

# System Deployment Guide

*Sun<sup>TM</sup> ONE Application Server*

**Version 7, Enterprise Edition**

817-2163-10  
September 2003

Sun Microsystems, Inc.  
4150 Network Circle  
Santa Clara, CA 95054 U.S.A.

Copyright © 2003 Sun Microsystems, Inc. All rights reserved.

THIS SOFTWARE CONTAINS CONFIDENTIAL INFORMATION AND TRADE SECRETS OF SUN MICROSYSTEMS, INC. USE, DISCLOSURE OR REPRODUCTION IS PROHIBITED WITHOUT THE PRIOR EXPRESS WRITTEN PERMISSION OF SUN MICROSYSTEMS, INC. U.S. Government Rights - Commercial software. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements. Use is subject to license terms.

This distribution may include materials developed by third parties.

Sun, Sun Microsystems, the Sun logo, Java, Sun™ ONE, the Java Coffee Cup logo and the Sun™ ONE logo are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

UNIX is a registered trademark in the U.S. and other countries, exclusively licensed through X/Open Company, Ltd.

This product is covered and controlled by U.S. Export Control laws and may be subject to the export or import laws in other countries. Nuclear, missile, chemical biological weapons or nuclear maritime end uses or end users, whether direct or indirect, are strictly prohibited. Export or reexport to countries subject to U.S. embargo or to entities identified on U.S. export exclusion lists, including, but not limited to, the denied persons and specially designated nationals lists is strictly prohibited.

---

Copyright © 2003 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, Etats-Unis. Tous droits réservés.

CE LOGICIEL CONTIENT DES INFORMATIONS CONFIDENTIELLES ET DES SECRETS COMMERCIAUX DE SUN MICROSYSTEMS, INC. SON UTILISATION, SA DIVULGATION ET SA REPRODUCTION SONT INTERDITES SANS L'AUTORISATION EXPRESSE, ÉCRITE ET PRÉALABLE DE SUN MICROSYSTEMS, INC. Droits du gouvernement américain, utilisateurs gouvernementaux - logiciel commercial. Les utilisateurs gouvernementaux sont soumis au contrat de licence standard de Sun Microsystems, Inc., ainsi qu'aux dispositions en vigueur de la FAR (Federal Acquisition Regulations) et des suppléments à celles-ci. L'utilisation est soumise aux termes de la Licence.

Cette distribution peut comprendre des composants développés par des tierces parties.

Sun, Sun Microsystems, le logo Sun, Java, Sun™ ONE, le logo Java Coffee Cup et le logo Sun™ ONE sont des marques de fabrique ou des marques déposées de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays.

UNIX est une marque déposée aux Etats-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company, Ltd.

Ce produit est soumis à la législation américaine en matière de contrôle des exportations et peut être soumis à la réglementation en vigueur dans d'autres pays dans le domaine des exportations et importations. Les utilisations, ou utilisateurs finaux, pour des armes nucléaires, des missiles, des armes biologiques et chimiques ou du nucléaire maritime, directement ou indirectement, sont strictement interdites. Les exportations ou réexportations vers les pays sous embargo américain, ou vers des entités figurant sur les listes d'exclusion d'exportation américaines, y compris, mais de manière non exhaustive, la liste de personnes qui font objet d'un ordre de ne pas participer, d'une façon directe ou indirecte, aux exportations des produits ou des services qui sont régis par la législation américaine en matière de contrôle des exportations et la liste de ressortissants spécifiquement désignés, sont rigoureusement interdites.

# Contents

<b>About This Guide</b>	<b>5</b>
Who Should Use This Guide	5
How This Guide is Organized	6
Using the Documentation	6
Documentation Conventions	8
General Conventions	9
Conventions Referring to Directories	10
Product Support	10
 <b>Chapter 1 Overview of Deployment</b>	 <b>11</b>
About Deployment	11
Throughput	12
Response Time	12
Availability	12
Phases of the Deployment Process	12
Planning Your Environment	13
Selecting a Topology	13
Running Tests	14
 <b>Chapter 2 Planning your Environment</b>	 <b>15</b>
Introducing HADB	15
Data Redundancy Units	16
Spare Nodes	16
Sample HADB Architecture	17
Establishing Performance Goals	18
Estimating Throughput	19
Estimating Load on Application Server Instances	19

