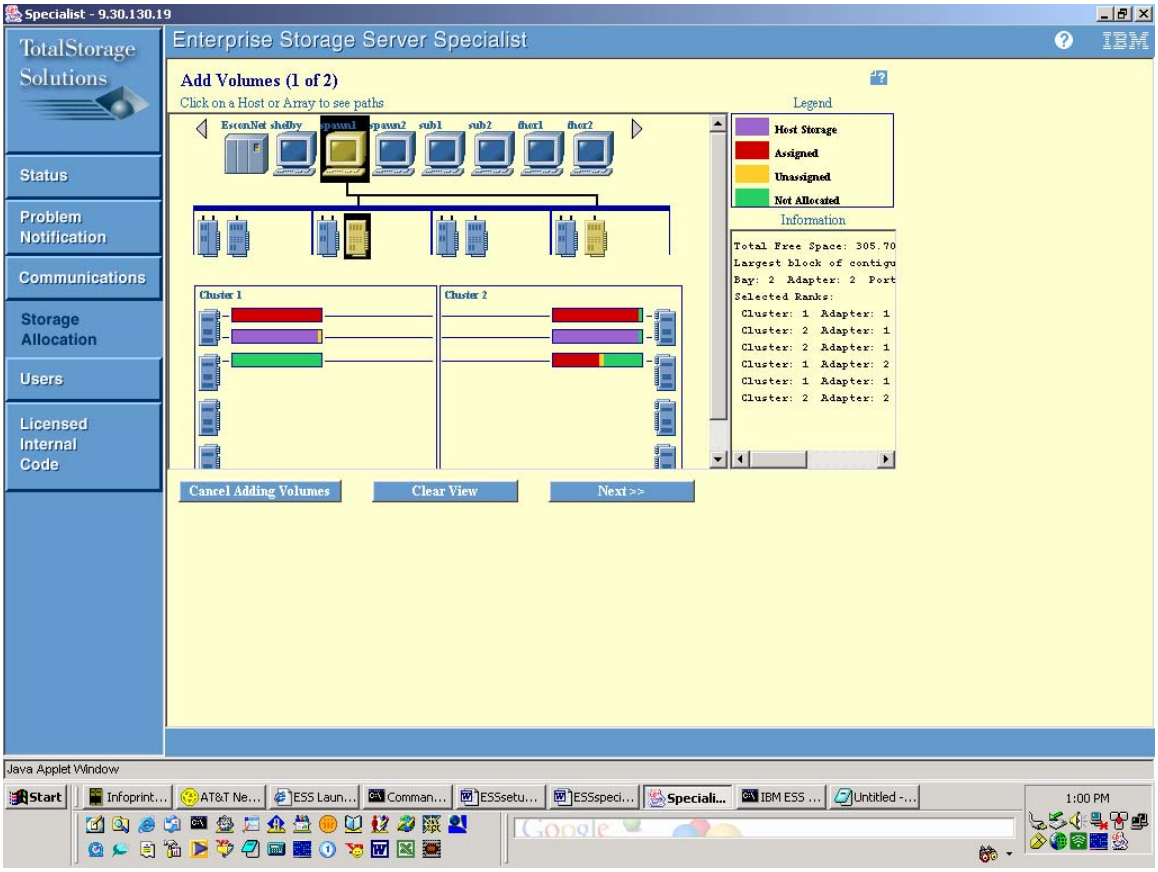


Figure 5-13 >Figure 12: Choosing the Shark FC HBA on the left

If you were to click on the Next button now, you would receive the following error message:

> Error 1650: Insufficient free space
> Volumes cannot be allocated on the set of selected storage areas.
> There is not enough unallocated space left in one or more selected areas. Please select the storage areas individually.

If you see this error message, click **OK**.

You must now select the storage rank that you will allocate space from. Click on the green box on the left side. For further information, see Figure 13.

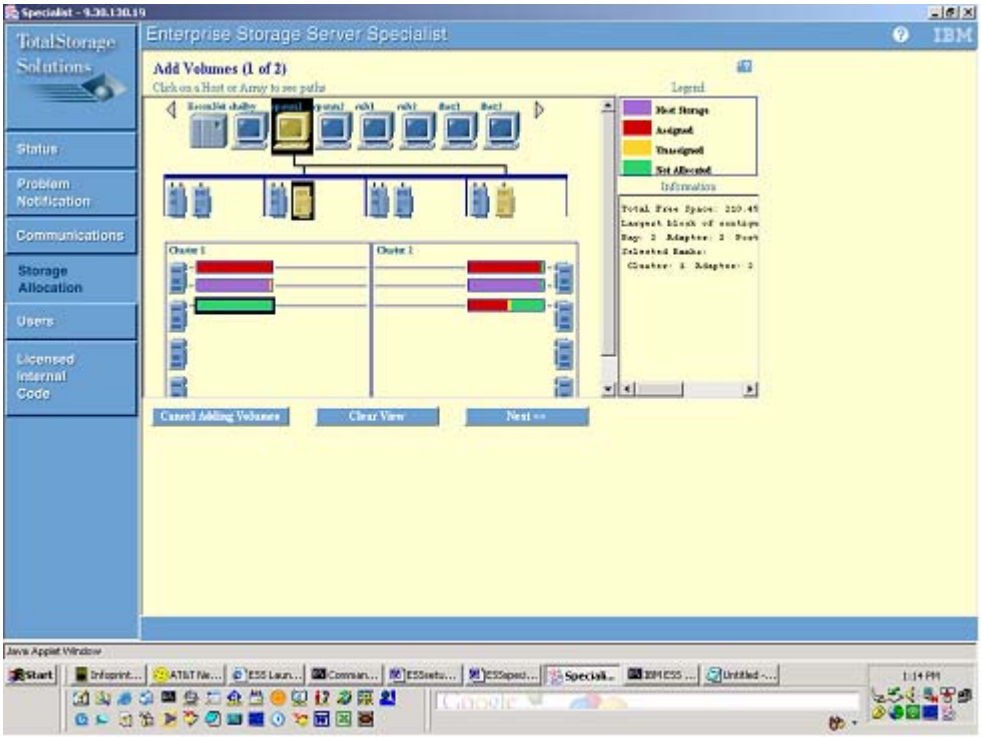


Figure 5-14 Selecting the entirely green colored box on row 3 on the left

When you are ready to proceed, click **Next**. You should see a window similar to the one in Figure 14.

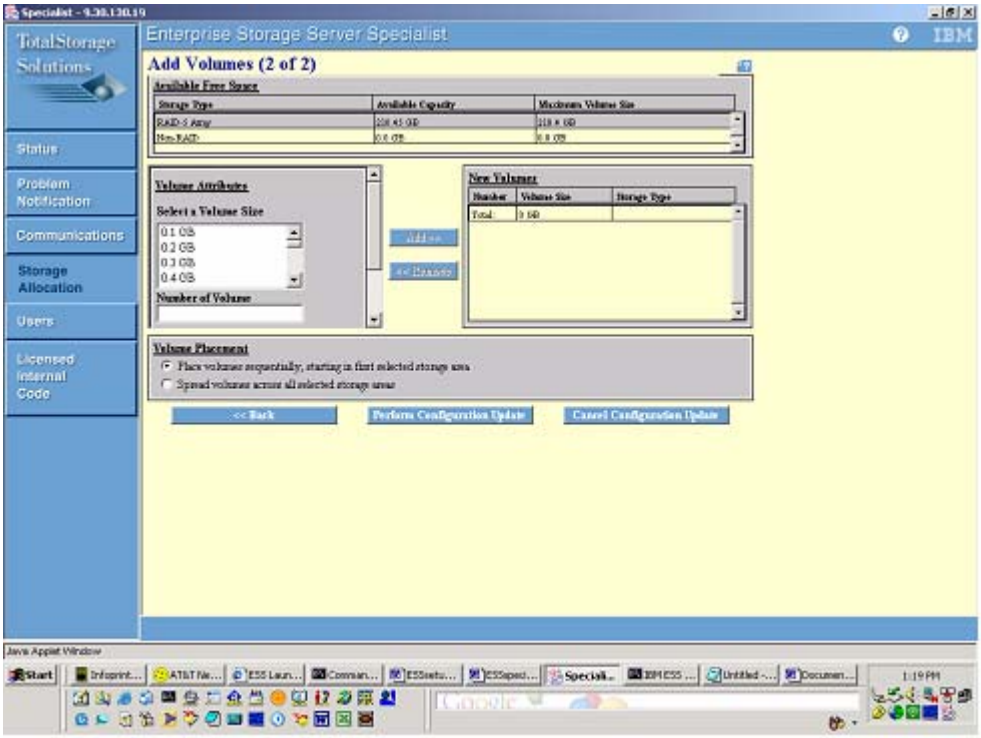


Figure 5-15 Add Volumes (2 of 2) panel

This panel identifies how much capacity is available for each RAID level (Raid 5 and non-RAID). You can specify the size of the volume, the number of volumes, and the placement of the volumes. After selecting these attributes, click **Add**, then **Perform Configuration Update**.

We will allocate two LUNs (one 7.0 GB and one 15.0 GB) to host spawn1. See Figures 15-17.

To define the first LUN, we specified 7.0 GB for the volume attribute and entered the value 1 for the attribute number of volumes. We also selected how we wanted the storage to be placed by choosing the second choice to the volume placement attribute. See Figure 15 for more information and specify the attributes for your own LUN..

After selecting all the attributes associated with this logical unit, click **Add**. The 7.0 GB volume is shown on the right side under the New Volumes heading in Figure 16.

For the second LUN, we chose the same parameters as before with the exception of the capacity of the logical unit. In this case, we specified its size to be 15.0 GB.

Specify the attributes associated with your second logical unit and click **Add**.

Figure 17 shows both the 7.0 GB and 15.0 GB LUNs definitions on the right side under the New Volumes heading prior to the commit phase. Click **Perform Configuration Update** to assign the volumes to the Solaris host. You will be warned about a time-intensive action.

After the configuration definition is committed, the ESS will begin formatting the LUNs and you will be presented with the following message:

```
2101: The specified volumes have been added successfully and are being
formatted. The Enterprise Storage Server is formatting the new volumes. Please
wait until formatting is complete before using the volumes. You can check the
progress of formatting on the Modify Volume Assignments panel (use the Refresh
Status button for the latest progress).
```

Proceed to the Modify Volume Assignments panel shown in Figure 18.

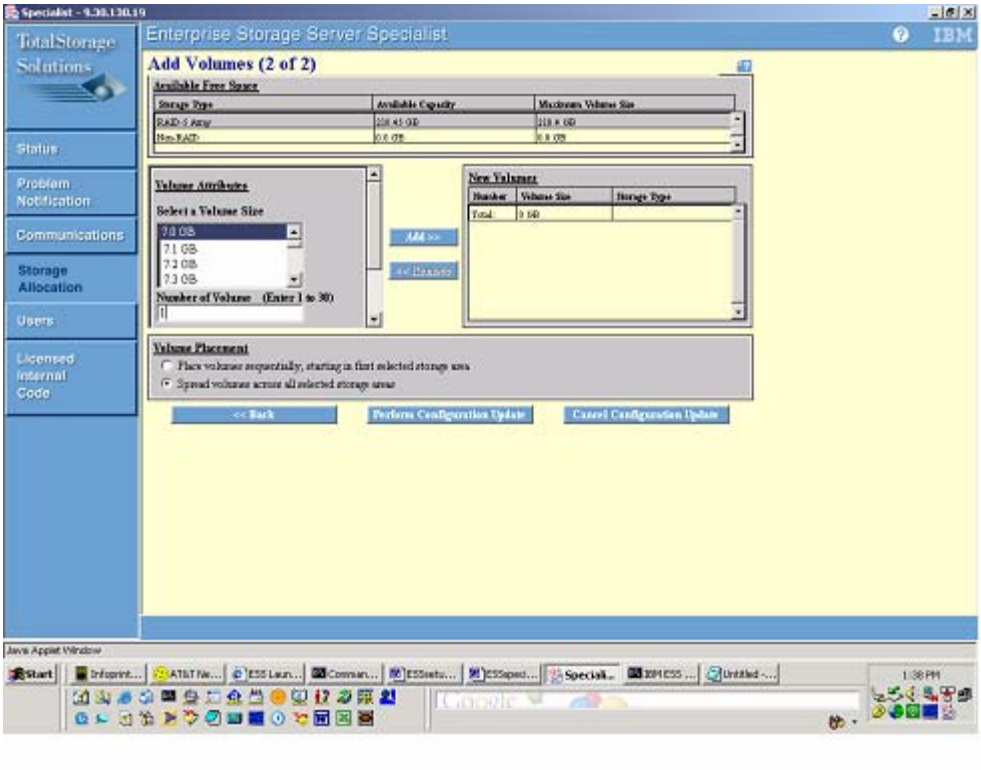


Figure 5-16 Allocating one 7.0 GB LUN to host spawn1

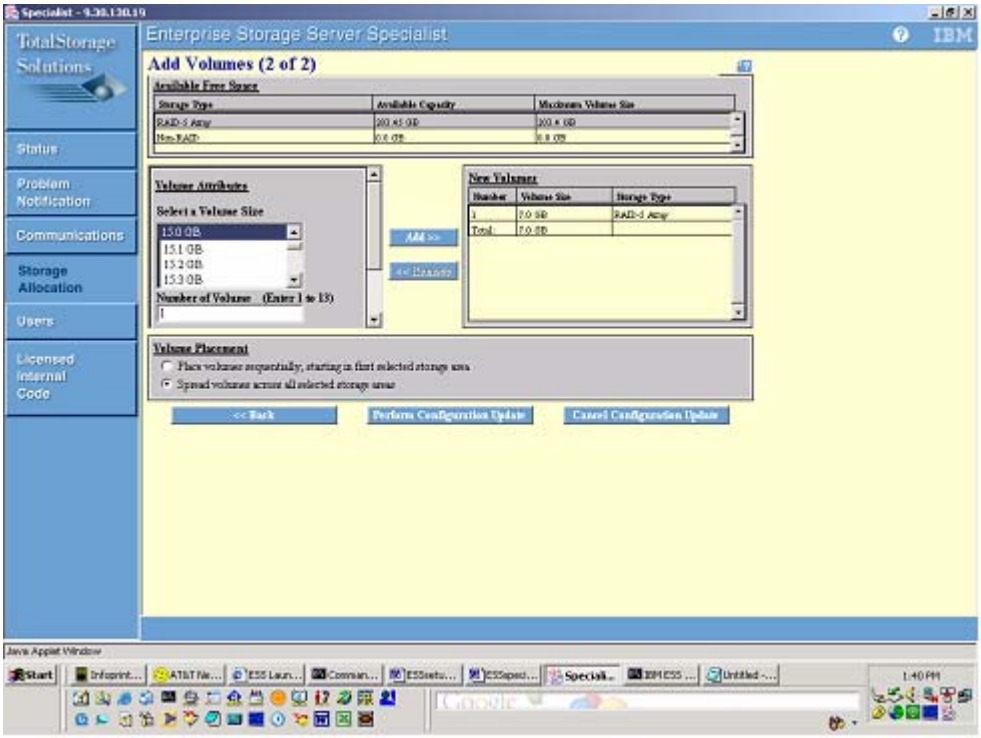


Figure 5-17 Allocating a 15.0 GB LUN to host spawn1

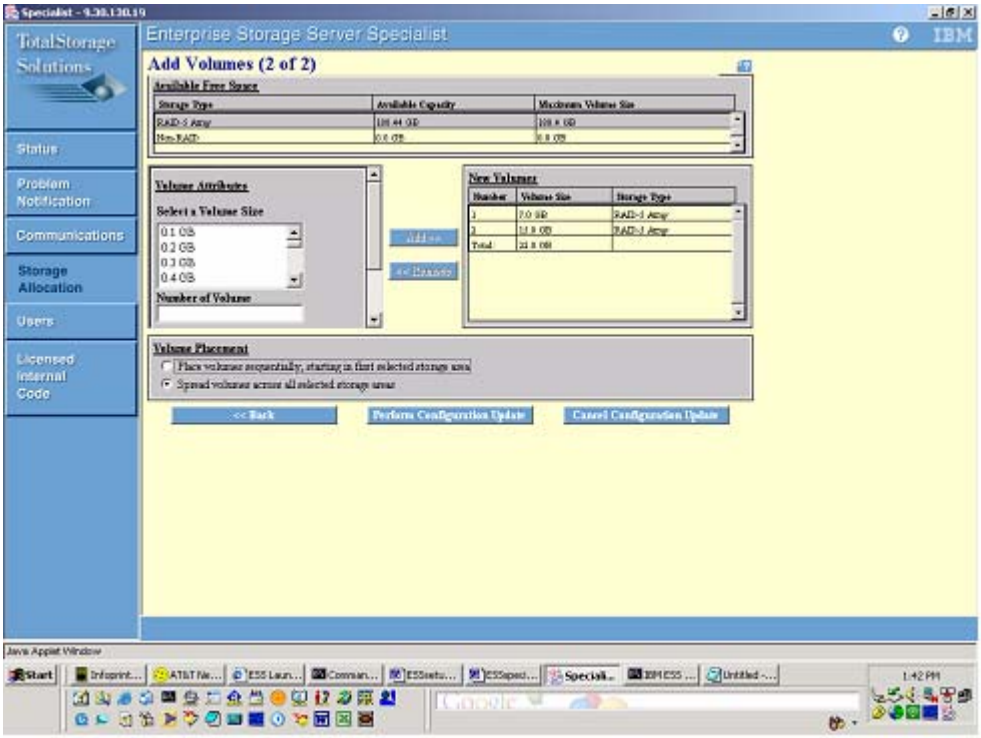


Figure 5-18 Two LUNs (one 7.0 GB & one 15.0 GB) prior to commit phase

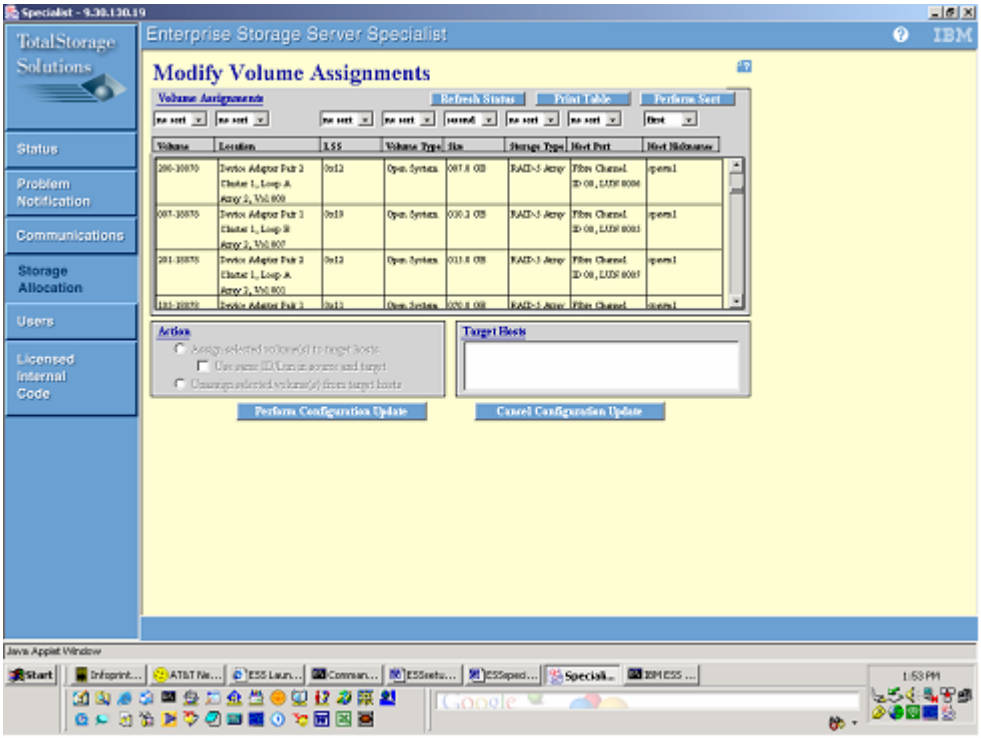


Figure 5-19 Modify Volume Assignments panel

Now we need to proceed to configuring the Solaris host from the host side so that it will recognize these two LUNs.

First, we will assume that you have booted up the Solaris operating system. In order to enable the operating system to recognize multiple LUNs behind a given SCSI target, you must modify the `/kernel/drv/sd.conf` file. See Figures 20 and 21 for the difference between the original default version and the revised version that permits support for more LUNs.

Next follow instructions found in Quick Steps 03-07 to install two IBM applications: IBM ESS Utility for Solaris and the ESS command-line interface.

If the file `/kernel/drv/scsi_vhci.conf` exists, DISABLE MPxIO as explained in step 09.

Because we have two LUNs of varying capacity, we will execute the `cfgadm -al | grep 5005` command found in step 13. Running the Solaris format command after the previous cfgadm command will enable us to determine if these two LUNs appear in the list of recognized devices. These disks will appear with the familiar `cXtYdZ` representation. And these disks do not have valid labels. If these two Shark LUNs do not appear on the list generated by the Solaris `format` command, you may have to use previous commands (such as `devfsadm` or the combination of `drvconfig` ; `disks` ; `devlinks`) to recognize these disks. Once these disks are visible upon executing the `format` command and are actually present (use the `format` command to do a non-destructive read from the device), then if the file `/kernel/drv/scsi_vhci.conf` exists, modify it to enable MPXIO. As soon as the Sharks LUNs are visible and MPxIO is enabled, the disks are no longer represented by Solaris in the standard `cXtYdZ` format. See Figure 24 to see the new format for designation of MPxIO-enabled LUNs.

Instructions for the location of the latest version of Sun's SAN Foundation Software (SFS) are provided in Quick Step 08. Download and install this application according to the instructions provided by Sun. Also apply any patches required by Sun.

Modify the file `/kernel/drv/scsi_vhci.conf` as indicated in step 17 to enable MPxIO at this time. Also append the section found in Quick Step 09 depending on which model of Shark is being used: 2105 model F20 or 2105 model 800. Note that there are five blank spaces between “IBM” and “2105”. See Figures 32 and 33 for more information.

Reboot the host. We recommend rebooting with the reconfiguration option to clean up the device tree. See Quick Step 25.

Use the Solaris `format` command to ensure that Shark devices are shown in the output list from the `format` command. Alternatively, execute the commands found in step 19 to verify Shark LUNs are visible to the operating system.

We need to enable access to both LUNs from the second HBA in node1. To enable node1_hba2 to access both these LUNs, we can either use the ESS Specialist or the ESS command-line interface.

Instructions for defining the second HBA in node1 to the Shark were provide in Quick Step 20. Verification that the node1_hba2 object has been properly defined can be determined by following instructions in Quick Step 21.

Alternatively, you can use the ESS Specialist to accomplish these tasks. We have already defined the second HBA using the ESS Specialist. Now, we need to assign the 7.0 and 15.0 GB volumes to node1_hba2. Navigate to the Modify Volume Assignments panel. (If you closed the ESS Specialist, restart it, click **ESS Specialist > Storage Allocation > Open Systems Storage**. Click **Modify Volume Assignments**, a panel similar to Figure 27 will be displayed.

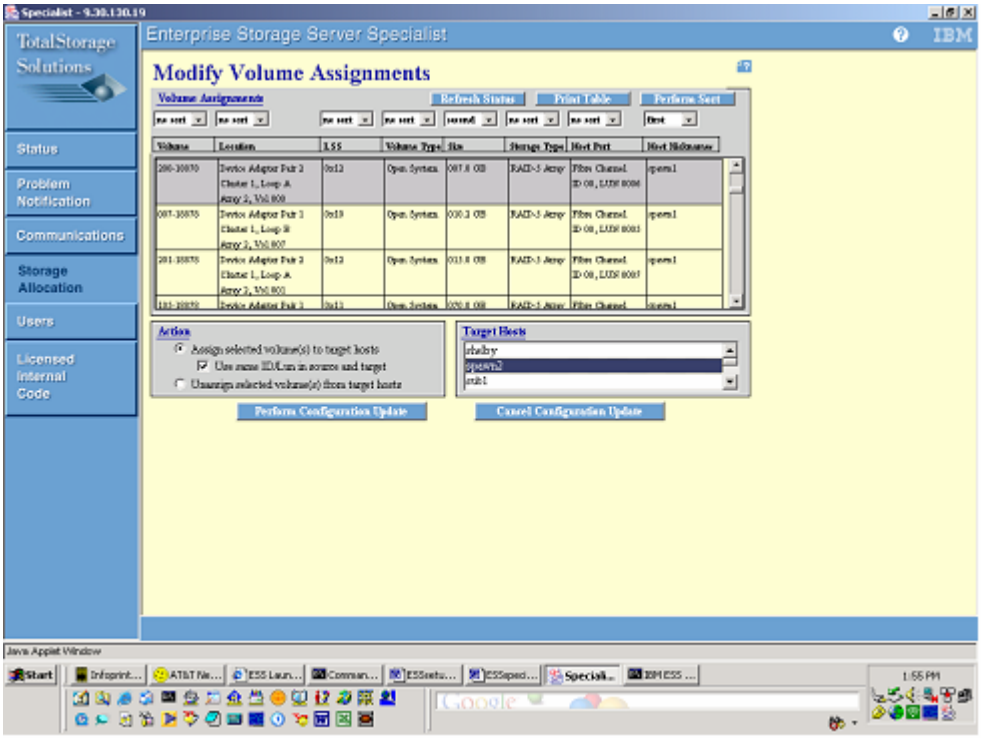


Figure 5-20 Permitting access to spawn2 to previously defined 7.0 GB LUN

The list boxes located below the title in Volume Assignments sub-panel permit sorting according the attribute of the column heading. Change the Host Nicknames list box to indicate first and also change the list box for the Size column to show second. Then click **Perform Sort** to perform the sort according to these two criteria.

Scroll down the list looking for the nickname node1_hba1. Select the row that shows node1_hba1 has a LUN assigned to it that is 7.0 GB in capacity.

Now, move to the Action sub-panel. Select **Assigned selected volume(s) to target hosts**. You will see a list of hosts appear in the Target Hosts sub-panel. Select **Use same ID/Lun in source and target**.

Move to the Target Hosts sub-panel. Select node1_hba2. Click **Perform Configuration Update**.

Repeat the same procedure for the 15.0 GB LUN to assign it to node1_hba2.

At this point, the Solaris host node1 has been configured so that both Fibre Channel HBAs can access the two Shark LUNs. And because of Sun's multipathing software, if access through one path is lost, node 1 still retains access to its storage.

We must define the second Solaris server to the Shark storage subsystem. And because this second Sun server also has two FC HBAs, we will need to define two objects to the Shark. Creating node2_hba1 and node2_hba1 can be accomplished by following the same steps above. You have your choice of using the ESS Specialist or the ESS command-line interface.

Both LUNs that node1 sees, the 7.0 GB LUN and the 15.0 GB LUN, also need to be seen and recognized by the second Solaris server, node2.

We can either associate both these LUNs using the ESS Specialist from the Modify Volume Assignments sub-panel or we can use the ESS command-line interface as was done in Quick Steps 22-24.

Perform the same series of steps outlined in this document to have this second Solaris host recognize the storage that is attached to.

We recommend executing the command found in Quick Step 19 prior to proceeding to the next section. Typical output from these commands are shown in sections11.4.1 through 11.4.3. Pay particular attention to the change from the familiar `cXtYdZ` nomenclature to identify disks in Solaris. With MPxIO enabled, the output of the new format is shown in Figure 24.

5.6.8 The shortcut

1. Assume the following:
    a. Solaris 8 or 9 is installed and running on host servers.
    b. Netscape or Internet Explorer is installed and running on Solaris hosts.
    c. At least two Sun Fibre Channel HBAs are installed in each server.
    d. IBM ESS is installed and configured including both Ethernet ports.
    e. A SAN fabric environment is in place using {Brocade, CNT, McData} switches.
2. Modify `/kernel/drv/sd.conf` to enable support for more LUNs behind a target.
3. Install ESSutl for Solaris.
4. Install the ESS command-line interface.
5. Test connectivity to the Shark using the ESS command-line interface.
6. Install the Sun SAN Foundation Suite (SFS).
7. Modify `/kernel/drv/scsi_vhci.conf` to enable MPXIO.
8. Modify `/kernel/drv/scsi_vhci.conf` to support third-party storage: Shark F20 or 800.
9. Create the following objects on the Shark using ESS Specialist or the command-line interface.
    a. node1_hba1
    b. node1_hba2
    c. node2_hba1
    d. node2_hba2
10. Configure Shark disk groups (ranks) on the Shark using ESS Specialist or the command-line interface.
11. Create Shark LUNs on the Shark using ESS Specialist or the command-line interface.
12. Associate LUNs to host server HBAs on the Shark using ESS Specialist or the command-line interface.
13. Reboot with reconfiguration to clean up the device tree: `reboot -- -r`
14. Execute `cfgadm –al | grep 5005` on the Solaris servers.
15. Run the Solaris `format` command to recognize the disks, label them, and partition them.
16. Execute the commands found in step 19 on the Solaris servers.
17. Now continue with the setup and configuration of Sun Cluster V3.x software.

5.6.9 Figure 20: Default /kernel/drv/sd.conf file

```
# Copyright (c) 1992, by Sun Microsystems, Inc.
#
#ident	"@(#)sd.conf	1.9	98/01/11 SMI"

name="sd" class="scsi" class_prop="atapi" target=0 lun=0;
name="sd" class="scsi" class_prop="atapi" target=1 lun=0;
name="sd" class="scsi" class_prop="atapi"	target=2 lun=0;
name="sd" class="scsi" class_prop="atapi"	target=3 lun=0;
name="sd" class="scsi"	target=4 lun=0;
name="sd" class="scsi"	target=5 lun=0;
name="sd" class="scsi"	target=6 lun=0;
name="sd" class="scsi"	target=8 lun=0;
name="sd" class="scsi"	target=9 lun=0;
name="sd" class="scsi"	target=10 lun=0;
name="sd" class="scsi"	target=11 lun=0;
name="sd" class="scsi"	target=12 lun=0;
name="sd" class="scsi"	target=13 lun=0;
name="sd" class="scsi"	target=14 lun=0;
name="sd" class="scsi"	target=15 lun=0;
```

5.6.10 Figure 21: Modified /kernel/drv/sd.conf

```
#
# Copyright (c) 1992, by Sun Microsystems, Inc.
#
#ident	"@(#)sd.conf	1.9	98/01/11 SMI"
# Start of lines added by SUNWscr
sd_retry_on_reservation_conflict=0;
# End   of lines added by SUNWscr
name="sd" class="scsi"	class_prop="atapi"  target=0 lun=0;
name="sd" class="scsi"	target=0 lun=1;
name="sd" class="scsi"	target=0 lun=2;
name="sd" class="scsi"	target=0 lun=3;
name="sd" class="scsi"	class_prop="atapi"  target=1 lun=0;
name="sd" class="scsi"		target=1 lun=1;
name="sd" class="scsi"	target=1 lun=2;
name="sd" class="scsi"		target=1 lun=3;
name="sd" class="scsi"	class_prop="atapi"  target=2 lun=0;
name="sd" class="scsi"		target=2 lun=1;
name="sd" class="scsi"		target=2 lun=2;
name="sd" class="scsi"		target=2 lun=3;
name="sd" class="scsi"	class_prop="atapi"  target=3 lun=0;
name="sd" class="scsi"		target=3 lun=1;
name="sd" class="scsi"		target=3 lun=2;
name="sd" class="scsi"		target=3 lun=3;
name="sd" class="scsi"		target=4 lun=0;
name="sd" class="scsi"		target=4 lun=1;
name="sd" class="scsi"		target=4 lun=2;
name="sd" class="scsi"		target=4 lun=3;
name="sd" class="scsi"		target=5 lun=0;
name="sd" class="scsi"		target=5 lun=1;
name="sd" class="scsi"		target=5 lun=2;
name="sd" class="scsi"		target=5 lun=3;
```

5.6.11 Figure 22: Modified /kernel/drv/scsi_vhci.conf for an IBM Shark F20

```
# Copyright 2001-2003 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
#pragma ident	"@(#)scsi_vhci.conf	1.8	03/06/16 SMI"
#
name="scsi_vhci" class="root";
#
# mpxio Global enable/disable configuration
# possible values are mpxio-disable="no" or mpxio-disable="yes"
mpxio-disable="no";
#
# Load Balancing global configuration
# possible values are load-balance="none" or load-balance="round-robin"
#
load-balance="round-robin";
#
# Automatic failback configuration
# possible values are auto-failback="enable" or auto-failback="disable"
auto-failback="disable";
#
# For enabling MPxIO support for 3rd party symmetric device need an
# entry similar to following in this file. Just replace the "SUN     SENA"
# part with the Vendor ID/Product ID for the device, exactly as reported by
# Inquiry cmd.
#
# device-type-scsi-options-list =
# "SUN     SENA", "symmetric-option";
#
# symmetric-option = 0x1000000;
#
 device-type-scsi-options-list =
 "IBM     2105F20", "symmetric-option";

symmetric-option = 0x1000000;
```

5.6.12 Figure 23: scsi_vhci.conf settings for an IBM Shark 800

```
# symmetric-option = 0x1000000;
 device-type-scsi-options-list =
 "IBM     2105800", "symmetric-option";

 symmetric-option = 0x1000000;
```

5.6.13 Figure 24: Output of FORMAT command

```
Searching for disks...done
AVAILABLE DISK SELECTIONS:
       0. c0t0d0 <SUN18G cyl 7506 alt 2 hd 19 sec 248>
          /pci@17c,700000/pci@1/scsi@2,1/sd@0,0
       1. c0t1d0 <SUN18G cyl 7506 alt 2 hd 19 sec 248>
          /pci@17c,700000/pci@1/scsi@2,1/sd@1,0
       2. c0t2d0 <SUN18G cyl 7506 alt 2 hd 19 sec 248>
          /pci@17c,700000/pci@1/scsi@2,1/sd@2,0
       3. c4t6005076300C08756000000000000001101d0 <IBM-2105F20-.593 cyl 8342 alt 2 hd 64 sec 256>
          /scsi_vhci/ssd@g6005076300c087560000000000001101
       4. c4t6005076300C08756000000000000001100d0 <IBM-2105F20-.593 cyl 8342 alt 2 hd 64 sec 256>
          /scsi_vhci/ssd@g6005076300c087560000000000001100
       5. c4t6005076300C08756000000000000001001d0 <IBM-2105F20-.593 cyl 11918 alt 2 hd 64 sec 256>
          /scsi_vhci/ssd@g6005076300c087560000000000001001
       6. c4t6005076300C08756000000000000001000d0 <IBM-2105F20-.593 cyl 11918 alt 2 hd 64 sec 256>
          /scsi_vhci/ssd@g6005076300c087560000000000001000
       7. c4t6005076300C08756000000000000001102d0 <IBM-2105F20-.593 cyl 8342 alt 2 hd 64 sec 256>
          /scsi_vhci/ssd@g6005076300c087560000000000001102
       8. c4t6005076300C08756000000000000001004d0 <IBM-2105F20-.593 cyl 10577 alt 2 hd 30 sec 64>
          /scsi_vhci/ssd@g6005076300c087560000000000001004
Specify disk (enter its number): selecting c0t0d0
[disk formatted]


FORMAT MENU:
        disk       - select a disk
        type       - select (define) a disk type
        partition  - select (define) a partition table
        current    - describe the current disk
        format     - format and analyze the disk
        repair     - repair a defective sector
        label      - write label to the disk
        analyze    - surface analysis
        defect     - defect list management
        backup     - search for backup labels
        verify     - read and display labels
        save       - save new disk/partition definitions
        inquiry    - show vendor, product and revision
        volname    - set 8-character volume name
        !<cmd>     - execute <cmd>, then return
        quit
format>
```

5.6.14 Figure 25: Sample answer file for ESSCLI command

```
!VERSION 1.0
-u user1 -p pass1
-d "ess=2105.18878"
#-fmt "server,model,mfg,wwn,codeec,cache,nvs,racks"
-s 10.1.1.11
-b 10.1.1.12
```

5.6.15 Figure 26: Korn Shell script to display MPxIO paths ONLINE status

```
@
#!/bin/ksh
###############################################################################################
# Program       :       mpxiopaths.ksh
# Purpose       :       Display multiple path status to LUNs.
# Author        :       Richard Heffel.
# Create Date   :       March 23, 2004.
# Copyright © International Business Machines; IBM
# Algorithm by Richard Heffel
# Modification History  :
# Date          Author          Reason
#
###############################################################################################
echo "Script to determine if multiple paths to LUNs are ONLINE/INACTIVE:"
echo ""
LUNs=`luxadm probe  | grep "Logical Path:" | wc -l`
luxadm probe -p
echo ""
echo "No. of LUNs : $LUNs \n"
echo "*********************************************************************************"
for i in `luxadm probe | grep Logical | sed 's/Logical Path://' `
do
     echo "luxadm display $i is : \n`luxadm display $i `"
#    LUN_PATH=`luxadm probe  | grep Logical | sed 's/Logical Path://' `
#    echo "MPxIO path info for : $LUN_PATH"
     echo ""
     echo "*********************************************************************************"
done
###############################################################################################
#  End: mpxiopaths.ksh
###############################################################################################
```

5.7 Configuring the Disk I/O Subsystem

Disk devices utilized in the Content Manager high availability environments described in this white paper were configured with Solaris Volume Manager (SVM) in the Solaris 9 Operating Environment and with Solstice DiskSuite in the Solaris 8 Operating Environment. From the Operating System and Sun Cluster point of view, the storage devices are seen as LUN devices. That is to say, the storage system is fault tolerant from the point of view of the operating system. Therefore, in this project, the disksets are configured SVM or DiskSuite were configured as RAID0 for simplicity, and file systems are partitioned with the soft partition feature of SVM/DiskSuite. For extensive SVM software array configuration (RAID1, RAID5, and RAID01/10), see the SVM administrative guide from http://docs.sun.com/.

Before shared volumes can be configured into the cluster environment the appropriate disk devices must be selected. Determining the available disk devices and their characteristics can be accomplished by using the Sun Cluster command `scdidadm –L`. This command will display all storage resources available to the cluster. Shared and local devices will be displayed in the output. Below is the output of the `scdidadm –L` command executed against a two-node cluster:

```
scdidadm -L
1     cldb1:/dev/rdsk/c0t0d0          /dev/did/rdsk/d1
2     cldb1:/dev/rdsk/c0t1d0          /dev/did/rdsk/d2
3     cldb1:/dev/rdsk/c0t2d0          /dev/did/rdsk/d3
4     cldb1:/dev/rdsk/c4t6005076300C0875600000000000001000d0 /dev/did/rdsk/d4
4     cldb2:/dev/rdsk/c4t6005076300C0875600000000000001000d0 /dev/did/rdsk/d4
5     cldb1:/dev/rdsk/c4t6005076300C0875600000000000001001d0 /dev/did/rdsk/d5
5     cldb2:/dev/rdsk/c4t6005076300C0875600000000000001001d0 /dev/did/rdsk/d5
6     cldb1:/dev/rdsk/c4t6005076300C0875600000000000001102d0 /dev/did/rdsk/d6
6     cldb2:/dev/rdsk/c4t6005076300C0875600000000000001102d0 /dev/did/rdsk/d6
7     cldb1:/dev/rdsk/c4t6005076300C0875600000000000001101d0 /dev/did/rdsk/d7
7     cldb2:/dev/rdsk/c4t6005076300C0875600000000000001101d0 /dev/did/rdsk/d7
8     cldb1:/dev/rdsk/c4t6005076300C0875600000000000001100d0 /dev/did/rdsk/d8
8     cldb2:/dev/rdsk/c4t6005076300C0875600000000000001100d0 /dev/did/rdsk/d8
9     cldb1:/dev/rdsk/c4t6005076300C0875600000000000001004d0 /dev/did/rdsk/d9
9     cldb2:/dev/rdsk/c4t6005076300C0875600000000000001004d0 /dev/did/rdsk/d9
10    cldb2:/dev/rdsk/c0t0d0          /dev/did/rdsk/d10
11    cldb2:/dev/rdsk/c0t1d0          /dev/did/rdsk/d11
12    cldb2:/dev/rdsk/c0t2d0          /dev/did/rdsk/d12
```

The `scdidadm` command shows there are two hosts participating in the cluster. Each node has three local disk devices and six shared disk devices. With this information decisions can be made as to which disk devices can be used for placement of binaries, home directories, and data that needs to shared, etc.

The following Solaris Volume Manager commands require a metadevice state database to be configured locally to each system. The database holds the metadevice configuration seen by each host and tracks changes made to the metadevice information. Any unused slice can be utilized for holding the state database. There is a minimum requirement of three state database replicas. The following command creates a metadevice state database with three replicas:

```
metadb –a –f c0t0d0 cltrt0d0s7 c2t0d0s
```

This command creates state database replicas on three different disk slices. Spreading the state database replicas across multiple disks and controllers will decrease the possibility of losing the state database information due to disk or HBA failures.

After the creation of the state database the cluster volume configuration can begin.

The Solaris Volume Manager command, `metaset` configures shared disksets. The diskset configuration allows two hosts that are physically attached to the same set of disk devices to manage the devices in cluster environments. The following command will create a diskset that can be managed by hosts in a Sun Cluster

environment. The command creates a diskset named home and names two hosts, cldb1 and cldb2, that can access the diskset.

```
metaset -s home -a –h cldb1 cldb2
```

Upon successful creation of the disk set, disk devices can be added to the diskset. The `metaset` command is used again to add disk devices to the newly created diskset. The syntax of the command follows. In this case a single disk device is being added to the diskset home.

```
metaset -s home -a /dev/did/rdsk/d9
```

Now that the diskset has been successfully created and a disk device has been added to the diskset using the `metaset` command, a metadevice can be created within the diskset. The metadevice is created with the Solaris Volume Manager `metainit` command. The syntax of the `metainit` command is:

```
metainit -s home d10 1 1 /dev/did/rdsk/d9s0
```

The `metainit` command above creates a metadevice called d10 on the device, /dev/did/rdsk/d9s0. The first number 1 indicates that this metadevice is being created with a single stripe. The second number one indicates only one slice is being used for this metadevice.

The Solaris Volume Manager configuration utilized in the Content Manager Sun Cluster high availability environment in this paper employs a feature of SVM known as soft partitions. Soft partitions allow a disk device to be subdivided into many smaller devices, more than the standard number of partitions the Solaris operating system provides. The command below shows a soft partition being created in the home diskset on the d10 metadevice created in the preceding `metainit` statement. This command creates a 20GB soft partition named D0 on the d10 metadevice.**:**

```
 metainit -s home d10 -p d0 20g
d10: Soft Partition is setup
```

The soft partition commands that follow are creating soft partitions on meta devices within their respect disk sets indicated by the –s argument of the `metainit` command. In all cases the steps above to determine available shared devices, creating disk sets, adding disk devices to the disksets, and creating meta devices within the disksets were successfully completed prior to creating the soft partitions.

```
 metainit -s lsdg d10 -p d0 20g
d10: Soft Partition is setup
 metainit -s lsdg d11 -p d0 20g
d11: Soft Partition is setup
 metainit -s rmdg d10 -p d0 20g
d10: Soft Partition is setup
 metainit -s rmdg d11 -p d0 20g
d11: Soft Partition is setup
 metainit -s rmdg d20 -p d0 20g
d20: Soft Partition is setup
 metainit -s rmdg d21 -p d0 20g
d21: Soft Partition is setup
```

When partition creation is complete, the Solaris Volume Manager metadevices can be manipulated as a regular block or character device. The SVM metadevices reside in the `/dev/md` directory rather than the `/dev/dsk` and `/dev/rdsk` directories. The dsk and rdsk devices are also preceded by their metaset name.

The `newfs` command can be used to create file systems on the raw devices in the same manner as a regular disk device.

```
newfs /dev/md/home/rdsk/d10
newfs /dev/md/lsdg/rdsk/d10
newfs /dev/md/lsdg/rdsk/d10
newfs /dev/md/rmdg/rdsk/d10
newfs /dev/md/rmdg/rdsk/d11
newfs /dev/md/rmdg/rdsk/d20
newfs /dev/md/rmdg/rdsk/d21
```

5.7.1 Configuring the global devices

Note: For the convenience of installation, the storage device for the library server and resource manager database can also be globally mounted at this time. For improved I/O performance in production environment, these file systems or devices can be configuration with HAStoragePlus as failover file system. See *DB2 Universal Database and High Availability on Sun Cluster 3.X* (http://www-306.ibm.com/software/data/pubs/papers/#suncluster).