Maximum Number of Concurrent Users .....	19
Think Time .....	20
Average Response Time .....	21
Requests Per Minute .....	22
Estimating Load on HADB .....	23
Understanding Session Persistence .....	23
Number of Requests per Minute Received by the HADB .....	24
Session Size Per Request .....	25
Comparison of Persistence Scope Options .....	26
Designing for Peak Load or Steady State Load .....	27
Frequency and Duration of Peak Load .....	28
Design Decisions to Make .....	28
Number of Applications Server Instances Needed .....	28
Number of HADB Nodes Required .....	29
HADB Storage Capacity Required .....	29
Whether You Want to Design for Peak Load or for Steady State Load .....	30
Planning Network Configuration to Meet Your Performance Goals .....	31
Estimating Bandwidth Requirements .....	31
Peak Load Times .....	31
Calculating Bandwidth Required .....	31
Peak Load .....	32
Subnets .....	33
Network Cards .....	33
Network Settings for HADB .....	33
Planning Availability .....	34
Adding Redundancy to the System .....	34
Failure Classes .....	34
Using Redundancy Units to Improve Availability .....	34
Using Spare Nodes to Improve Fault Tolerance .....	35
Planning Failover Capacity .....	35
Using Multiple Clusters to Improve Availability .....	35
<b>Chapter 3 Selecting a Topology .....</b>	<b>37</b>
Common Requirements .....	38
General Requirements .....	38
HADB Nodes and Machines .....	39
Load Balancer Configuration .....	39
Co-located Topology .....	40
Reference Co-located Topology .....	40
Configuration Settings for Reference Co-located Topology .....	42
Variation to Reference Co-located Topology .....	43
Configuration Settings for Variation to the Reference Co-located Topology .....	45
Separate Tier Topology .....	46

Reference Configuration . . . . .	46
Configuration Settings for Reference Separate Tier Topology . . . . .	48
Variation to Reference Separate Tier Topology . . . . .	49
Configuration Settings for Variation to Reference Separate Tier Topology . . . . .	51
Comparison of the Topologies . . . . .	52
Determining Which Topology to Use . . . . .	53
<b>Index . . . . .</b>	<b>55</b>



# About This Guide

This guide provides a foundation for evaluating your system needs to ensure that you deploy the Sun™ Open Network Environment (Sun ONE) Application Server 7, Enterprise Edition in a manner that best suits your business needs. General issues and concerns that you must be aware of when deploying an application server are also discussed.

This preface includes the following sections:

- [Who Should Use This Guide](#)
- [How This Guide is Organized](#)
- [Using the Documentation](#)
- [Product Support](#)

## Who Should Use This Guide

The information in this guide is intended for system administrators who want to deploy and support a large system that has high availability requirements.

This guide assumes you are familiar with the following:

- Installation of enterprise-level software products
- UNIX® operating system
- Client/server programming model
- Internet and World Wide Web
- Availability and clustering concepts

# How This Guide is Organized

This guide is divided into three chapters. Read the chapters in the order they are presented as each chapter builds on the previous one.

- [Chapter 1, “Overview of Deployment”](#) provides an introduction to deployment and discusses phases of deployment.
- [Chapter 2, “Planning your Environment”](#) discusses the steps you need to determine the environment that best suits your business needs.
- [Chapter 3, “Selecting a Topology”](#) contains examples of application server topologies, and helps you determine the topology that best suits your business needs.

# Using the Documentation

The Sun ONE Application Server 7, Enterprise Edition manuals are available as online files in Portable Document Format (PDF) and Hypertext Markup Language (HTML).

The following table lists tasks and concepts described in the Sun ONE Application Server manuals.

**Table 1** Sun ONE Application Server Documentation Roadmap

For information about	See the following
Late-breaking information about the software and the documentation.	<i>Release Notes</i>
Comprehensive, table-based summary of supported hardware, operating system, JDK, and JDBC/RDBMS.	<i>Platform Summary</i>
Sun ONE Application Server 7 overview, including the features available with each product edition.	<i>Product Overview</i>
Diagrams and descriptions of server architecture and the benefits of the Sun ONE Application Server architectural approach.	<i>Server Architecture</i>
New enterprise, developer, and operational features of Sun ONE Application Server 7.	<i>What's New</i>
How to get started with the Sun ONE Application Server 7 product. Includes a sample application tutorial.	<i>Getting Started Guide</i>



**Table 1** Sun ONE Application Server Documentation Roadmap (*Continued*)

For information about	See the following
Installing the Sun ONE Application Server software and its components, such as sample applications, the Administration interface, and the high-availability components. Instructions for implementing a basic high-availability configuration are included.	<i>Installation Guide</i>
Evaluating your system needs and enterprise to ensure that you deploy Sun ONE Application Server in a manner that best suits your site. General issues and concerns that you must be aware of when deploying an application server are also discussed.	<i>System Deployment Guide</i>
Best practices for HTTP session availability that application architects and developers can use.	<i>Application Design Guidelines for Storing Session State</i>
Creating and implementing Java™ 2 Platform, Enterprise Edition (J2EE™ platform) applications intended to run on the Sun ONE Application Server 7 that follow the open Java standards model for J2EE components such as servlets, Enterprise JavaBeans™ (EJBs™), and JavaServer Pages™ (JSPs™). Includes general information about application design, developer tools, security, assembly, deployment, debugging, and creating lifecycle modules. A comprehensive Sun ONE Application Server glossary is included.	<i>Developer's Guide</i>
Creating and implementing J2EE web applications that follow the Java™ Servlet and JavaServer Pages (JSP) specifications on the Sun ONE Application Server 7. Discusses web application programming concepts and tasks, and provides sample code, implementation tips, and reference material. Topics include results caching, JSP precompilation, session management, security, deployment, SHTML, and CGI.	<i>Developer's Guide to Web Applications</i>
Creating and implementing J2EE applications that follow the open Java standards model for enterprise beans on the Sun ONE Application Server 7. Discusses Enterprise JavaBeans (EJB) programming concepts and tasks, and provides sample code, implementation tips, and reference material. Topics include container-managed persistence, read-only beans, and the XML and DTD files associated with enterprise beans.	<i>Developer's Guide to Enterprise JavaBeans Technology</i>
Creating Application Client Container (ACC) clients that access J2EE applications on the Sun ONE Application Server 7.	<i>Developer's Guide to Clients</i>
Creating web services in the Sun ONE Application Server environment.	<i>Developer's Guide to Web Services</i>
Java™ Database Connectivity (JDBC™), transaction, Java Naming and Directory Interface™ (JNDI), Java™ Message Service (JMS), and JavaMail™ APIs.	<i>Developer's Guide to J2EE Services and APIs</i>
Creating custom NSAPI plug-ins.	<i>Developer's Guide to NSAPI</i>

**Table 1** Sun ONE Application Server Documentation Roadmap (*Continued*)

For information about	See the following
Information and instructions on the configuration, management, and deployment of the Sun ONE Application Server subsystems and components, from both the Administration interface and the command-line interface. Topics include cluster management, the high-availability database, load balancing, and session persistence. A comprehensive Sun ONE Application Server glossary is included.	<i>Administrator's Guide</i>
Editing Sun ONE Application Server configuration files, such as the <code>server.xml</code> file.	<i>Administrator's Configuration File Reference</i>
Configuring and administering security for the Sun ONE Application Server operational environment. Includes information on general security, certificates, and SSL/TLS encryption. HTTP server-based security is also addressed.	<i>Administrator's Guide to Security</i>
Configuring and administering service provider implementation for J2EE™ Connector Architecture (CA) connectors for the Sun ONE Application Server 7. Topics include the Administration Tool, Pooling Monitor, deploying a JCA connector, and sample connectors and sample applications.	<i>J2EE CA Service Provider Implementation Administrator's Guide</i>
Migrating your applications to the new Sun ONE Application Server 7 programming model, specifically from iPlanet Application Server 6.x and from Netscape Application Server 4.0. Includes a sample migration.	<i>Migrating and Redeploying Server Applications Guide</i>
How and why to tune your Sun ONE Application Server to improve performance.	<i>Performance Tuning Guide</i>
Information on solving Sun ONE Application Server problems.	<i>Troubleshooting Guide</i>
Messages that you may encounter while running Sun ONE Application Server 7. Includes a description of the likely cause and guidelines on how to address the condition that caused the message to be generated.	<i>Error Message Reference</i>
Utility commands available with the Sun ONE Application Server; written in manpage style.	<i>Utility Reference Manual</i>
Using the Sun™ Open Net Environment (Sun ONE) Message Queue software.	The Sun ONE Message Queue documentation at:  <a href="http://docs.sun.com/db?p=prod/sl.slmsgqu">http://docs.sun.com/db?p=prod/sl.slmsgqu</a>