The following user directories were made on the shared devices created above: `db2inst1`, `db2inst2`, `db2fenc1`, `db2efnc2`, `icmadmin`, `rmadmin`, and `icmconct`. The `/db/lsdb` directory holds the Content Manager library server database and the `/db/lslog` contains the library server database logs. The `/db/rmdb` and `/db/rmlog` directories contain the Content Manager resource manager database and logs respectively.

```
mkdir -p /global/home
mkdir -p /db/lsdb
mkdir -p /db/lslog
mkdir -p /db/rmdb
mkdir -p /db/rmlog

rsh cldb2 mkdir -p /global/home
rsh cldb2 mkdir -p /db/lsdb
rsh cldb2 mkdir -p /db/lslog
rsh cldb2 mkdir -p /db/rmdb
rsh cldb2 mkdir -p /db/rmlog
```

Below are the `/etc/vfstab` entries for the newly created shared files systems. These entries need to go into the vfstab tab of each node participating in the cluster.

```
/dev/md/home/dsk/d10 /dev/md/home/rdsk/d10 /global/home ufs 2 yes global,logging
/dev/md/rmdg/dsk/d10 /dev/md/rmdg/rdsk/d10 /db/rmdb ufs 2 yes global,logging
/dev/md/rmdg/dsk/d11 /dev/md/rmdg/rdsk/d11 /db/rmlog ufs 2 yes global,logging
/dev/md/rmdg/dsk/d20 /dev/md/rmdg/rdsk/d20 /db/lsdb ufs 2 yes global,logging
/dev/md/rmdg/dsk/d21 /dev/md/rmdg/rdsk/d21 /db/lslog ufs 2 yes global,loggingInstalling the CM LS/RM Database Cluster
```

5.8 Set up user and group ids

Make sure `/global/home` is globally mounted between cldb1 and cldb2, the two database server nodes.

```
/global/home      (/dev/md/home/dsk/d10):40952078 blocks  2489072 files
```

First, disable the NIS name lookup service, modifying the appropriate entry in the `nsswitch.conf` file:

```
/etc/nsswitch.conf
# ORG:
# passwd:     files nis
# group:      files nis
# NOW:
passwd:     files
group:      files
```

Note: Sun Cluster automatically configures the `nsswitch.conf` with cluster entry for hosts and netmask items in `nsswitch.conf`.

Define group and group IDs:

```
Group ID:
> groupadd -g 100 db2as
> groupadd -g 101 db2adm1
> groupadd -g 102 db2udf1
> groupadd -g 201 db2adm2
> groupadd -g 202 db2udf2
```

Next, define all user IDs and create their home directories on global shared file space:

```
>#  useradd -u 101 -g db2adm1 -G db2as -d /global/home/db2inst1 -m -s /bin/ksh db2inst1 64 blocks
> useradd -u 201 -g db2adm2 -G db2as -d /global/home/db2inst2 -m -s /bin/ksh db2inst2 64 blocks
> useradd -u 102 -g db2udf1 -d /global/home/db2fenc1 -m -s /bin/ksh db2fenc1  64 blocks
> useradd -u 202 -g db2udf2 -d /global/home/db2fenc2 -m -s /bin/ksh db2fenc2  64 blocks
> rsh cldb2 useradd -u 101 -g db2adm1 -G db2as -d /global/home/db2inst1 -s /bin/ksh db2inst1
> rsh cldb2 useradd -u 201 -g db2adm2 -G db2as -d /global/home/db2inst2 -s /bin/ksh db2inst2
> rsh cldb2 useradd -u 102 -g db2udf1 -d /global/home/db2fenc1 -s /bin/ksh db2fenc1
> rsh cldb2 useradd -u 202 -g db2udf2 -d /global/home/db2fenc2 -s /bin/ksh db2fenc2
> useradd -u 103 -g db2adm1 -G db2as -d /global/home/icmadmin -m -s /bin/ksh icmadmin  64 blocks
> useradd -u 203 -g db2adm2 -G db2as -d /global/home/rmadmin -m -s /bin/ksh rmadmin  64 blocks
> rsh cldb2 useradd -u 103 -g db2adm1 -G db2as -d /global/home/icmadmin -s /bin/ksh icmadmin
> rsh cldb2 useradd -u 203 -g db2adm2 -G db2as -d /global/home/rmadmin -s /bin/ksh rmadmin
> useradd -u 106 -d /global/home/icmconct -m -s /bin/ksh icmconct   64 blocks
> rsh cldb2 useradd -u 106 -d /global/home/icmconct -s /bin/ksh icmconct
```

5.9 DB2 UDB installation

The DB2 UDB application will be installed locally on both cluster nodes (cldb1 and cldb2).

As super user on each node, run `db2_install` and choose DB2.ESE to install the complete binary set. Accept the default location, `/opt`, as the base directory. The DB2 binary will be installed in `/opt/IBM/db2/V8.1`.

```
bash-2.05# rlogin clapp1
Password:
Last login: Thu Nov 20 14:40:48 from campi
Sun Microsystems Inc.   SunOS 5.9       Generic May 2002
# cd /net/clapp2/stage/source
# pwd
/net/clapp2/stage/source
# cd db2v8
# ls
007_ESE_SUN_3264_NLV  C47KOML.tar          FP4_SUN
# cd *NLV
# ls
doc         ese.tar.Z   readme.jp   readme.pl   readme.tw
ese         readme.cn   readme.kr   readme.ru   readme.txt
# cd ese
# ls
db2           db2setup      readme.cn     readme.pl     readme.txt
db2_deinstall  doc          readme.jp     readme.ru
db2_install    doc.cmn      readme.kr     readme.tw
# ./db2_install
```

```
Specify one or more of the following keywords,
separated by spaces, to install DB2 products.

 Keyword        Product Description
 DB2.ESE        DB2 Enterprise Server Edition for SUNOS
 DB2.ADMCL      DB2 Administration Client for SUNOS
 DB2.ADCL       DB2 Application Development Client for SUNOS

Enter "help" to redisplay product names.

Enter "quit" to exit.

************************************************************
DB2.ADCL

Default directory for installation of products - /opt
```

5.10 DB2 fix pack installation

After installing DB2 locally on both cluster nodes, install DB2 the latest DB2 fix pack on both cluster nodes. As super user, on each node, install the fix pack according to the installation instructions.

5.11 Update kernel parameters and reboot the systems

Set up the global environment and update `/etc/system` with the IPC settings recommended by `db2osconf`.

```
#cd /opt/IBM/db2/V8.1/bin
#vi ./db2osconf
set msgsys:msginfo_msgmax = 65535
set msgsys:msginfo_msgmnb = 65535
set msgsys:msginfo_msgmni = 5120
set msgsys:msginfo_msgtql = 5120
set semsys:seminfo_semmni = 6144
set semsys:seminfo_semmns = 12902
set semsys:seminfo_semmnu = 6144
set semsys:seminfo_semume = 240
set shmsys:shminfo_shmmax = 26386934169
set shmsys:shminfo_shmmni = 6144
set shmsys:shminfo_shmseg = 240
Total kernel space for IPC:
0.84MB (shm) + 3.84MB (sem) + 1.90MB (msg) == 6.58MB (total)
```

5.12 Globalize /var/db2 and /var/lum

Some profile registry values and environment variables are stored in the files in `/var/db2`, and the DB2 license key is kept in `/var/lum`. To keep information synchronized on both cluster nodes, `/var/db2` and `/var/lum` can be placed on the cluster filesystem.

When running DB2 UDB V8 ESE, each physical node has to have its own copy of `/var/db2/global.reg` to keep track its registry information. Therefore, the file `/var/db2/global.reg` has to be placed on the local filesystem even if `/var/db2` is shared among physical nodes. This file can be placed in the `/opt/IBM/db2/V8.1/ha/sc30` directory on each of the physical cluster nodes, where `/opt/IBM/db2/V8.1/ha/sc30` is on the local file system on each of the physical cluster nodes.

Assuming `/global/home` is the mount point for a global mounted cluster filesystem, execute the following command as super user on one node in the cluster, for example: clust1:

```
# mkdir -p /global/home/var
# mv /var/db2 /global/home/var/.
# mkdir -p /global/home/var/lum

note: /var/lum does not exist before adding license file with "db2licm". If /var/lum exists, use the following command instead:

# mv /var/lum /global/home/var
```

Create a symbolic link from both cluster nodes. As super user, execute the following commands on each node:

```
# cp /var/db2/global.reg /opt/IBM/db2/V8.1/ha/sc30/global.reg
# rm -rf /var/db2
# ln -s /global/home/var/db2 /var/db2
# rm -rf /var/lum
# ln -s /global/home/var/lum /var/lum
```

Create a symbolic link from `/global/home` to point `global.reg` back to a local filesystem. Execute this command only once as super user from the `/global/home` directory:

```
# cd /global/home/var/db2
# rm global.reg
# ln -s /opt/IBM/db2/V8.1/ha/sc30/global.reg global.reg
```

5.13 Update the DB2 product license key

With `/var/lum` on the cluster filesystem, as super user execute the following command on one node in the cluster to update the DB2 product license key. **$DBSW** is the path to the DB2 installer directory:

```
# /opt/IBM/db2/V8.1/adm/db2licm -a $DBSW/db2/license/db2dlm.lic
```

5.14 Create the two DB2 database instances:

As super user, on one node, execute the following command to create DB2 ESE instances:

```
#/opt/IBM/db2/V8.1/instance/db2icrt -u db2fenc1 db2inst1

Sun Microsystems Inc.   SunOS 5.9       Generic May 2002
DBI1070I Program db2icrt completed successfully.

#/opt/IBM/db2/V8.1/instance/db2icrt -u db2fenc2 db2inst2

Sun Microsystems Inc.   SunOS 5.9       Generic May 2002
DBI1070I Program db2icrt completed successfully.
```

After instance creation and before the library server and resource manager databases are brought under Sun Cluster contro, you should verify the contents of the db2nodes.cfg file. This file defines the servers that are participating in the instance. For initial configuration, the db2nodes.cfg file should reflect where the particular instance will run by default. During a failover the db2nodes.cfg file will be updated to the server name that it has failed over to. For the Content Manager high availability configuration described here the db2nodes.cfg file needs only the required fields of *nodenum* and *hostname.*

Below is the content of the db2nodes.cfg file that can be used for initial configuration. If the library server is run under **db2inst1** and the default server the library server runs on is server **cldb1** then the contents of the db2nodes.cfg file should initiallly look like this:

```
0       cldb1
```

The db2inst2 `db2nodes.cfg` file should reflect the server the resource manager database initially configured to run on.

5.15 Update svcename and /etc/services for DB2 instances

As super user, su to DB2 instance owner. Then update the database manager port definition for client connections on one node.

```
# su – db2inst1 –c “db2 update dbm cfg using svcename db2inst1”
# su – db2inst2 –c “db2 update dbm cfg using svcename db2inst2”
```

On both cluster nodes, update `/etc/services` to reflect the port definition for client connections and the reserved FCM ports.

```
# DB2 Ports

db2inst1              50000/tcp          # db2inst1 connection port
DB2_db2inst1                  50010/tcp          # FCM Begin
DB2_db2inst1_END              50020/tcp          # FCM End
db2inst2              51000/tcp          # db2inst2 connection port
DB2_db2inst2                  51010/tcp          # FCM Begin
DB2_db2inst2_END              51020/tcp          # FCM End
```

5.16 .rhosts file for instance owner

Log on to each of the cluster nodes as the instance owner, and create an `.rhosts` file in the instance owner’s home directory with the following entries:

```
# cat .rhosts
cldb1
cldb2
lsdbsrv
rmdbsrv
```

5.17 Enable the DB2 instance for high availability

1. Copy $INSTALLPATH/ha/sc30 to the instance home directory under ~$DB2INSTANCE/sqllib. As super user, execute the following command on node cldb1:

```
# cd /opt/IBM/db2/V8.1
# tar -cvf - ha/sc30 | (cd ~db2inst1/sqllib; tar -xvf -)
# tar -cvf - ha/sc30 | (cd ~db2inst2/sqllib; tar -xvf -)
```

2. Change the group and user ownership

```
# cd ~db2inst1
#chown -R db2inst1:db2iadm1 ha
# cd ~db2inst2
# chown -R db2inst2:db2iadm2 ha
```

3. Register the instance with the Sun Cluster and associate the HA-IP address. As super user, execute the following command on node clust1:

```
# /opt/IBM/db2/V8.1/ha/sc30/util/regdb2udb -a db2inst1 -h lsdbsrv
# /opt/IBM/db2/V8.1/ha/sc30/util/regdb2udb -a db2inst2 -h rmdbsrv
```

4. Bring them online.

```
# /opt/IBM/db2/V8.1/ha/sc30/util/onlinedb2udb -a db2inst1 -h lsdbsrv
# /opt/IBM/db2/V8.1/ha/sc30/util/onlinedb2udb -a db2inst2 -h rmdbsrv
```

5.18 Instances high availability failover validation

Note: See *DB2 Universal Database and High Availability on Sun Cluster 3.X* (http://www-306.ibm.com/software/data/pubs/papers/#suncluster).

The cluster nodes configured to host the library server and resource manager databases employed Solaris IPMP, IP multipathing, to facilitate failover over of the database back ends. The IPMP was configured using a single network interface. Below is the `/etc/hostname.`*`network_interface_name`* file used to enable IPMP on the cldb1 node. A similar file was used on cldb2 to enable IPMP.

```
cldb1 -failover group sc_ipmp0
```

This creates a failover group named sc_ipmp0.

5.19 Content Manager pre-installation tasks

The following tasks must be completed before installing Content Manager.

5.19.1 JDK Installation

Before we can start the Content Manager installation, we need see if the correct JDK package is installed: The Content Manager V8.2 installation procedure requires the Java JDK version 1.3.1; otherwise the install program will not start. At runtime the Content Manager library server does not require any Java package to be available.

On Solaris 9, J2SDK 1.4 is part of the operating system installation and J2SDK 1.4 shares package names with J2SDK 1.3.1. The impact of using 'pkgadd' to install J2SDK 1.3.1 packages is hard to predict. Therefore, we chose to use the self-extracted version of J2SDK downloaded from http://java.sun.com/.

```
# pkginfo | grep -i j3
system      SUNWj3dev                 J2SDK 1.4 development tools
system      SUNWj3dmo                 J2SDK 1.4 demo programs
system      SUNWj3dvx                 J2SDK 1.4 development tools (64-bit)
system      SUNWj3irt                 JDK 1.4 I18N run time environment
system      SUNWj3man                 J2SDK 1.4 man pages
system      SUNWj3rt                  J2SDK 1.4 runtime environment
system      SUNWj3rtx                 J2SDK 1.4 runtime environment (64-bit)
```

Install the self-extracting JDK1.3.1_04, and update the $PATH variable before starting the Content Manager installer.

```
> pwd
/usr/j2sdk-1_3_1_04

> sh /net/clapp2/stage/source/jdk/j2sdk-1_3_1_04-solaris-sparc.sh

> ls -l /usr/java
lrwxrwxrwx   1 root     other          6 Nov 19 23:14 /usr/java -> ./j2se/

> rm /usr/java

> ln -s ./j2sdk1_3_1_04 java

> ls -l /usr/java
lrwxrwxrwx   1 root     other         15 Dec  2 11:15 /usr/java -> ./j2sdk1_3_1_04/

> export PATH=/usr/java/bin:$PATH
```

5.19.2 DB2 instance owner environment setting for Content Manager

Now set up the users db2inst1, and db2inst2 profiles by updating the ~db2inst1 and ~db2inst2/sqllib/profiles

with the Content Manager environment settings needed. Complete the following tasks:

1. Update the file `~db2inst1/sqllib/userprofile`. It should contain the following information:

```
ICMROOT=/opt/IBMicm
ICMDLL=/export/global/home/db2fenc1
ICMCOMP=/opt/SUNWspro/bin
CMCOMMON=/opt/IBMcmb/cmgmt
PATH=$PATH:$ICMROOT/bin/DB2
LD_LIBRARY_PATH=$ICMROOT/lib:$ICMROOT/inso:$LD_LIBRARY_PATH
export ICMROOT ICMDLL ICMCOMP CMCOMMON PATH LD_LIBRARY_PATH
```

Note: Do not modify `/export/home/db2inst1/sqllib/db2profile` itself, because this file can be overwritten by the install script of a DB2 fix pack. Instead, put any necessary modifications in the users profile it self. The DB2profile, when invoked, will execute the user profile as well. In this way all user environment specific settings are added automatically to the DB2 ones. The Content Manager library server instance at runtime will run as a fenced process using the db2fenc1 user ID. This user ID will inherit the environment from the db2inst1 user.

2. Update `~db2inst2/sqllib/userprofile`.

```
ICMROOT=/opt/IBMicm
ICMDLL=/export/global/home/db2fenc2
ICMCOMP=/opt/SUNWspro/bin
CMCOMMON=/opt/IBMcmb/cmgmt
PATH=$PATH:$ICMROOT/bin/DB2
LD_LIBRARY_PATH=$ICMROOT/lib:$ICMROOT/inso:$LD_LIBRARY_PATH
export ICMROOT ICMDLL ICMCOMP CMCOMMON PATH LD_LIBRARY_PATH
```

Next, update the DB2 instance `profile.env` file for db2inst1 and db2inst2. If the data is not already in the file, you need to add two lines to the `/home/db2inst1/sqllib/profile.env` file. The first line needs to contain all library directories that you want DB2 to know about. The second line is the DB2 environment variable list that must contain LIBPATH variables. For example:

```
DB2LIBPATH=/opt/lib:/opt/IBMicm/lib
DB2ENVLIST='DB2LIBPATH ICMROOT ICMDLL ICMCOMP CMCOMMON '

DB2COMM='tcpip '
DB2AUTOSTART='TRUE '
```

The setup for the second instance ID, db2inst2, is much simpler because the resource manager processes do not run on the database nodes. Set::

```
DB2COMM='tcpip '
DB2AUTOSTART='TRUE '
```

Finally, update the user profiles and make sure the Content Manager administrator user IDs are able to connect and use their respective databases. Update the `.profile` files for icmadmin and rmadmin, adding the following line to their `/export/home/icmadmin/.profile` and `/export/home/rmadmin/.profile` files.

```
. /export/home/db2inst1/sqllib/db2profile
```

Note the space between the period (.) and the first slash (/). Adding this line establishes the DB2 environment and associates the users with the db2inst1

5.19.3 Prepare the database path for library server database and resource manager database creation

Determine where the DB2 table spaces should go, and define the correct filesystem path by setting the DFTDBPATH environment variable. This will ensure that the Content Manager library server and resource manager databases will be created on the correct filesystem.

Note: For the convenience of installation, the storage device for the library server and resource manager database can also be globally mounted at this time. For improved I/O performance in production environment, these file systems or devices can be configuration with HAStoragePlus as failover file system. See *DB2 Universal Database and High Availability on Sun Cluster 3.X* (http://www-306.ibm.com/software/data/pubs/papers/#suncluster).

```
> df –lk
/global/home      (/dev/md/home/dsk/d10):41023032 blocks  2489374 files
/db/rmdb          (/dev/md/rmdg/dsk/d10):41266798 blocks  2489979 files
/db/rmlog         (/dev/md/rmdg/dsk/d11):41266798 blocks  2489979 files
/db/lslog         (/dev/md/lsdg/dsk/d11):41266798 blocks  2489979 files
/db/lsdb          (/dev/md/lsdg/dsk/d10):41266798 blocks  2489979 files
```

For the library server database:

As root:

```
> chown db2inst1:db2adm1 /db/lsdb
> chown db2inst1:db2adm1 /db/lslog
```

As db2inst1:

```
> id db2inst1
uid=101(db2inst1) gid=101(db2adm1)
> db2 update dbm cfg using DFTDBPATH /db/lsdb
```

As root, disable the resource groups:

```
> scswitch -n -j db2_db2inst1_0-rs
> scswitch -n -j lsdbsrv
```

As root, stop the db2inst1 instance:

```
> scswitch -z -g db2_db2inst1_0-rg -h ""
```

For the resource manager database:

As root:

```
> chown db2inst2:db2adm2 /db/rmdb
> chown db2inst2:db2adm2 /db/rmlog
```

As db2inst2:

```
> id db2inst2
uid=201(db2inst2) gid=201(db2adm2)
> db2 update dbm cfg using DFTDBPATH /db/rmdb
```

As root, stop the db2inst2 instance:

```
> scswitch -z -g db2_db2inst2_0-rg -h ""
```

5.19.4 Install the Content Manager V8 library server and library server database

Now we are ready to start installing the Content Manager library server. Again, let's begin with checking the configuration now of the icmadmin and rmadmin environment settings. We did this before for db2inst2 and db2inst1. Set the following environment variables in the database manager:

`DFTDBPATH=/db2/lsdb` for db2inst1, the Content Manager library server instance ID
`DFTDBPATH=/db2/rmdb` for db2inst2, the Content Manager resource manager instance ID

This ensures that the Content Manager library server database table spaces are created on stated multi-host file path.

Background: During runtime, it is more important to make sure the Content Manager library server environment settings are correct than settings for the Content Manager resource manager. This is so because the library server stored procedures must access various configuration files located on the filesystem. The resource manager code is executed exclusively on the WebSphere Application Server application server nodes. Normally the Content Manager root directory **`/opt/IBMicm/`** is placed on local disks, but one could think of a common root directory and place it on one of the cluster file systems that is shared between the database backend cluster nodes. This would reduce the effort to keep both CMROOT directories in synch. In our test environment we have tested both variants and both worked fine. In this paper we describe the second variant.

The database cluster nodes will be set up in a hybrid active-active high availability configuration. Both the library server and resource manager will run using independent DB2 instances on two different nodes. We would like for both nodes to back each other up. If one node fails, the other takes over the resource manager and library server resource load. For the DB2 high availability setup, we will configure **db2inst1** to run in resource group **resgrp1**, and **db2inst2** to run in resource group **regrp2**. Resource group **resgrp1** will be active on node1 and passive on node2. Resource group resgrp2 will be active on node2 and passive on node1.

For the installation we will use the following approach:

1. Install the Content Manager library server only on node1 using db2inst1 as the instance ID, and repeating the procedure on node2 using the same instance ID.
2. Then we will reverse the roles, and install the Content Manager resource manager database first on node2 using db2inst2 , and repeating the procedure on node1 using the same instance ID.

Complete the following steps:

1. Log on to node cldb1 as root.

2. Make sure the instance db2inst1 is running on cldb1, use the following command:

```
#scswitch –z -g db2_db2inst1_0-rg –h cldb1
```

3. Source Content Manager and DB2 environment:

```
# . ~db2inst1/.profile
```

4. Start the Content Manager setup program. (Make sure the DISPLAY is set properly for GUI display.)

```
# /net/clapp2/stage/source/cm_sun/cm/English/setup.exe &
```

5. On the “Select the features for Content Manager V8" panel you checkmark the library server box only, nothing else is necessary. This will ensure only the CM LS database will be created.

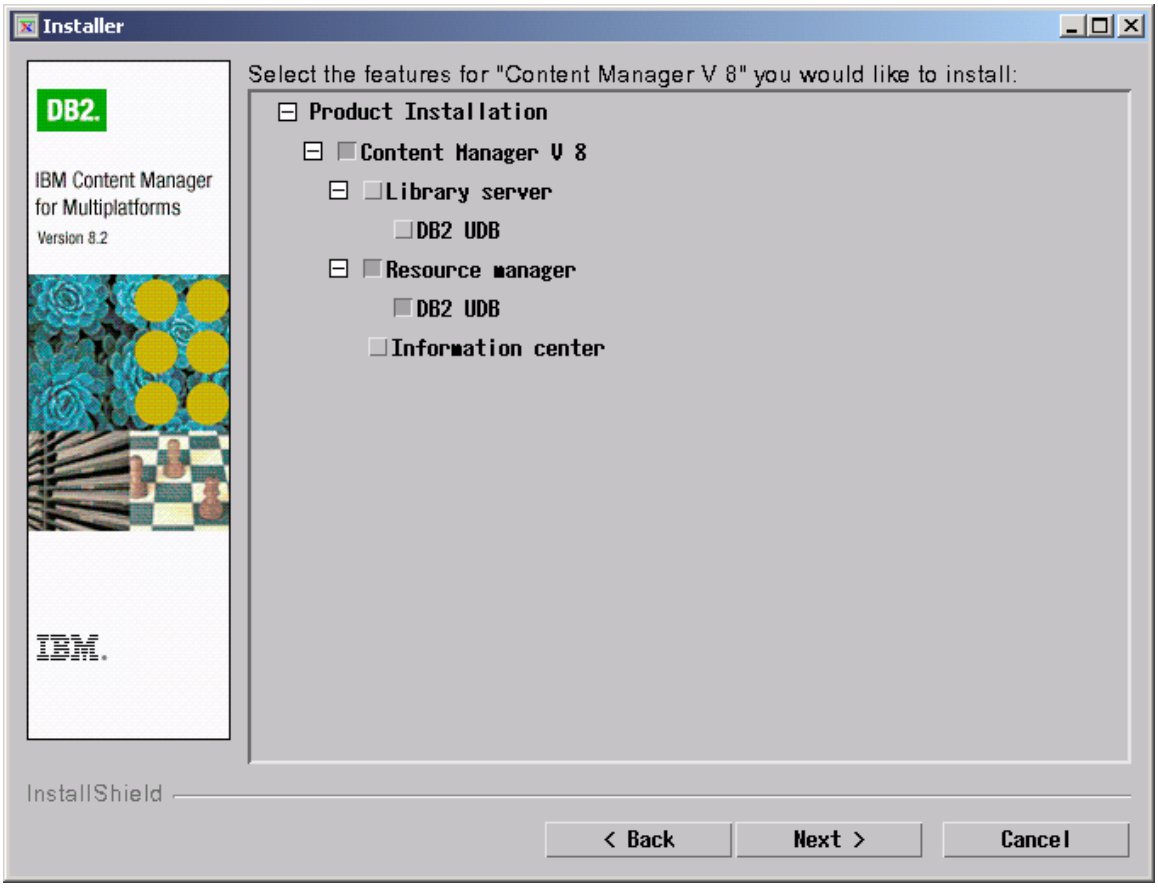


Figure 5-21 CM V8 main installation panel.

Note: When going through the installation steps, use the configuration data we created during the planning session.

6. Next specify the identification and authentication information for the library server.

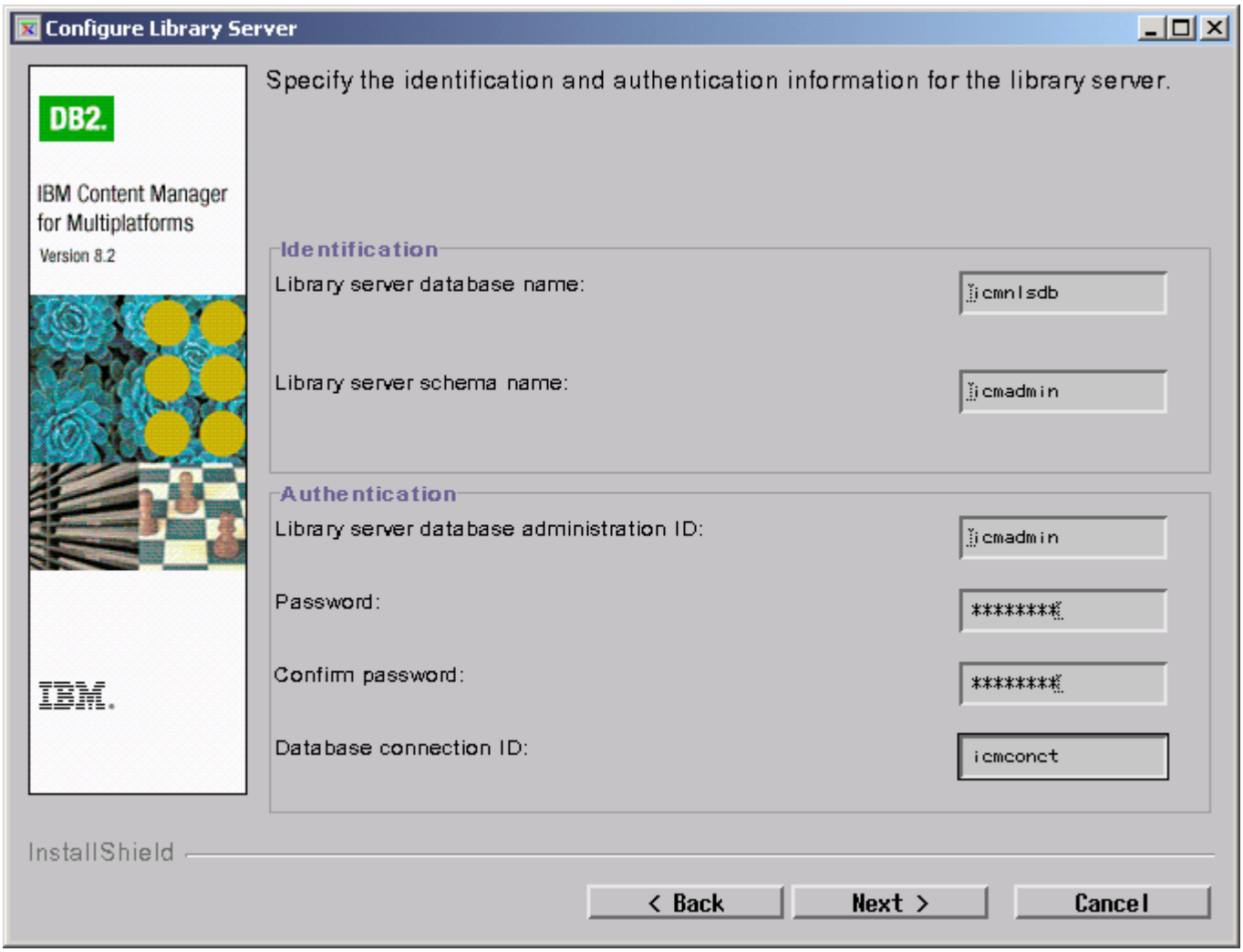


Figure 5-22 CM V8 library server configuration parameters.

7. Finally click next to start the actual installation process.

Upon successful completion, the next step would be to repeat the same library server installation procedure on the second node:

1. failover the instance “db2inst1” from **cldb1** to **cldb2**:

   #scswitch –z -g db2_db2inst1_0-rg –h cldb2

2. logon to node: **cldb2** as root

   Now repeat steps 3-5 on node: cldb2.

3. On the main panel select to install the library server only.
4. Then specify the identification and authentication information for the library server.
5. On next panel use the default settings for the library server options

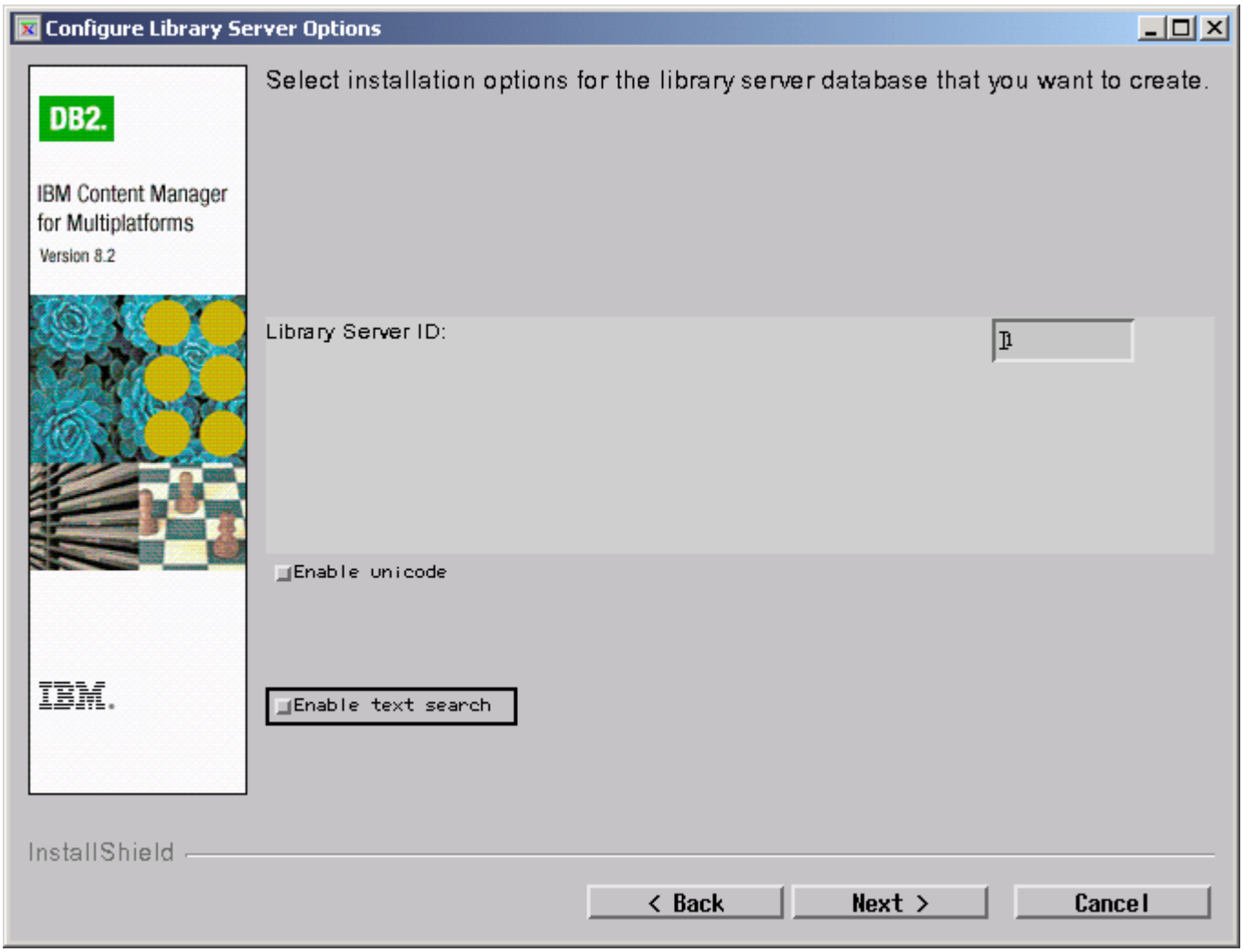


Figure 5-23 CM V8 library server configuration parameters.

6. Then repeat the installation

During the second Content Manager library server installation, when asked, choose **not** to overwrite the existing library server database.

At this point the Content Manager V8 library server is installed and can be tested. Some basic functional tests can be done using the Content Manager System administration client.

5.19.5 Install Content Manager V8 resource manager database

In order to install the Content Manager resource manager database we need to run the Content Manager V8 installation program (setup.exe) again. This time we use the db2inst2 and rmadmin user IDs.
Complete the following steps:

1. Log on to node cldb2 as root.
2. make sure the instance db2inst2 is running on cldb1:
3. `#scswitch -z -g db2_db2inst2_0-rg -h cldb2`
4. Source Content Manager and DB2 environment:
5. `# . ~db2inst2/.profile`
6. Start the Content Manager setup program. (Make sure the DISPLAY is set properly for GUI display.)
7. `# /net/clapp2/stage/source/cm_sun/cm/English/setup.exe &`
8. Select only the resource manager nothing else is necessary.

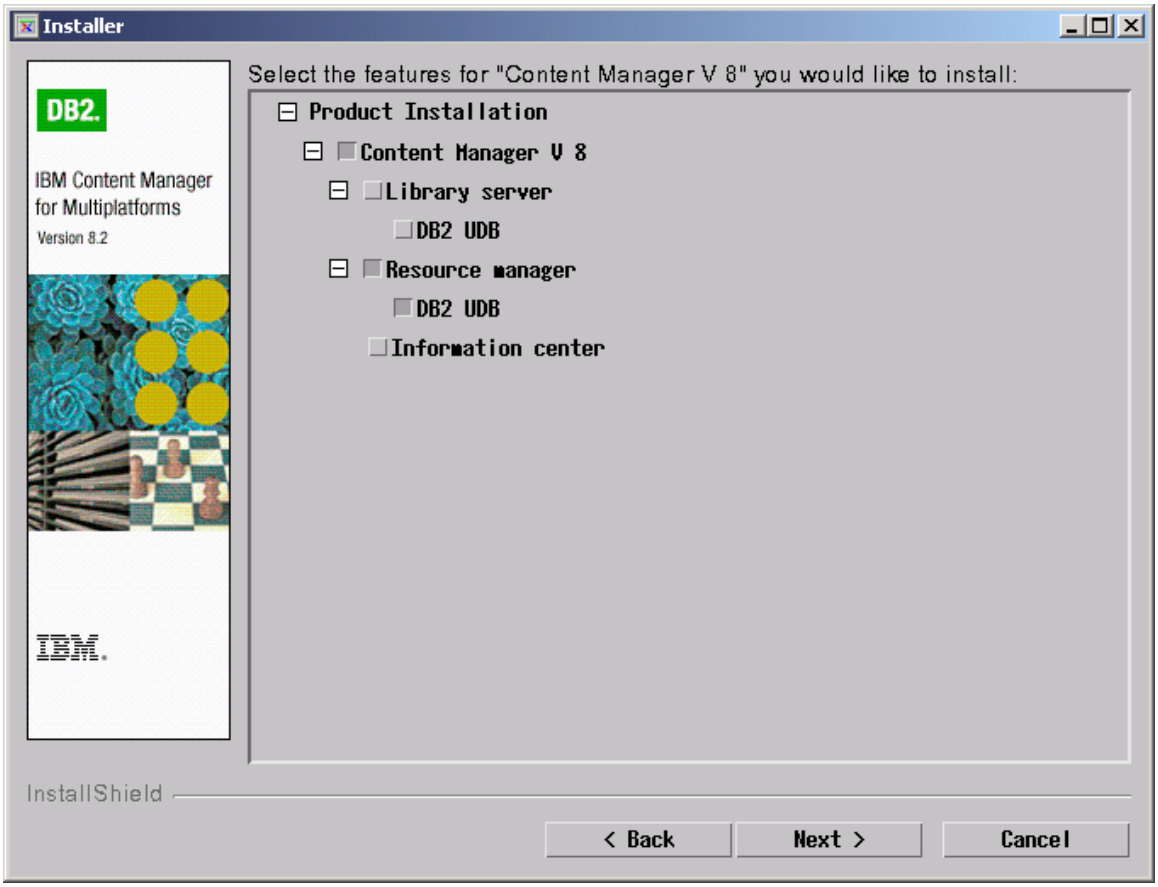


Figure 5-24 CM v8 main installation panel.

9. In the next panel provide the resource manager identification and authentication information

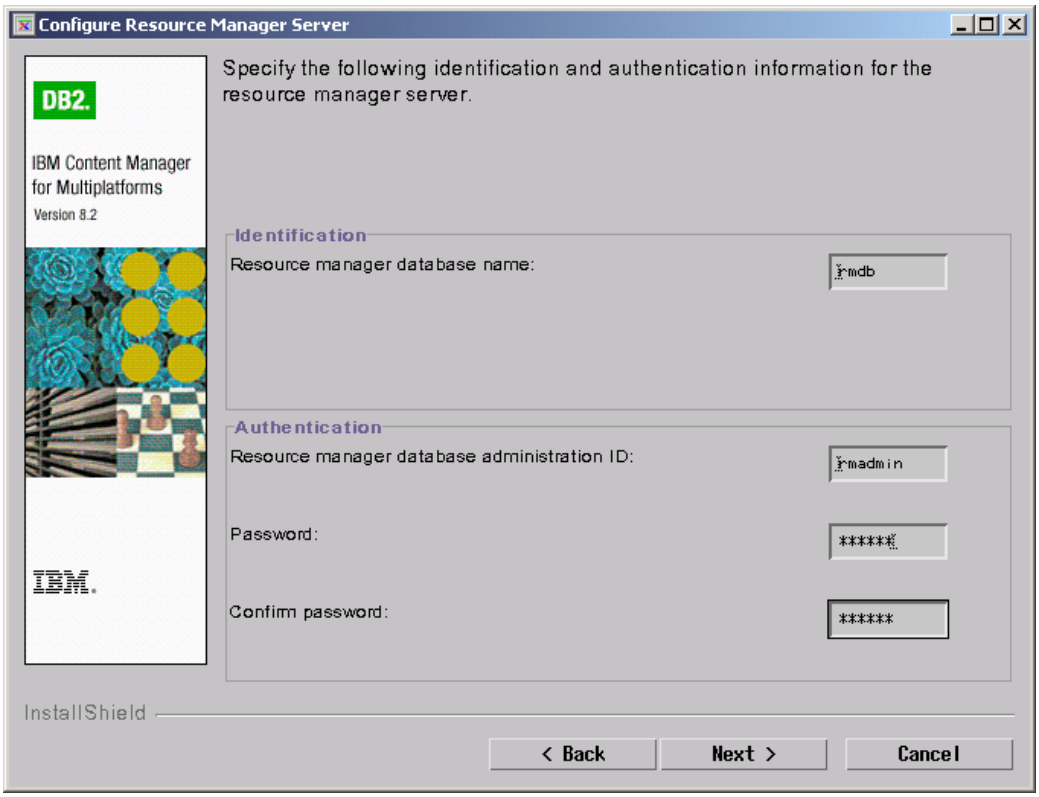


Figure 5-25 CM resource manager identification and authentication information.

10. Followed by the installations options and WebSphere configuration information.

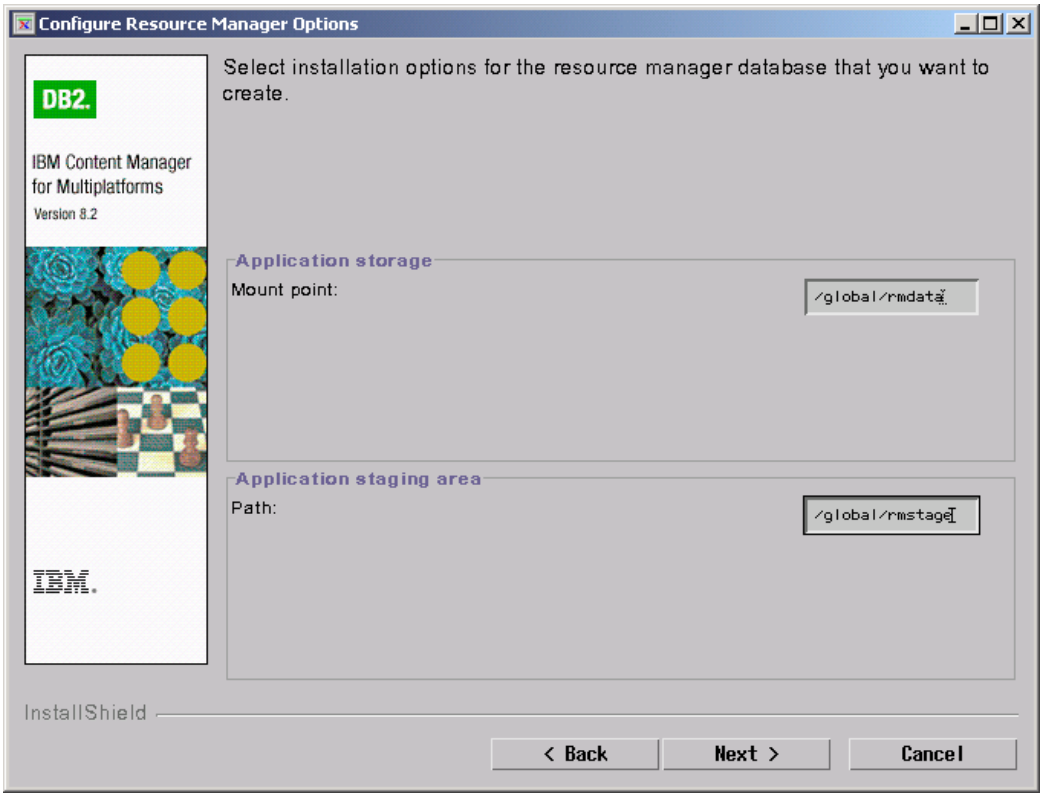


Figure 5-26 CM resource manager installation options..

11. Followed by the installations options and WebSphere configuration information.

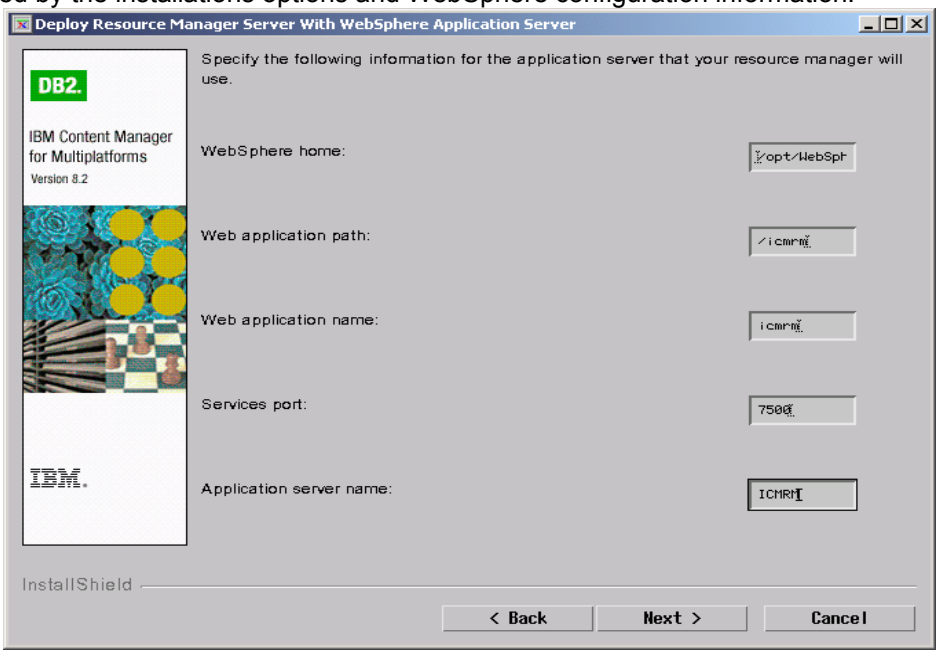


Figure 5-27 WebSphere configuration information

11. Ignore warnings , the resource manager web application will be deployed in a later step.

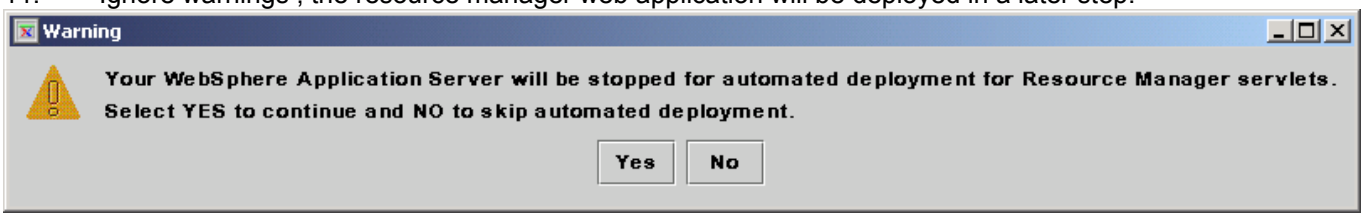


Figure 5-28 Warnings

12. Complete the WebSphere configuration information.

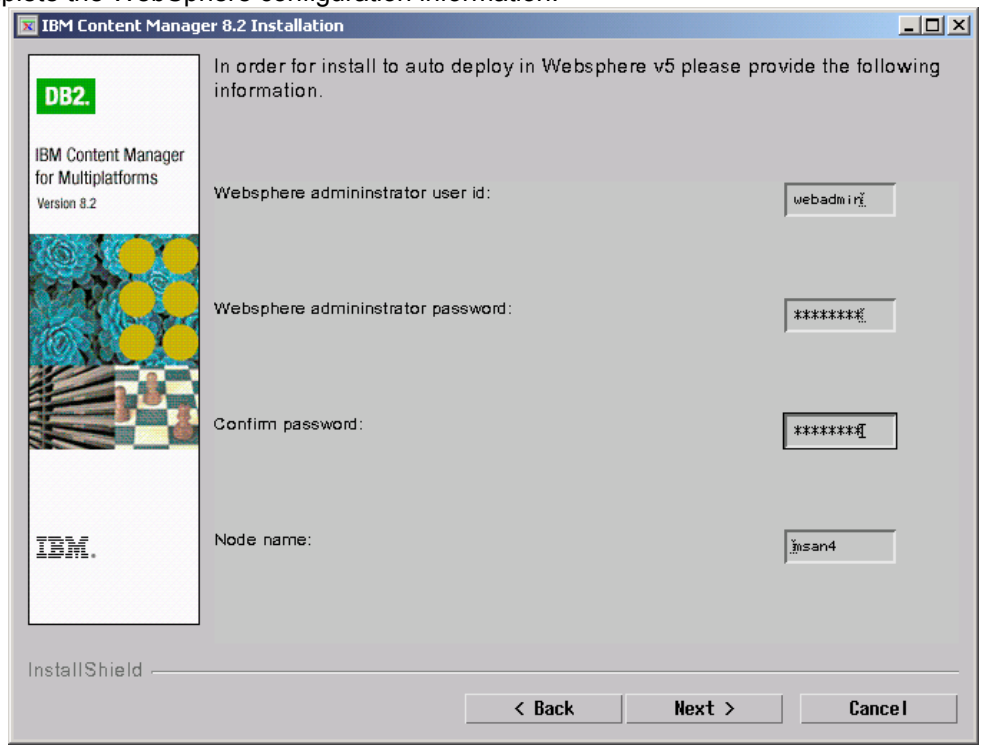


Figure 5-29 WebSphere configuration information

13. Provide the library server connection information.

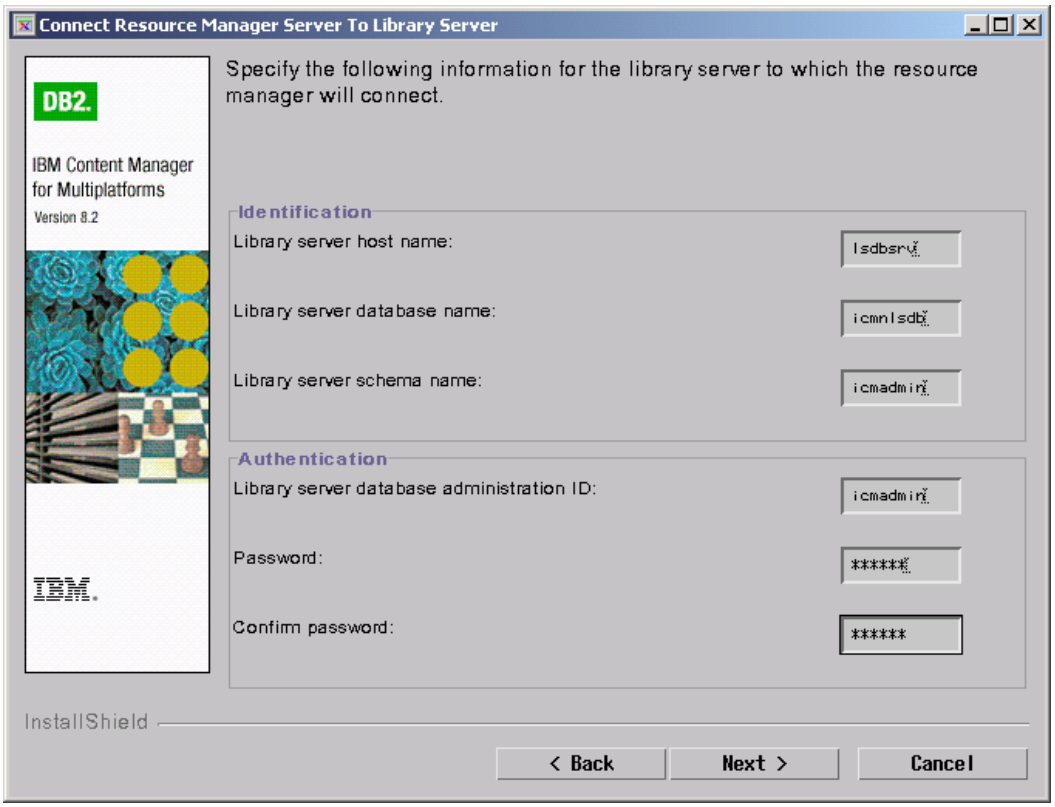


Figure 5-30 Connection information for the resource manager to log onto the library server data base.

13. Complete the LDAP configuration information.

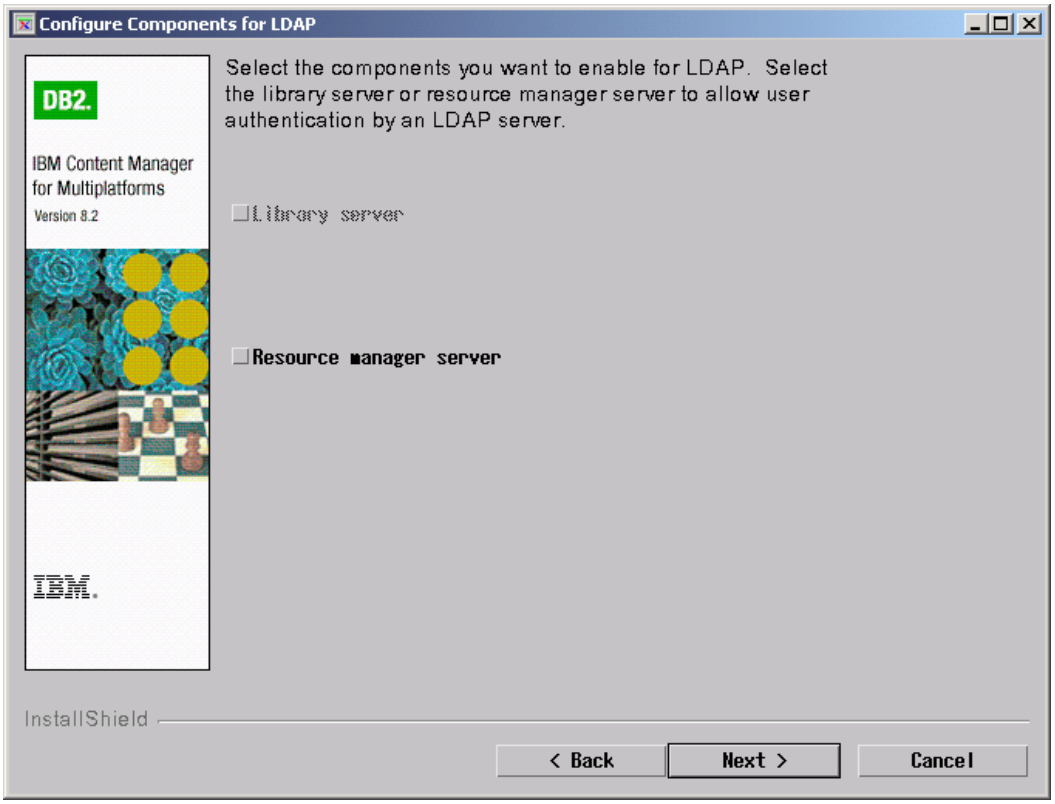


Figure 5-31 Enabling the CM components for LDAP server

14. Start the actual resource manager installation.

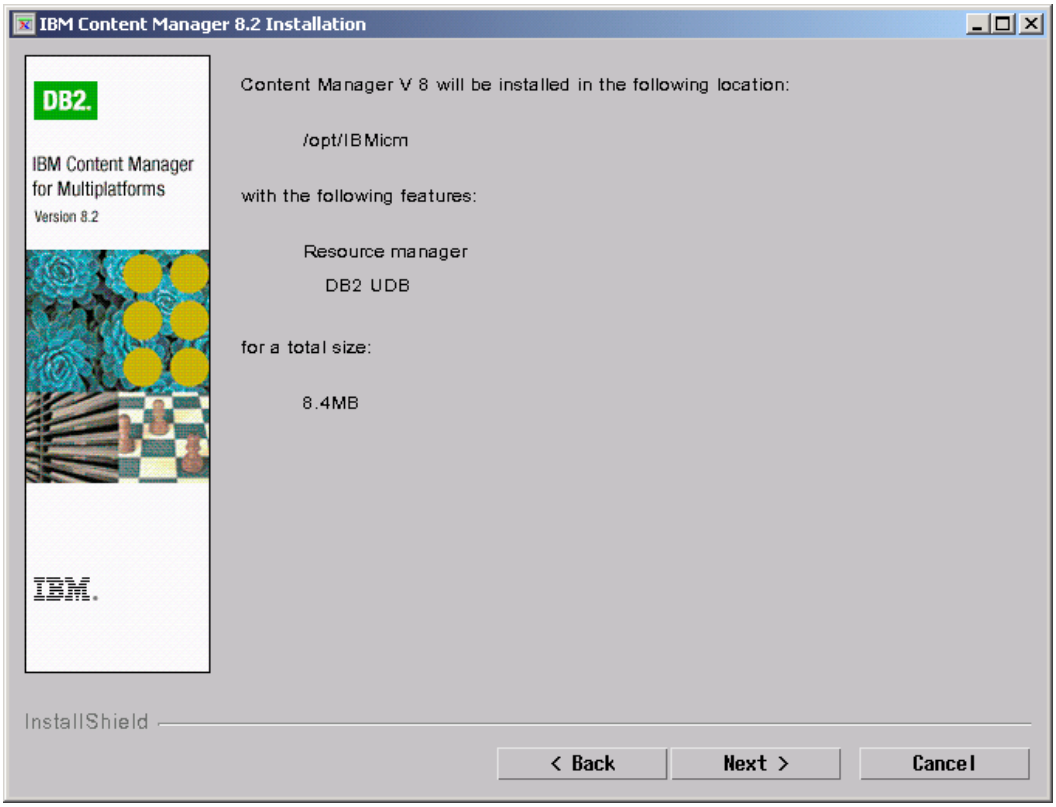


Figure 5-32 Resource manager installation

The automatic deployment of the resource manager application might fail, which is fine. We will deploy the icmrm application in later steps.

Access to the resource manager database is only through the DB2 JDBC client APIs, which will be installed on the WebSphere Application Server nodes. The Content Manager resource manager Web application environment will be configured on the application WebSphere Application Server nodes through WebSphere itself.

The Content Manager installer will unpack the Content Manager resource manager Web application packages in the `/opt/IBMcm/config` directory. Two important files, `icmrm.ear` and `icmrm.war` will be extracted into this directory. These files contain the Content Manager resource manager binaries with respective resources and need to be copied to the application server nodes, which will be executed in one of the coming installation steps.

Note: Another ways to share the CM installation root directory is to provide to the application server nodes global shared access to /`opt/IBMcm/config` directory, which can be mounted as a cluster filesystem.

6. Installing the Content Manager V8 resource manager Web application

6.1 Summary of installation steps

What needs to be done	LS	RMDB	App1	App2	Comment
Install DB2 client API (ADCL)			X	X	We might have to install the DB2 server SW DB2 FP4
Install WebSphere Application Server V5.0.2 and HTTPd			X	X	Use WebSphere Application Server V5.0.2 with fix pack 2 or WebSphere Application Server V5.1 IBM HTTPd 1.3.28 is part of the WebSphere Application Server bundle.
Configure SSL for HTTPd			X	X	The Content Manager system administration client requires SSL to communicate with the resource manager.
Install Content Manager resource manager Web application			X	X	Set up the rmadmin environment. Catalog the resource manager and library server databases. Install the Content Manager resource manager using the resource manager install script. Deploy the Content Manager resource manager Web application into the WebSphere Application Server cell
Set up Content Manager resource manager Web application cluster			X	X	Install WebSphere Application Server Network Deployment V5.0.2. Configure the resource manager cell. Add nodes. Configure the resource manager cluster. Add cluster members.
Test Content Manager high availability installation			X	X	Perform basic function tests against resource manager cluster. Install Content Manager First Steps. Use the Content Manager Windows client and system administration client to test the installation.

Note: The resource manager hostname must use a cluster-managed virtual IP address (VIP) like, such as `rmweb`. During the installation of the resource manager Web application, choose to **not** overwrite the existing resource manager database.

Note: The `rmweb` will eventually be hosted by the NetDispatcher (or a hardware IP sprayer) as the entry point to the Content Manager application. During the high availability Content Manager installation, all reference to this application entry point should be to `rmweb` instead of the IP address directly. At this point (before NetDispatcher is configured), the `rmweb` is resolved to `clapp1` for convenience. To se this up, modify the `/etc/hosts` files on all nodes:

```
10.1.19.103     clapp1   rmweb
```

During NetDispatcher configuration, this entry will be modified to resolve `rmweb` to VIP address hosted by the NetDispatcher:

```
10.1.19.103      clapp1
10.1.19.100      rmweb   # VIP: HA-IP for Resource Manager Web Interface
```

6.2 Installing the Content Manager resource manager cluster

The Content Manager high availability setup requires modifications to the documented Content Manager V8 installation approach. These modifications are necessary in order to accommodate the more complex nature of the Content Manager high availability environment. This is most apparent with the Content Manager resource manager Installation tasks, which have been split into two distinct phases:

1 Resource manager database creation phase
2 Resource manager Web application deployment phase

During phase 1, we installed the Content Manager resource manager database on the nodes CLDB1 and CLDB2 by running the Content Manager installation script twice.

During that process, not only the resource manager database was created, but also all relevant executable files were also unpacked into the <CMROOT> directory. The 2 relevant files, with respect to the resource manager, are the ICMRM EAR and WAR archives, which can be found in the `config` subdirectory. These files contain the actual Content Manager resource manager Web application binary code and respective runtime resources.

During this second phase of the Content Manager resource manager installation, we must now set up the WebSphere Application Server Web application server cluster and then deploy the resource manager Web application into it.

6.2.1 Prepare <CMROOT>

There are two ways to expanded <CMROOT> (`/opt/IBMicm`) on the system hosting the WebSphere Application Server application server:

1. **Manual approach:** Copy the entire <CMROOT> directory tree from one of the database nodes like: CLDB1 to CLAPP1 and CLAPP2.
2. **Automated approach:** On the WebSphere Application Server nodes, re-execute the Content Manager installation program (`setup.exe`) to create the directory structure and have the ear and war files unpacked in the `config` directory.

   Prerequisite: The DB2 engine packages must be installed on the host.

   Restriction: The installation program will fail at the step to configure the remote resource manager database. As a result, it also prevents the automatic deployment of the resource manager application (`icmrm`). However, the <CMROOT> directory will still be populated with the expanded binaries.

After <CMROOT> is populated, the next step is to use the Java command line tool to deploy the Content Manager resource manager Web application: `<CMROOT>\config\WAS50DeployRM.class`

We used the automated approach followed with `WAS50DeployRM.class`. See the series of screen shots in Install CM V8 Resource Manager DB starting on page 87. The installation will terminate with the following error message when configuring the resource manager database. This error can be ignored because the resource manager database has already been configured.

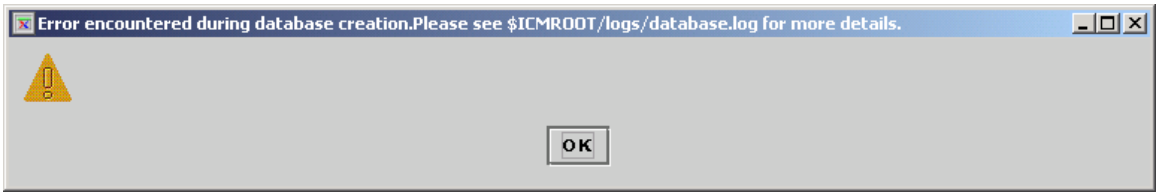


Figure 6-1 OK

From the ease of operation point of view, the manual approach would be a better choice.

The WebSphere Application Server must be installed before the resource manager Web application can be deployed.

6.2.2 Install WebSphere Application Server V5 with fix pack 2

To install WebSphere Application Server V5 with fix pack 2:

1. Log on to CLAPP1.
2. Install WebSphere Application Server V5, with the IBM HTTP Server V1.3.28. The applications are bundled together. For the initial installation, accept all defaults settings.
3. Configure SSL for the HTTP server and enable https. (This step is required for the Content Manager system administration client to be able to manage the Content Manager resource manager.)
4. Perform some simple functional tests to check that both the HTTP server and WebSphere Application Server V5 are working properly. For example:
    a. Log on to the WebSphere Application Server V5 administration console by pointing the browser to: `http://cmha3:9090/admin/`. Make sure that the administration console works normally.
    b. Start the default application server, `server1`, and issue a HTTP request against the snoop servlet: http://cmha3/snoop/ and https://clapp1:9090/admin/. Make sure everything works normally.
5. Repeat steps 1-4 on the second node, CLAPP2.

At this point we are ready to deploy the Content Manager resource manager Web application.

6.2.3 Preliminary configuration steps

Perform the following configuration steps:

1. Configure the shared global resource manager data storage area (LBOS) and create a soft partition on `rmstg` for `/global/rmdata`.

2. Execute the following command:

```
metainit -s rmstg d20 -p d0 20g
newfs /dev/md/rmstg/rdsk/d20
```

3. Modify `/etc/vfstab` on nodes CLAPP1 and CLAPP2:

```
#
/dev/md/rmdata/dsk/d10 /dev/md/rmdata/rdsk/d10 /global/rmdata ufs 2 yes global,logging
/dev/md/rmstg/dsk/d20 /dev/md/rmstg/rdsk/d20 /global/rmdata ufs 2 yes global,logging
# df
/               (/dev/dsk/c0t0d0s0 ):31343274 blocks  3465250 files
/proc           (/proc             ):       0 blocks    29948 files
/etc/mnttab     (mnttab            ):       0 blocks        0 files
/dev/fd         (fd                ):       0 blocks        0 files
/var/run        (swap              ):31308576 blocks  1600047 files
/tmp            (swap              ):31308576 blocks  1600047 files
/global/.devices/node@2(/dev/did/dsk/d4s4 ):  976248 blocks   242465 files
/global/.devices/node@1(/dev/did/dsk/d1s4 ):  976248 blocks   242465 files
/global/rmstage    (/dev/md/rmstg/dsk/d10):41263174 blocks  2489939 files
/global/rmdata     (/dev/md/rmstg/dsk/d20):41266798 blocks  2489979 files
```

4. Log on to the node CLDB1 and stop the IBM HTTP server by entering the following command:

```
/opt/IBMHttpServer/bin/apachectl stop
```

5. Create and configure a DB2 client instance (db2clnt) on each node. Repeat the following steps on both CLAPP1 and CLAPP2. (The home directory of db2clnt can be created on the global file system `/global/home`. If so, the following configuration steps only need to be done once on either CLAPP1 or CLAPP2.)

```
# useradd -u 105 -d /export/home/db2clnt -s /bin/ksh -c "DB2 Client" -m db2clnt
64 blocks

# grep db2 /etc/passwd

db2clnt:x:105:1:DB2 Client:/export/home/db2clnt:/bin/ksh
# /opt/IBM/db2/V8.1/instance/db2icrt -s client db2clnt
DBI1070I Program db2icrt completed successfully.
```

6. Log in as db2clnt and catalog the library server and resource manager databases on both nodes. Repeat the following steps on both CLAPP1 and CLAPP2. (The home directory of db2clnt can be created on the global file system `/global/home`. If so, the following configuration steps only need to be done once on either CLAPP1 or CLAPP2.) To create the catalogs, you can create a DB2 script, catall.sql, and execute it or manually perform the following steps on a command line. Below is the contents of the catall.sql script

```
$ more catall.sql
catalog tcpip node lsdbsrv remote lsdbsrv server db2inst1;
catalog database icmnlsdb as icmnlsdb at node lsdbsrv;
catalog tcpip node rmdbsrv remote rmdbsrv server db2inst2;
catalog database rmdb as rmdb at node rmdbsrv;

$ db2 -tvf catall.sql
```

7. Execute the following command:

```
$ db2 catalog tcpip node lsdbsrv remote lsdbsrv server db2inst1
You should see a message like this one:
DB20000I  The CATALOG TCPIP NODE command completed successfully.
DB21056W  Directory changes may not be effective until the directory cache is refreshed.
```

8. Execute the following command:

```
$ db2 catalog database icmnlsdb as icmnlsdb at node lsdbsrv
You should see a message like this one:
DB20000I  The CATALOG TCPIP NODE command completed successfully.
DB21056W  Directory changes may not be effective until the directory cache is refreshed.
```

9. Execute the following command: `$ db2 list db directory`

```
 System Database Directory
 Number of entries in the directory = 1
Database 1 entry:
 Database alias                 = ICMNLSDB
 Database name                   = ICMNLSDB
 Node name                      = LSDBSRV
 Database release level           = a.00
 Comment                        =
 Directory entry type             = Remote
 Catalog database partition number    = -1
```

10. Now test the connection:

```
$ db2 connect to icmnlsdb user icmadmin using ibmdb2
 Database Connection Information
 Database server        = DB2/SUN 8.1.4
 SQL aucldb1ization ID   = ICMADMIN
 Local database alias   = ICMNLSDB
```

11. Repeat the same procedure for the resource manager database:

```
$ db2 catalog tcpip node rmdbsrv remote rmdbsrv server db2inst2
DB20000I  The CATALOG TCPIP NODE command completed successfully.
DB21056W  Directory changes may not be effective until the directory cache is
refreshed.

$ db2 catalog database rmdb as rmdb at node rmdbsrv
DB20000I  The CATALOG DATABASE command completed successfully.
DB21056W  Directory changes may not be effective until the directory cache is
refreshed.
```

12. Check that it is properly configured:

```
$ db2 list db directory
 System Database Directory
 Number of entries in the directory = 2
 Database 1 entry:
 Database alias                 = ICMNLSDB
 Database name                  = ICMNLSDB
 Node name                      = LSDBSRV
 Database release level         = a.00
 Comment                        =
 Directory entry type           = Remote
 Catalog database partition number = -1

 Database 2 entry:
 Database alias                 = RMDB
 Database name                  = RMDB
 Node name                      = RMDBSRV
 Database release level         = a.00
 Comment                        =
 Directory entry type           = Remote
 Catalog database partition number = -1
```

13. Now test the connection:

```
$ db2 connect to rmdb user rmadmin using ibmdb2

 Database Connection Information
 Database server        = DB2/SUN 8.1.4
 SQL aucldb1ization ID  = RMADMIN
 Local database alias   = RMSDB
```

6.2.4 Deploy the Content Manager resource manager Web Application

To deploy the resource manager Web application, use the following procedure:

1. Prepare <CMROOT> as discussed in Prep <CMROO>: /opt/IBMicm on page 95.

2. We are deploying the resource manager Web application into the unmanaged WebSphere Application Server cell first, so start the WebSphere default administration server, server1.

3. Execute the following command:

```
# /opt/WebSphere/AppServer/bin/startServer.sh server1
```

4. As root, change to the install image directory and source the proper environment.

```
# . ~db2inst2/.profile
```

5. Make sure that the ICM_LS_HOSTNAME environment variable is set properly.

```
>grep lsdbsrv * CMcfg.params:
ICM_LS_HOSTNAME=lsdbsrv
```

Execute the `WAS50DeployRM.class` Java program to deploy the resource manager (icmrm). Make sure to type in the path for DB2 JDBC driver as the following:

```
Enter the absolute path of the DB2 JDBC Driver :[Example : /opt/IBM/db2/V8.1/java/db2java.zip]:
/opt/IBM/db2/V8.1/java/db2java.zip
```

The prompted value is an “Example” not the “Default”, so just clicking the “Enter” key will not assign the value. If the value wasn’t assigned during this manual deployment, the DB2 JDBC driver variable can still be assigned through WAS Admin Console.

```
# cd /opt/IBMicm/config
```

and run the resource manager command line deployment tool by using the following command:

```
# java WAS50DeployRM
```

```
This utility facilitates deployment of RM for
WebSphere Application Server [Ver 5] only

Licensed Materials - Property of IBM IBM Content Manager for Multiplatforms V8.2 (program number 5724-B19) (c )
 Copyright IBM Corp. 1994, 2003.  All Rights Reserved. US Government Users Restricted Rights - Use, duplication
 or disclosure restricted by GSA ADP Schedule Contract with IBM Corporation
Enter the name for each item or enter a null (no entry) for the default name.

What procedure would you like to perform ?
 1. Install
 2. Uninstall
 3. Exit
[Example : 1]:
1
1. Install

Enter the IBM WebSphere Application Server Home :[Default: /opt/WebSphere/AppServer]:

/opt/WebSphere/AppServer
Enter the WebSphere Application Server Node name :[Default: clapp1]:

clapp1
Enter the desired Application Server name :[Default: icmrm]:
icmrm
Enter the desired Enterprise Application name :[Default: icmrm]:
icmrm
Enter the desired Context root for the web module:[Default: /icmrm]:

/icmrm
Enter the WAS Administrative Server username :[Default : ]:
webadmin
webadmin
Enter the WAS Administrative Server password :[Default : ]:
password
Enter the location of the RM icmrm.ear and icmrm.war files :[Default : /opt/IBMicm/config]:

/opt/IBMicm/config
Enter the Output Directory of the RM icmrm.ear and icmrm.war files :[Default : /tmp]:

/tmp
Enter the absolute path of the RM icmrm.jacl file :[Default : /opt/IBMicm/config/icmrm.jacl]:

/opt/IBMicm/config/icmrm.jacl
Enter the database name of the resource manager
 (the default is "RMDB"):
RMDB
Enter the user ID of the resource manager
 (the default is "RMADMIN"):
RMADMIN
Enter the password of the resource manager
 (the default is "password"): ibmdb2
Enter the RM Database type.
 Enter 1 for DB2, or 2 for Oracle :[Default: 1]
DB2
Enter the fully qualified host name of the resource manager  rmweb
rmweb
Enter the absolute path of the DB2 JDBC Driver :[Example : /opt/IBM/db2/V8.1/java/db2java.zip]:
/opt/IBM/db2/V8.1/java/db2java.zip
/opt/IBM/db2/V8.1/java/db2java.zip

The following are input parameters specified:

WebSphere Application Server Home : /opt/WebSphere/AppServer
WebSphere Application Server Node name : clapp1
RM Application Server name : icmrm
RM Enterprise Application name : icmrm
RM Application Context root : /icmrm
WAS Administrative Server username : webadmin
RM icmrm.ear and icmrm.war file location : /opt/IBMicm/config
RM icmrm.ear and icmrm.war files output directory: /tmp
Absolute path of the RM icmrm.jacl file : /opt/IBMicm/config/icmrm.jacl
Resource manager database name: RMDB
Resource manager database user ID: RMADMIN
RM Database type : DB2
Resource manager host name:: rmweb
DB2 JDBC Driver path : /opt/IBM/db2/V8.1/java/db2java.zip
```

```
Please ensure that the WebSphere administrative server is running.
Please review the input parameters specified above
(Enter 1 to Continue, Enter anything else to start over again): 1
./icmrmcfg5.sh 'RMDB' 'RMADMIN' ******** 'DB2' 'rmweb' " " '/opt/IBMicm/config' '/opt/IBMicm/config/icmrm.jac
l' install '/opt/WebSphere/AppServer' 'webadmin' ******** 'clapp1' 'icmrm' 'icmrm' '/icmrm' '/opt/WebSphere/AppS
erver/installedApps' '/opt/IBM/db2/V8.1/java/db2java.zip' " '/tmp' 'disabled' 'com.sun.jndi.ldap.LdapCtxFactor
y' 'STANDARD_LDAP' 'ldap\://kozina.stl.ibm.com' 'ignore' 'none' 'cn=root' 'password' 'o=IBM,c=US' 'SUBTREE_SCOP
E' 'cn' " '389' 'DN' '5000' '15'
RMACTION : install
Copying war and ear file
-rwxrwxrwx   1 root     other    2611392 Dec  3 16:31 /tmp/icmrm.ear
-rwxrwxrwx   1 root     other    2660202 Dec  3 16:31 /tmp/icmrm.war
Updating Context Root
UpdatingWarFile
WEB-INF/classes/com/ibm/mm/icmrm/ICMRM.properties
UpdatingloggingFile
UpdatingLdapPropFile
is 18
updateProps
storeToFile
UpdatingEarFile
successfully built EAR file in /tmp Directory
WASX7023E: Error creating "SOAP" connection to host "localhost"; exception information: com.ibm.WebSphere.manag
ement.exception.ConnectorNotAvailableException
WASX7213I: This scripting client is not connected to a server process; please refer to the log file /opt/WebSph
ere/AppServer/logs/wsadmin.traceout for additional information.

main: Args passed in and processed:
main:   action        =  install
main:   wasHome       =  /opt/WebSphere/AppServer
main:   nodeName      =  clapp1
main:   serverName    =  icmrm
main:   appName       =  icmrm
main:   earLocation   =  /tmp/icmrm.ear
main:   installDir    =  /opt/WebSphere/AppServer/installedApps
       real location used /opt/WebSphere/AppServer/installedApps/clapp1
main:   db2JdbcDriver  =  /opt/IBM/db2/V8.1/java/db2java.zip
main:   oraJdbcDriver  =

main: invoking validateNode.
WASX7017E: Exception received while running file "/opt/IBMicm/config/icmrm.jacl"; exception information: com.ib
m.bsf.BSFException: error while eval'ing Jacl expression: com.ibm.ws.scripting.ScriptingException: WASX7070E: T
he configuration service is not available.
You must restart the  WebSphere administrative server and IBM HTTP Server before using Resource Manager.
```

6.2.5 Test the installed resource manager Web application

Perform some simple functional tests to check that both the HTTP server and WebSphere Application Server V5 are working properly. For example:

1. Restart the IBM HTTP Server
   # . /opt/IBMHttpServer/bin/apachectl restart
2. Update the Resource Manager passwords
   Edit the following file: /opt/WebSphere/AppServer/installedApps/(node)/icmrm.ear/icmrm.war/WEB-INF/classes/com/ibm/mm/icmrm/ICMRM.properties
   Set the line DBPassword to clear text. For example DBPassword=password
   Connect to the RMDB
   `Issue the following db2 command: db2 update rmaccess set acc_password='password'`
3. Log on to the WebSphere Application Server V5 console pointing the browser to: http://clapp1:9090/admin/ and test WebSphere Application Server administration.
4. Start the default application server, icmrm, and issue a HTTP request against the snoop servlet:
   http://clapp1/icmrm/snoop/
   http://clapp1/icmrm/ICMResourceManager?order=heartbeat&libname=ICMNLSDB
   https://clapp1/icmrm/ICMResourceManager?order=heartbeat&libname=ICMNLSDB
5. The following page should be returned to your browser.

```
IBM Content Manager V8
Your request: heartbeat
Return Code: 0
Message text: ICM0000: OK
```

At this point the Content Manager resource manager Web application is deployed and configured. Using the administration console, verify that a new Web application server called icmrm has been created and that this server is associated with the icmrm.war Web container. We will need both later as a template for our resource manager cluster members (also called *clones*).

6.2.6 Installing Content Manager V8.2 fix pack 3

To install fix pack 3, complete the following steps:

First apply the fix pack for the Content Manager library server on CLDB1.

1. Go to the primary Content Manager database node (CL1) and halt the entire database cluster.
2. Stop the library server monitor:

```
/etc/rc.cmlsproc –shutdown
```

3. Verify that ~db2inst1/sqllib/db2nodes.cfg points to CLDB1 and then manually restart the library server database.
4. As root, source the icmadmin user profile:

```
# . ~icmadmin/.profile
```

5. Run the fix pack installation program. Provide information for the Library Server as Well as the Resource Manager, marking the Resource Manager database as local. Ignore any Resource Manager error messages.

```
./updateCM_SUN_ENU –W wasRunscript.active=false –W  rmconnchkse1.active=false
–W exitSequence.active  –W  uninstallYourself.active=false
```

6. This will unpack the fix pack binaries and the data that will be used in the next sequence of steps .
7. Update the input file `updateCM.cfg`, which contains the parameter for the installer. Set:

```
LSDB_COUNT=1
RMDB_COUNT=0
RM_AS_COUNT=0
```

8. Update the Content Manager library server:

```
./updateProduct –d –c /opt/IBMicm/Fixpack03/updateCM.cfg –l /opt/IBMicm/Fixpack03/config_rm.log
```

9. Verify that the Content Manager library server database was updated:

```
db2 select LSCURRENTVERSION from ICMSTSYSCONTROL
```

Apply the fix pack to the Content Manager resource manager.

1. Tar entire /opt/IBMicm directory on CLDB1 and extract it to the same location on CLAPP1.
2. Go to the primary application server node (App1) and bring the resource manager database server up under HACMP cluster control.
3. By default, the rmadmin user will not exist on the WebSphere Application Server cluster nodes. You must add it. You should use the same user and group ID that are used on the library server and resource manager cluster nodes. The rmadmin user should source the db2clnt profile by adding the following line to ~rmadmin/.profile:

```
. ~db2clnt/.profile
```

4. Update the input file `updateCM.cfg`, which contains the parameter for the installer. Set:

```
LSDB_COUNT=0
RMDB_COUNT=1
RM_AS_COUNT=1
```

`Verify that AS_NODE_001 exists. If not, the Resource Manager settings were not specified during the initial updateCM_SUN_ENU step.`

Start the WebSphere Application Server server1:

/opt/WebSphere/AppServer/bin/StartServer.sh server1

5. Update the Content Manager resource manager:

```
./updateProduct –d –c /opt/IBMicm/Fixpack03/updateCM.cfg –l /opt/IBMicm/Fixpack03/config_rm.log
```

6. Verify that the Content Manager resource manager database was updated:

```
db2 select * from RMVERSION
```

Restart the entire WebSphere Application Server cluster and test the system.

6.2.7 Installing Content Manager V8.2 fix pack 3 in a pre-existing high availability environment

This section applies only to pre-existing high availability configurations that have already cloned the icmrm application.

It is much easier to install a fixpack before cloning the icmrm application in Websphere Network Deployment. Although it is possible to update a cloned application, the process can be difficult and is error prone. For this reason, we recommend uninstalling any existing clones, applying the fixpack, and then regenerating the clones.

DO NOT delete the existing RMDB, /global/rmdata/, or /global/rmstaging or you may lose data.
To remove the clones:

1. Remove CLAPP1 and CLAPP2 from the cluster. From CLAPP1 and CLAPP2, excecute:

   /opt/WebSphere/AppServer/bin/removeNode.sh

2. Delete the icmrm application from the Websphere Application Server Network Deployment Manager:
   Under **Applications > Enterprise Applications**, select icmrm and click uninstall.
   Save changes and regenerate the plugin.
3. Restart the Websphere Application Server Network Deployment manager:

   /opt/WebSphere/DeploymentManager/bin/stopManager.sh
   /opt/WebSphere/DeploymentManager/bin/startManager.sh

4. Start Server1 on CLAPP1:
   /opt/WebSphere/AppServer/bin/startServer.sh server1

5. Remove the icmrm application:
   Under **Applications > Enterprise Applications**, select icmrm and click uninstall.
   Save changes and regenerate the plug-in.
6. Restart server1.
7. Proceed as normal at section 6.2.4 Deploy the Content Manager resource manager Web Application

6.2.8 Deploying the icmrm application on CLAPP2

Note: We do not have to repeat the deployment of the resource manager Web application using the installation program on this node, we will use the WebSphere Application Server Network Deployment Manager to do so after we have configured the resource manager cluster.

On the second node, we must only setup the DB2 runtime environment for the resource manager application to be able to connect to the library server and resource manager database.

To configure the second node:

1. Add user db2clnt to the other application server.
2. Create a soft link, (ln –s), so the directory `/export/home/db2clnt` points to `/global/home/db2clnt`
   Alternatively, update `/export/home` settings files to point to `/global/home`:

```
~/sqllib/db2cshrc: setenv INSTHOME /export/home/db2clnt
~/sqllib/db2profile: INSTHOME=/export/home/db2clnt
~/.profile
# The following three lines have been added by UDB DB2.
if [ -f /export/home/db2clnt/sqllib/db2profile ]; then
   . /export/home/db2clnt/sqllib/db2profile
fi
```

3. Next, update the DB2 directory entry by adding the catalog information that db2clnt user ID will need to connect to the library server and resource manager database .
   We created a short script for this:

```
$ more catall.sql
catalog tcpip          node lsdbsrv remote lsdbsrv server db2inst1;
catalog database       icmnlsdb as icmnlsdb at node lsdbsrv;
catalog tcpip          node rmdbsrv remote rmdbsrv server db2inst2;
catalog database       rmdb as rmdb at node rmdbsrv;

$ db2 -tvf catall.sql

$ db2 connect to rmdb user rmadmin using ibmdb2
 Database Connection Information
 Database server        = DB2/SUN 8.1.4
 SQL aucldb1ization ID  = RMADMIN
 Local database alias   = RMDB
```

4. If CLAPP1 and CLAPP2 do not share the same global /opt/IBMicm directory, tar the entire /opt/IBMicm directory from CLAPP1 and extract it to the same path on CLAPP2.

6.3 Installing WebSphere Network Deployment

Beginning with WebSphere Application Server V5, the notion of unmanaged and managed WebSphere Application Server cells was introduced.

An unmanaged WebSphere Application Server V5 Cell is a basic, single node WebSphere Application Server V5 installation that is part of the default configuration and managed by the default WebSphere Application Server V5 Administrative Server, server1.

This default administrative server, is limited in its capabilities to administer a WebSphere Application Server realm containing multiple Web application servers and respective Web containers.

A managed WebSphere Application Server V5 Cell is a complex, possibly distributed, heterogeneous, clustered WebSphere Application Server V5 realm that is managed by the WebSphere Application Server Network Deployment Administrative Server, dmgr. A managed node is a node that was added to a WebSphere Application Server Network Deployment managed cell. Only with a WebSphere Application Server Network Deployment administrative server one can fully manage a WebSphere Application Server cell.

On managed nodes, the default administration server (server1), is disabled as soon as a specific node is added (using the `addNode` command) to a administrated WebSphere Application Server cell.

The WebSphere Application Server V5 Deployment Manager is for all practical purposes the administrative instance of a WebSphere Application Server V5 cell. For this reason, a Content Manager high availability cluster needs to have a fully functional WebSphere Application Server Network Deployment V5 administrative realm. A multi-node Content Manager resource manager configuration can not be fully managed by a default WebSphere Application Server V5 administrative console. The last statement is also true for a typical Content Manager configuration, where the Content Manager resource manager and the Content Manager eClient are deployed into the same WebSphere Application Server cell.

6.3.1 Installing WebSphere Application Server Network Deployment V5

Perform a normal WebSphere Application Server Network Deployment V5 installation. After completing the installation, apply the WebSphere Application Server Network Deployment V5.0.2 update. See Installation of WebSphere Application Server Network Deployment V5.0 with fix pack 2 on page 161.

During the WebSphere Application Server Network Deployment installation you have to specify the cell that the WebSphere Application Server Network Deployment server will administer. We chose RMCELL as the cell name.

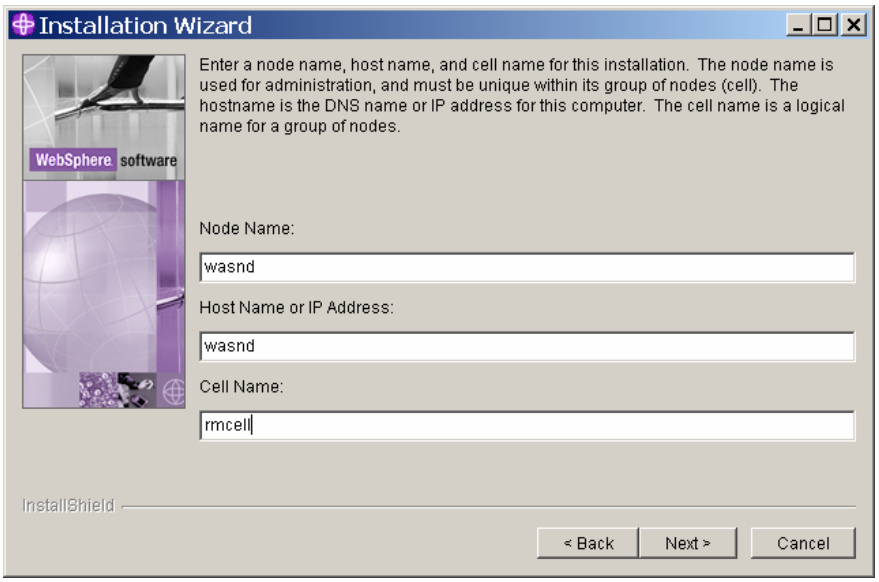


Figure 6-2. WebSphere Application Server Network Deployment installation and cell creation

Note: If you are using WebSphere Application Server V5.1, you can skip the update step. For a production system, consider making the WebSphere Application Server Network Deployment administration server itself highly available. In our scenario we installed the Network Deployment manager on one of the cluster nodes, CLAPP1.