## Documentation Conventions

This section describes the types of conventions used throughout this guide:

- [General Conventions](#)

- [Conventions Referring to Directories](#)

## General Conventions

The following general conventions are used in this guide:

- **File and directory paths** are given in UNIX<sup>®</sup> format (with forward slashes separating directory names).

- **URLs** are given in the format:

`http://server.domain/path/file.html`

In these URLs, *server* is the server name where applications are run; *domain* is your Internet domain name; *path* is the server's directory structure; and *file* is an individual filename. Italic items in URLs are placeholders.

- **Font conventions** include:

- The `monospace` font is used for sample code and code listings, API and language elements (such as function names and class names), file names, pathnames, directory names, and HTML tags.
- *Italic* type is used for code variables.
- *Italic* type is also used for book titles, emphasis, variables and placeholders, and words used in the literal sense.
- **Bold** type is used as either a paragraph lead-in or to indicate words used in the literal sense.

- **Installation root directories** for most platforms are indicated by *install\_dir* in this document. Exceptions are noted in [“Conventions Referring to Directories” on page 10](#).

By default, the location of *install\_dir* on **most** platforms is:

`/opt/SUNWappserver7`

For the platforms listed above, *default\_config\_dir* and *install\_config\_dir* are identical to *install\_dir*. See [“Conventions Referring to Directories” on page 10](#) for exceptions and additional information.

- **Instance root directories** are indicated by *instance\_dir* in this document, which is an abbreviation for the following:

`default_config_dir/domains/domain/instance`

## Conventions Referring to Directories

By default, when using the Solaris™ 8 and 9 installation, the application server files are spread across several root directories. These directories are described in this section.

- **For Solaris 8 and 9 installations**, this guide uses the following document conventions to correspond to the various default installation directories provided:
  - *install\_dir* refers to `/opt/SUNWappserver7`, which contains the static portion of the installation image. All utilities, executables, and libraries that make up the application server reside in this location.
  - *default\_config\_dir* refers to `/var/opt/SUNWappserver7/domains` which is the default location for any domains that are created.
  - *install\_config\_dir* refers to `/etc/opt/SUNWappserver7/config`, which contains installation-wide configuration information such as licenses and the master list of administrative domains configured for this installation.

## Product Support

If you have general feedback on the product or documentation, please send this to [appserver-feedback@sun.com](mailto:appserver-feedback@sun.com).

If you have problems with your system, contact customer support using one of the following mechanisms:

- The online support web site at:  
<http://www.sun.com/supporttraining/>
- The telephone dispatch number associated with your maintenance contract

Please have the following information available prior to contacting support. This helps to ensure that our support staff can best assist you in resolving problems:

- Description of the problem, including the situation where the problem occurs and its impact on your operation
- Machine type, operating system version, and product version, including any patches and other software that might be affecting the problem
- Detailed steps on the methods you have used to reproduce the problem
- Any error logs or core dumps

# Overview of Deployment

This chapter describes what you need to know to set up your Sun™ Open Net Environment (Sun ONE) Application Server 7, Enterprise Edition the way that best meets your requirements.

This chapter contains the following sections:

- [About Deployment](#)
- [Phases of the Deployment Process](#)

## About Deployment

Successful deployment of complex applications on the Sun ONE Application Server 7, Enterprise Edition requires that you consider practical aspects of the environment. In general, you should begin by assessing your goals for performance and availability. You should then plan the hardware, network, and resource configuration accordingly.

Here are the important goals that you should consider while planning the deployment:

- Throughput
- Response time
- Availability

You gather information related to these goals and then analyze it to establish a processing threshold for your site.

In considering performance, consider both—application server instances and the Sun ONE High Availability Database (HADB).

The HADB uses the patented Always-On technology and works as the persistence store to provide high availability for web applications. It offers an ideal platform for delivering all types of session state persistence within an enterprise application server environment. For more information on configuring HADB, see the *Sun One Application Server Administrator's Guide*.

## Throughput

Throughput is the number of requests that Sun ONE Application Server can service in a given time period, for example per minute. You should estimate the maximum number of operations and transactions that the system needs to perform under peak load conditions. It is also useful to determine the operations and transactions per minute under steady state (typical) load conditions. This will help you to determine the network bandwidth needed, the number of application server instances required, and the number of HADB nodes required.

You should also consider plans to increase capacity in future.

## Response Time

You should determine the acceptable response time from the system under heavy load. This has a direct bearing on hardware capacity planning.

## Availability

Will your system be running 24 x 7? If there is a failure in the system, will your users notice it? Do you have a subset of applications that need to be available all the time whereas other applications will run only periodically? The answers to these questions determine your availability needs. You will have to build redundancy into the system to meet availability needs and to avoid single points of failure.

# Phases of the Deployment Process

The deployment process primarily comprises the following three phases, each one building on the previous one:

- [Planning Your Environment](#)

- [Selecting a Topology](#)
- [Running Tests](#)

## Planning Your Environment

Planning your environment is the first phase of deployment and consists of determining how the Sun ONE Application Server 7, Enterprise Edition fits into your overall enterprise. Central to planning your environment is the assessment of the goals discussed in [“About Deployment” on page 11](#). You establish performance goals related to throughput and response time. You also determine your availability goals.

Based on the performance and availability goals, you consider the network requirements and the infrastructure requirements including hardware, storage, and network requirements.

You may realize during this process that you should change the structure and components of your network to accommodate your Sun ONE Application Server 7, Enterprise Edition needs. Or, if your network structure is not flexible at this time, use the environment planning process to determine how you can best deploy Sun ONE Application Server 7, Enterprise Edition to fit within the existing network setup.

This phase is discussed in detail in [Chapter 2, “Planning your Environment.”](#)

## Selecting a Topology

Once you have determined the performance, availability, network, and infrastructure requirements, you then select a topology that best meets your performance needs. A topology is the schematic arrangement of the Sun ONE Application Server components and the communication flow between these components. This document describes two topologies (and their variations) that you can choose from.

This phase is discussed in detail in [Chapter 3, “Selecting a Topology.”](#)

## Running Tests

Once you configure the Sun ONE Application Server 7, Enterprise Edition, you deploy a representative sampling of applications and run tests on these applications to check whether the performance meets your performance goals. Wherever you identify bottlenecks, use this phase to fine-tune the system and improve performance.

This phase is highly dependent on your particular environment and is, therefore, not covered in this guide.



# Planning your Environment

Planning your environment is one of the first phases of deployment. In this phase, you first decide your performance and availability goals. You then make decisions about the hardware, network, and storage requirements accordingly.

The main objective of this phase is to determine the environment that best meets your business requirements.

This chapter contains the following sections:

- [Introducing HADB](#)
- [Establishing Performance Goals](#)
- [Planning Network Configuration to Meet Your Performance Goals](#)
- [Planning Availability](#)

## Introducing HADB

Sun<sup>TM</sup> Open Net Environment (Sun ONE) Application Server 7, Enterprise Edition supports persistence of HTTP sessions. The high-availability database (HADB) bundled with the Sun ONE Application Server works as the persistence store to provide high availability for applications.

An HADB node consists of a set of processes, a dedicated area of shared memory, and one or more secondary storage devices. It is used for storing and updating session data. There are two types of nodes:

- **Active Nodes:** An active node is a node that stores data.
- **Spare Nodes:** A spare node does not contain any data initially, but can start performing as an active node if an active node becomes unavailable.

Each active node must have a mirror node; therefore, active nodes occur in pairs. In addition, to maximize HADB availability, you should include two spare nodes for each pair so that if an active node fails, a spare node can take over while the failed node is repaired.

## Data Redundancy Units

HADB nodes are organized into two Data Redundancy Units (DRUs) that mirror each other. Each DRU consists of half of the active nodes and spare nodes, and contains one complete copy of the data. To ensure fault tolerance, the computers that support one DRU must be completely self-supported with respect to power (use of uninterruptible power supplies is needed), processing units, and storage. If a power failure occurs in one DRU, the nodes in the other DRU can continue servicing requests until the power returns.