After the WebSphere Application Server Network DeploymentV5 binaries are installed :

1. Change the WebSphere Application Server V5 administrative ports. This will allow the use of both application servers, server1 and dplmgr, if necessary. The latter will be used to manage the WebSphere Application Server V5 cell which we will create. Set the Network Deployment ports to: 9090, and 9043 (secure). Server1 ports are changed to: 9190, and 9143 (secure)
2. Start the deployment manager process on the WebSphere Application Server Network Deployment node

```
> ./startManager.sh
ADMU0116I: Tool information is being logged in file
/opt/WebSphere/DeploymentManager/logs/dmgr/startServer.log
ADMU3100I: Reading configuration for server: dmgr
ADMU3200I: Server launched. Waiting for initialization status.
ADMU3000I: Server dmgr open for e-business; process id is 13729
```

3. Create the resource manager cell RMCELL using the WebSphere Application Server administrative console. Select **Main > System Administration > Cell** and expand the local topology pane.
4. Verify with the administrative console: `http://clapp1:9090/admin`

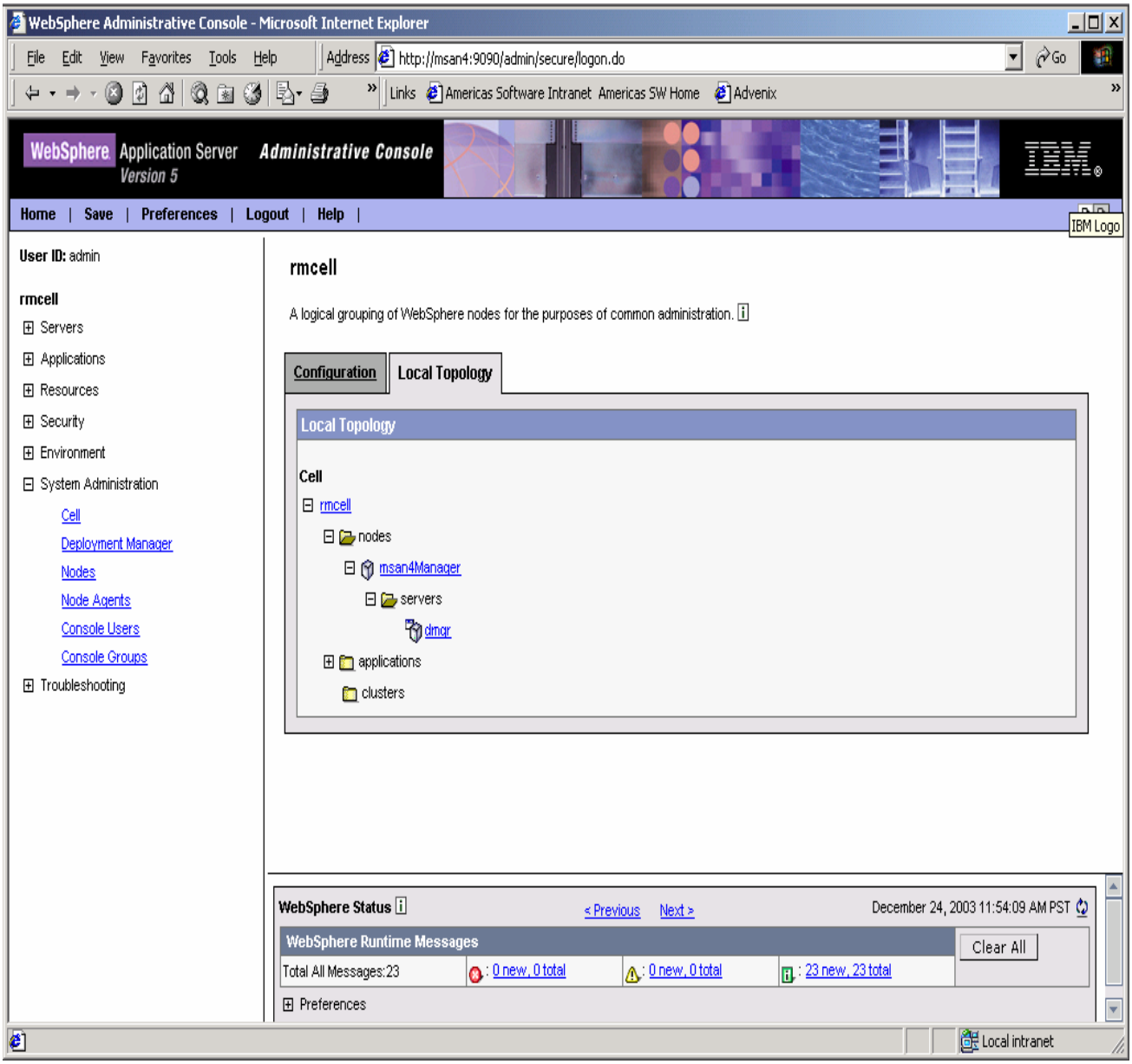


Figure 6-3. WebSphere Application Server Network Deployment Console

Now the two designated WebSphere nodes must be added to the cell. To do this, logon to node CLAPP1, use the `addNode` command, which can be found in the `<WASROOT>\bin` directory of every WebSphere Application Server V5 installation and add the node.

Execute the `./addNode.sh localhost 8879 -includeapps` command and wait for the command to complete. You should see output like:

```
ADMU0300I: Congratulations! Your node clapp1 has been successfully incorporated into the rmcell cell
……..
ADMU0003I: Node clapp1 has been successfully federated. ,
```

Note: The command requires the parameter for the hostname of the computer running the WebSphere Application Server deployment manager. In our case we have chosen to install the WebSphere Application Server Network Deployment on CLAPP1. Also, the -includeapps option will enforce that the previously-deployed resource manager Web application server icmrm will be propagated into the new cell. We will use this server later as a template.

Use the WebSphere Application Server administrative console to verify that the node was added properly. Restarting the Network Deployment manager may be necessary to verify that the node was added properly.

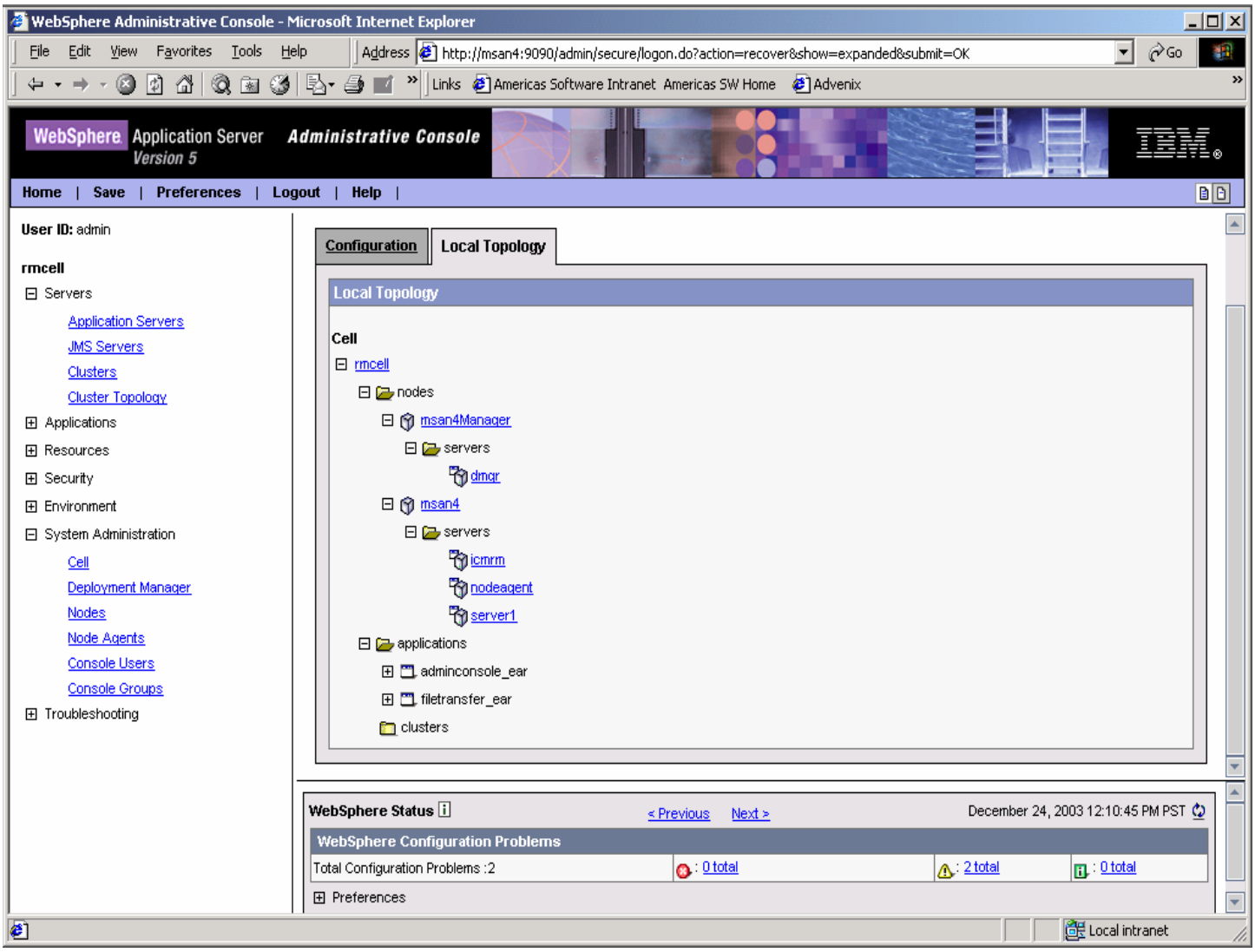


Figure 6-4. WebSphere Application Server Network Deployment cell and first Node

1. Repeat the same steps on the second node (CLAPP2).
2. Now add the second node, CLAPP2 to the resource manager cell:

```
Adding “clapp2” to “rmcell”
# pwd  /opt/WebSphere/AppServer/bin
# ./serverStatus.sh –all
# ./addNode.sh clapp1 8879
…….
ADMU9990I: ADMU0300I: Congratulations! Your node clapp2 has been successfully incorporated     into the rmcell cell…….
ADMU0003I: Node clapp2 has been successfully federated.
```

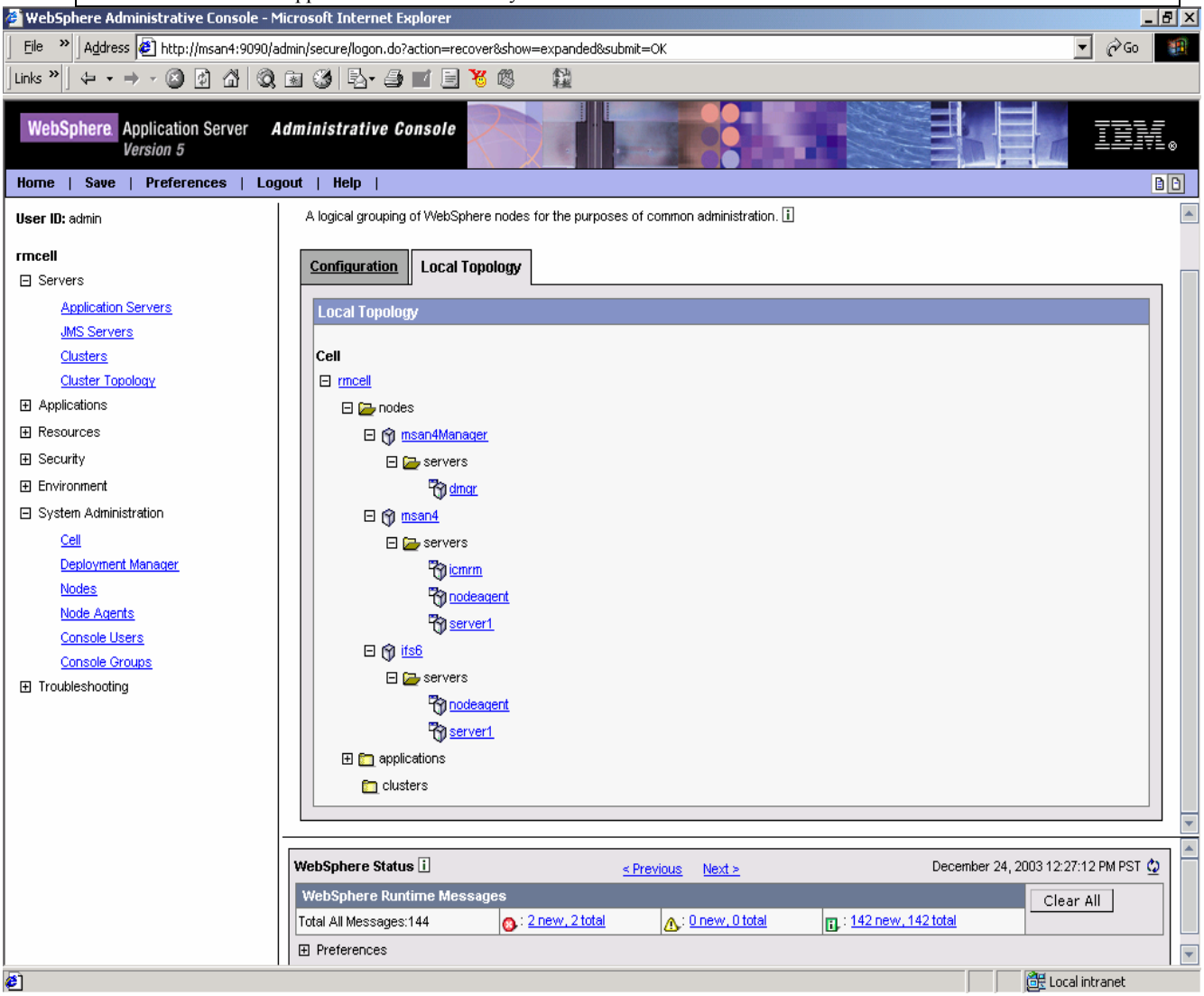


Figure 6-5. WebSphere Application Server Network Deployment: both nodes configured to the resource manager cell

6.3.2 Creating the WebSphere Application Server resource manager cluster: step-by-step approach

Next, we will configure the resource manager cluster and will add cluster members (application servers) to it.

Background: It is important to note that with WebSphere Application Server V5, the cloning and cloning semantic has changed.

New in WebSphere Application Server V5 is the cluster, with its cluster member concept. In contrast to the Application Group, and clone concept that existed before. One could imagine a cluster to be a superset of the Application group, with similar capabilities and more flexible enhancements.

Like in WebSphere Application Server V4, WebSphere Application Server V5 uses the existing application server as a template to create cluster members. A cluster with cluster members all alike, are similar to the WebSphere Application Server V4 Application Group and clones. The difference is that, cluster members do not have to be derived from the same application server template, and even if cluster members are derived from the same template they can be individually configured. If new cluster members are added, using the same application server template and configured identically this is equivalent to configuring Web application clones. Finally, by adding multiple cluster members on different nodes, one can configure a cluster of clones that provides both: failover, and horizontal and vertical application scaling capabilities.

In the following installation, all the Content Manager resource manager cluster members will use exactly the same configuration, and in order to do so we must ensure the two nodes are configured identically. This means that the DB2, WebSphere Application Server, and Content Manager file locations and environment variables must match on both nodes.

More specifically, we will create the RMCl01 cluster will add a total of four resource manager clones to it, two for each node:

1. WLM_icmrm on CLAPP1
2. WLM_icmrmvc02 on CLAPP1
3. WLM_icmrm2 on CLAPP2
4. WLM_icmrm2vc02 on CLAPP2

This design will provide intra-node and inter-node high availability. In addition, if the underlying workstations are powerful enough this configuration will also provide horizontal and vertical load balancing, allowing better scalability of the configuration and better exploitation of the hardware resources available.

Complete the following steps:

1. In the WebSphere Application Server Network Deployment administrative console, select **Servers > Clusters**
2. Click **New** in the right pane.
3. On the next panel, provide the cluster name (RMCL01). Select the existing server option and select rmcell/spawn/icmrm from the list. Click **Next**.
4. Name the first Cluster Server (WLM_icmrm). Click **Apply** and **Next**.
5. Click **Finish**
6. Add more cluster members as needed.

The following sequence of panels shows the basic flow through the most relevant WebSphere Application Server Network Deployment administrative console panels.

Adding a new cluster member:

Select **Server > Cluster > Cluster Members**. Add a cluster member by providing a name, selecting a node and clicking **Apply**.

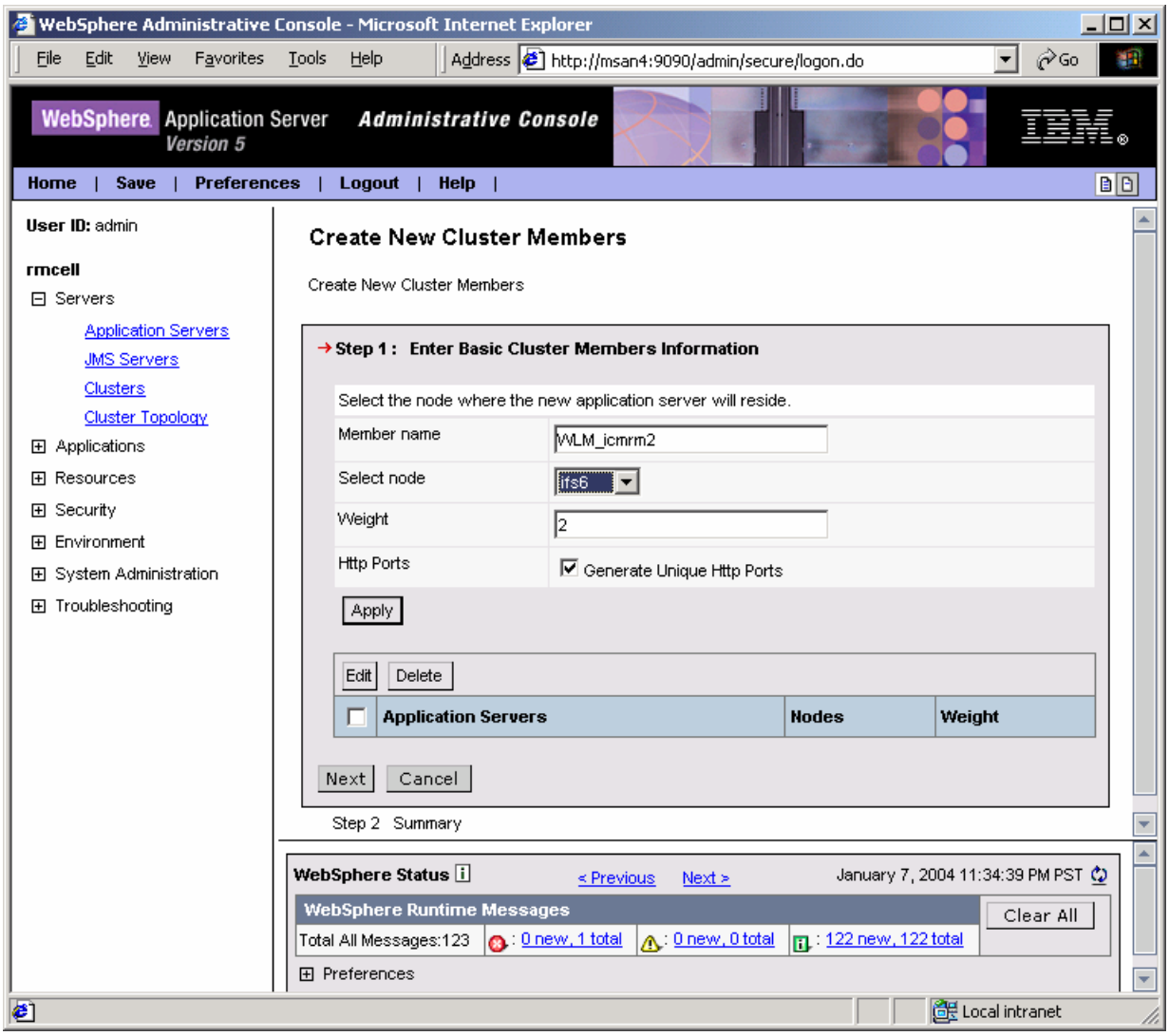


Figure 6-6. Cluster: Adding cross node cluster members

Click **Next**. On the next panel, check the parameters and click **Finish**.

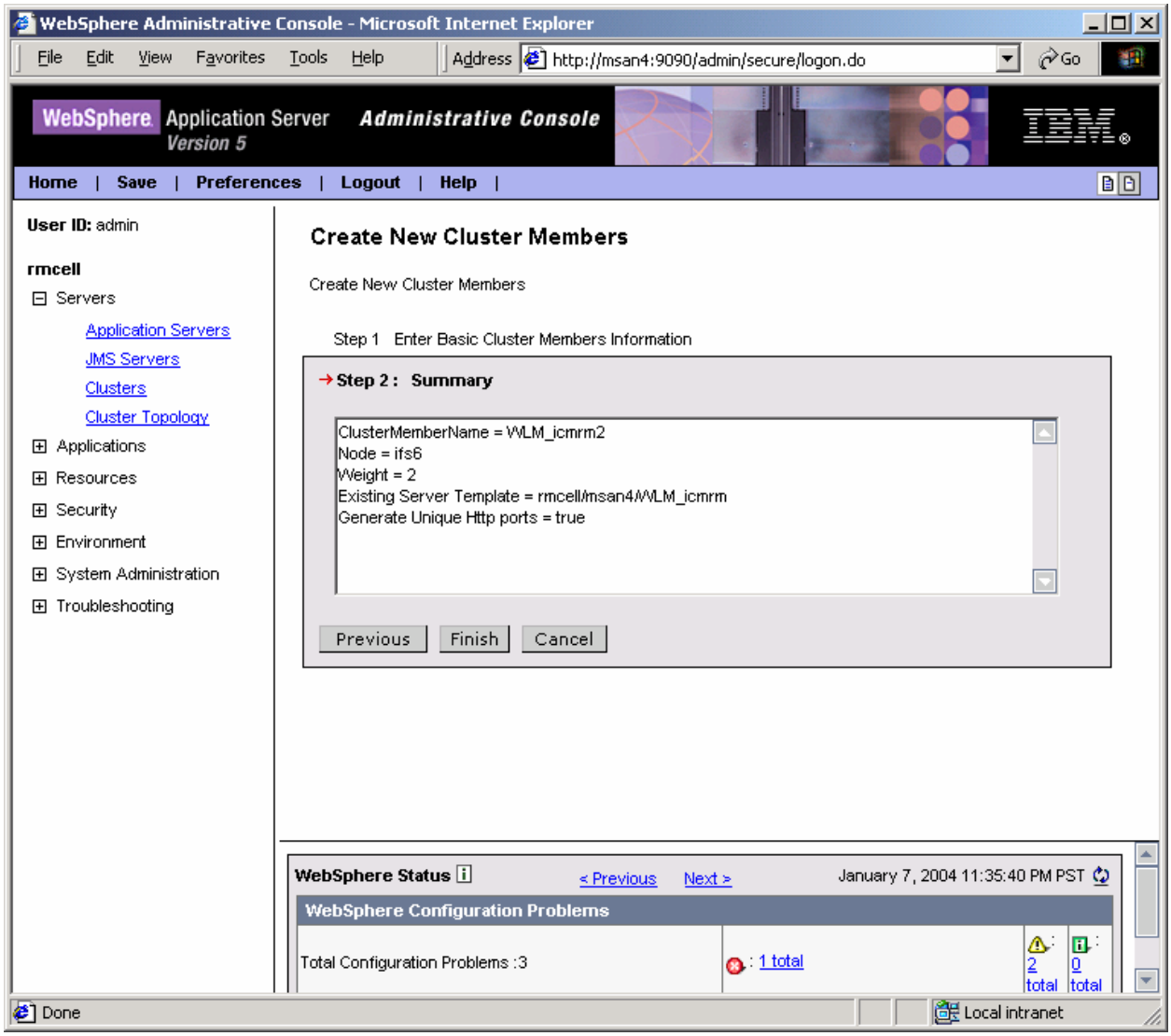


Figure 6-7. Cluster: Adding cross-node cluster members

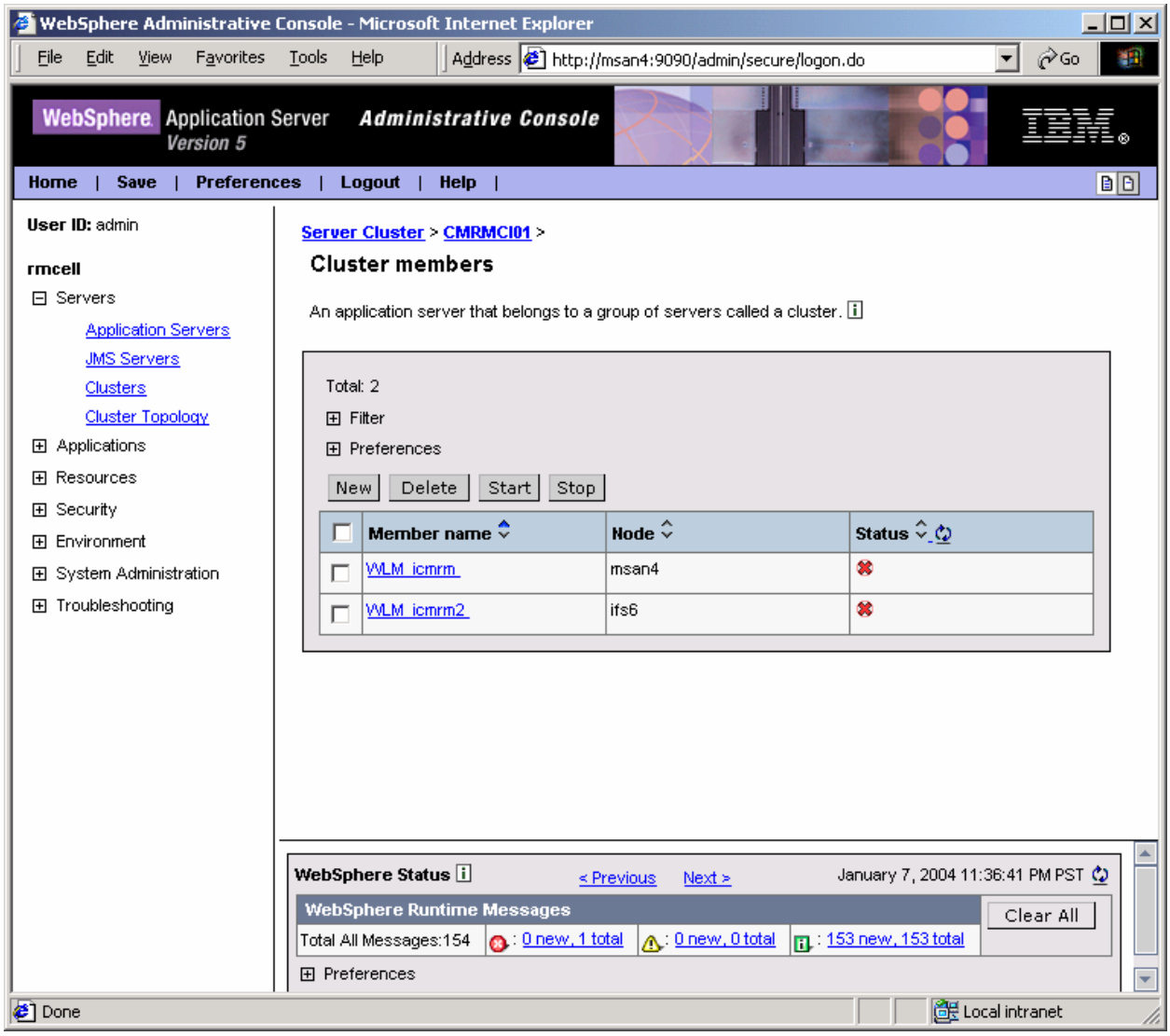


Figure 6-8. WebSphere Application Server V5 Cross-node resource manager cluster members.

Repeat this procedure until the entire cluster is configured.

Configure the second resource manager cluster member (first vertical clone) on CLAPP1.

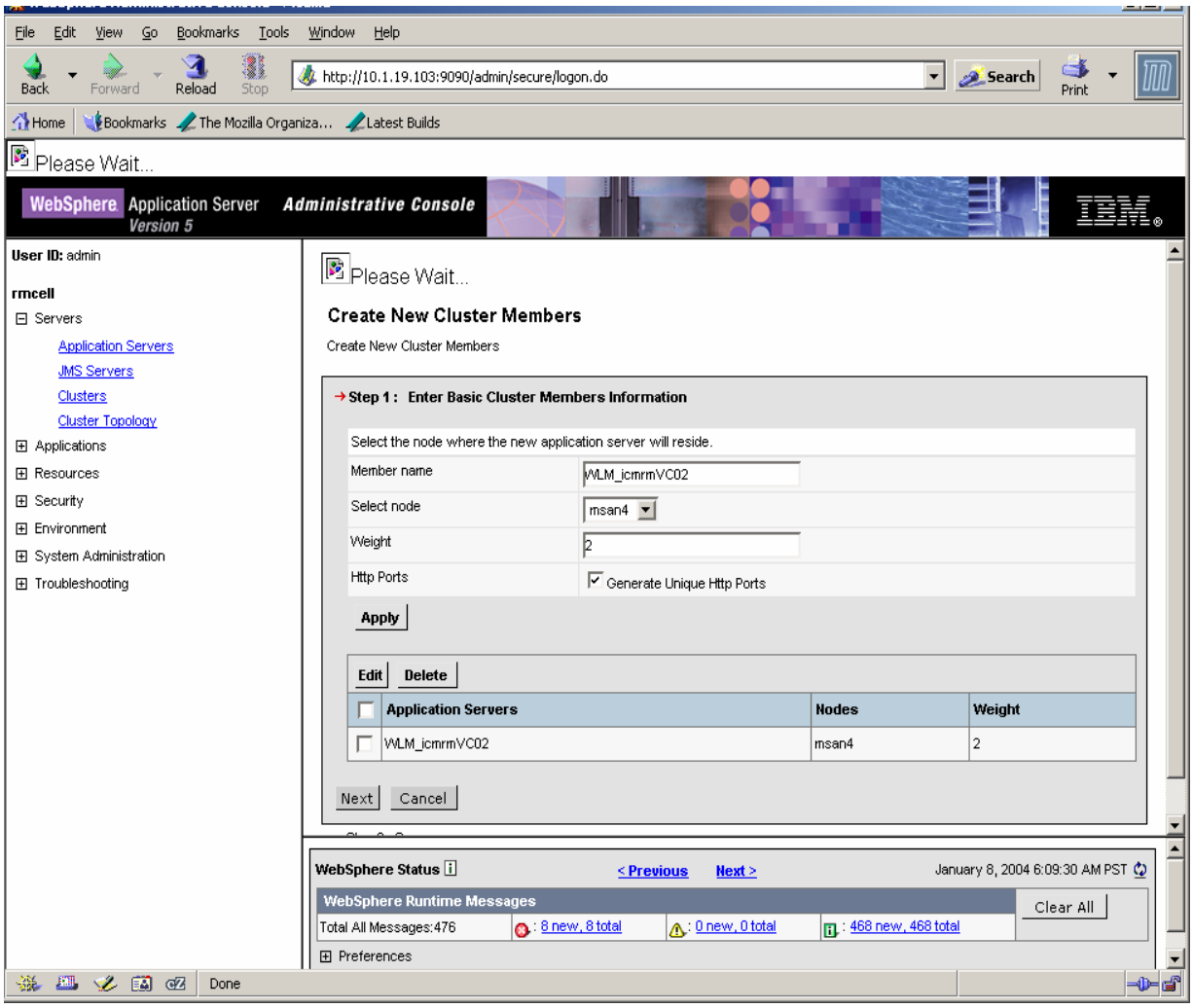


Figure 6-9. Vertical cloning

Click on the Apply button, and then click on Next button

Finally, check that all the parameters are correct, and then click on the Finish button.

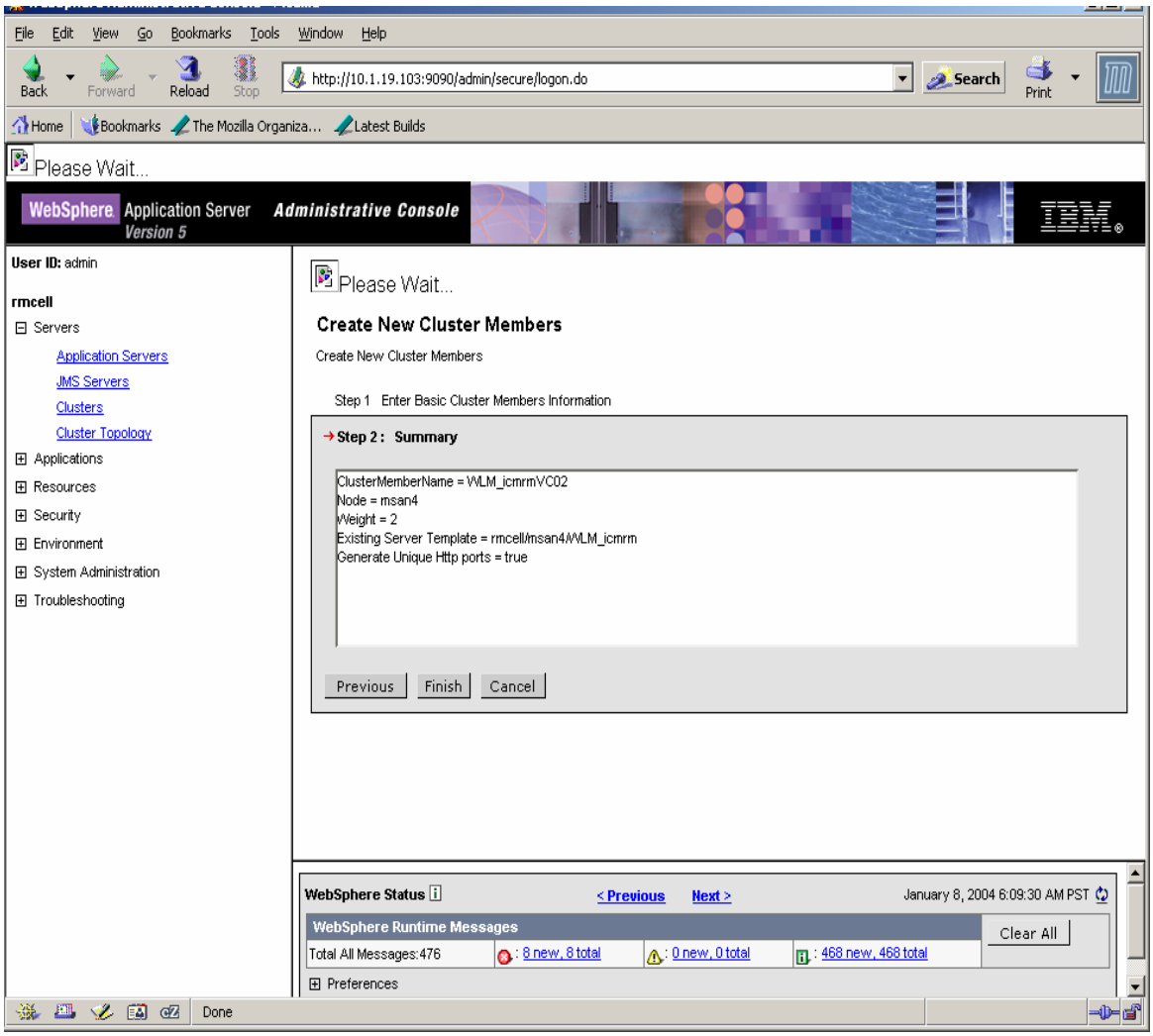


Figure 6-10. Vertical Cloning

Now repeat the procedure for the last clone.
Step through the same panel, provide the same input data then click on Apply and Next,

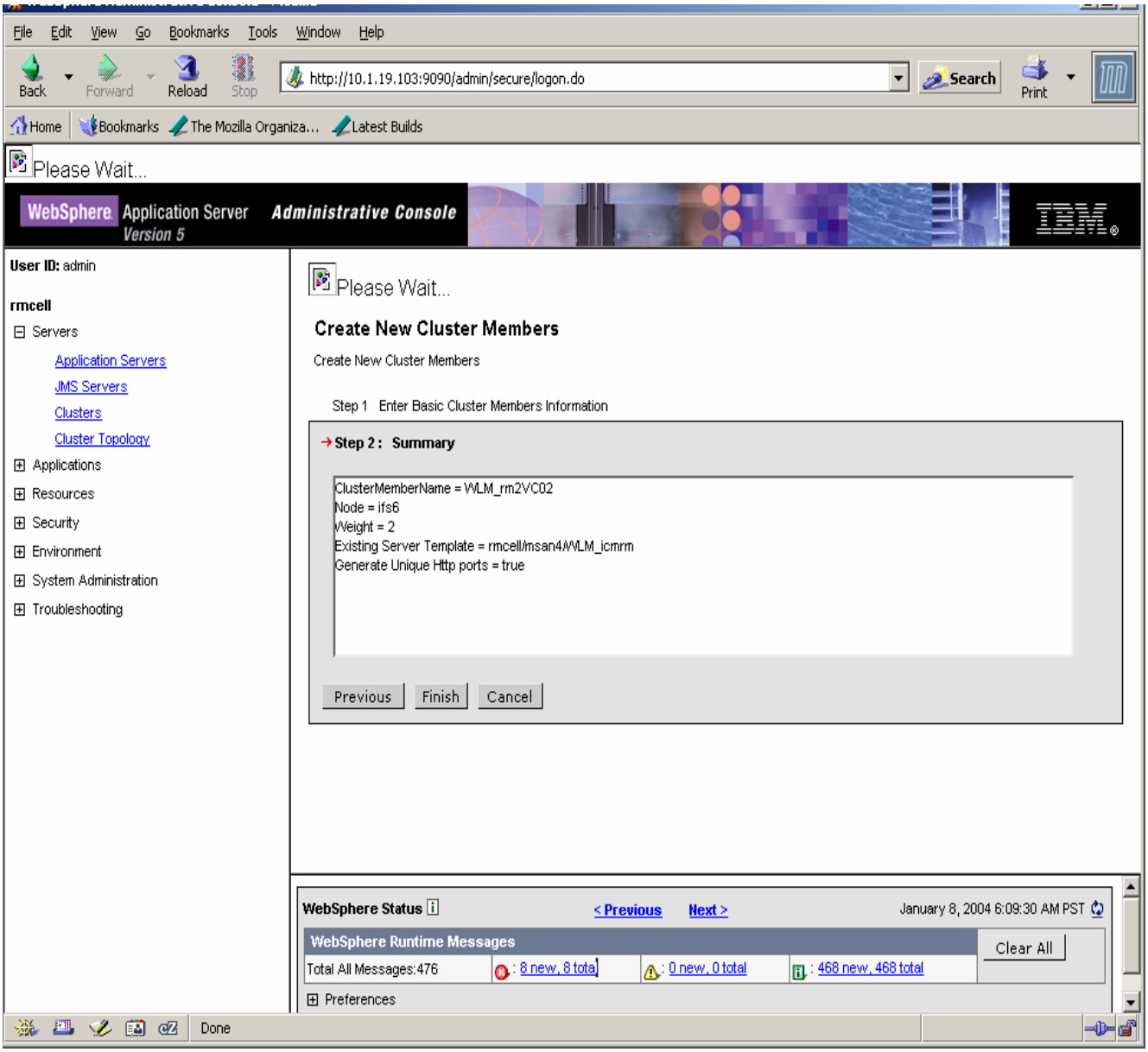


Figure 6-11. Vertical Cloning

Finally accept the default settings and click **Finish**.

The **Servers > Cluster** panel should contain now the four resource manager cluster members:

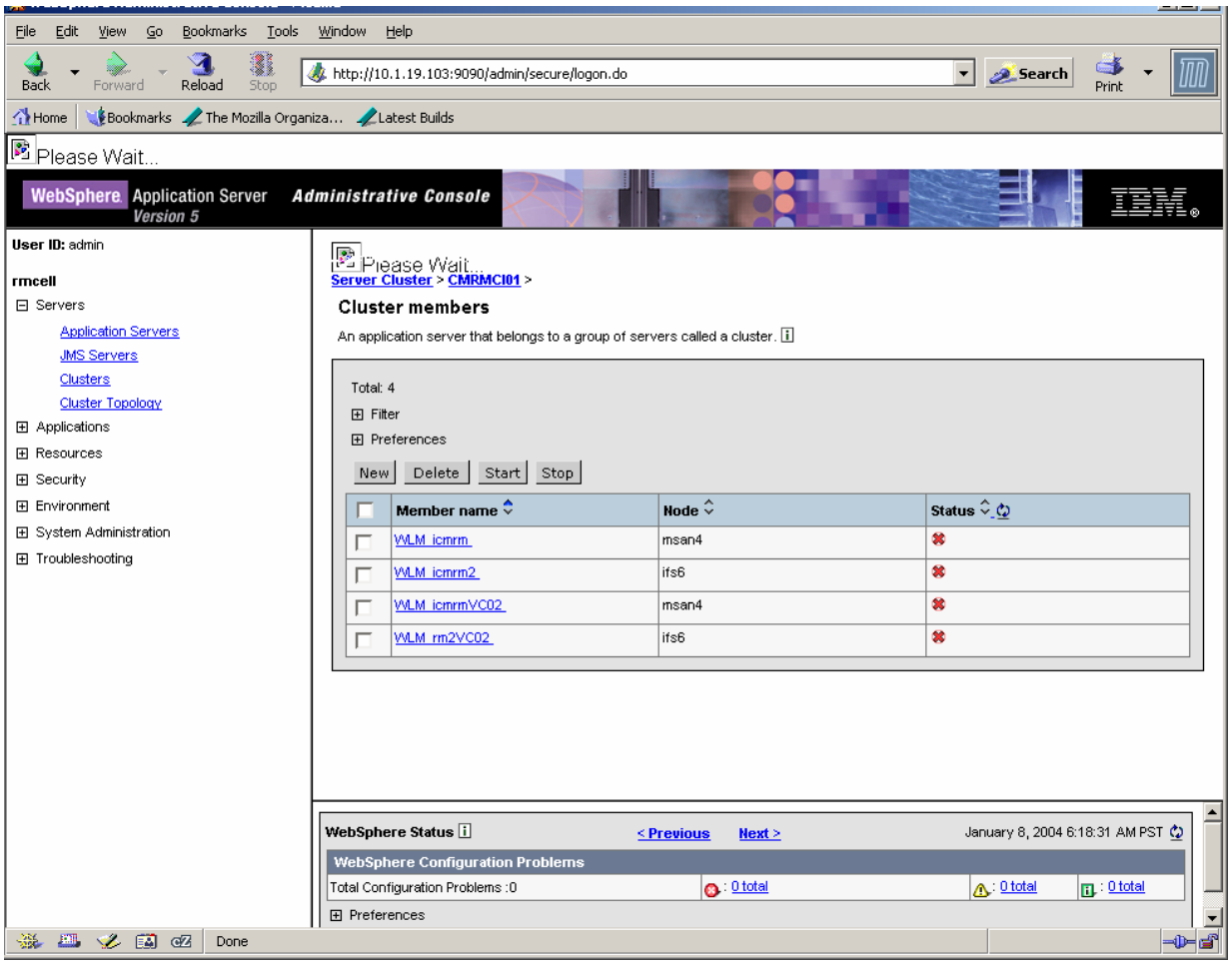


Figure 6-12. Vertical Cloning

The next steps are:

1. Associate the new cluster with the icmrm.war Web container, which represent the Content Manager resource manager application.
2. Save the created configuration into the master configuration file in order to become available throughout the cluster.
3. Specify DB2 as a JDBC provider.
4. Update the master HTTP Server plug-in.
5. Fully synchronize the whole cluster.

Go to **Applications > Enterprise Application > icmrm > Map modules to Application Servers**. You should see a panel similar to the one in Figure 6-13.

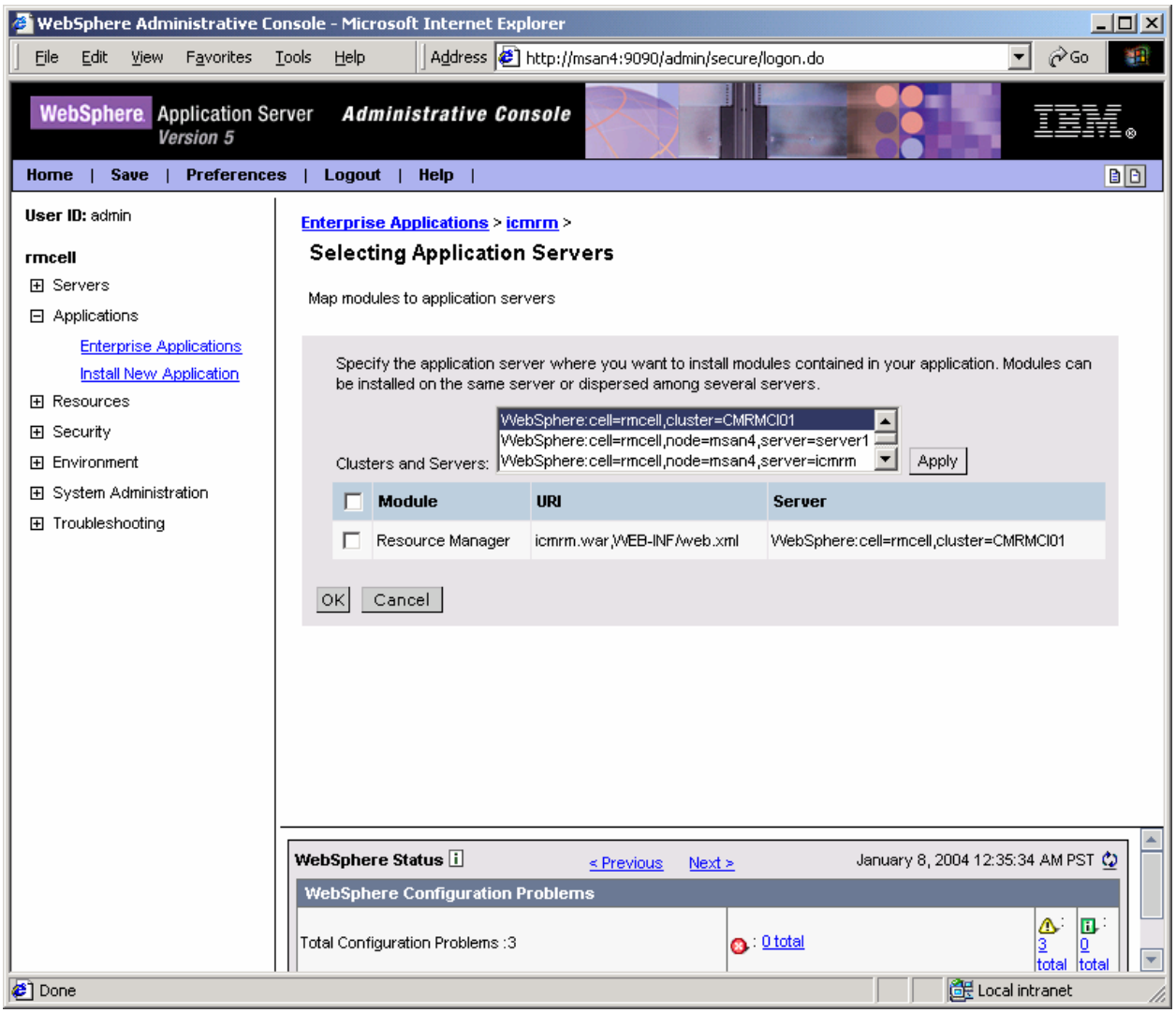


Figure 6-13. Binding the Content Manager resource manager EAR to the resource manager Cluster

In Figure 6-13 you will see the resource manager module with its URI bound to the server. The server column should have the RMCELL and CMRMCL01 cluster listed. Select the cluster entry, click on the Resource Manager module, and click **Apply**. Select the Resource Manager module again and click **OK**.

Note that any customizations to the resource manager WAR files on CLAPP1 and CLAPP2 may be overwritten if they are not reflected in the master copy used by WebSphere Application Server Network Deployment.

Now update the HTTP transport configuration if necessary.
ICMRM should have HTTP port 9081 or 9082 configured.

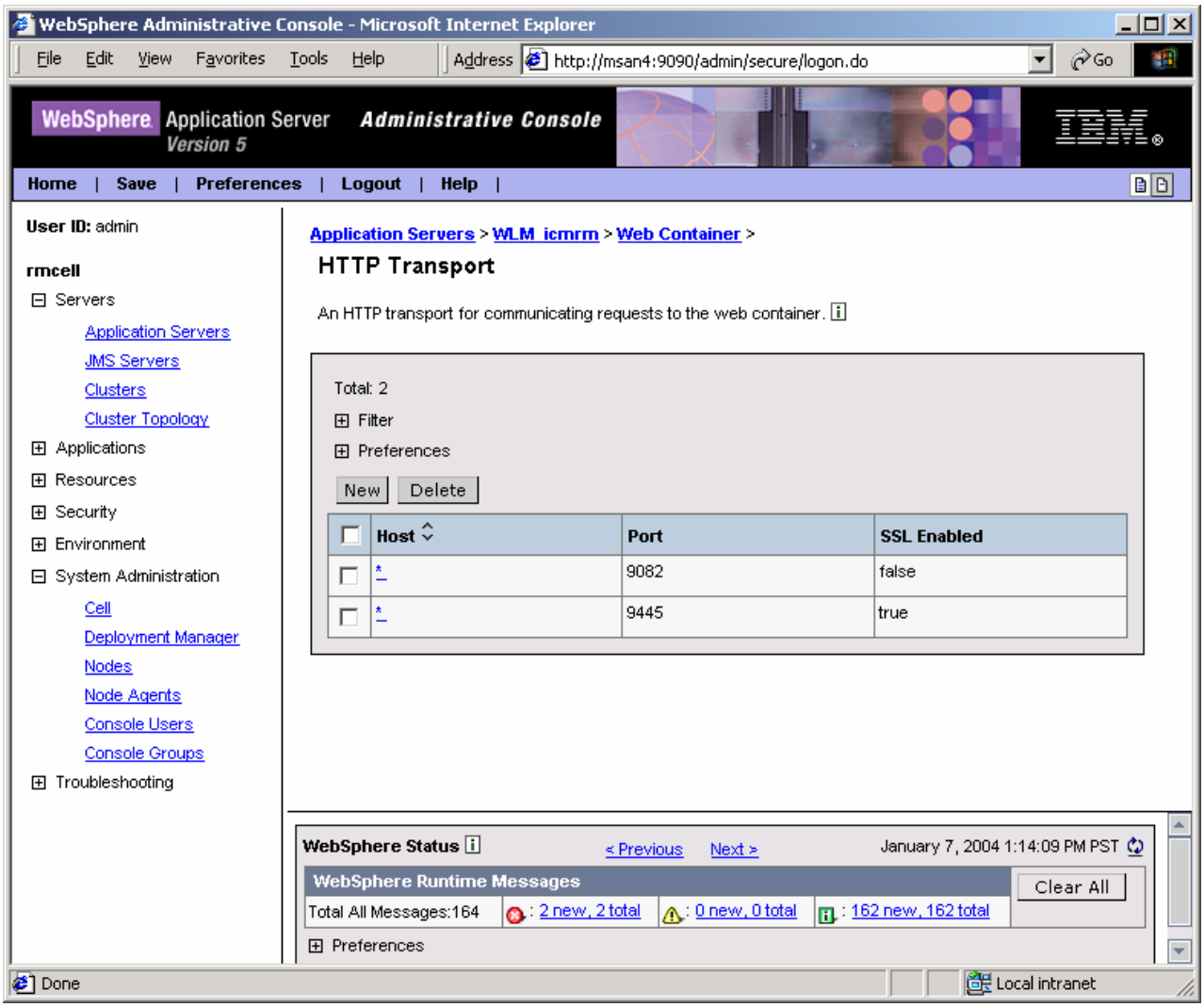


Figure 6-14. Resource manager cell and cluster

Now update the database provider for the icmrm application.

Click **Resources > JDBC Providers**
Clear the node field and click **Apply**. Next, click **New**. Select “DB2 Legacy CLI-based Type 2 JDBC Driver” and click **OK**. Finally, verify that the class path is correct and click **OK**.

Then save the HTTP plug-in configuration.

Click **Environment > Update Web Server Plug-in**.

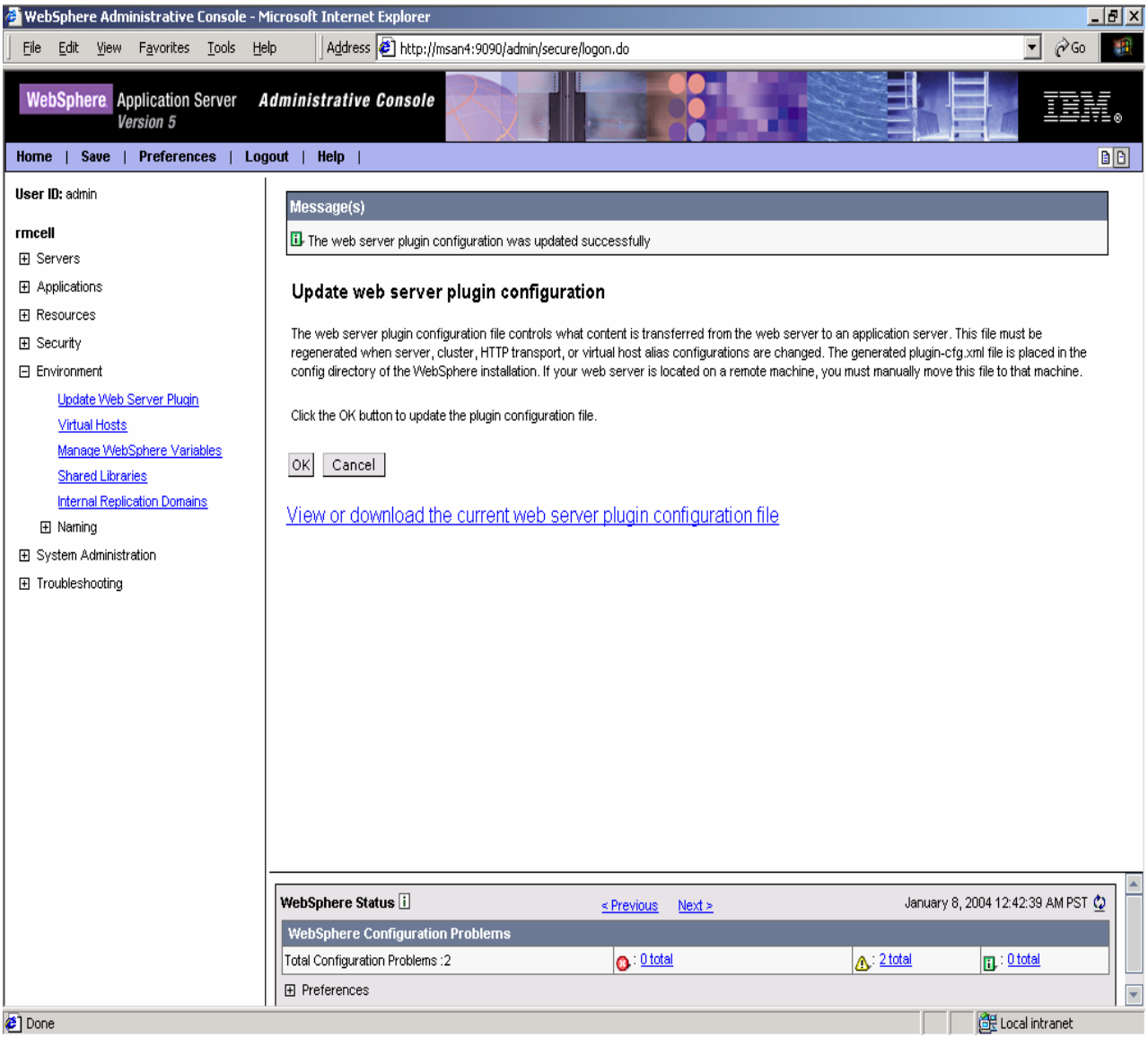


Figure 6-15. WebSphere Application Server V5 resource manager horizontal cluster

Finally, synchronize all the nodes in the cell.

Go to **System Administration Nodes**, select all nodes, and click **Full Resynchronize**.

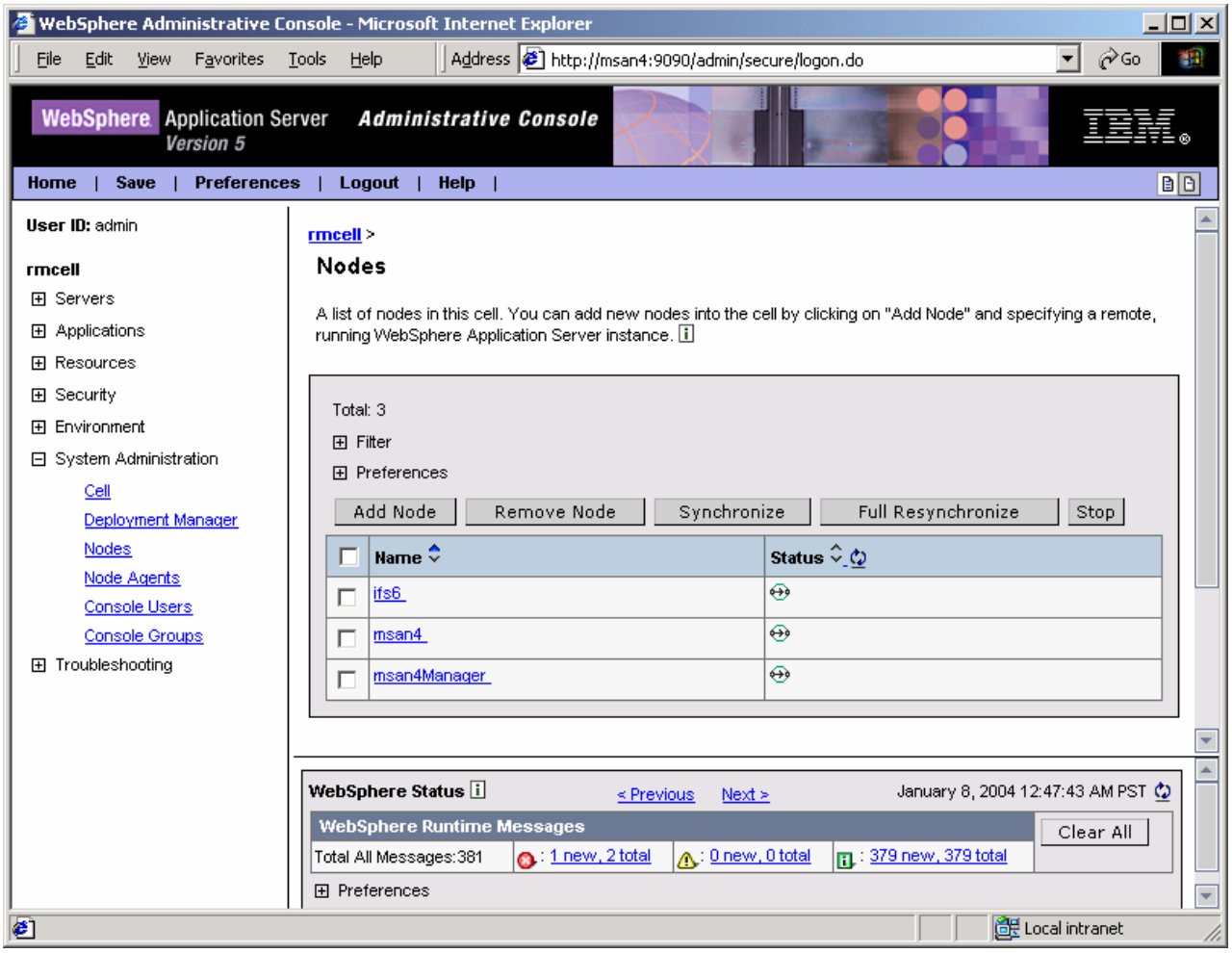


Figure 6-16. WebSphere Application Server V5 resource manager horizontal cluster

At this point, you should stop all WebSphere Application Server application servers, including the Network Deployment manager, the node agents, and the HTTP servers.

6.3.3 Customize the startServer.sh

In order to start application servers with the right environment when using the WebSphere Application Server administrative console, one approach is to source the proper environment from the `startServer.sh` for both the deployment manager and application servers.

For the icmrm application, we have added `. /export/home/db2clnt/.profile` to the following file on both application servers:

```
/opt/WebSphere/AppServer/bin/startServer.sh
```

Add the following lines to `startServer.sh` on both WebSphere Application Server nodes.

```
#!/bin/sh
. /export/home/db2clnt/.profile
```

6.3.4 Start the application servers

The application servers can be started using the following scripts on Clapp1 and Clapp2.

On Clapp1:

```
# cat /opt/WebSphere/AppServer/startICMRM.sh
# !/bin/sh

#The following two lines should be used only on the
#WebSphere Application Server Network Deployment machine
cd /opt/WebSphere/DeploymentManager/bin
./startServer.sh dmgr

cd /opt/WebSphere/AppServer/bin
./startServer.sh nodeagent
./startServer.sh WLM_icmrm
./startServer.sh WLM_vc02
```

On Clapp2:

```
# cat /opt/WebSphere/AppServer/startICMRM.sh
# !/bin/sh

cd /opt/WebSphere/AppServer/bin
./startServer.sh nodeagent
./startServer.sh WLM_icmrm2
./startServer.sh WLM_icmrm2vc02
```

To start these application servers automatically after reboot, add an entry to `/etc/inittab` to call `/opt/WebSphere/AppServer/startICMRM.sh`.

At this point, the HTTP servers, dmgr, node agents, and the entire cluster can be started. With `dmgr` and `nodeagent` running, the application cluster can also be started from the WebSphere Application Server Network Deployment administrative console. With everything running, the next step is to verify the installation.

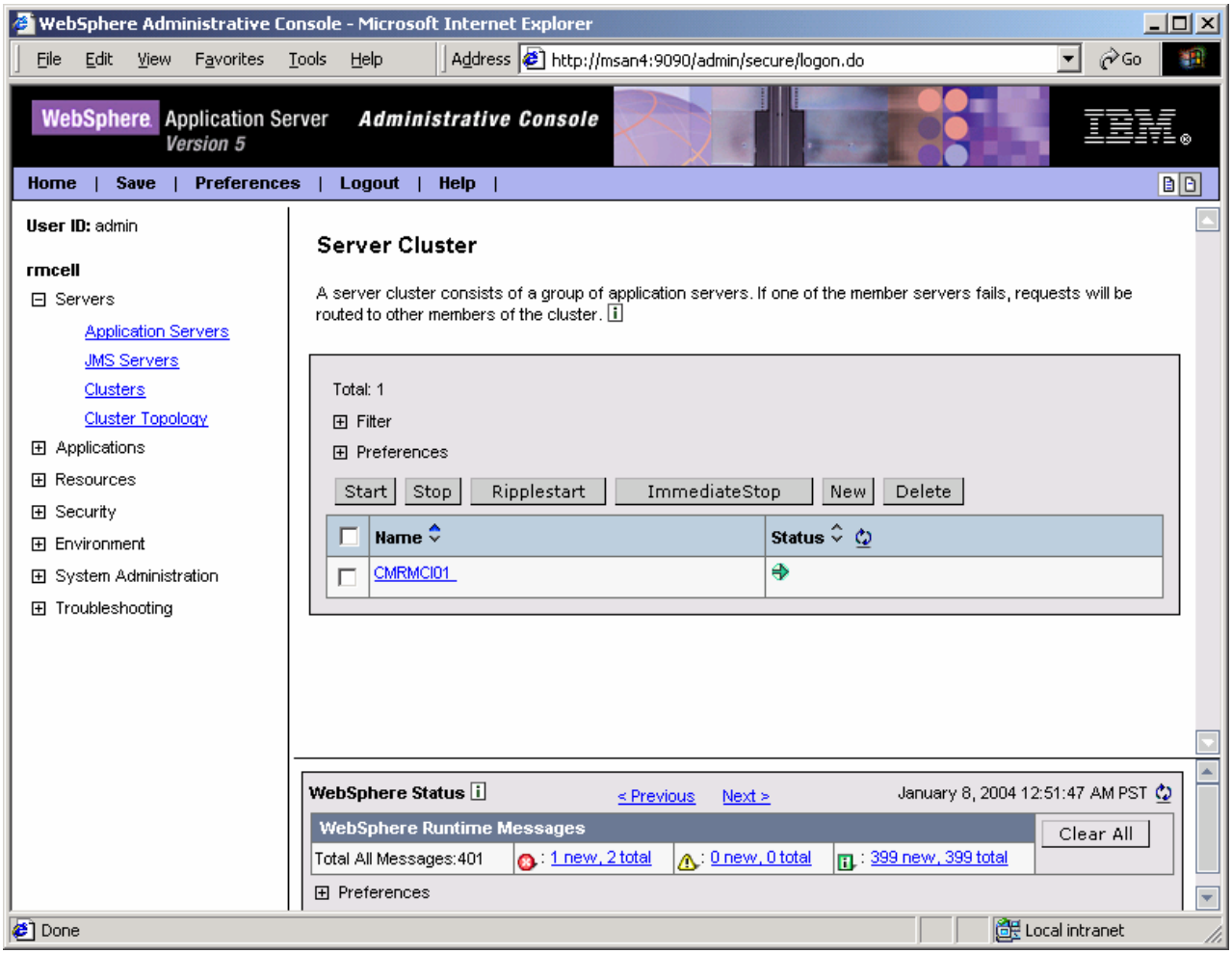


Figure 6-17. WebSphere Application Server V5 resource manager horizontal cluster

6.3.5 Resource manager cluster function and verification tests

Once the cluster is started, all Content Manager resource manager instances will connect to both the library server and resource manager databases. By default, each resource manager creates a pool of three connections. After the startup, check the database to see if applications have open connections to the database and, if so, how many there are.

Figure 6-18 shows a snapshot of our system.

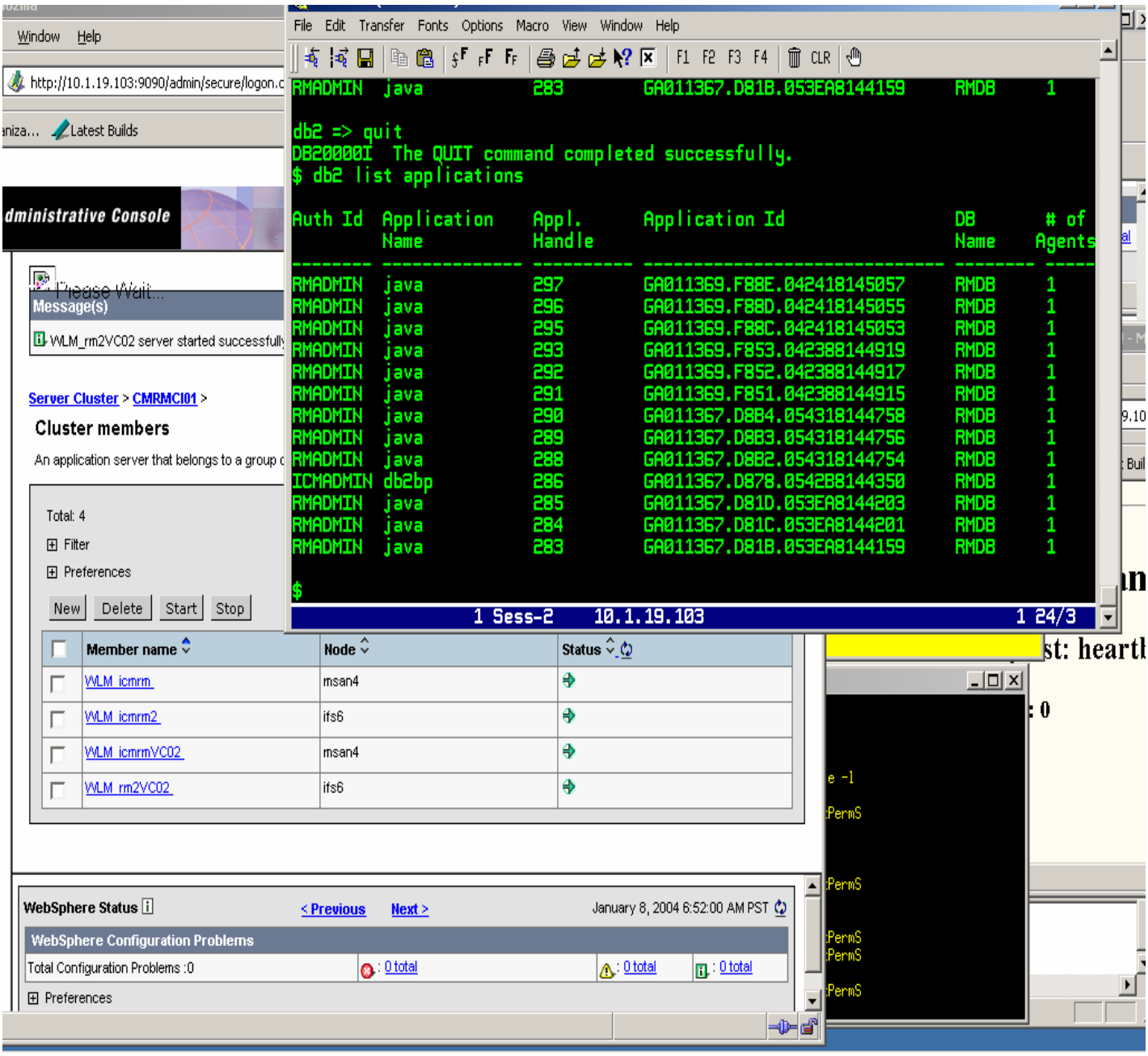


Figure 6-18. Vertical cloning

From the DB2 command line window issue a

```
db2 list applications
```

command, this will list all active connections to the DB2 database instance.

In Figure 6-18 you can see in the telnet session window that RDADMIN was holding a total of 12 open connections to the resource manager database. That means that all four resource manager clones were able start and to connect to the same resource manager database.

To see if the resource manager cluster is functional across all nodes, use the ICMResourceManager or the resource manager's own version of the snoop servlet.

The first servlet is the main entry point into the resource manager and functions like a request dispatcher. Resource manager requests are HTTP requests using an specific protocol. The general layout of this protocol is:

```
ICMREsourceManager?order=heartbeat&libname=ICMNLSDB
```

We will use the simple heart beat request and the snoop servlet:

1. Start the cluster with all cluster members active
2. Then open Browser and point the URL to: http://clapp1/icmrm/snoop

Figure 6-19 shows a successful response.

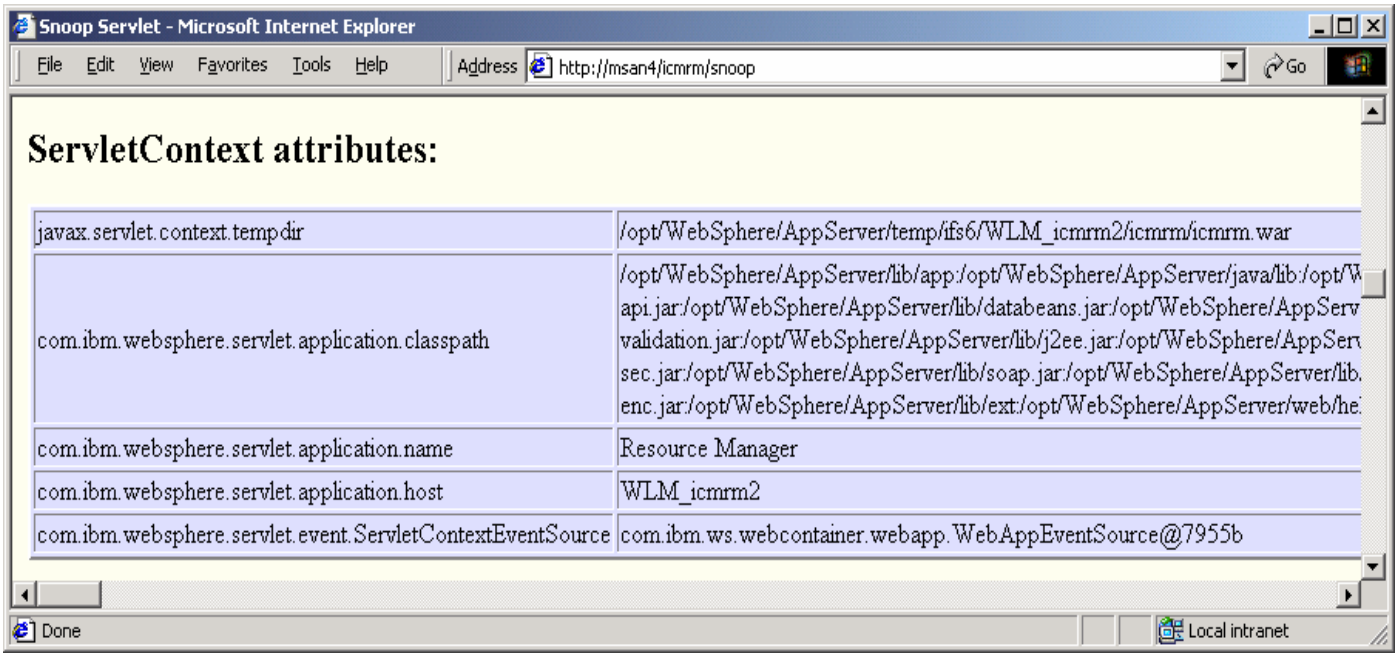


Figure 6-19. WebSphere Application Server V5 resource manager horizontal cluster

3. Repeatedly and quickly click **Refresh** in the browser. You should see that the application host displayed by the HTTP response changes occasionally, showing WLM_icmrm, WLM_icmrm2, WLM_icmrmcv01, and WLM_icmrmcvo2. This test shows that the WebSphere Application Server HTTP plugin on CLAPP1 is distributing the load throughout the cluster across the nodes.

4. Use the WebSphere Application Server administrative console to stop all but one cluster member. For example, let's say you want to keep only WLM_icmrm alive.
   Click **Server Cluster >CMRMCL01** and select all the other cluster members. Click **Stop**.
5. Wait until the other cluster members are stopped, then continue to use the Browser to issue HTTP requests as before. In the snoop response page you should now see that after a few requests the application host is WLM_icmrm.

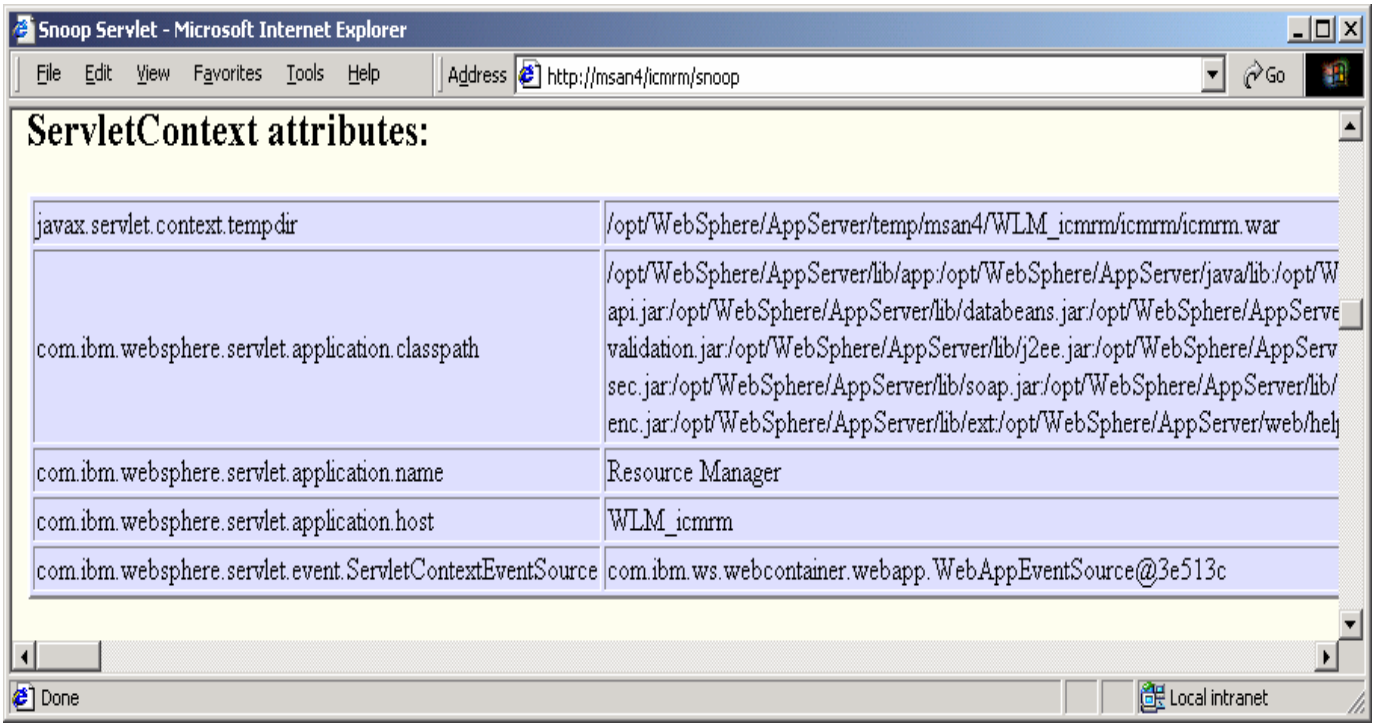


Figure 6-20. WebSphere Application Server V5 resource manager horizontal cluster

The next test will exercise the main resource manager servlet, ICMResourceManager. In order to demonstrate that the resource manager cluster members (clones) are working properly, we will send heartbeat orders to the resource manager and check that these requests are services by all cluster members across the two nodes.

1. Log on to the first node, CLAPP1 and change to the resource manager Web application installation directory: `<WASROOT>/installapps/<cellname><RM EAR>/icmrm.war/WEB-INF` and open `icmrm_logging.xml`. At the bottom of the file, change the trace level by changing the priority value from `INFO` to `DEBUG`.

```
<!-- These are the available priority's.
  1) FATAL     - Only log if the servlet is terminating unexpectedly
  2) ACTION    - Messages that describe an action the System Administrator
                 need to take. These are not errors but conditions like
                 short on space.
  3) ERROR     - Indicates a request was unable to be fulfilled or an
                 internal error.
  4) WARN      - Warnings of unexpected behavior.
  5) INFO      - Informational start/stop messages
  6) BEGINEND  - Time markers begin and end for performance measures
  7) REQUEST   - Detailed information on the incomming request.
  8) RESPONSE  - Detailed information on the outgoing response.
  9) TRACE     - General flow messages.
 10) DEBUG     - Detailed debugging information, plus all other levels.
<priority value="FATAL"    class="com.ibm.mm.icmrm.util.ICMRMPriority"/>
<priority value="ACTION"   class="com.ibm.mm.icmrm.util.ICMRMPriority"/>
<priority value="ERROR"    class="com.ibm.mm.icmrm.util.ICMRMPriority"/>
<priority value="WARN"     class="com.ibm.mm.icmrm.util.ICMRMPriority"/>
<priority value="INFO"     class="com.ibm.mm.icmrm.util.ICMRMPriority"/>
<priority value="BEGINEND" class="com.ibm.mm.icmrm.util.ICMRMPriority"/>
<priority value="REQUEST"  class="com.ibm.mm.icmrm.util.ICMRMPriority"/>
<priority value="RESPONSE" class="com.ibm.mm.icmrm.util.ICMRMPriority"/>
<priority value="TRACE"    class="com.ibm.mm.icmrm.util.ICMRMPriority"/>
<priority value="DEBUG"    class="com.ibm.mm.icmrm.util.ICMRMPriority"/>
-->
<priority value="INFO"  <- set to DEBUG
      class="com.ibm.mm.icmrm.util.ICMRMPriority"/>
   <appender-ref ref="ASYNC"/>
```

2. Repeat the procedure on CLAPP2
3. Start the resource manager cluster using the WebSphere Application Server administrative console.
4. On both nodes open a telnet window and enter the following command:
   `tail -f  <WASROOT>/logs/icmrm/icmrm.logfile`
5. Position both windows so that you can see the trace logs created by the respective resource manager clone as requests are coming in.
6. Point the browser to:
   http://clapp1/icmrm/ICMResourceManager?order=heartbeat&libname=ICMNLSDB
7. Click **Enter** several times in a quick series. It must be fast In order to generate a load heavy enough that it is too much for a single clone to cope with. Otherwise if the load is too light and can be services by one cluster member and the WebSphere Application Server HTTP plug-in will not distribute the request across the cluster. On the screen you should now see that all the different cluster members are servicing the requests as they come in.

The `icmrm.logfile` logs should look like the ones in Figure 6-21:

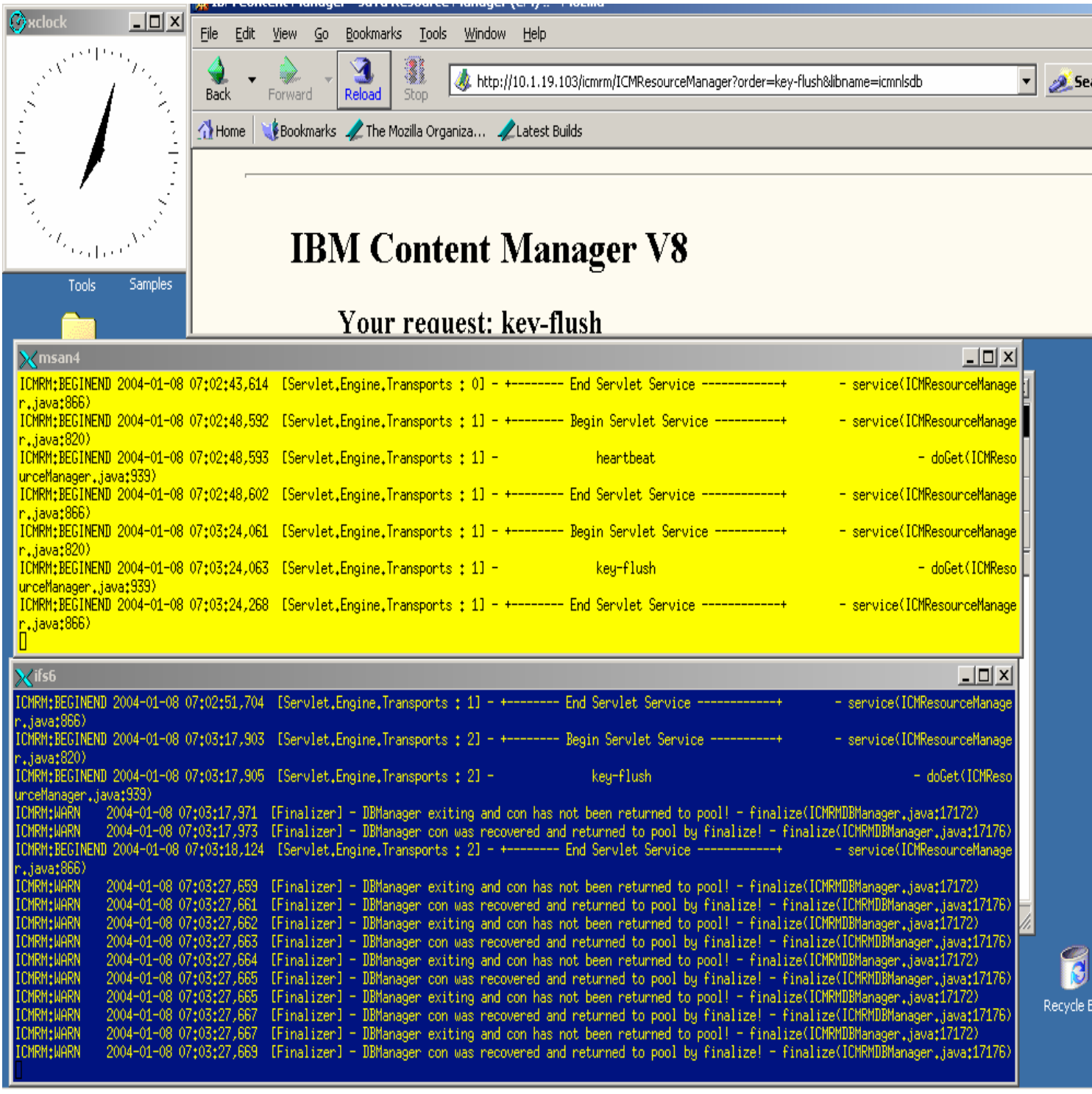


Figure 6-21. Vertical cloning resource manager verification

6.3.6 Troubleshooting resource manager database login problems

A. When testing this configuration, the resource manager database logon might not work due to a password decryption error. This happens every time the secret key file ICMRMKeyStore gets out of sync.

In order to logon to DB2 you need a valid user ID and password. For the resource manager, the Content Manager library server user ID and password is stored in the RMAccess table. The resource manager administrative password is stored in the ICMRM.properties file. At installation, both passwords are in clear text, and at the first startup, the resource manager servlet generates a secret key, encrypts the password and stores them back in the RMAccess tables and the ICMRM.properties file. The secret key itself is stored in the ICMRMKeyStore file. If the key and the properties file get out of sync, the resource manager logon will fail and the servlet will not start. It is therefore important to keep these files in sync all the times on both nodes, for all the resource manager clones.

To recover from this situation do the following:

1. Stop the resource manager clones.
2. Delete the ICMRMKeyStore file on one of the clusters (for example cluster A) .
3. Reset the passwords in the ICMRM.properties file (for example cluster A) , RMACCESS table, ACC_PASSWORD column with unencrypted passwords.
4. Restart one of the resource manager clusters (for example cluster A).
5. Copy from cluster A machine to cluster B machine the ICMRMKeyStore and ICMRM.properties files.
6. Start cluster B.

B. While starting the RM clones from the WebSphere application server administrative console, WAS may not read the db2java.zip file from the db2 jdbc path. This can be determined if in the icmrm.logfile the following error is seen “java.sql.SQLException: No suitable driver**”.**

To work around this condition, try starting the RM clones from the command line.

1. Source the db2 profile to ensure db2java.zip is in the classpath.
2. Start the clones - # /opt/WebSphere/AppServer/bin/startServer.sh (clone_name)

7. Enabling the Content Manager daemons for high availability

This section describes the installation and configuration steps necessary to set up the Content Manager daemons.

To make full use of all Content Manager functions in a production environment, all related daemons must be:

- Set up properly
- Fully functional
- Configured for failover in case of a node failure

These daemons are:

- Library server monitor
- Asynchronous recovery
- Migrator
- Stager
- Purger
- Replicator

The library server monitor is responsible for:

- Monitoring resource manager availability.
- Updating notify and suspend flags for document routing.
- Counting concurrent users and writing event records.
- Processing TIE updates for Oracle.

```
Syntax:  icmplsap [SERVICE] dbname [userid [password]]
```

If the parameter `SERVICE` is specified then the assumption is that this program is started as a Windows service. If defined as a service, the service name must match the database name. This is to ensure that it will be possible to have one service for each database. Finally if `userid` is specified, then the password is required

The library server monitor is responsible for polling the status of available resource mangers in the network. It sends out heartbeat requests and tracks the status in the resource manager table. Resource managers that are marked *down* will not be considered for store and retrieve operations for client requests. This daemon is pivotal to the Content Manager replication services and serves as the trigger for the failover mechanism in Content Manager.

During failover, when the database is down, the library server monitor will loop and retry a database logon every 60 seconds. The retry loop will only be started in the event of "DB not stared" or "connection down". Otherwise, it will report the error and exit. When the backup node has restarted the library server database, it will then continue normal operation.

On the resource manager side, the asynchronous recovery and migrator daemons are jointly unified in the migrator application. This service must always be turned on, as it is part of the Content Manager V8 transaction model. Even if object migration is not required, the migrator must run to make sure the cumulated transactions state is cleared and the transaction logs can be purged. The asynchronous recovery logic, as part of the migrator application, does this by comparing the library server transaction table (ICMSTTXLT) with the transaction table of the respective resource manager (RMTRACKING). Both tables can grow quite large if old transaction records are not purged.

The other resource manager daemons, stager, purger, and replicator, do exactly what their names imply. These daemons must be configured to be active during normal operations.

The resource manager daemons must be configured to run on the resource manager Web application nodes. The library server monitor should run on the Content Manager database nodes.

Finally, in order to enable the Content Manager daemons for high availability in a Sun Cluster environment, the daemons need to be defined as cluster managed resources in the Sun Cluster Manager. This is usually done by creating a set of wrapper scripts that map to a specific protocol (start, stop, and probe). These scripts are usually created by the Sun Sysplex Agent Builder, a graphical tool that can automatically generate standard wrapper scripts. In our case we have built and customized such a set of scripts for the resource type IBM.cmrmproc. With these scripts the resource manager daemons are put under the control of Sun Cluster Manager.

The steps on how to generate the script will be explained later. Let's assume we have all that we need to enable the Content Manager daemons for high availability, and start with their setup.

7.1 Configure the Content Manager library server monitor for high availability

The library server monitor must be configured to run on the two Content Manager database nodes.

Complete the following steps:

1. Log on to the first database node, CLDB1:
   Edit `/etc/rc.cmlsproc` and change `WHOLOGIN='whoami'` to `WHOLOGIN='logname'`.

   (Note: `whoami` is not on the default path.)

2. Switch user to "root
   ```
   su - $LSCMADM -c "$ICMPLSAP $LSCMDBNAME >/dev/null 2>&1&" ;;"
   ```

   Note: This keeps the process foregrounded so that the init process can monitor it.

3. Edit `/etc/inittab` to start `/etc/rc.cmlsproc` with *respawn* for action:

   ```
   cldb1/cldb2:
   # tail -5 /etc/inittab
   rb:6:wait:/sbin/uadmin 2 1 >/dev/msglog 2<>/dev/msglog </dev/console
   sc:234:respawn:/usr/lib/saf/sac -t 300
   co:234:respawn:/usr/lib/saf/ttymon -g -h -p "`uname -n` console login: " -T xterms -d /dev/console -l console -m
   ldterm,ttcompat
   #fmc:234:respawn:/opt/IBM/db2/V8.1/bin/db2fmcd #DB2 Fault Monitor Coordinator
    cmls:3:respawn:/etc/rc.cmlsproc > /dev/console 2>&1 # Autostart CM82 LS processes
   ```

4. Repeat steps 1-3 on CLDB2.

7.2 Configure the Content Manager resource manager daemons for high availability

Each of the Content Manager resource manager daemons will be monitored by Sun Cluster software. These processes will be restarted or failed-over to the backup nodes in the cluster.

7.2.1 Configure the Content Manager resource manager daemons to be started from either side of the cluster

Before we can let Sun Cluster monitor and automatically manage failover for the Content Manager resource manager daemons, we need to configure the system so that the resource manager daemons can be started from either side of the cluster. In order to do that, we need to globalize the environment.

1. Mount `<CMROOT>: /opt/IBMicm` as a cluster file system between CLAPP1 and CLAPP2. Files in /o`pt/IBMicm/config` and `/opt/IBMicm/lib` are needed by the resource manager daemons.
2. Add entries to `/etc/vfstab` for clapp1 and clapp2:

```
# grep /opt/IBMicm /etc/vfstab
/dev/md/rmstg/dsk/d31 /dev/md/rmstg/rdsk/d31 /opt/IBMicm ufs 2 yes global,logging
# mount /opt/IBMicm
```

3. Add port entries to /etc/services on both CLAPP1 and CLAPP2.

```
RMMigrator_RMDB 7500/tcp                #Resource Manager Migrator
RMPurger_RMDB   7501/tcp                #Resource Manager Purger
RMReplicator_RMDB       7502/tcp        #Resource Manager Replicator
RMStager_RMDB   7503/tcp                #Resource Manager Stager
RMAR_RMDB       7504/tcp                #Resource Manager Async Recovery
```

4. Comment out the `/etc/rc.cmrmproc` entry in `/etc/inittab` on both CLAPP1 and CLAPP2. The Content Manager resource manager daemons will be started and monitored by the Sun Cluster Software. Disable the startup from `/etc/inittab` to avoid conflict.

```
# cmrm:3:once:/etc/rc.cmrmproc > /dev/console 2>&1 # Autostart CM82 RM background processes
```

5. Make sure `/etc/rc.cmrmproc` exists on both CLAPP1 and CLAPP2.

   The `/etc/rc.cmrmproc` shipped with the Content Manager is the control program to start/stop the Content Manager resource manager daemons. The file is installed to `/etc/rc.cmrmpro`c during the resource manager setup. Depending on how CLAPP1 and CLAPP2 are configured, `/etc/rc.cmrmpro` may not be present. If that's the case, the file can be copied from CLDB1 or CLDB2 where resource manager codes are extracted during the resource manager database setup.

```
bash-2.05# rcp /etc/rc.cmrmproc clapp1:/etc/rc.cmrmproc
bash-2.05# rcp /etc/rc.cmrmproc clapp2:/etc/rc.cmrmproc
```

6. Initialize resource manager daemons environment. The resource manager daemons can be started with explicitly by passing in the required command line parameters. For example, to start the RMMigrator, enter the following command:

```
/etc/rc.cmrmproc -db RMDB1 -app icmrm1 -insthome /export/home/db2clnt -was
/opt/WebSphere/AppServer -proc RMMigrator
```

The /etc/rc.cmrmproc calls the customized /opt/IBMicm/config/setprocenv.sh to get default set of settings for background processes. Modify /opt/IBMicm/config/setprocenv.sh with the following values:

```
# Resource Manager Database Name
dbname=RMDB
# Resource Manager Application Name
rmappname=icmrm
# nodename is required if using  WAS5.X
nodename=clapp1
# WebSphere home installation directory
was_home=/opt/WebSphere/AppServer
# db2 instance home use for RM install
insthome=/export/home/db2clnt
```

7. Assign the proper value to ICMROOT (add this variable):

```
# The install root for ICM
ICMROOT=/opt/IBMicm
```

$ICMROOT is used in later part of the script to obtain the proper CLASSPATH setting:

```
export CLASSPATH=$CLASSPATH:$ICMROOT/lib/NLVLog.jar
```

Note: The <node name> in our case **"clapp1"** needs to match with the node name of the "resource manager template web application" at the time the WebSphere Application Cluster was configured, because the that node name will be part of the installation path into which the resource manager web application will be installed on all cluster nodes. With WAS Deployment Manager, the /opt/WebSphere/AppServer/installedApps/clapp1 path will be the default path on all WebSphere Application Server Cluster nodes which will be used to deploy the resource manager clones. In our case, the cluster will be synchronized across the CLAPP1 and CLAPP2 nodes.

8. Verify that the individual resource manager daemons start and stop. Use /etc/rc.cmrmproc on both CLAPP1 and CLAPP2. For example:

```
# /etc/rc.cmrmproc -act start -proc RMMigrator
# /etc/rc.cmrmproc -act stop -proc RMMigrator
```

9. Start the Content Manager resource manager daemons using etc/rc.cmrmproc:

```
# /etc/rc.cmrmproc
2004-01-09 23:38:52,468 ICM0000: Licensed Materials - Property of IBM IBM Content Manager for Multiplatforms V8.2
(program number 5724-B19) (c ) Copyright IBM Corp. 1994, 2002, 2003.  All Rights Reserved. US Government Users
Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corporation
RMMigrator --> RMDB
RMPurger --> RMDB
RMReplica --> RMDB
RMStager --> RMDB
```

10. Stop the Content Manager resource manager daemon s using `etc/rc.cmrmproc`:

```
# /etc/rc.cmrmproc -act stop
PROCACTION: stop
2004-01-09 23:47:55,659 ICM0000: Licensed Materials - Property of IBM IBM Content Manager for Multiplatforms V8.2
(program number 5724-B19) (c ) Copyright IBM Corp. 1994, 2002, 2003.  All Rights Reserved. US Government Users
Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corporation
RMMigrator --> RMDB
input: clapp1 7500 shutdown
sending shutdown to process
client waiting for status info
client got down
RMPurger --> RMDB
input: clapp1 7501 shutdown
sending shutdown to process
client waiting for status info
client got down
RMReplica --> RMDB
input: clapp1 7502 shutdown
sending shutdown to process
client waiting for status info
client got down
RMStager --> RMDB
input: clapp1 7503 shutdown
sending shutdown to process
client waiting for status info
client got down
```

7.2.2 Building the Content Manager resource manager agents

This section briefly explains how to use the Sun Cluster Agent Builder to create the wrapper script to start and stop the Content Manager resource manager daemons under the control of the Sun Cluster Manager.

A set of scripts (the high availability agent) was first constructed with the Sun Cluster build-in Agent Builder for a single resource manager daemon: RMMigrator. After this agent was validated, the scripts and resource type definition file were enhanced to accept additional variables, *dbname* and *procname,* for each resource. All four resource manager daemons are made highly available with this customized high availability agent.

In addition, we also created a utility, regcmrmproc, to automate the resource manager daemon resource registration process with Sun Cluster.

The agent is packaged as a tar ball available for ftp download. You can also follow the process outlined in this section to build your own agent.

7.2.2.1 Building high availability agent for RMMigrator using Sun Cluster Agent Builder

1. Start the agent builder executable :

```
# /usr/cluster/bin/scdsbuilder&
```

2. For the Working Directory value, choose a shared location that is accessible from both ND1 and ND2 like /global/home.

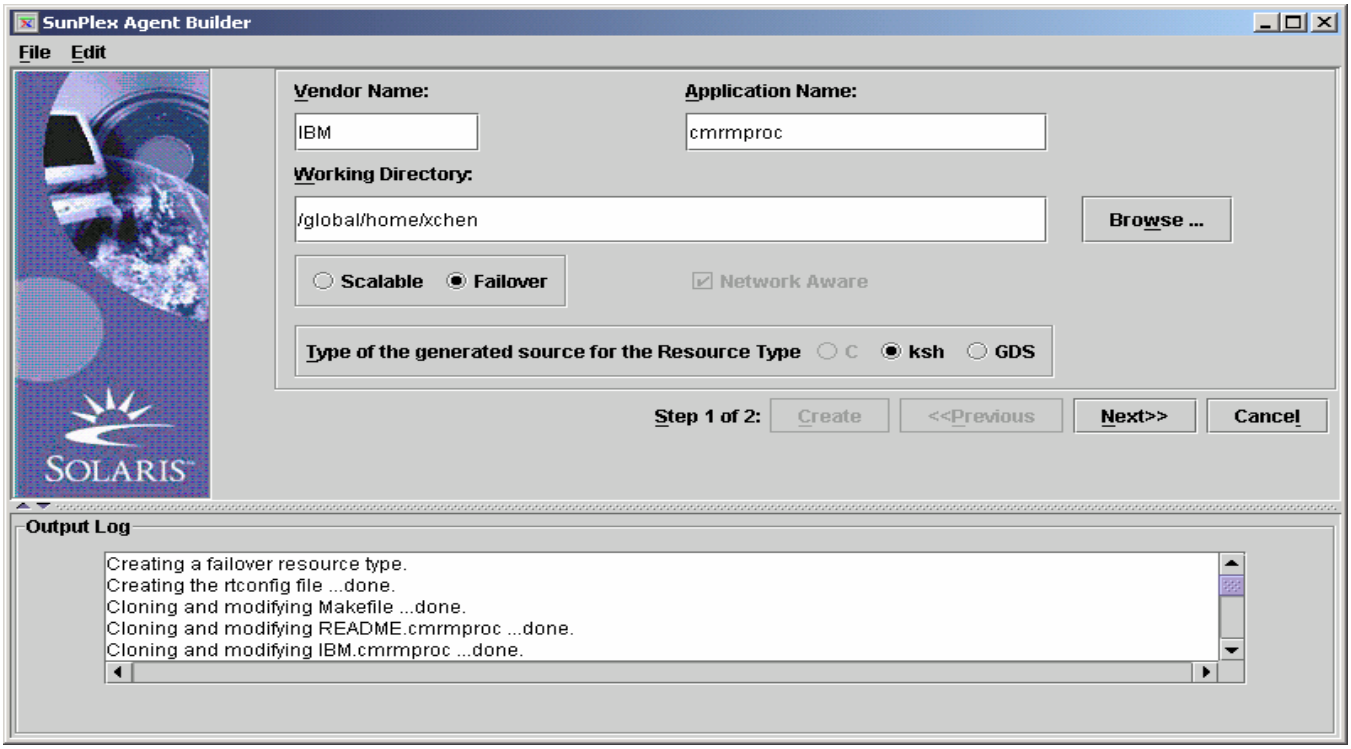


Figure 7-1 Building the agent

3. Enter the necessary information. All the fields are self explanatory. Only three commands are needed: start, stop, and probe.
   Probe

We chose to use Sun Cluster default probe. The return value of the following command can also be used as a probe:

```
/usr/ucb/ps -auxww | grep java | awk "/$dbname/" | grep $baseComp > /dev/null 2>&1
```

Start

```
/etc/rc.cmrmproc -act start -proc $baseComp
```

Stop

```
/etc/rc.cmrmproc -act stop -proc $baseComp
```

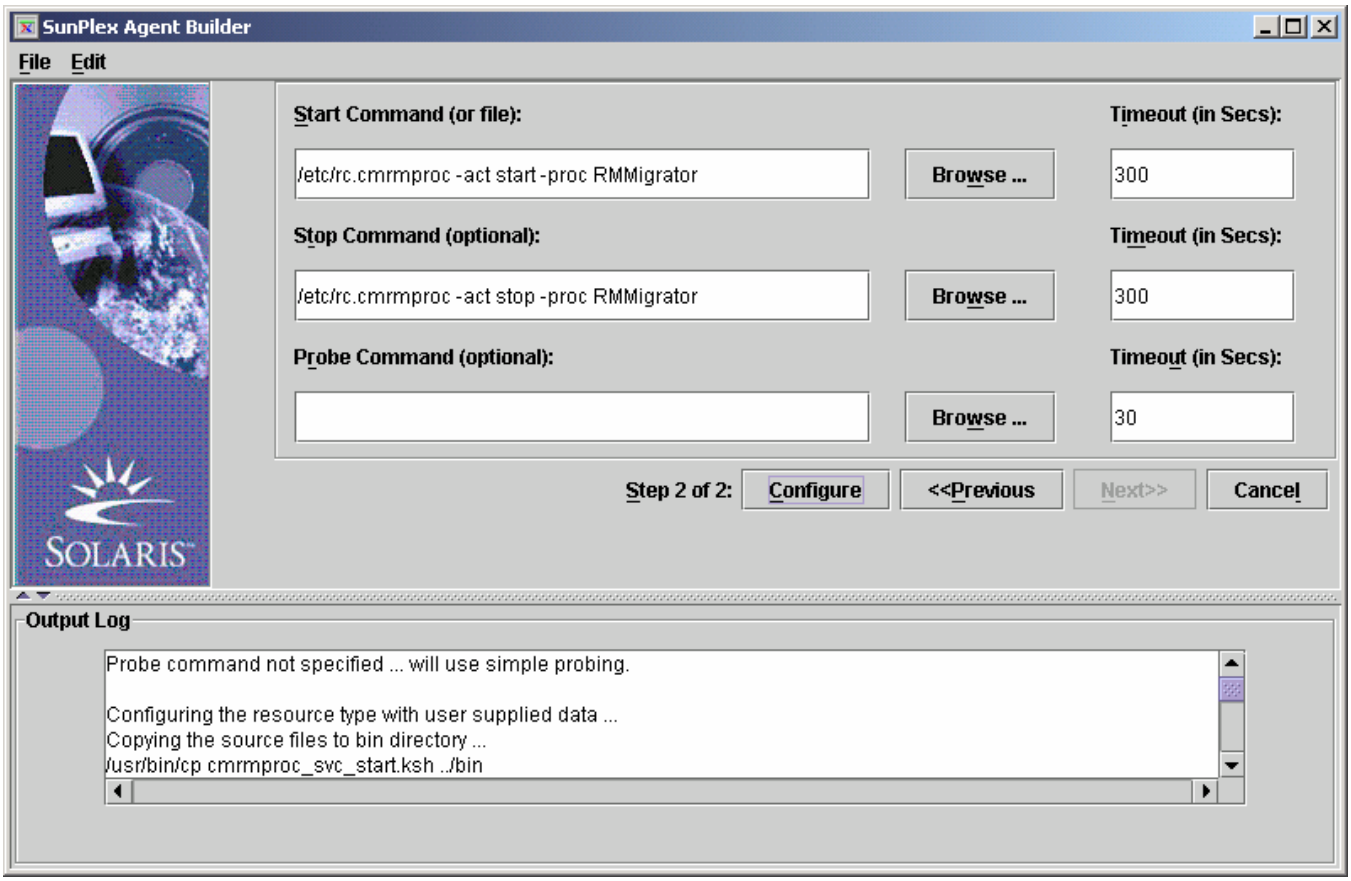


Figure 7-2 Building the Agent

Follow the wizard to the end and the shell script will be generated.

7.2.2.2 Adding additional variables to support all four resource manager daemons

1. Modify IBM.cmrmproc to include the extended properties. Append the following definition to the resource definition file `/global/home/IBMcmrmproc/etc/IBM.cmrmproc`:

```
#
# Extension Properties
#
# Not to be edited by end user

{
  PROPERTY = DBNAME;
  EXTENSION;
  STRINGARRAY;
  TUNABLE = AT_CREATION;
  DESCRIPTION = "Resource Manager DB to Connect to";
}

{
  PROPERTY = PROCNAME;
  EXTENSION;
  STRINGARRAY;
  TUNABLE = AT_CREATION;
  DESCRIPTION = "RM Background Process Name";
}
```

For now, the resource will accept additional *dbname* and *procname* variables. The `/etc/rc.cmrmproc` will use `/opt/IBMicm/config/setprocenv.sh` to obtain other resource manager variables.

The agent can be further extended to take all resource manager variables to support installation of multiple resource manager databases.

Even in a standard environment, since `/etc/rc.cmrmproc` will call the single copy of `/opt/IBMicm/config/setprocenv.sh` as the default, we need to pay attention to namespace collision when starting resource manager daemons with `/etc/rc.cmrmproc`

2. Modify the start, stop, and validate methods to support the other RMProcs. Customize the scripts of these control methods, using examples from the appendix as a guide:

```
START                           =       cmrmproc_svc_start.ksh;
STOP                            =       cmrmproc_svc_stop.ksh;
VALIDATE                        =       cmrmproc_validate.ksh;
```

3. Make sure the scripts are executable:

```
# chmod u+x /global/home/IBMcmrmproc/bin/*.ksh
```

4. Modify the resource definition to reflect the base runtime directory:

```
# vi /global/home/IBMcmrmproc/etc/IBM.cmrmproc
#RT_BASEDIR=/opt/IBMcmrmproc/bin;
RTBASEDIR=/global/home/IBMcmrmproc/bin;
```

7.2.2.3 Registering the Resource Manager Daemons

Add the Resource Manager type:
scrgadm -a -t IBM.cmrmproc -f /global/home/IBMcmrmproc/etc/IBM.cmrmproc

Register the group and resource for each of the 4 processes: RMMigrator, RMPurger, RMReplicator, and RMStager, and then activate them:

```
# scrgadm -a -g RMDB_RMMigrator_rg
# scrgadm -a -j RMDB_RMMigrator_rs -g RMDB_RMMigrator_rg -t IBM.cmrmproc -x DBNAME=RMDB -x
PROCNAME=RMMigrator
# scswitch -Z -g RMDB_RMMigrator _rg

# scrgadm -a -g RMDB_ RMPurger _rg
# scrgadm -a -j RMDB_ RMPurger _rs -g RMDB_ RMPurger _rg -t IBM.cmrmproc -x DBNAME=RMDB -x
PROCNAME= RMPurger
# scswitch -Z -g RMDB_ RMPurger_rg

# scrgadm -a -g RMDB_ RMReplicator _rg
# scrgadm -a -j RMDB_ RMReplicator _rs -g RMDB_ RMReplicator _rg -t IBM.cmrmproc -x DBNAME=RMDB -x
PROCNAME= RMReplicator
# scswitch -Z -g RMDB_ RMReplicator_rg

# scrgadm -a -g RMDB_ RMStager _rg
# scrgadm -a -j RMDB_ RMStager _rs -g RMDB_ RMStager _rg -t IBM.cmrmproc -x DBNAME=RMDB -x
PROCNAME= RMStager
# scswitch -Z -g RMDB_ RMStager_rg
```

7.2.2.4 Check the resource status:

```
bash-2.05# scstat -g

-- Resource Groups and Resources --

         Group Name          Resources
         ----------          ---------
 Resources: RMDB_RMMigrator_rg  RMDB_RMMigrator_rs
 Resources: RMDB_RMPurger_rg    RMDB_RMPurger_rs
 Resources: RMDB_RMReplicator_rg RMDB_RMReplicator_rs
 Resources: RMDB_RMStager_rg    RMDB_RMStager_rs


-- Resource Groups --

         Group Name          Node Name           State
         ----------          ---------           -----
     Group: RMDB_RMMigrator_rg  clapp1             Online
     Group: RMDB_RMMigrator_rg  clapp2             Offline

     Group: RMDB_RMPurger_rg    clapp1             Online
     Group: RMDB_RMPurger_rg    clapp2             Offline

     Group: RMDB_RMReplicator_rg clapp1            Online
     Group: RMDB_RMReplicator_rg clapp2            Offline

     Group: RMDB_RMStager_rg    clapp1             Online
     Group: RMDB_RMStager_rg    clapp2             Offline


-- Resources --

         Resource Name       Node Name           State     Status Message
         -------------       ---------           -----     --------------
  Resource: RMDB_RMMigrator_rs  clapp1             Online    Online
  Resource: RMDB_RMMigrator_rs  clapp2             Offline   Offline

  Resource: RMDB_RMPurger_rs    clapp1             Online    Online
  Resource: RMDB_RMPurger_rs    clapp2             Offline   Offline

  Resource: RMDB_RMReplicator_rs clapp1            Online    Online
  Resource: RMDB_RMReplicator_rs clapp2            Offline   Offline

  Resource: RMDB_RMStager_rs    clapp1             Online    Online
  Resource: RMDB_RMStager_rs    clapp2             Offline   Offline


 Verify that all resource manager daemons are running:
 # /usr/ucb/ps -auxww | grep -i java | awk "/RMDB/"
```

7.2.2.5 Test the high availability enabled the RMMigrator process

Test1: Kill -9: In place Auto Restart

In this test, the process is terminated using “kill –9” to verify the auto restart protection provided by Sun Cluster through the high availability agent.

```
# /usr/ucb/ps -auxww | grep java  | awk "/RMMigrator/"
root     24759  0.3  0.321355247728 ?        S 04:07:22  0:07
/opt/WebSphere/AppServer/java/bin/../bin/sparc/native_threads/java -Xms128m -Xmx256m
com.ibm.mm.icmrm.process.RMMigratorControl 7500 180 RMDB
# kill -9 24759
# /usr/ucb/ps -auxww | grep java  | awk "/RMMigrator/"
root     24879  6.4  0.318012038184 ?        O 04:09:38  0:06
/opt/WebSphere/AppServer/java/bin/../bin/sparc/native_threads/java -Xms128m -Xmx256m
com.ibm.mm.icmrm.process.RMMigratorControl 7500 180 RMDB
```

We observed that right after the original RMMigrator (pid: 24759) was terminated, the new RMMigrator (PID: 24879) was initiated.

Test2: Simulated failover

In this test, the RMMigrator process is switched over from the running system to the backup system. It simulated an unrecoverable hardware failure.

```
# scstat -g
# scswitch -z -g RMDB_RMMigrator_rg -h clapp2
# scstat -g
```

At the beginning of the test, the RMMigrator was running on clapp.1. After the scswitch command, the RMMigrator resource was terminated on clapp1 and brought online on clapp2. Also, the system log shows the agent kickstarted the RMMigrator resource on Clapp2 with `cmrmproc_svc_start.ksh`.

8. Install and configure WebSphere NetDispatcher

Perform the following steps on ND1 and ND2:

1. Update the /etc/hosts file. Add the following line to all hosts:
   ```
   10.1.19.100      rmweb
   ```

   For example:
   ```
   # more /etc/hosts
   127.0.0.1       localhost
   10.1.19.110     ND1   loghost
   10.1.19.103     clapp1
   10.1.19.102     cldb1
   10.1.19.104     cldb2
   10.1.19.105     clapp2
   10.1.19.201     lsdbsrv
   10.1.19.202     rmdbsrv
   10.1.19.110     ND1
   10.1.19.111     ND2
   10.1.19.112     off5c
   10.1.19.100     rmweb
   ```
2. Enter the following commands to start the installation:

   ```
   # cd  /net/clapp2/stage/source/was51edge
   # ./install
   ```
3. Follow the prompts to complete the installation.

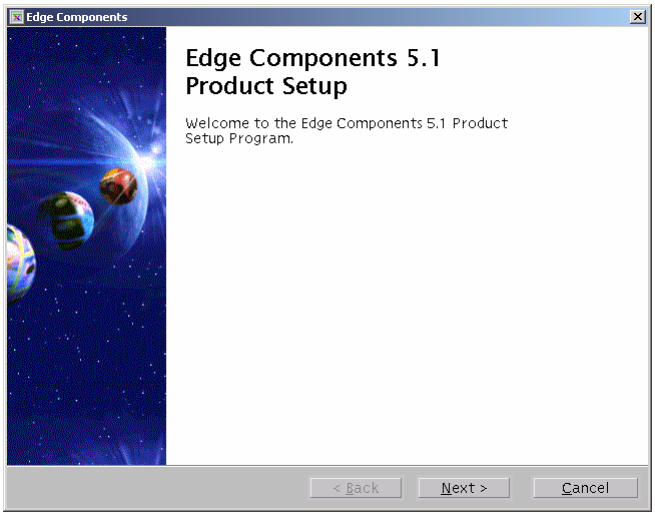


Figure 8-1 WebSphere Edge Components installation

Select **Dispatcher**.

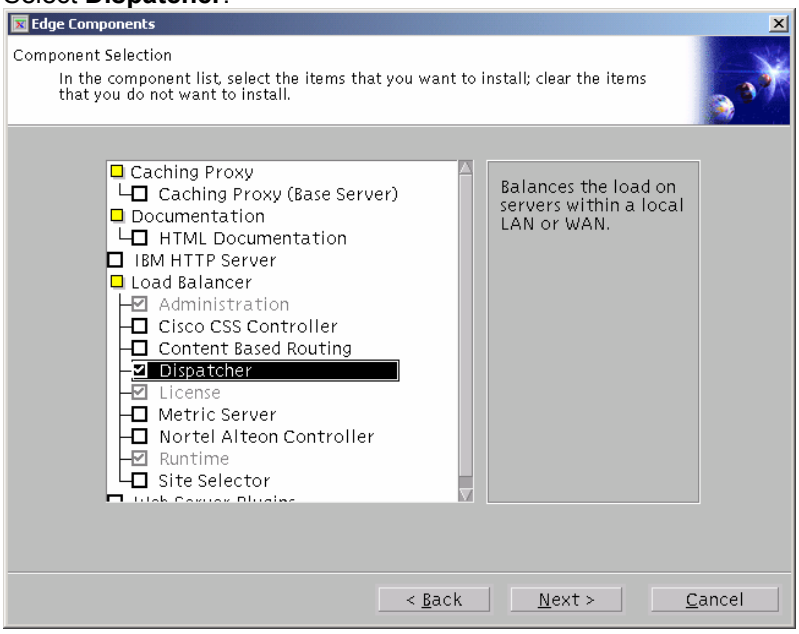


Figure 8-2 Detdispatcher

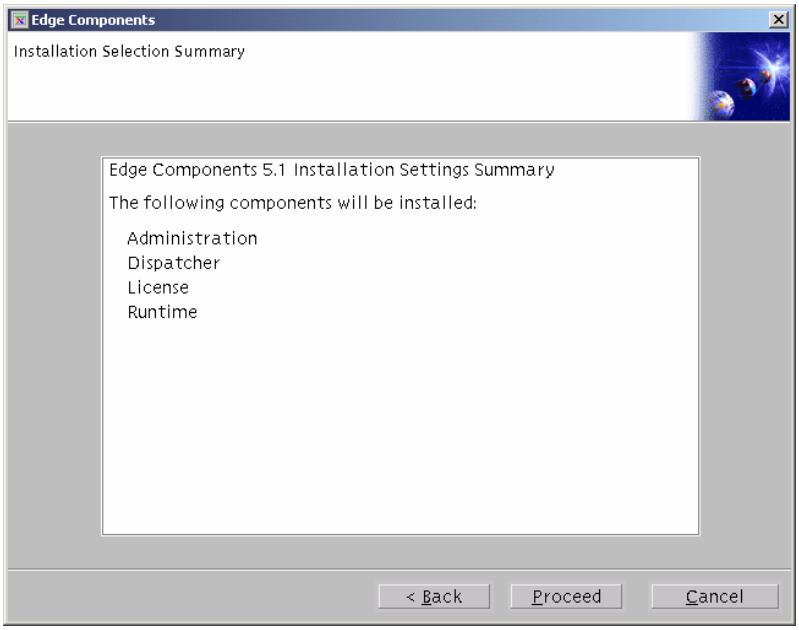


Figure 8-3 Install NetDispacther

This part provides procedures for building basic demonstration networks using the WebSphere Application Server Edge components.

8.1.1 Configure the Dispatcher using the command line

Reference: *WebSphere Application Server Concepts,Planning,and Installation for Edge Components Version 5.1*, Chapter 16. Build a Load Balancer network.

1. Modify the `/etc/hosts` files for ND1, ND2, clapp1, clapp2, and all client machines to add an entry for rmweb:
   ```
   10.1.19.100      rmweb
   ```
2. Modify the `/etc/hosts` files for ND1 and ND2 to add an entries for clapp1 and clapp2:
   ```
   10.1.19.103      clapp1
   10.1.19.104      clapp2
   ```
3. Add an alias for rmweb to the loopback interface on clapp1.

   "NOTE: Make sure that this configuration is also valid after reboot."
   # ifconfig lo0:1 plumb rmweb up
   # ifconfig –a
   lo0: flags=1000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1 inet 127.0.0.1 netmask ff000000
   lo0:1: flags=1000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1 inet 10.1.19.100 netmask ff000000
   hme0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2 inet 10.1.19.105 netmask ffffff00 broadcast 10.1.19.255
   ether 8:0:20:c5:bc:1f qfe1: flags=1008843<UP,BROADCAST,RUNNING,MULTICAST,PRIVATE,IPv4> mtu 1500 index 3
   inet 172.16.1.2 netmask ffffff80 broadcast 172.16.1.127
   ether 0:3:ba:22:a5:73 qfe0: flags=1008843<UP,BROADCAST,RUNNING,MULTICAST,PRIVATE,IPv4> mtu 1500 inndex 4
   inet 172.16.0.130 netmask ffffff80 broadcast 172.16.0.255
   ether 0:3:ba:22:a5:72 clprivnet0:
   flags=1009843<UP,BROADCAST,RUNNING,MULTICAST,MULTI_BCAST,PRIVATE,IPv4> mtu 1500 index 5
    inet 172.16.193.2 netmask ffffff00 broadcast 172.16.193.255
   ether 0:0:0:0:0:2

4. Repeat step 2 on CLAPP2.

On ND1, configure the Dispatcher:

```
# dsserver

# dscontrol executor start
Loaded kernel successfully.

# dscontrol cluster add rmweb
Cluster rmweb has been added.

# dscontrol port add rmweb:80
Port 80 successfully added to cluster rmweb.

# dscontrol server add rmweb:80:clapp1
Server clapp1 was added to port 80 of cluster rmweb.

# dscontrol server add rmweb:80:clapp2
Server clapp2 was added to port 80 of cluster rmweb.

# dscontrol executor configure rmweb
Created new logical interface hme0:1
Address 10.1.19.100 has been configured.

# dscontrol manager start
The manager has been started.

# dscontrol advisor start http 80
Advisor 'http' has been started on port 80.
Proportions 49 50 1 0 for cluster rmweb successfully set.
```

8.2 Enable NetDispatcher for high availability

On ND1, enter the following commands:

```
# dscontrol cluster set rmweb primaryhost ND1
Cluster field(s) successfully set.

# dscontrol highavailability heartbeat add ND1 ND2
Heartbeat '10.1.19.110' to '10.1.19.111' successfully added.

# dscontrol highavailability backup add primary auto 4567
Backup information successfully added.

# dscontrol highavailability status

High Availability Status:
Role ..................... Primary
Recovery strategy .... Auto
State ..................... Active
Sub-state ..... Not Synchronized
Primary host ..... 10.1.19.110
Port ..... 4567
Preferred target ..... n/a
Heartbeat Status:
Count ..... 1
Source/destination ... 10.1.19.110/10.1.19.111
Reachability Status:
Count ..... 0
```

On ND2, enter the following commands:

```
# dsserver

# dscontrol executor start
Loaded kernel successfully.

# dscontrol cluster add rmweb
Cluster rmweb has been added.
# dscontrol cluster set rmweb primaryhost ND1
Cluster field(s) successfully set.

# dscontrol port add rmweb:80
Port 80 successfully added to cluster rmweb.

# dscontrol server add rmweb:80:clapp1
Server clapp1 was added to port 80 of cluster rmweb.

# dscontrol server add rmweb:80:clapp2
Server clapp2 was added to port 80 of cluster rmweb.

# dscontrol manager start
The manager has been started.

# dscontrol advisor start http 80
Advisor 'http' has been started on port 80.
Proportions 49 50 1 0 for cluster rmweb successfully set.

# ifconfig –a
lo0: flags=1000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
hme0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 3
        inet 10.1.19.111 netmask ffffff00 broadcast 10.1.19.255
        ether 8:0:20:a0:57:8e

# dscontrol highavailability heartbeat add ND2 ND1
Heartbeat '10.1.19.111' to '10.1.19.110' successfully added.

# dscontrol highavailability backup add backup auto 4567
Backup information successfully added.

# dscontrol highavailability status
High Availability Status:
Role ..... Backup
Recovery strategy .... Auto
State ..... Standby
\ub-state ..... Synchronized
Primary host ..... 10.1.19.110
Port ..... 4567
Preferred target ..... 10.1.19.110
Heartbeat Status:
Count ..... 1
Source/destination ... 10.1.19.111/10.1.19.110
Reachability Status:
Count ..... 0
```

8.2.1 Create NetDispatcher control scripts

NetDispatcher must have an IP Sprayer the can dispatch HTTP requests to the Content Manager resource manager cluster across nodes. If a node fails, NetDispatcher will detect this fact and automatically direct following requests to the surviving nodes.

This means that NetDispatcher itself can become a single point of failure, so it needs to be configured for high availability. NetDispatcher high availability is achieved by using a second, standby NetDispatcher node paired together with the heartbeat and failover services. This means that our Content Manager high availability environment requires a pair of NetDispatcher nodes, linked together by the heartbeat service. With this configuration, the backup node will take the place of the main one if the latter fails.

For the failover to work, a set of scripts are required, which allow starting, stopping and setting a specific NetDispatcher node as standby. These scripts are user-defined but can be created by modifying samples that come with the product. These samples are in the `/opt/ibm/edge/lb/servers/samples` directory, and must be copied to the `/opt/ibm/edge/lb/servers/bin` directory after they have been modified.

The appendix on page 154 contains customized goActive, goStandby, goInOp, highavailChange scripts used in this project.

8.2.2 Enable NetDispatcher for HTTPS forwarding

On both ND1 and ND2, enter the following commands:

```
# dscontrol port add rmweb:443
Port 443 successfully added to cluster rmweb.

# dscontrol server add rmweb:443:clapp1
Server clapp1 was added to port 443 of cluster rmweb.

# dscontrol server add rmweb:443:clapp2
Server clapp2 was added to port 443 of cluster rmweb.

# dscontrol advisor start https 443
Advisor 'https' has been started on port 443.
```

8.2.3 Starting the WebSphere Application Server NetDispatcher automatically after reboot

Add the following line to the end of `/etc/inittab` on node ND1:

```
nd:3:once:/opt/ibm/edge/lb/servers/bin/ND1.dispatch.sh> /dev/console 2>&1 # Autostart NetDispatcher
```

Add the following line to the end of `/etc/inittab` on node ND2:

```
nd:3:once:/opt/ibm/edge/lb/servers/bin/ND2.dispatch.sh> /dev/console 2>&1 # Autostart NetDispatcher
```

On node ND1, enter the following command:

```
# cat /opt/ibm/edge/lb/servers/bin/ND1.dispatch.sh
```

You should see something like this:

```
#/bin/sh
cd /opt/ibm/edge/lb/servers/bin
dsserver
dscontrol executor start
scontrol cluster add rmweb
dscontrol cluster set rmweb primaryhost ND1
dscontrol port add rmweb:80
dscontrol server add rmweb:80:clapp1
dscontrol server add rmweb:80:clapp2
dscontrol port add rmweb:443
dscontrol server add rmweb:443:clapp1
dscontrol server add rmweb:443:clapp2
dscontrol manager start
dscontrol advisor start http 80
dscontrol advisor start https 443
dscontrol highavailability heartbeat add ND1 ND2
dscontrol highavailability backup add primary auto 4567
dscontrol highavailability status
```

On node ND2, enter the following commands:

```
# pwd
/opt/ibm/edge/lb/servers/bin

# cat ND2.dispatch.sh

# cd /opt/ibm/edge/lb/servers/bin
dsserver
dscontrol executor start
dscontrol cluster add rmweb
dscontrol cluster set rmweb primaryhost ND1
dscontrol port add rmweb:80
dscontrol server add rmweb:80:clapp1
dscontrol server add rmweb:80:clapp2
dscontrol port add rmweb:443
dscontrol server add rmweb:443:clapp1
dscontrol server add rmweb:443:clapp2
dscontrol manager start
dscontrol advisor start http 80
dscontrol advisor start https 443
dscontrol highavailability heartbeat add ND2 ND1
dscontrol highavailability backup add backup auto 4567
dscontrol highavailability status
```

Make sure the WebSphere Application Server Deployment Manager Daemon (dmgr), the WebSphere Application Server Node Agent (nodeagent), and the WebSphere Application Server resource manager Web applications WLM_icmrm and WLM_icmrmVC02 are running.

Use the following command to verify that all server are running:

```
# /opt/WebSphere/AppServer/bin/serverStatus.sh –all
ADMU0116I: Tool information is being logged in file
        /opt/WebSphere/AppServer/logs/serverStatus.log
ADMU0500I: Retrieving server status for all servers
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: server1
ADMU0506I: Server name: icmrm
ADMU0506I: Server name: nodeagent
ADMU0506I: Server name: WLM_icmrm
ADMU0506I: Server name: WLM_icmrmVC02
```

8.3 Verification

Issue the following URL Query:

```
http:// rmweb/icmrm/ICMResourceManager?order=heartbeat&libname=icmnlsdb
```

The response should be:

```
IBM Content Manager V8
Your request: heartbeat
Return Code: 0
Message text: ICM0000: OK
CM Windows Client:
```

If you do not get this response, see Troubleshooting resource manager database login problems on page 132.

9. Deploying the Content Manager V8.2 eClient in a WebSphere Application Server Cluster

The Content Manager eClient is a Web application much like the resource manager is. As such the eClient installation can be done the same way we did the Content Manager resource manager installation.

Create a new WebSphere cluster to host the eClient and deploy the eClient using the method described in Installing the Content Manager V8 resource manager Web application on page 94.

What needs to be done:	CIdb1	CIdb2	CIApp1	CIApp2	Comment
Fourth step on application nodes:					
Create eClient WebSphere Application Server cluster			X	X	Reverse order, eClient prime = resource manager secondary
Install Information Integrator for Content			X	X	
Install Content Manager eClient binaries			X	X	
Test and tune the cluster					

10. Appendix

10.1 Fully-configured WebSphere http plug-in.xml

```
A fully configured CM V8 RM Cluster using WebSphere V5.0.2 FP2
<?xml version="1.0" encoding="ISO-8859-1"?>
<Config ASDisableNagle="false" IISDisableNagle="false"
    IgnoreDNSFailures="false" RefreshInterval="60" ResponseChunkSize="64">
    <Log LogLevel="Error" Name="/opt/WebSphere/DeploymentManager/logs/http_plugin.log"/>
    <Property Name="ESIEnable" Value="true"/>
    <Property Name="ESIMaxCacheSize" Value="1024"/>
    <Property Name="ESIInvalidationMonitor" Value="false"/>
    <VirtualHostGroup Name="default_host">
        <VirtualHost Name="*:9080"/>
        <VirtualHost Name="*:80"/>
        <VirtualHost Name="*:9443"/>
        <VirtualHost Name="*:9081"/>
        <VirtualHost Name="*:9081"/>
        <VirtualHost Name="*:9444"/>
        <VirtualHost Name="*:9082"/>
        <VirtualHost Name="*:9082"/>
    </VirtualHostGroup>
    <ServerCluster CloneSeparatorChange="false"
        LoadBalance="Round Robin" Name="CMRMCl01"
        PostSizeLimit="10000000" RemoveSpecialHeaders="true" RetryInterval="60">
        <Server CloneID="v7p2pu21" ConnectTimeout="0"
            ExtendedHandshake="false" LoadBalanceWeight="2"
            MaxConnections="0" Name="clapp1_WLM_icmrm" WaitForContinue="false">
            <Transport Hostname="clapp1" Port="9082" Protocol="http"/>
            <Transport Hostname="clapp1" Port="9445" Protocol="https">
                <Property Name="keyring" Value="/opt/WebSphere/DeploymentManager/etc/plugin-
key.kdb"/>
                <Property Name="stashfile" Value="/opt/WebSphere/DeploymentManager/etc/plugin-
key.sth"/>
                <Property Name="certLabel" Value="selfsigned"/>
            </Transport>
        </Server>
        <Server CloneID="v7q6ho1c" ConnectTimeout="0"
            ExtendedHandshake="false" LoadBalanceWeight="2"
            MaxConnections="0" Name="clapp2_WLM_icmrm2" WaitForContinue="false">
            <Transport Hostname="clapp2" Port="9083" Protocol="http"/>
            <Transport Hostname="clapp2" Port="9446" Protocol="https">
                <Property Name="keyring" Value="/opt/WebSphere/DeploymentManager/etc/plugin-
key.kdb"/>
                <Property Name="stashfile" Value="/opt/WebSphere/DeploymentManager/etc/plugin-
key.sth"/>
                <Property Name="certLabel" Value="selfsigned"/>
            </Transport>
        </Server>
        <Server CloneID="v7qt5d4r" ConnectTimeout="0"
            ExtendedHandshake="false" LoadBalanceWeight="2"
            MaxConnections="0" Name="clapp1_WLM_icmrmVC02" WaitForContinue="false">
            <Transport Hostname="clapp1" Port="9083" Protocol="http"/>
            <Transport Hostname="clapp1" Port="9446" Protocol="https">
                <Property Name="keyring" Value="/opt/WebSphere/DeploymentManager/etc/plugin-
key.kdb"/>
                <Property Name="stashfile" Value="/opt/WebSphere/DeploymentManager/etc/plugin-
key.sth"/>
                <Property Name="certLabel" Value="selfsigned"/>
            </Transport>
        </Server>
        <Server CloneID="v7qtga77" ConnectTimeout="0"
            ExtendedHandshake="false" LoadBalanceWeight="2"
            MaxConnections="0" Name="clapp2_WLM_rm2VC02" WaitForContinue="false">
```

```
            <Transport Hostname="clapp2" Port="9084" Protocol="http"/>
            <Transport Hostname="clapp2" Port="9447" Protocol="https">
                <Property Name="keyring" Value="/opt/WebSphere/DeploymentManager/etc/plugin-
key.kdb"/>
                <Property Name="stashfile" Value="/opt/WebSphere/DeploymentManager/etc/plugin-
key.sth"/>
                <Property Name="certLabel" Value="selfsigned"/>
            </Transport>
        </Server>
        <PrimaryServers>
            <Server Name="clapp1_WLM_icmrm"/>
            <Server Name="clapp2_WLM_icmrm2"/>
            <Server Name="clapp1_WLM_icmrmVC02"/>
            <Server Name="clapp2_WLM_rm2VC02"/>
        </PrimaryServers>
    </ServerCluster>
    <ServerCluster CloneSeparatorChange="false"
        LoadBalance="Round Robin" Name="dmgr_clapp1Manager_Cluster"
        PostSizeLimit="10000000" RemoveSpecialHeaders="true" RetryInterval="60">
        <Server ConnectTimeout="0" ExtendedHandshake="false"
            MaxConnections="0" Name="clapp1Manager_dmgr" WaitForContinue="false"/>
        <PrimaryServers>
            <Server Name="clapp1Manager_dmgr"/>
        </PrimaryServers>
    </ServerCluster>
    <ServerCluster CloneSeparatorChange="false"
        LoadBalance="Round Robin" Name="server1_clapp1_Cluster"
        PostSizeLimit="10000000" RemoveSpecialHeaders="true" RetryInterval="60">
        <Server ConnectTimeout="0" ExtendedHandshake="false"
            MaxConnections="0" Name="clapp1_server1" WaitForContinue="false">
            <Transport Hostname="clapp1" Port="9080" Protocol="http"/>
            <Transport Hostname="clapp1" Port="9443" Protocol="https">
                <Property Name="keyring" Value="/opt/WebSphere/DeploymentManager/etc/plugin-
key.kdb"/>
                <Property Name="stashfile" Value="/opt/WebSphere/DeploymentManager/etc/plugin-
key.sth"/>
                <Property Name="certLabel" Value="selfsigned"/>
            </Transport>
        </Server>
        <PrimaryServers>
            <Server Name="clapp1_server1"/>
        </PrimaryServers>
    </ServerCluster>
    <ServerCluster CloneSeparatorChange="false"
        LoadBalance="Round Robin" Name="icmrm_clapp1_Cluster"
        PostSizeLimit="10000000" RemoveSpecialHeaders="true" RetryInterval="60">
        <Server ConnectTimeout="0" ExtendedHandshake="false"
            MaxConnections="0" Name="clapp1_icmrm" WaitForContinue="false">
            <Transport Hostname="clapp1" Port="9081" Protocol="http"/>
            <Transport Hostname="clapp1" Port="9444" Protocol="https">
                <Property Name="keyring" Value="/opt/WebSphere/DeploymentManager/etc/plugin-
key.kdb"/>
                <Property Name="stashfile" Value="/opt/WebSphere/DeploymentManager/etc/plugin-
key.sth"/>
                <Property Name="certLabel" Value="selfsigned"/>
            </Transport>
        </Server>
        <PrimaryServers>
            <Server Name="clapp1_icmrm"/>
        </PrimaryServers>
    </ServerCluster>
    <ServerCluster CloneSeparatorChange="false"
        LoadBalance="Round Robin" Name="server1_clapp2_Cluster"
        PostSizeLimit="10000000" RemoveSpecialHeaders="true" RetryInterval="60">
        <Server ConnectTimeout="0" ExtendedHandshake="false"
            MaxConnections="0" Name="clapp2_server1" WaitForContinue="false">
            <Transport Hostname="clapp2" Port="9080" Protocol="http"/>
            <Transport Hostname="clapp2" Port="9443" Protocol="https">
```

```
                <Property Name="keyring" Value="/opt/WebSphere/DeploymentManager/etc/plugin-
key.kdb"/>
                <Property Name="stashfile" Value="/opt/WebSphere/DeploymentManager/etc/plugin-
key.sth"/>
                <Property Name="certLabel" Value="selfsigned"/>
            </Transport>
        </Server>
        <PrimaryServers>
            <Server Name="clapp2_server1"/>
        </PrimaryServers>
    </ServerCluster>
    <UriGroup Name="default_host_CMRMCl01_URIs">
        <Uri AffinityCookie="JSESSIONID"
            AffinityURLIdentifier="jsessionid" Name="/icmrm/*"/>
    </UriGroup>
    <Route ServerCluster="CMRMCl01"
        UriGroup="default_host_CMRMCl01_URIs" VirtualHostGroup="default_host"/>
    <UriGroup Name="default_host_server1_clapp1_Cluster_URIs">
        <Uri AffinityCookie="JSESSIONID"
            AffinityURLIdentifier="jsessionid" Name="/snoop/*"/>
    </UriGroup>
    <Route ServerCluster="server1_clapp1_Cluster"
        UriGroup="default_host_server1_clapp1_Cluster_URIs" VirtualHostGroup="default_host"/>
    <RequestMetrics armEnabled="false" newBehavior="false"
        rmEnabled="false" traceLevel="HOPS">
        <filters enable="false" type="URI">
            <filterValues enable="false" value="/servlet/snoop"/>
            <filterValues enable="false" value="/webapp/examples/HitCount"/>
        </filters>
        <filters enable="false" type="SOURCE_IP">
            <filterValues enable="false" value="255.255.255.255"/>
            <filterValues enable="false" value="254.254.254.254"/>
        </filters>
    </RequestMetrics>
</Config>
```

10.2 WebSphere Application Server NetDispatcher control scripts

10.2.1 goActive script

```
 #!/bin/ksh
# 5630-A36,  (C) COPYRIGHT International Business Machines Corp., 1998, 2002
# All Rights Reserved * Licensed Materials - Property of IBM
#
# goActive script
#
# Configure this script when using the high availability feature of
# Network Dispatcher.
#
# This script is executed when Network Dispatcher goes into the
# 'Active' state and begins routing packets.
#
# This script must be placed in Network Dispatcher's bin directory
# (/opt/ibm/edge/lb/servers/bin/) and it needs to have root execute permission.
#
# le0=Ethernet, hme0=fast Ethernet, ge0=Gigabit Ethernet
# The netmask must be the netmask of your LAN.  It may be hexadecimal or octal notation.
#
#LB_LOGDIR=/opt/ibm/edge/lb/servers/logs/dispatcher/
CLUSTER=rmweb
# For Solaris 8, remove the ':alias' at the end of INTERFACE
# Please do not remove interfaces (like: lo0, hme0), remove aliases only.
#INTERFACE=hme0:1
INTERFACE=hme0
LOOPBACK=lo0:1
NETMASK=255.255.255.0
# For Solaris 8, Interface Configuration Commands and Parameters:
INTERFACEPARMS='addif '$CLUSTER' netmask '$NETMASK
 LOOPBACKPARMS='unplumb'
#   date >> $LB_LOGDIR/ha.log
#   print "This machine is Active.  Aliasing cluster address(es) to NIC \n" >>
$LB_LOGDIR/ha.log
ifconfig $LOOPBACK $LOOPBACKPARMS
# where LOOPBACK maps to CLUSTER address
#
ifconfig $INTERFACE $INTERFACEPARMS up :
```

10.2.2 goStandby script

```
#!/bin/ksh
# 5630-A36,  (C) COPYRIGHT International Business Machines Corp., 1998, 2002
# All Rights Reserved * Licensed Materials - Property of IBM
#
# goStandby script
#
# Configure this script when using the high available feature of
# Network Dispatcher.
#
# This script is executed when Network Dispatcher goes into the
# 'Standby' state.  Monitoring the health of the 'Active' machine
# but not routing packets.
#
# This script must be placed in Network Dispatcher's bin directory (by default this
# is /opt/ibm/edge/lb/servers/bin/) and it needs to have root execute permission.
#
# le0=Ethernet, hme0=fast Ethernet, ge0=Gigabit Ethernet
#
#LB_LOGDIR=/opt/ibm/edge/lb/servers/logs/dispatcher
CLUSTER=rmweb
# For Solaris 8, remove the ':alias' at the end of INTERFACE
# Please do not remove interfaces (like: lo0, hme0), remove aliases only.
#INTERFACE=hme0:1
INTERFACE=hme0
LOOPBACK=lo0:1
NETMASK=255.255.255.0
# For Solaris 8, Interface Configuration Commands and Parameters:
 INTERFACEPARMS='removeif '$CLUSTER
 LOOPBACKPARMS='plumb '$CLUSTER' netmask '$NETMASK
#   date >> $LB_LOGDIR/ha.log
#   print "Going into Standby mode. \n" >> $LB_LOGDIR/ha.log
#   print "Deleting the device alias(es) and adding the loopback aliases.\n" >>
LB_LOGDIR/ha.log
ifconfig $INTERFACE $INTERFACEPARMS
ifconfig $LOOPBACK $LOOPBACKPARMS up
```

10.2.3 goInOp script

```
#!/bin/ksh
# 5630-A36,  (C) COPYRIGHT International Business Machines Corp., 1998, 2002
# All Rights Reserved * Licensed Materials - Property of IBM
#
# goInOp script
#
# Configure this script when using the high availability feature of
# Network Dispatcher and optionally when using Network Dispatcher in a
# standalone environment.
#
# This script is executed when the Network Dispatcher executor is stopped
# (and before the executor is initially started).
#
# This script must be placed in Network Dispatcher's bin directory
# (/opt/ibm/edge/lb/servers/bin/) and it needs to have root execute permission.
# Modify CLUSTER and INTERFACE to match your environment.
#
# le0=Ethernet, hme0=fast Ethernet, ge0=Gigabit Ethernet
#
#LB_LOGDIR=/opt/ibm/edge/lb/servers/logs/dispatcher
CLUSTER=rmweb
# For Solaris 8, remove the ':alias' at the end of INTERFACE
# Please do not remove interfaces (like: lo0, hme0), remove aliases only.
# INTERFACE=hme0:1
INTERFACE=hme0
LOOPBACK=lo0:1
# For Solaris 8, Interface Configuration Commands and Parameters:
 INTERFACEPARMS='removeif '$CLUSTER
 LOOPBACKPARMS='unplumb'
#
#   date >> $LB_LOGDIR/ha.log
#   print "Executor has stopped. Removing loopback and device aliases. \n" >>
$LB_LOGDIR/ha.log
ifconfig $LOOPBACK $LOOPBACKPARMS
ifconfig $INTERFACE $INTERFACEPARMS
```

10.2.4 highavailChange script

```
#!/bin/ksh
# 5630-A36,  (C) COPYRIGHT International Business Machines Corp., 2001, 2002
# All Rights Reserved * Licensed Materials - Property of IBM
#
# highavailChange script
#
# Configure this script only if you are using the Highavailability
# feature of Dispatcher.
#
# Configuration of this script is optional.
#
# This script must be placed in Dispatcher's bin directory
# (by default this is /opt/ibm/edge/lb/servers/bin/) and it needs to have
# root execute permission.
#
# This script is executed whenever the Highavailability state
# changes within Dispatcher such that one of the go* scripts is
# called. The single parameter passed to this script is the name
# of the go* script just run by Dispatcher.
#
# This script may use this information in many ways, such as to alert
# an Administrator or simply record the event.  Below is an example
# of how to record the data when Dispatcher has detected a problem.
DATE=`date`
OUTPUT="$DATE LB just ran $1."
echo $OUTPUT >> /opt/ibm/edge/lb/servers/logs/lb.log
echo $OUTPUT | mail root@localhost
```

10.3 Installation of WebSphere Application Server Network Deployment V5.0 with fix pack 2

```
bash-2.05# export WAS_HOME="/opt/WebSphere/DeploymentManager"
bash-2.05# export JAVA_HOME="/opt/WebSphere/DeploymentManager/java"
bash-2.05# ls
docs/                fixpacks/            java_tmp/            updateSilent.sh*     utils/
earLauncher/         installer.jar        lib/                 updateWizard.sh*
version.properties
bash-2.05# env
PWD=/net/clapp2/stage/source/Wasv5/xchen/was50_nd_fp2
TZ=US/Pacific
HZ=100
HOSTNAME=clapp1
WAS_HOME=/opt/WebSphere/DeploymentManager
MACHTYPE=sparc-sun-solaris2.9
MAIL=/var/mail/root
OLDPWD=/net/clapp2/stage/source/Wasv5/xchen
JAVA_HOME=/opt/WebSphere/DeploymentManager/java
DISPLAY=172.16.2.190:0.0
LOGNAME=root
SHLVL=1
SHELL=/sbin/sh
HOSTTYPE=sparc
OSTYPE=solaris2.9
HOME=/
TERM=vt100
PATH=/usr/sbin:/usr/bin:/usr/cluster/bin:/usr/X/bin:
_=/usr/bin/env
bash-2.05# ./updateWizard.sh
```

Figure 10-1 WAS ND fix pack installation

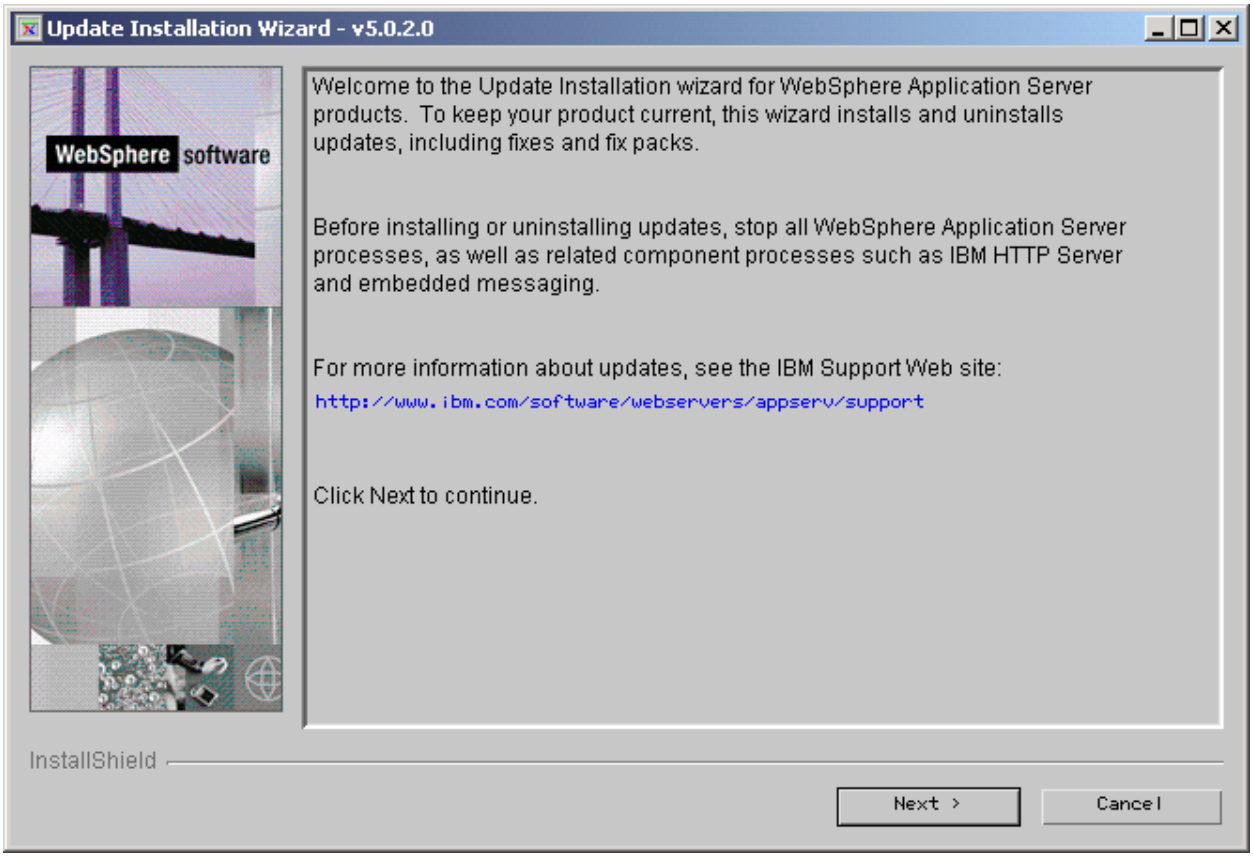


Figure 10-2 WAS FP2 installation

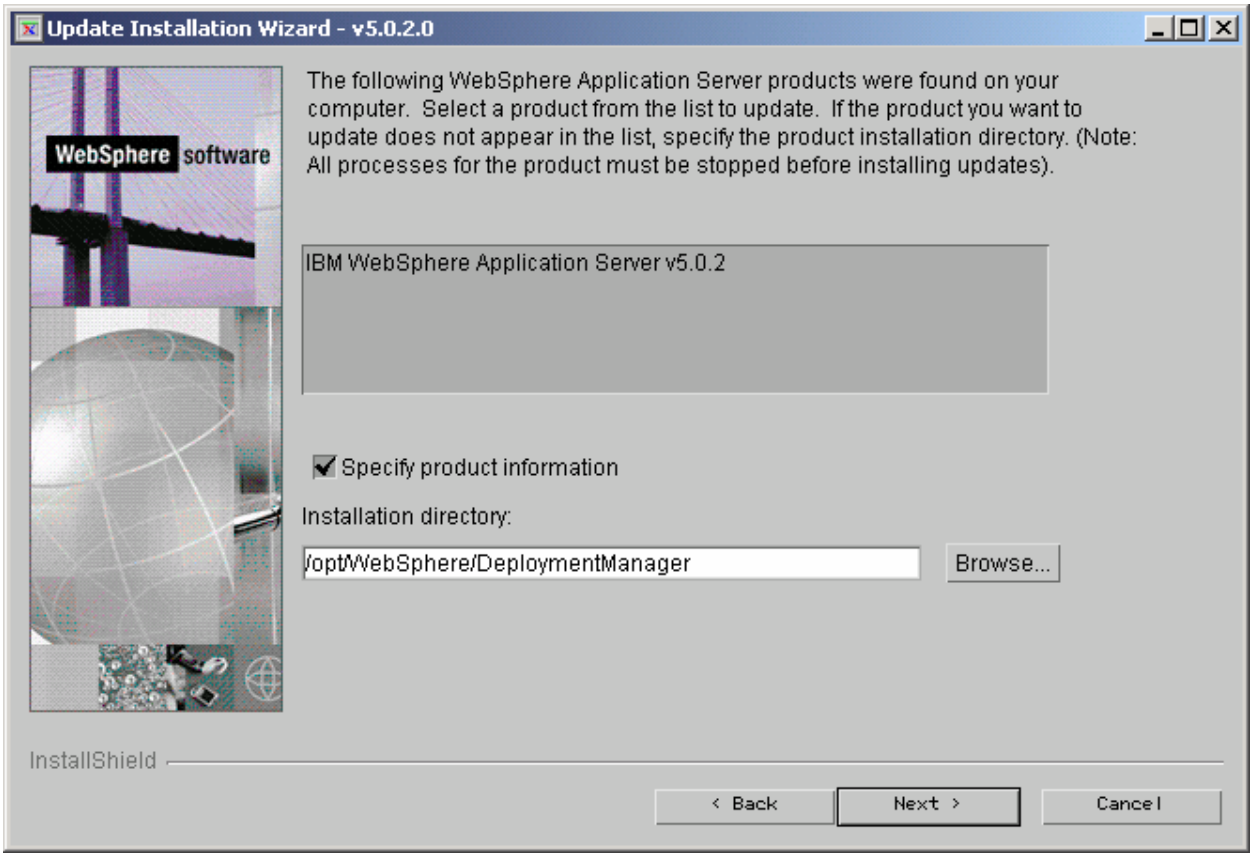


Figure 10-3 WAS FP installation

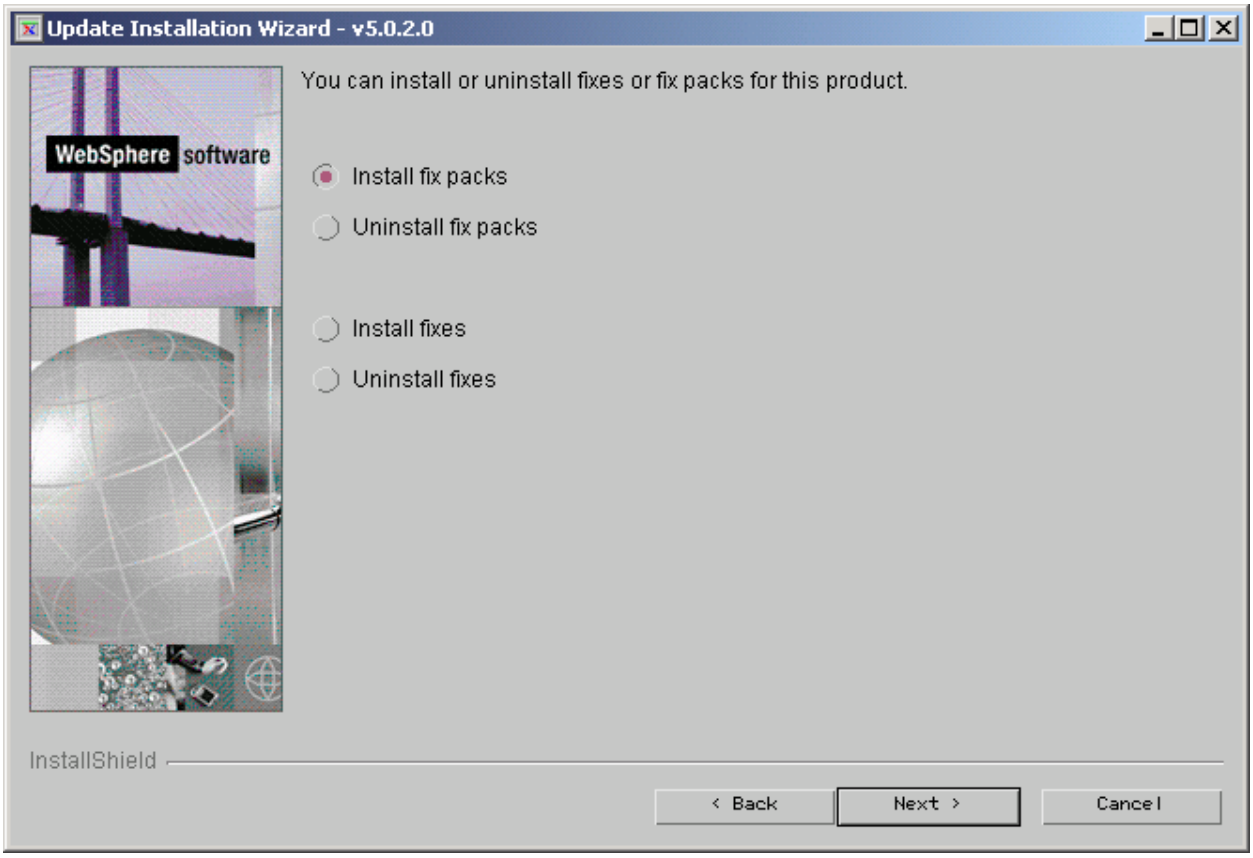


Figure 10-4 WAS FP2 installation

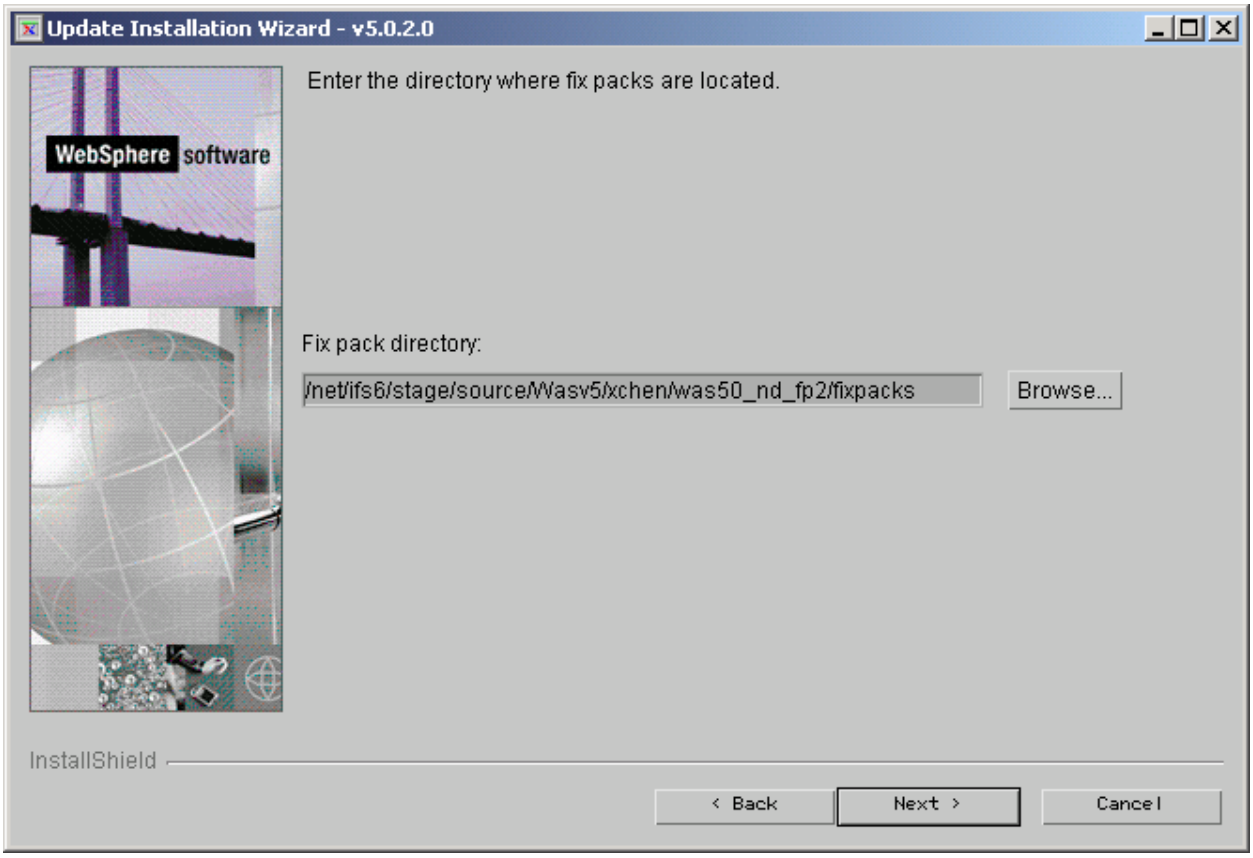


Figure 10-5 WAS FP2 installation

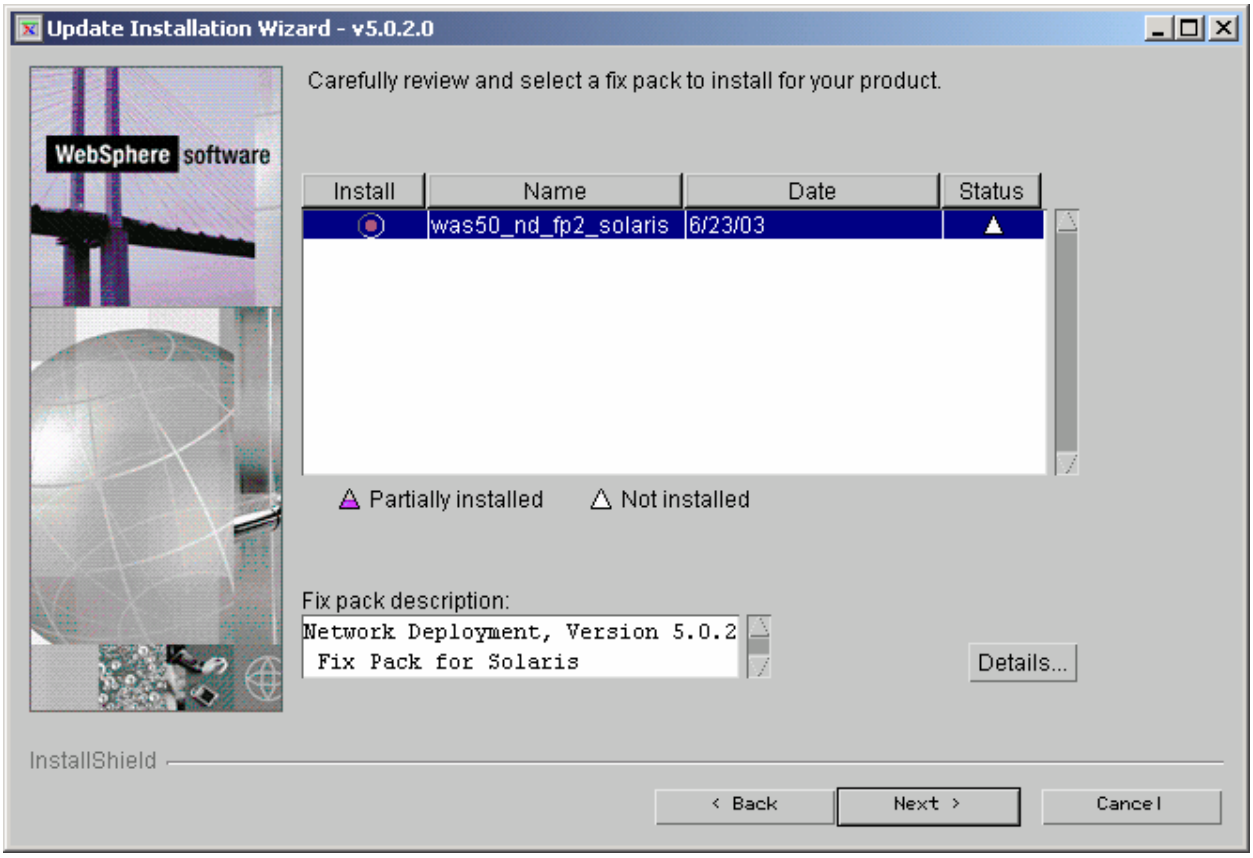


Figure 10-6 WAS F2 installation

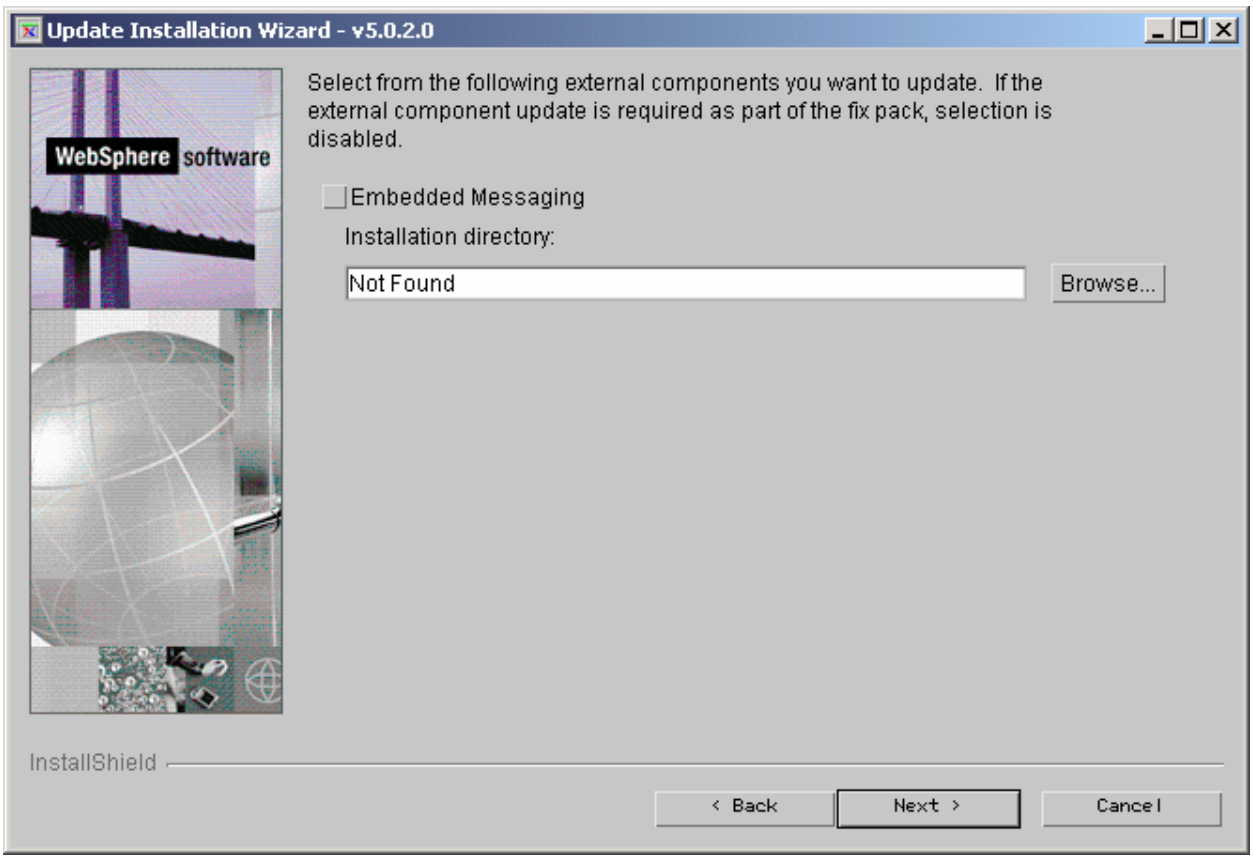


Figure 10-7 WAS F2 installation

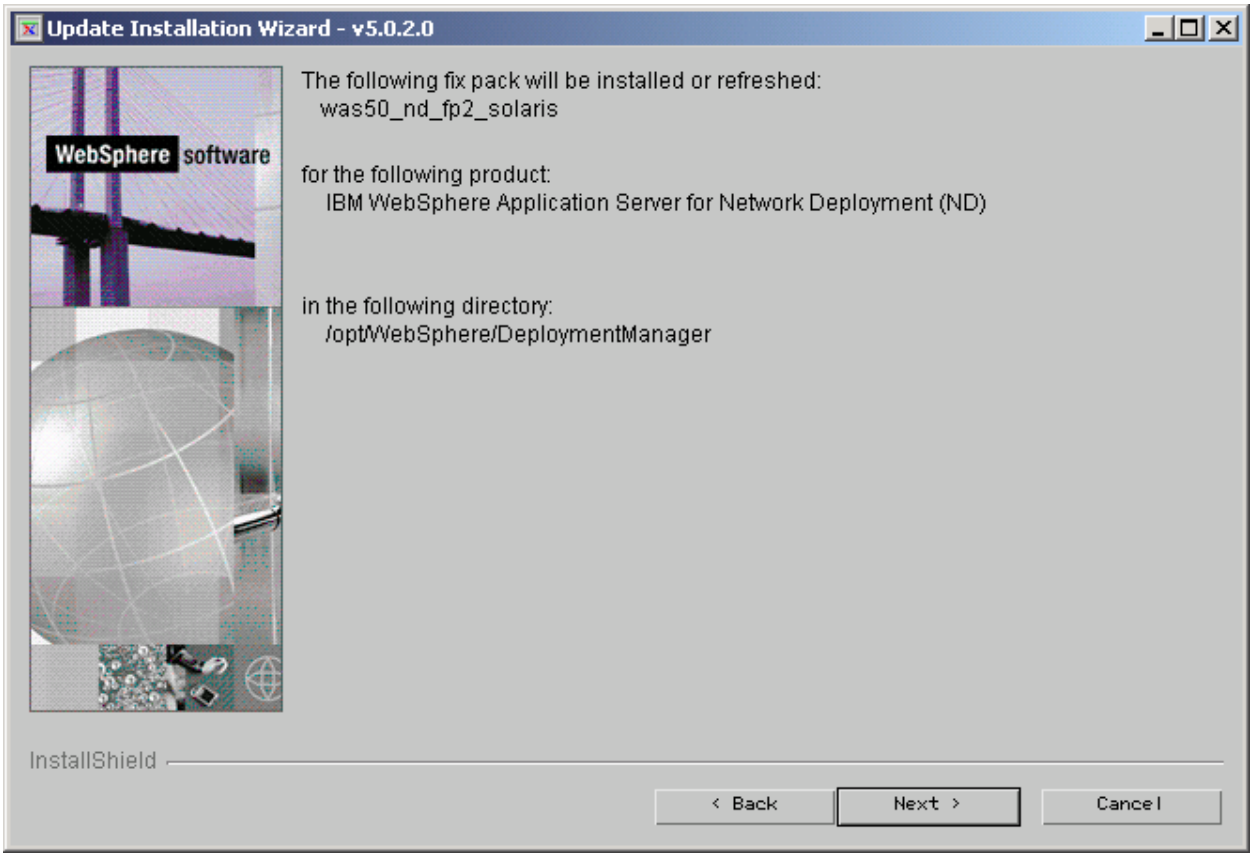


Figure 10-8 WAS F2 installation

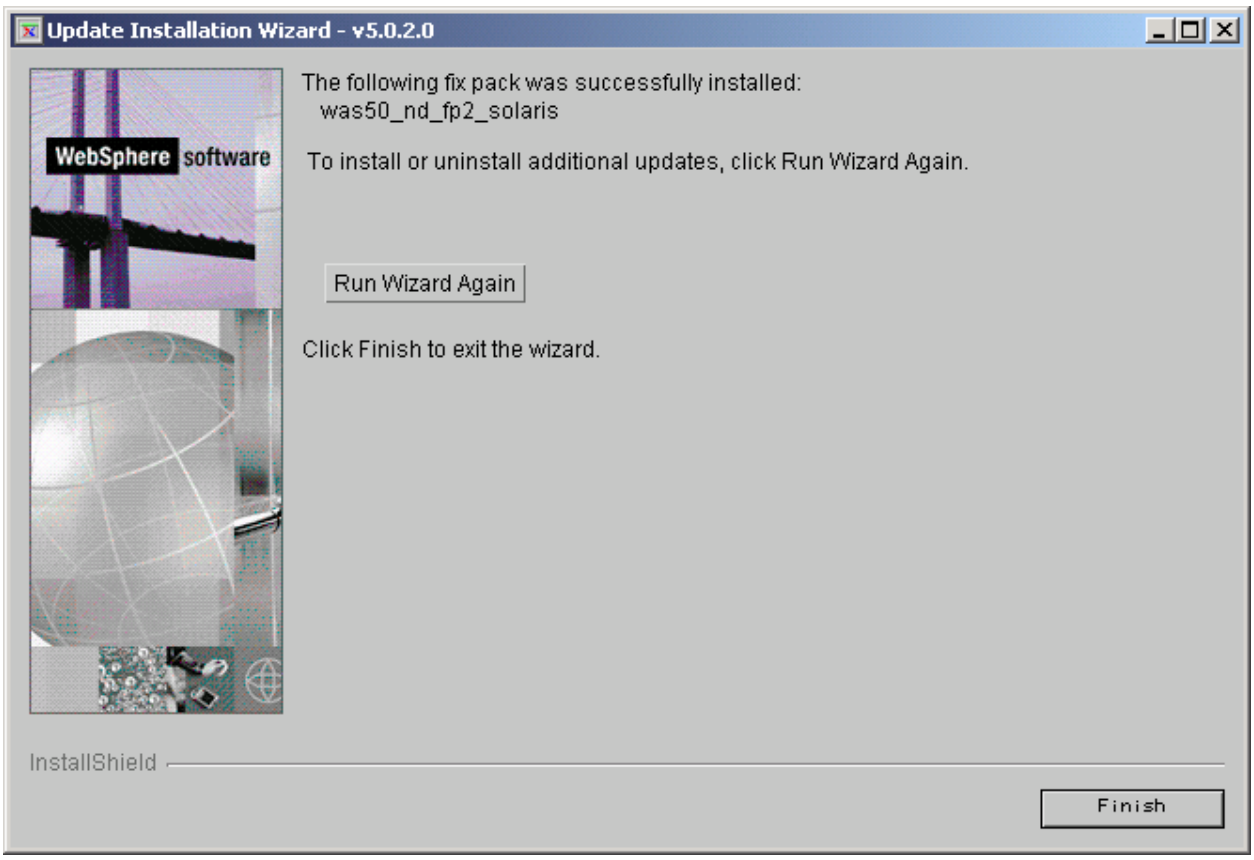


Figure 10-9 WAS F2 installation

10.4 Starting Applications after reboot

The following table lists the 'how' each application/process will be started after a system reboot or failover event.

Component	/etc/inittab	/etc/rc3.d	Cluster	comment
Content Manager library server Database			X	
Content Manager resource manager Database			X	
Content Manager Library Server Monitor	X			
Resource manager: Migrator, Purger, Replicator, Stager			X	
WebSphere Deployment Manager	X			
WebSphere Node agent	X			
WLM_icmrm WLM_vc02 WLM_icmrm2 WLM_icmrm2vc02	X			
HTTP Server		X		
NetDispatcher	X			

Sample start/stop control script for HTTP Server. Similar scripts can be used for other processes. This sample script can be linked as `S14IBMHttp` and `K14IBMHttp` in `/etc/rc3.d`.

```
#!/bin/sh
#
# Apache control script designed to allow an easy command line interface
# to controlling Apache.  Written by Marc Slemko, 1997/08/23
#
# The exit codes returned are:
# 0 - operation completed successfully
# 1 -
# 2 - usage error
# 3 - httpd could not be started
# 4 - httpd could not be stopped
# 5 - httpd could not be started during a restart
# 6 - httpd could not be restarted during a restart
# 7 - httpd could not be restarted during a graceful restart
# 8 - configuration syntax error
#
# When multiple arguments are given, only the error from the _last_
# one is reported.  Run "apachectl help" for usage info
#
#
# |||||||||||||||||||| START CONFIGURATION SECTION  ||||||||||||||||||||
# --------------------                              --------------------
#
# the path to your PID file
PIDFILE=/opt/IBMHttpServer/logs/httpd.pid
#
# the path to your httpd binary, including options if necessary
HTTPD="/opt/IBMHttpServer/bin/httpd -d /opt/IBMHttpServer"
#
```

```
# a command that outputs a formatted text version of the HTML at the
# url given on the command line.  Designed for lynx, however other
# programs may work.
LYNX="lynx -dump"
#
# the URL to your server's mod_status status page.  If you do not
# have one, then status and fullstatus will not work.
STATUSURL="http://localhost/server-status"
#
# --------------------                                    --------------------
# ||||||||||||||||||||   END CONFIGURATION SECTION  ||||||||||||||||||||

ERROR=0
ARGV="$@"
if [ "x$ARGV" = "x" ] ; then
    ARGS="help"
fi

for ARG in $@ $ARGS
do
    # check for pidfile
    if [ -f $PIDFILE ] ; then
  PID=`cat $PIDFILE`
  if [ "x$PID" != "x" ] && kill -0 $PID 2>/dev/null ; then
      STATUS="httpd (pid $PID) running"
      RUNNING=1
  else
      STATUS="httpd (pid $PID?) not running"
      RUNNING=0
  fi
    else
  STATUS="httpd (no pid file) not running"
  RUNNING=0
    fi

    case $ARG in
    start)
  if [ $RUNNING -eq 1 ]; then
      echo "$0 $ARG: httpd (pid $PID) already running"
      continue
  fi
  if $HTTPD ; then
      echo "$0 $ARG: httpd started"
  else
      echo "$0 $ARG: httpd could not be started"
      ERROR=3
  fi
  ;;
    stop)
  if [ $RUNNING -eq 0 ]; then
      echo "$0 $ARG: $STATUS"
      continue
  fi
  if kill $PID ; then
      echo "$0 $ARG: httpd stopped"
  else
      echo "$0 $ARG: httpd could not be stopped"
      ERROR=4
  fi
  ;;
    restart)
  if [ $RUNNING -eq 0 ]; then
      echo "$0 $ARG: httpd not running, trying to start"
      if $HTTPD ; then
          echo "$0 $ARG: httpd started"
      else
          echo "$0 $ARG: httpd could not be started"
          ERROR=5
```

```
    fi
  else
    if $HTTPD -t >/dev/null 2>&1; then
       if kill -HUP $PID ; then
           echo "$0 $ARG: httpd restarted"
       else
           echo "$0 $ARG: httpd could not be restarted"
           ERROR=6
       fi
    else
       echo "$0 $ARG: configuration broken, ignoring restart"
       echo "$0 $ARG: (run 'apachectl configtest' for details)"
       ERROR=6
    fi
  fi
  ;;
    graceful)
  if [ $RUNNING -eq 0 ]; then
    echo "$0 $ARG: httpd not running, trying to start"
    if $HTTPD ; then
       echo "$0 $ARG: httpd started"
    else
       echo "$0 $ARG: httpd could not be started"
       ERROR=5
    fi
  else
    if $HTTPD -t >/dev/null 2>&1; then
       if kill -USR1 $PID ; then
           echo "$0 $ARG: httpd gracefully restarted"
       else
           echo "$0 $ARG: httpd could not be restarted"
           ERROR=7
       fi
    else
       echo "$0 $ARG: configuration broken, ignoring restart"
       echo "$0 $ARG: (run 'apachectl configtest' for details)"
       ERROR=7
    fi
  fi
  ;;
    status)
  $LYNX $STATUSURL | awk ' /process$/ { print; exit } { print } '
  ;;
    fullstatus)
  $LYNX $STATUSURL
  ;;
    configtest)
  if $HTTPD -t; then
     :
  else
     ERROR=8
  fi
  ;;
    *)
  echo "usage: $0 (start|stop|restart|fullstatus|status|graceful|configtest|help)"
  cat <<EOF

start      - start httpd
stop       - stop httpd
restart    - restart httpd if running by sending a SIGHUP or start if not running
fullstatus - dump a full status screen; requires lynx and mod_status enabled
status     - dump a short status screen; requires lynx and mod_status enabled
graceful   - do a graceful restart by sending a SIGUSR1 or start if not running
configtest - do a configuration syntax test
help       - this screen

EOF
  ERROR=2
```

```
    ;;
    esac
done
exit $ERROR

## ====================================================================
## The Apache Software License, Version 1.1
##
## Copyright (c) 2000-2002 The Apache Software Foundation.  All rights
## reserved.
##
## Redistribution and use in source and binary forms, with or without
## modification, are permitted provided that the following conditions
## are met:
##
## 1. Redistributions of source code must retain the above copyright
##    notice, this list of conditions and the following disclaimer.
##
## 2. Redistributions in binary form must reproduce the above copyright
##    notice, this list of conditions and the following disclaimer in
##    the documentation and/or other materials provided with the
##    distribution.
##
## 3. The end-user documentation included with the redistribution,
##    if any, must include the following acknowledgment:
##       "This product includes software developed by the
##        Apache Software Foundation (http://www.apache.org/)."
##    Alternately, this acknowledgment may appear in the software itself,
##    if and wherever such third-party acknowledgments normally appear.
##
## 4. The names "Apache" and "Apache Software Foundation" must
##    not be used to endorse or promote products derived from this
##    software without prior written permission. For written
##    permission, please contact apache@apache.org.
##
## 5. Products derived from this software may not be called "Apache",
##    nor may "Apache" appear in their name, without prior written
##    permission of the Apache Software Foundation.
##
## THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED
## WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
## OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
## DISCLAIMED.  IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR
## ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
## SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
## LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF
## USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
## ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
## OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT
## OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
## SUCH DAMAGE.
## ====================================================================
##
## This software consists of voluntary contributions made by many
## individuals on behalf of the Apache Software Foundation.  For more
## information on the Apache Software Foundation, please see
## <http://www.apache.org/>.
##
## Portions of this software are based upon public domain software
## originally written at the National Center for Supercomputing Applications,
## University of Illinois, Urbana-Champaign.
##
```

11. Resource manager high availability agent for Sun Cluster V3.1

11.1.1 Resource type definition:

 # cat /opt/IBMicm/ha/sc3.x/etc/IBM.cmrmproc

```
# Sun Cluster Data Services Builder template version 1.0
# Registration information and Paramtable for cmrmproc
#
# NOTE: Keywords are case insensitive, i.e. users may use any
# capitalization style they wish
#

RESOURCE_TYPE = "cmrmproc";
VENDOR_ID = IBM;
RT_DESCRIPTION = "cmrmproc server for Sun Cluster";

RT_VERSION ="1.0";
API_VERSION = 2;
FAILOVER = FALSE;

#XCB
#RT_BASEDIR=/opt/IBMcmrmproc/bin;
RT_BASEDIR=/opt/IBMicm/ha/sc3.x/bin;
#XCE
PKGLIST = IBMcmrmproc;

START                           =       cmrmproc_svc_start.ksh;
STOP                            =       cmrmproc_svc_stop.ksh;

VALIDATE                        =       cmrmproc_validate.ksh;
UPDATE                          =       cmrmproc_update.ksh;

MONITOR_START                   =       cmrmproc_mon_start.ksh;
MONITOR_STOP                    =       cmrmproc_mon_stop.ksh;
MONITOR_CHECK                   =       cmrmproc_mon_check.ksh;

# The paramtable is a list of bracketed resource property declarations
# that come after the resource-type declarations
# The property-name declaration must be the first attribute
# after the open curly of a paramtable entry
#
# The following are the system defined properties. Each of the system defined
# properties have a default value set for each of the attributes. Look at
# man rt_reg(4) for a detailed explanation.
#
{
  PROPERTY = Start_timeout;
  MIN = 60;
  DEFAULT = 300;
}
{
  PROPERTY = Stop_timeout;
  MIN = 60;
  DEFAULT = 300;
}
{
  PROPERTY = Validate_timeout;
  MIN = 60;
  DEFAULT = 300;
}
{
        PROPERTY = Update_timeout;
```

```
 MIN = 60;
       DEFAULT = 300;
}
{
 PROPERTY = Monitor_Start_timeout;
 MIN = 60;
 DEFAULT = 300;
}
{
 PROPERTY = Monitor_Stop_timeout;
 MIN = 60;
 DEFAULT = 300;
}
{
 PROPERTY = Monitor_Check_timeout;
 MIN = 60;
 DEFAULT = 300;
}
{
       PROPERTY = FailOver_Mode;
       DEFAULT = SOFT;
       TUNABLE = ANYTIME;
}
{
       PROPERTY = Network_resources_used;
       TUNABLE = WHEN_DISABLED;
 DEFAULT = "";
}
{
 PROPERTY = Cldb1ough_Probe_Interval;
 MAX = 3600;
 DEFAULT = 60;
 TUNABLE = ANYTIME;
}
{
 PROPERTY = Retry_Count;
 MAX = 10;
 DEFAULT = 2;
 TUNABLE = ANYTIME;
}
{
 PROPERTY = Retry_Interval;
 MAX = 3600;
 DEFAULT = 300;
 TUNABLE = ANYTIME;
}

{
 PROPERTY = Port_list;
 DEFAULT = ;
 TUNABLE = AT_CREATION;
}

{
       PROPERTY = Scalable;
 DEFAULT=false;
       TUNABLE = AT_CREATION;
}

{
       PROPERTY = Load_balancing_policy;
       DEFAULT = LB_WEIGHTED;
       TUNABLE = AT_CREATION;
}

{
       PROPERTY = Load_balancing_weights;
       DEFAULT = "";
```

```
        TUNABLE = ANYTIME;
}

#
# Extension Properties
#

# Not to be edited by end user

{
  PROPERTY = Confdir_list;
  EXTENSION;
  STRINGARRAY;
  DEFAULT = "";
  TUNABLE = AT_CREATION;
  DESCRIPTION = "The Configuration Directory Path(s)";
}

# These two control the restarting of the fault monitor itself
# (not the server daemon) by PMF.
{
  PROPERTY = Monitor_retry_count;
  EXTENSION;
  INT;
  DEFAULT = 4;
  TUNABLE = ANYTIME;
  DESCRIPTION = "Number of PMF restarts allowed for the fault monitor";
}

{
  PROPERTY = Monitor_retry_interval;
  EXTENSION;
  INT;
  DEFAULT = 2;
  TUNABLE = ANYTIME;
  DESCRIPTION = "Time window (minutes) for fault monitor restarts";
}

# Time out value for the probe
{
  PROPERTY = Probe_timeout;
  EXTENSION;
  INT;
  MIN = 2;
  DEFAULT = 30;
  TUNABLE = ANYTIME;
  DESCRIPTION = "Time out value for the probe (seconds)";
}

# Child process monitoring level for PMF (-C option of pmfadm)
# Default of -1 means: Do NOT use the -C option to PMFADM
# A value of 0-> indicates the level of child process monitoring
# by PMF that is desired.
{
  PROPERTY = Child_mon_level;
  EXTENSION;
  INT;
  DEFAULT = -1;
  TUNABLE = ANYTIME;
  DESCRIPTION = "Child monitoring level for PMF";
}

#
# Extension Properties
#
# Not to be edited by end user

{
```

```
   PROPERTY = DBNAME;
   EXTENSION;
   STRINGARRAY;
   TUNABLE = AT_CREATION;
   DESCRIPTION = "Resource Manager DB to Connect to";
}

{
   PROPERTY = PROCNAME;
   EXTENSION;
   STRINGARRAY;
   TUNABLE = AT_CREATION;
   DESCRIPTION = "RM Background Process Name";
}
```

11.1.2 Registration utility:

 # cat /opt/IBMicm/ha/sc3.x/util/regcmrmproc

```
#!/bin/ksh
#-------------------------------------------------------------------------
#
# A sample script to register and create appropriate resources and
# resources group for all four RM background processes with SC3.x
#
# The sample script takes a single variable: the resource manager
# database name, and it relys on /opt/IBMicm/config/setprocenv.sh
# for other configuratio parameter setting.
#
# Using the same frame work presented by this sample script,
# the following variables can also be included:
# -dbname         --> Resource Manager Database Name                  #
# -rmappname      --> Resource Manager Application Name               #
# -nodename       --> If using WAS5.x, set the WAS nodename           #
# -was_home       --> WebSphere home installation directory           #
# -insthome       --> db2 instance home use for RM install            #
#
# Yet, besides DBNAME and PROCNAME, other parameters need to be added
# to the Resource Type Definition file (IBM.cmrmproc) as extension of
# tunable parameters. Agent scripts should be change accordingly.
#
#-------------------------------------------------------------------------
#
PATH=/bin:/usr/bin:/usr/cluster/bin:/usr/sbin:$PATH
export PATH
VERBOSE=false

###############################################################################
function usage
{
    printf "Usage: %s -d resource_manager_database_name\n"
}
###############################################################################

function parse_args # [args ...]
{
    typeset opt

    while getopts 'd:v' opt
    do
        case "$opt" in
        v)
            # Set verbose to true
            VERBOSE=true
        ;;
        d)
            # RMDB for the RM background processes
            DBNAME=$OPTARG
        ;;
        *)
            usage
            exit 1
        ;;
        esac
    done
}

###############################################################################
# main {
###############################################################################
unset DBNAME
parse_args "$@"
```

```
if [ -z $DBNAME ]; then
    usage
    exit 1
fi

echo "\n"
echo "About to register ${DBNAME?} resource manager background
echo processes with Sun Cluster 3.x"
echo "\n"
echo "Please hit enter to continue, break to terminate:"
read reponse1

# Register the IBM.cmrmproc Resource Type
# Resource Type
RT="IBM.cmrmproc"
# RTR file path
RTR_FILE="/opt/IBMicm/ha/sc3.x/etc/IBM.cmrmproc"

echo "Adding type $RT as defined in $RTR_FILE"
scrgadm -a -t $RT -f $RTR_FILE > /dev/null 2> /dev/null
rc=$?
if [ ${rc?} -ne 0 ]; then
    # Anything other than 0 is an error
    # However, we will ignore rc=13, since that implies type already exists
    if [ ${rc?} -ne 13 ]; then
        echo "Unable to add resource type $RT to the cluster. Exiting"
        exit 1
    fi
fi

# Create resource group and resource for each of the RM background processes
for proc in RMMigrator RMPurger RMReplicator RMStager
do
        # Create the failover resource group for this RMPROC
        scrgadm -a -g ${DBNAME}_${proc}_rg 2> /dev/null > /dev/null
  echo ${DBNAME}_${proc}_rg
        rc=$?
        if [ $rc -ne 0 ]; then
            # However, we will ignore rc=3, since that implies group already exists
            if [ ${rc?} -ne 3 ]; then
                echo "Unable to add resource group ${DBNAME}_${proc}_rg to the cluster.
Exiting"
                exit 1
            fi
        fi

  # Create the resource for this RMPROC
        scrgadm -a -j ${DBNAME}_${proc}_rs -g ${DBNAME}_${proc}_rg -t $RT -x DBNAME=${DBNAME?}
-x PROCNAME=${proc?} 2> /dev/null > /dev/null
        rc=$?
        if [ $rc -ne 0 ]; then
            if [ ${rc?} -ne 3 ]; then
                echo "Unable to add resource ${DBNAME}_${proc}_rs to the cluster. Exiting"
                exit 1
            fi
        fi
done

echo "Completed.  Use scstat and scrgadm commands to view resultant cluster configuration"
```

11.1.3 /opt/IBMicm/ha/sc3.x/bin/cmrmproc_svc_start.ksh

```
#!/bin/ksh
# Sun Cluster Data Services Builder template version 1.0
#
# Start Method for cmrmproc.
#
# The data service is started under the control of PMF. Prior to starting the
# process(es) for cmrmproc some sanity checks are done. The PMF tries to start the
# service specified number of times and if the number of attempts exceeds this
# value within a specified interval of time the PMF reports a failure to start
# the service. The number of times to retry and the interval in which it is to
# be tried are both properties of the resource set in RTR file.

################################################################################
# Parse program arguments.
#
function parse_args # [args ...]
{
  typeset opt

  while getopts 'R:G:T:' opt
  do
          case "$opt" in

          R)
          # Name of the cmrmproc resource.
          RESOURCE_NAME=$OPTARG
          ;;

          G)
          # Name of the resource group in which the resource is
          # configured.
          RESOURCEGROUP_NAME=$OPTARG
          ;;

          T)
          # Name of the resource type.
          RESOURCETYPE_NAME=$OPTARG
          ;;

          *)
          logger -p ${SYSLOG_FACILITY}.err -t \
          [$RESOURCETYPE_NAME,$RESOURCEGROUP_NAME,$RESOURCE_NAME] \
          "ERROR: Option $OPTARG unknown"
          exit 1
          ;;

          esac
  done

}

#XCB
##########################################################
# error_exit
##########################################################
error_exit()
    {
        typeset exit_code="${rc:-1}"
        logger -p err -t "$SYSLOG_TAG" " $0 exiting with $exit_code"
        exit $exit_code
    }


##########################################################
# read_parameters()
```

```
#
# Read the required properties from the resource
# Always return succesful since we will verify in another
# function
############################################################
read_parameters()
    {
          scha_resource_get -O Extension -R ${RESOURCE_NAME?} -G ${RESOURCEGROUP_NAME?} DBNAME
| tail -1 | read DBNAME
        scha_resource_get -O Extension -R ${RESOURCE_NAME?} -G ${RESOURCEGROUP_NAME?} PROCNAME
| tail -1 | read PROCNAME
        return 0
    }

############################################################
# validate_parameters()
############################################################
validate_parameters()
    {
        rc=0
        if [ -z "$DBNAME" ]; then
            logger -p err -t "$SYSLOG_TAG" "Validation failed. DBNAME is not set"
            rc=1
        fi
        if [ -z "$PROCNAME" ]; then
            logger -p err -t "$SYSLOG_TAG" " Validation failed. PROCNAME is not set"
            rc=1
        fi
        return $rc
    }

#XCE
################################################################################
# MAIN
################################################################################

export PATH=/bin:/usr/bin:/usr/cluster/bin:/usr/sbin:/usr/proc/bin:$PATH

# Obtain the syslog facility to use. This will be used to log the messages.
SYSLOG_FACILITY=`scha_cluster_get -O SYSLOG_FACILITY`

# Parse the arguments that have been passed to this method
parse_args "$@"

PMF_TAG=$RESOURCEGROUP_NAME,$RESOURCE_NAME,0.svc
SYSLOG_TAG=$RESOURCETYPE_NAME,$RESOURCEGROUP_NAME,$RESOURCE_NAME

# We need to know the full path for the gethostnames utility which resides
# in the directory <RT_BASEDIR>. Get this from the RT_BASEDIR property of the
# resource type.
RT_BASEDIR=`scha_resource_get -O RT_BASEDIR -R $RESOURCE_NAME \
  -G $RESOURCEGROUP_NAME`

hostnames=`$RT_BASEDIR/gethostnames -R $RESOURCE_NAME -G $RESOURCEGROUP_NAME \
  -T $RESOURCETYPE_NAME`

#XCB
        read_parameters || error_exit $?
        validate_parameters || error_exit $?

start_cmd_args="/etc/rc.cmrmproc -act start -db $DBNAME -proc $PROCNAME"


#XCE

start_cmd_prog=`echo $start_cmd_args | nawk '{print $1}'`

if [[ ! -f $start_cmd_prog || ! -x $start_cmd_prog ]]; then
```

```
  logger -p ${SYSLOG_FACILITY}.err -t [$SYSLOG_TAG] \
         "${ARGV0} File $start_cmd_prog is missing or not executable"
  exit 1
fi

# Get the value for retry count from the RTR file.
RETRY_CNT=`scha_resource_get -O Retry_Count -R $RESOURCE_NAME \
  -G $RESOURCEGROUP_NAME`

# Get the value for retry interval from the RTR file. The value for the
# RETRY_INTERVAL in the RTR file will be in seconds. Convert this value from
# seconds to minutes for passing on to pmfadm. Note that this is necessarily
# a conversion with round-up, eg. 59 seconds --> 1 minute.
((RETRY_INTRVAL = (`scha_resource_get -O Retry_Interval -R $RESOURCE_NAME -G \
  $RESOURCEGROUP_NAME` + 59) / 60 ))

# User added code -- BEGIN vvvvvvvvvvvvvvv
# User added code -- END   ^^^^^^^^^^^^^^^

# start the daemon under the control of Sun Cluster Process Monitor
# Facility. Let it crash and restart up to $RETRY_COUNT times in a period of
# $RETRY_INTERVAL; if it crashes more often than that, the process monitor
# facility will cease trying to restart it
pmfadm -c $PMF_TAG -n $RETRY_CNT -t $RETRY_INTRVAL $start_cmd_args

# Log a message indicating that cmrmproc has been started.
if [ $? -eq 0 ]; then
  logger -p ${SYSLOG_FACILITY}.info \
         -t [$SYSLOG_TAG] \
         "${ARGV0} cmrmproc successfully started"
fi

# User added code -- BEGIN vvvvvvvvvvvvvvv
# User added code -- END   ^^^^^^^^^^^^^^^

exit 0
```

11.1.4 /opt/IBMicm/ha/sc3.x/bin/cmrmproc_svc_stop.ksh

```
#!/bin/ksh
# Sun Cluster Data Services Builder template version 1.0
#
# Stop method for cmrmproc
#

###############################################################################
# Parse program arguments.
#
function parse_args # [args ...]
{
  typeset opt

  while getopts 'R:G:T:' opt
  do
          case "$opt" in
          R)
          # Name of the cmrmproc resource.
          RESOURCE_NAME=$OPTARG
          ;;

          G)
          # Name of the resource group in which the resource is
          # configured.
          RESOURCEGROUP_NAME=$OPTARG
          ;;

          T)
          # Name of the resource type.
          RESOURCETYPE_NAME=$OPTARG
          ;;

          *)
          logger -p ${SYSLOG_FACILITY}.err \
          -t [$RESOURCETYPE_NAME,$RESOURCEGROUP_NAME,$RESOURCE_NAME] \
          "ERROR: Option $OPTARG unknown"
          exit 1
          ;;

          esac
  done

}

#XCB
#############################################################
# error_exit
#############################################################
error_exit()
    {
        typeset exit_code="${rc:-1}"
        logger -p err -t "$SYSLOG_TAG" " $0 exiting with $exit_code"
        exit $exit_code
    }

#############################################################
# read_parameters()
#
# Read the required properties from the resource
# Always return succesful since we will verify in another
# function
#############################################################
read_parameters()
    {
```

```
        scha_resource_get -O Extension -R ${RESOURCE_NAME?} -G ${RESOURCEGROUP_NAME?} DBNAME |
tail -1 | read DBNAME
        scha_resource_get -O Extension -R ${RESOURCE_NAME?} -G ${RESOURCEGROUP_NAME?} PROCNAME
| tail -1 | read PROCNAME
        return 0
    }

############################################################
# validate_parameters()
############################################################
validate_parameters()
    {
        rc=0
        if [ -z "$DBNAME" ]; then
            logger -p err -t "$SYSLOG_TAG" "Validation failed. DBNAME is not set"
            rc=1
        fi
        if [ -z "$PROCNAME" ]; then
            logger -p err -t "$SYSLOG_TAG" " Validation failed. PROCNAME is not set"
            rc=1
        fi
        return $rc
    }

#XCE



##############################################################################
# MAIN
##############################################################################

export PATH=/bin:/usr/bin:/usr/cluster/bin:/usr/sbin:/usr/proc/bin:$PATH

# Obtain the syslog facility to use. This will be used to log the messages.
SYSLOG_FACILITY=`scha_cluster_get -O SYSLOG_FACILITY`

# Parse the arguments that have been passed to this method.
parse_args "$@"

# get the Timeout value allowed for stop method from the RTR file
STOP_TIMEOUT=`scha_resource_get -O STOP_TIMEOUT -R $RESOURCE_NAME \
  -G `$RESOURCEGROUP_NAME`

# We will try to wait wait for 80% of the stop_method_timeout value when we
# send a SIGTERM through PMF to the Data service. This is to make sure that
# the application stops in a decent manner. If the application does not
# respond favourably to this then we use SIGKILL to stop the data service
# and this will be done for a 15% of the Stop_method_timeout value. However,
# if the data service has not stopped by now, we conclude that there was a
# Failure in the stop method and exit non-zero. The remaining 5% of the
# stop method timeout is for other needs.
((SMOOTH_TIMEOUT=$STOP_TIMEOUT * 80/100))

((HARD_TIMEOUT=$STOP_TIMEOUT * 15/100))

PMF_TAG=$RESOURCEGROUP_NAME,$RESOURCE_NAME,0.svc
SYSLOG_TAG=$RESOURCETYPE_NAME,$RESOURCEGROUP_NAME,$RESOURCE_NAME

# We need to know the full path for the gethostnames utility which resides
# in the directory <RT_BASEDIR>. Get this from the RT_BASEDIR property of the
# resource type.
RT_BASEDIR=`scha_resource_get -O RT_BASEDIR -R $RESOURCE_NAME \
        -G $RESOURCEGROUP_NAME`

hostnames=`$RT_BASEDIR/gethostnames -R $RESOURCE_NAME -G $RESOURCEGROUP_NAME \
        -T $RESOURCETYPE_NAME`
```

```
# XCB
        read_parameters || error_exit $?
        validate_parameters || error_exit $?

stop_cmd_args="/etc/rc.cmrmproc -act stop -db $DBNAME -proc $PROCNAME"
#stop_cmd_args="/etc/rc.cmrmproc -act stop -proc RMMigrator"
# XCE

stop_cmd_prog=`echo $stop_cmd_args | nawk '{print $1}'`

typeset -i SEND_KILL=0

# User added code -- BEGIN vvvvvvvvvvvvvvv
# User added code -- END   ^^^^^^^^^^^^^^^

pmfadm -q $PMF_TAG
if [[ $? == 0 ]]; then
  if [[ -f $stop_cmd_prog && -x $stop_cmd_prog ]]; then
          # User added code -- BEGIN vvvvvvvvvvvvvvv
          # User added code -- END   ^^^^^^^^^^^^^^^

          pmfadm -s $PMF_TAG
          if [[ $? != 0  ]]; then
                  logger -p ${SYSLOG_FACILITY}.info \
                          -t [$SYSLOG_TAG] \
                          "${ARGV0} Failed to take cmrmproc out of PMF \
                          control; trying to send SIGKILL now"
                  SEND_KILL=1
          else
                  # User added code -- BEGIN vvvvvvvvvvvvvvv
                  # User added code -- END   ^^^^^^^^^^^^^^^

                  # execute the user specified stop_cmd using hatimerun
                  hatimerun -k KILL -t $SMOOTH_TIMEOUT $stop_cmd_args
                  if [[ $? != 0 ]]; then
                          logger -p ${SYSLOG_FACILITY}.err \
                                  -t [$SYSLOG_TAG] \
                                  "${ARGV0} Failed to stop cmrmproc using \
                                  the custom stop command; trying \
                                  SIGKILL now."
                  fi

                  # Regardless of whether the command succeeded or not we
                  # send KILL signal to the pmf tag. This will ensure
                  # that the process tree goes away if it still exists.
                  # If it doesn't exist by then, we return NOERR.
                  SEND_KILL=1
          fi
  else
          # User added code -- BEGIN vvvvvvvvvvvvvvv
          # User added code -- END   ^^^^^^^^^^^^^^^

          # Send a SIGTERM signal to the Data service and wait for
          # 80% of the total timeout value.
          pmfadm -s $PMF_TAG -w $SMOOTH_TIMEOUT TERM
          # We compare the exit status of pmfadm to be greater than 2
          # because "exit status = 1" means nametag doesn't exist, which
          # is a OK, for the stop method has to be idempotent.
          if [[ $? -ge 2 ]]; then
                  logger -p ${SYSLOG_FACILITY}.err \
                          -t [$SYSLOG_TAG] \
                          "${ARGV0} Failed to stop cmrmproc with SIGTERM; \
                          retry with SIGKILL"
                  SEND_KILL=1;
          fi
  fi

  pmfadm -q $PMF_TAG
```

```
  if [[ $SEND_KILL == 1 && $? == 0 ]]; then
          # User added code -- BEGIN vvvvvvvvvvvvvvv
          # User added code -- END   ^^^^^^^^^^^^^^^

          # Since the Data service did not stop with a SIGTERM we will
          # use a SIGKILL now and wait for another 15% of total timeout.
          pmfadm -s $PMF_TAG -w $HARD_TIMEOUT KILL
          # We compare the exit status of pmfadm to be greater than 2
          # because "exit status = 1" means nametag doesn't exist, which
          # is a OK, for the stop method has to be idempotent.
          if [[ $? -ge 2 ]]; then
                  logger -p ${SYSLOG_FACILITY}.err -t [$SYSLOG_TAG] \
                  "${ARGV0} Failed to stop cmrmproc; exiting UNSUCCESFUL"
                  exit 1
          fi
  fi
else
  # The Data service is not running as of now. Log a message and
  # exit suceess.
  logger -p ${SYSLOG_FACILITY}.err -t [$SYSLOG_TAG] \
          "cmrmproc not already running"

  # Even if the cmrmproc is not running, we exit success, for idempotency of
  # the STOP method.
  exit 0
fi

# User added code -- BEGIN vvvvvvvvvvvvvvv
# User added code -- END   ^^^^^^^^^^^^^^^

# could successfully stop cmrmproc. Log a message and exit success.
logger -p ${SYSLOG_FACILITY}.info -t [$SYSLOG_TAG] "cmrmproc successfully stopped"
exit 0
```

11.1.5 /opt/IBMicm/ha/sc3.x/bin/cmrmproc_validate.ksh

```
#!/bin/ksh
# Sun Cluster Data Services Builder template version 1.0
#
# Validate method for cmrmproc.
# ex: When the resource is being created command args will be
#
# cmrmproc_validate -c -R <..> -G <...> -T <..> -r <sysdef-prop=value>...
#       -x <extension-prop=value>.... -g <resourcegroup-prop=value>....
#
# when the resource property is being updated
#
# cmrmproc_validate -u -R <..> -G <...> -T <..> -r <sys-prop_being_updated=value>
#   OR
# cmrmproc_validate -u -R <..> -G <...> -T <..> -x <extn-prop_being_updated=value>
#

################################################################################
# Parse program arguments.
#
function parse_args # [args ...]
{
  typeset opt

  while getopts 'cur:x:g:R:T:G:' opt
  do
          case "$opt" in

          R)
          # Name of the cmrmproc resource.
          RESOURCE_NAME=$OPTARG
          ;;

          G)
          # Name of the resource group in which the resource is
          # configured.
          RESOURCEGROUP_NAME=$OPTARG
          ;;

          T)
          # Name of the resource type.
          RESOURCETYPE_NAME=$OPTARG
          ;;

          r)
          # We are not accessing any system defined property.
          # So, this is a no-op
          ;;

          g)
          # This is a no-op as we are not bothered about
          # Resource group properties
          ;;

          c)
          # This is a no-op as this just a flag which indicates
          # that the validate method is being called while
          # creating the resource.
          ;;

          u)
          # This is a flag to indicate the updating of property
          # of an existing resource.
          UPDATE_PROPERTY=1
          ;;
```

```
        x)
        ;;

        *)
        logger -p ${SYSLOG_FACILITY}.err \
        -t [$RESOURCETYPE_NAME,$RESOURCEGROUP_NAME,$RESOURCE_NAME] \
        "ERROR: Option $OPTARG unknown"
        exit 1
        ;;

        esac
 done
}


#XCB
############################################################
# error_exit
############################################################
error_exit()
    {
        typeset exit_code="${rc:-1}"
        logger -p err -t "$SYSLOG_TAG" " $0 exiting with $exit_code"
        exit $exit_code
    }

############################################################
# read_parameters()
#
# Read the required properties from the resource
# Always return succesful since we will verify in another
# function
############################################################
read_parameters()
    {
        scha_resource_get -O Extension -R ${RESOURCE_NAME?} -G ${RESOURCEGROUP_NAME?} DBNAME |
tail -1 | read DBNAME
        scha_resource_get -O Extension -R ${RESOURCE_NAME?} -G ${RESOURCEGROUP_NAME?} PROCNAME
| tail -1 | read PROCNAME
        return 0
    }

############################################################
# validate_parameters()
############################################################
validate_parameters()
    {
        rc=0
        if [ -z "$DBNAME" ]; then
            logger -p err -t "$SYSLOG_TAG" "Validation failed. DBNAME is not set"
            rc=1
        fi
        if [ -z "$PROCNAME" ]; then
            logger -p err -t "$SYSLOG_TAG" " Validation failed. PROCNAME is not set"
            rc=1
        fi
        return $rc
    }

#XCE



################################################################################
# MAIN
################################################################################
```

```
export PATH=/bin:/usr/bin:/usr/cluster/bin:/usr/sbin:/usr/proc/bin:$PATH

# Obtain the syslog facility to use. This will be used to log the messages.
SYSLOG_FACILITY=`scha_cluster_get -O SYSLOG_FACILITY`

UPDATE_PROPERTY=0

# Parse the arguments that have been passed to this method.
parse_args "$@"

SYSLOG_TAG=$RESOURCETYPE_NAME,$RESOURCEGROUP_NAME,$RESOURCE_NAME

if [[ -f /opt/IBMcmrmproc/bin/hasp_check && \
  -x /opt/IBMcmrmproc/bin/hasp_check ]]; then
  /opt/IBMcmrmproc/bin/hasp_check "$@"
  hasp_status=$?
else
  # the binary doesn't exist so we cannot call it
  hasp_status=2
fi

case "$hasp_status" in

  1)
  logger -p ${SYSLOG_FACILITY}.err -t [$SYSLOG_TAG] \
          "${ARGV0} Internal Error. Failed to check status of \
IBM.HAStoragePlus resource."
  exit 1
  ;;

  2)
  logger -p ${SYSLOG_FACILITY}.info -t [$SYSLOG_TAG] \
          "${ARGV0} This resource doesn't depend on any \
IBM.HAStoragePlus resources. Proceeding with the normal checks."

  ;;

  3)
  logger -p ${SYSLOG_FACILITY}.err -t [$SYSLOG_TAG] \
          "${ARGV0} One or more of the IBM.HAStoragePlus resources \
that this resource depends on is in a different RG. \
Failing validate method configuration checks."
  exit 1
  ;;

  4)
  logger -p ${SYSLOG_FACILITY}.err -t [$SYSLOG_TAG] \
          "${ARGV0} One or more of the IBM.HAStoragePlus resources \
that this resource depends on is not online anywhere. \
Failing validate method."
  exit 1
  ;;

  5)
  logger -p ${SYSLOG_FACILITY}.info -t [$SYSLOG_TAG] \
          "${ARGV0} All the IBM.HAStoragePlus resources that this \
resource depends on are not online on the local node. \
Skipping the checks for the existence and permissions of the start/stop/probe \
commands."
  exit 0
  ;;

  6)
  logger -p ${SYSLOG_FACILITY}.info -t [$SYSLOG_TAG] \
          "${ARGV0} All the IBM.HAStoragePlus resources that this \
resource depends on are online on the local node. \
Proceeding with the checks for the existence and permissions of the \
start/stop/probe commands."
```

```
  ;;

  *)
  logger -p ${SYSLOG_FACILITY}.err -t [$SYSLOG_TAG] \
         "${ARGV0} Unknown status code $hasp_status"
  ;;

esac


#XCB
        #read_parameters || error_exit $?
        #validate_parameters || error_exit $?
#XCE

#start_cmd_args="/etc/rc.cmrmproc -act start -proc RMMigrator"
start_cmd_args="/etc/rc.cmrmproc -act start -db $DBNAME -proc $PROCNAME"

start_cmd_prog=`echo $start_cmd_args | nawk '{print $1}'`
if [[ ! -f $start_cmd_prog || ! -x $start_cmd_prog ]]; then
  logger -p ${SYSLOG_FACILITY}.err -t [$SYSLOG_TAG] \
         "${ARGV0} File $start_cmd_prog is missing or not executable"
  exit 1
fi

#stop_cmd_args="/etc/rc.cmrmproc -act stop -proc RMMigrator"
stop_cmd_args="/etc/rc.cmrmproc -act stop -db $DBNAME -proc $PROCNAME"


stop_cmd_prog=`echo $stop_cmd_args | nawk '{print $1}'`
if [[ ! -z $stop_cmd_prog ]]; then
  if [[ ! -f $stop_cmd_prog || ! -x $stop_cmd_prog ]]; then
          logger -p ${SYSLOG_FACILITY}.err -t [$SYSLOG_TAG] \
                 "${ARGV0} File $stop_cmd_prog is missing or "
                 "not executable"
          exit 1
  fi
fi

probe_cmd_args=""
probe_cmd_prog=`echo $probe_cmd_args | nawk '{print $1}'`
if [[ ! -z $probe_cmd_prog ]]; then
  if [[ ! -f $probe_cmd_prog || ! -x $probe_cmd_prog ]]; then
          logger -p ${SYSLOG_FACILITY}.err -t [$SYSLOG_TAG] \
                 "${ARGV0} File $probe_cmd_prog is missing or "
                 "not executable"
          exit 1
  fi
fi

# User added code -- BEGIN vvvvvvvvvvvvvvv
# User added code -- END   ^^^^^^^^^^^^^^^

# Log a message indicating that validate method was successful.
logger -p ${SYSLOG_FACILITY}.info \
  -t [$SYSLOG_TAG] \
  "${ARGV0} Validate method for resource "$RESOURCE_NAME \
  " completed successfully"

exit 0
```

Figures from section 5.6

11.2 Figure 07: luxadm –e port command and output

```
# luxadm -e port

Found path to 2 HBA ports

/devices/pci@11c,600000/SUNW,qlc@1/fp@0,0:devctl                  CONNECTED
/devices/pci@11c,600000/SUNW,qlc@1,1/fp@0,0:devctl                CONNECTED
```

11.3 Figure 08: luxadm –e dump_map command and output

```
# luxadm -e dump_map /devices/pci@11c,600000/SUNW,qlc@1/fp@0,0:devctl
Pos  Port_ID Hard_Addr Port WWN         Node WWN         Type
0    610613  0         5005076300cb8756 5005076300c08756 0x0  (Disk device)
1    610713  0         5005076300cf8756 5005076300c08756 0x0  (Disk device)
2    610813  0         210000e08b0873ed 200000e08b0873ed 0x1f (Unknown Type)
3    610913  0         210100e08b2873ed 200100e08b2873ed 0x1f (Unknown Type)
4    610a13  0         210000e08b083ded 200000e08b083ded 0x1f (Unknown Type)
5    610b13  0         210100e08b283ded 200100e08b283ded 0x1f (Unknown Type)
6    610c13  0         210000e08b093836 200000e08b093836 0x1f (Unknown Type)
7    610d13  0         210100e08b293836 200100e08b293836 0x1f (Unknown Type)
8    610e13  0         210100e08b2838ed 200100e08b2838ed 0x1f (Unknown Type)
9    611113  0         10000000c92ab429 20000000c92ab429 0x1f (Unknown Type)
10   610f13  0         210000e08b0838ed 200000e08b0838ed 0x1f (Unknown Type,Host Bus Adapter)

# luxadm -e dump_map /devices/pci@11c,600000/SUNW,qlc@1,1/fp@0,0:devctl
Pos  Port_ID Hard_Addr Port WWN         Node WWN         Type
0    610613  0         5005076300cb8756 5005076300c08756 0x0  (Disk device)
1    610713  0         5005076300cf8756 5005076300c08756 0x0  (Disk device)
2    610813  0         210000e08b0873ed 200000e08b0873ed 0x1f (Unknown Type)
3    610913  0         210100e08b2873ed 200100e08b2873ed 0x1f (Unknown Type)
4    610a13  0         210000e08b083ded 200000e08b083ded 0x1f (Unknown Type)
5    610b13  0         210100e08b283ded 200100e08b283ded 0x1f (Unknown Type)
6    610c13  0         210000e08b093836 200000e08b093836 0x1f (Unknown Type)
7    610d13  0         210100e08b293836 200100e08b293836 0x1f (Unknown Type)
8    610f13  0         210000e08b0838ed 200000e08b0838ed 0x1f (Unknown Type)
9    611113  0         10000000c92ab429 20000000c92ab429 0x1f (Unknown Type)
10   610e13  0         210100e08b2838ed 200100e08b2838ed 0x1f (Unknown Type,Host Bus Adapter)
```

11.4 Figure 31: Modified /kernel/drv/sd.conf

```
# Copyright (c) 1992, by Sun Microsystems, Inc.
#ident  "@(#)sd.conf   1.9     98/01/11 SMI"
# Start of lines added by SUNWscr
sd_retry_on_reservation_conflict=0;
# End   of lines added by SUNWscr
name="sd" class="scsi"        class_prop="atapi"    target=0 lun=0;
name="sd" class="scsi"        target=0 lun=1;
name="sd" class="scsi"        target=0 lun=2;
name="sd" class="scsi"        target=0 lun=3;
name="sd" class="scsi"        class_prop="atapi"    target=1 lun=0;
name="sd" class="scsi"        target=1 lun=1;
name="sd" class="scsi"        target=1 lun=2;
name="sd" class="scsi"        target=1 lun=3;
name="sd" class="scsi"        class_prop="atapi"    target=2 lun=0;
name="sd" class="scsi"        target=2 lun=1;
name="sd" class="scsi"        target=2 lun=2;
name="sd" class="scsi"        target=2 lun=3;
name="sd" class="scsi"        class_prop="atapi"    target=3 lun=0;
name="sd" class="scsi"        target=3 lun=1;
name="sd" class="scsi"        target=3 lun=2;
name="sd" class="scsi"        target=3 lun=3;
name="sd" class="scsi"        target=4 lun=0;
name="sd" class="scsi"        target=4 lun=1;
name="sd" class="scsi"        target=4 lun=2;
```

```
name="sd" class="scsi"        target=4 lun=3;
name="sd" class="scsi"        target=5 lun=0;
name="sd" class="scsi"        target=5 lun=1;
name="sd" class="scsi"        target=5 lun=2;
name="sd" class="scsi"        target=5 lun=3;
name="sd" class="scsi"        target=6 lun=0;
name="sd" class="scsi"        target=6 lun=1;
name="sd" class="scsi"        target=6 lun=2;
name="sd" class="scsi"        target=6 lun=3;
name="sd" class="scsi"        target=8 lun=0;
name="sd" class="scsi"        target=8 lun=1;
name="sd" class="scsi"        target=8 lun=2;
name="sd" class="scsi"        target=8 lun=3;
name="sd" class="scsi"        target=9 lun=0;
name="sd" class="scsi"        target=9 lun=1;
name="sd" class="scsi"        target=9 lun=2;
name="sd" class="scsi"        target=9 lun=3;
name="sd" class="scsi"        target=10 lun=0;
name="sd" class="scsi"        target=10 lun=1;
name="sd" class="scsi"        target=10 lun=2;
name="sd" class="scsi"        target=10 lun=3;
name="sd" class="scsi"        target=11 lun=0;
name="sd" class="scsi"        target=11 lun=1;
name="sd" class="scsi"        target=11 lun=2;
name="sd" class="scsi"        target=11 lun=3;
name="sd" class="scsi"        target=12 lun=0;
name="sd" class="scsi"        target=12 lun=1;
name="sd" class="scsi"        target=12 lun=2;
name="sd" class="scsi"        target=12 lun=3;
name="sd" class="scsi"        target=13 lun=0;
name="sd" class="scsi"        target=13 lun=1;
name="sd" class="scsi"        target=13 lun=2;
name="sd" class="scsi"        target=13 lun=3;
name="sd" class="scsi"        target=14 lun=0;
name="sd" class="scsi"        target=14 lun=1;
name="sd" class="scsi"        target=14 lun=2;
name="sd" class="scsi"        target=14 lun=3;
name="sd" class="scsi"        target=15 lun=0;
name="sd" class="scsi"        target=15 lun=1;
name="sd" class="scsi"        target=15 lun=2;
name="sd" class="scsi"        target=15 lun=3;
```

11.4.1 Output from IBM ESS utility command: ls2015

```
Hostname   SD#   hdisk     Location                                  LUN SN    S  Connection
Size LSS  Vol  Rank
--------   ---   -----     --------                                  ------    -  ----------
---- ---  ---  ----
submariner sd71  c4t6005076300C0875600000000000001005d0
/scsi_vhci/ssd@g6005076300c08756000000000001005 00518878   Y  R1-B2-H2-ZA  100.0 16   005
1001
submariner sd70  c4t6005076300C0875600000000000001006d0
/scsi_vhci/ssd@g6005076300c08756000000000001006 00618878   Y  R1-B4-H2-ZA  100.0 16   006
1001
submariner sd73  c4t6005076300C0875600000000000001007d0
/scsi_vhci/ssd@g6005076300c08756000000000001007 00718878   Y  R1-B2-H2-ZA   10.2 16   007
1001
submariner sd69  c4t6005076300C0875600000000000001103d0
/scsi_vhci/ssd@g6005076300c08756000000000001103 10318878   Y  R1-B4-H2-ZA   70.0 17   003
1101
submariner sd68  c4t6005076300C0875600000000000001104d0
/scsi_vhci/ssd@g6005076300c08756000000000001104 10418878   Y  R1-B4-H2-ZA   70.0 17   004
1101
submariner sd72  c4t6005076300C0875600000000000001105d0
/scsi_vhci/ssd@g6005076300c08756000000000001105 10518878   Y  R1-B4-H2-ZA   70.0 17   005
1101
```

11.4.2 Output from IBM ESS utility command: lsess

```
Disk        Location                                  LUN SN   Type   Size  LSS Vol Rank C/A S
Connection
------     ------------                               -------- ------- ----- --- --- ---- --- - --
---------
c4t6005076300C0875600000000000001005d0 /scsi_vhci/ssd@g6005076300c08756000000000001005 [FC]
00518878 2105F20 100.0  16 005 1001 1/1 Y R1-B2-H2-ZA
c4t6005076300C0875600000000000001006d0 /scsi_vhci/ssd@g6005076300c08756000000000001006 [FC]
00618878 2105F20 100.0  16 006 1001 1/1 Y R1-B4-H2-ZA
c4t6005076300C0875600000000000001007d0 /scsi_vhci/ssd@g6005076300c08756000000000001007 [FC]
00718878 2105F20  10.2  16 007 1001 1/1 Y R1-B2-H2-ZA
c4t6005076300C0875600000000000001103d0 /scsi_vhci/ssd@g6005076300c08756000000000001103 [FC]
10318878 2105F20  70.0  17 003 1101 2/1 Y R1-B4-H2-ZA
c4t6005076300C0875600000000000001104d0 /scsi_vhci/ssd@g6005076300c08756000000000001104 [FC]
10418878 2105F20  70.0  17 004 1101 2/1 Y R1-B4-H2-ZA
c4t6005076300C0875600000000000001105d0 /scsi_vhci/ssd@g6005076300c08756000000000001105 [FC]
10518878 2105F20  70.0  17 005 1101 2/1 Y R1-B4-H2-ZA
```

11.4.3 Output from Solaris luxadm probe command

```
No Network Array enclosures found in /dev/es

Found Fibre Channel device(s):
  Node WWN:5005076300c08756  Device Type:Disk device
    Logical Path:/dev/rdsk/c4t6005076300C0875600000000000001005d0s2
  Node WWN:5005076300c08756  Device Type:Disk device
    Logical Path:/dev/rdsk/c4t6005076300C0875600000000000001006d0s2
  Node WWN:5005076300c08756  Device Type:Disk device
    Logical Path:/dev/rdsk/c4t6005076300C0875600000000000001103d0s2
  Node WWN:5005076300c08756  Device Type:Disk device
    Logical Path:/dev/rdsk/c4t6005076300C0875600000000000001104d0s2
  Node WWN:5005076300c08756  Device Type:Disk device
    Logical Path:/dev/rdsk/c4t6005076300C0875600000000000001105d0s2
  Node WWN:5005076300c08756  Device Type:Disk device
    Logical Path:/dev/rdsk/c4t6005076300C0875600000000000001007d0s2
```

12. Terms and Acronyms

Term	Description
CM, ICM	IBM Content Manager
CM eClient	This is the a Middle-tier Web Application included with the IBM Content Manager product
CM Windows Client	A-Windows based document management client for Content Manager
HA	High Availability
LB	Load Balancing
DS	Disaster Recovery
MSCS	Microsoft Windows Cluster Services
Node	A workstation which is part of a clustered environment
HIS	IBM HTTP Server
WES	WebSphere Edge Services
WAS	IBM WebSphere Application Server
WAS ND	IBM WebSphere Application Server Network Deployment
WAS Edge	IBM WebSphere Edge Services Network Dispatcher

13. Online references

The following links provide additional information about setting up Content Manager high availability. You can download the Acrobat Reader at no charge from www.adobe.com.

IBM

- Content Manager product page
- Enterprise Storage Server (2105) Technical Support
- Tivoli Storage Manager product page
- DB2 Universal Database and High Availability on Sun Cluster 3.X
- Content Manager Backup/Recovery and High Availability: Strategies, Options, and Procedure
- IBM Content Manager V8.2 Performance Tuning Guide
- IBM WebShere Portal for Multiplatforms V5 Handbook
- Configuring Highly Available p690 Clusters Using HACMP 4.5
- Content Manager Backup/Recovery and High Availability: Strategies, Options, and Procedures
- IBM WebSphere V5.0 Performance, Scalability, and High Availability: WebSphere Handbook Series
- IBM WebSphere V5.1 Performance, Scalability, and High Availability WebSphere Handbook Series
- ESSutl for Solaris
- ESS CLI on Solaris
- IBM TotalStorage Enterprise Storage Server Command-Line Interface User's Guide
- IBM TotalStorage Enterprise Storage Server Host Systems Attachment Guide
- IBM TotalStorage Enterprise Storage Server Introduction and Planning Guide
- IBM TotalStorage Enterprise Storage Server Implementing the ESS in Your Environment - SG24-5420-01
- IBM TotalStorage Enterprise Storage Server Model 800 Performance Monitoring and Tuning Guide
- Fault Tolerant Storage Multipathing and Clustering Solutions for Open Systems for the IBM ESS - SG24-6295-00
- DB2 UDB Backup and Recovery with ESS Copy Services
- IBM TotalStorage Enterprise Storage Server Implementing ESS Copy Services in Open Environments
- Storage Management for IBM DB2 UDB: Snapshot Backup and Recovery With the IBM TotalStorage Enterprise Storage Server
- IBM TotalStorage Solutions for Disaster Recovery
- HA Considerations in a WAS V5.0 Deployment, by Tom Alcott, alcott@us.ibm.com

Sun Microsystems

- Sun product documentation
- Sun Cluster 3.1 Software Installation Guide
- SAN Foundation Software (SFS), SAN V4.4 software and associated patches
  Additional information about SAN Solutions V4.4:
  http://www.sun.com/products-n-solutions/hardware/docs/Network_Storage_Solutions/SAN
  http://www.sun.com/storage/san/open_san/
  http://www.sun.com/storage/software/storage_mgmt/traffic_manager/index.xml
  http://www.sun.com/products-n-solutions/hardware/docs/pdf/817-3674-10.pdf
  http://www.sun.com/products-n-solutions/hardware/docs/pdf/816-1420-11.pdf
  http://www.sun.com/products-n-solutions/hardware/docs/pdf/817-0385-11.pdf
- **Sun StorEdge Traffic Manager Software (STMS)**