Machines that host HADB Nodes must be added in pairs, with one machine in each DRU.

## Spare Nodes

A spare node is an additional HADB node connected to a DRU. A spare node initially does not contain data, but constantly monitors for failure of active nodes in the DRU. If an active node fails, the spare node takes over the functions of the failed node while the failed node is being repaired.

If you do not use spare nodes, if a machine goes down, the other node in the mirror takes over and continues service. However, the capacity reduces because that machine is no longer available to service requests. Depending on the impact of losing one machine, this may make your system effectively unavailable because the other machines become overloaded. Also, your system will be running without fault tolerance till you repair the machine because there is no mirror node to replicate the data. For high availability, it is required to minimize the time during which the system functions with only a single node.

Spare nodes allow a single machine to go down and still maintain overall level of service. Spare nodes are not mandatory, but should be used if you require high availability. You should allocate one machine for each DRU to act as a spare machine, so that if one of your machines goes down, your system can continue without getting overloaded. A spare node also makes it easy for you to perform planned maintenance on the machines that host the active nodes.

As a general rule, you should have a spare machine with enough application server instances and HADB nodes to replace any machine that becomes unavailable.

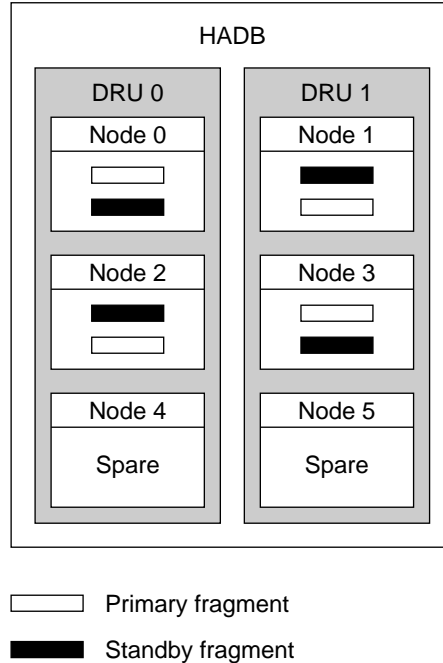
For example, if you have a co-located deployment with four Sun Fire™ V480 servers, where each server has one application server instance and two HADB data nodes, you should allocate two more servers as spare machines (one machine per DRU). Each spare machine should run one application server instance and two spare HADB nodes.

As another example, suppose you have a separate tier deployment where the HADB tier has two Sun Fire™ 280R servers, each running two HADB data nodes. If you want to maintain full capacity for this system even if a machine becomes unavailable, you should have one spare machine for the instances tier and one spare machine for the HADB tier. The spare machine for the instances tier should have as many instances as the other machines in the instances tier, and the spare machine for the HADB tier should have as many HADB nodes as the other machines in the HADB tier.

More information about the co-located and the separate tier deployment topologies is provided in [Chapter 3, “Selecting a Topology”](#).

## Sample HADB Architecture

The following figure shows the architecture of a database with four active nodes and two spare nodes. Nodes 0 and 1 are a mirror node pair, as are nodes 2 and 3.

**Figure 2-1** HADB Architecture with Four Active Nodes and Two Spare Nodes

## Establishing Performance Goals

As explained in [Chapter 1, “Overview of Deployment”](#), one of your main goals in deployment is to maximize performance. This principally translates into maximizing throughput and reducing response time.

Beyond these basic goals, you should establish specific goals by determining the following information.

- What capacity of requests, or throughput, can the system support?
- How many concurrent users can the system support?
- What is an acceptable average response time for requests submitted by your users?
- What is the average think time between requests?

These factors are interrelated. If you know any three of these four pieces of information, you can always calculate the fourth.

Some of the metrics described in this chapter can be calculated using a remote browser emulator (RBE) tool, or web site performance and benchmarking software, that simulates your enterprises web application activity. Typically, RBE and benchmarking products generate concurrent HTTP requests and then report back the response time and number of requests per minute. You can then use these figures to calculate server activity.

The results of the calculations described in the following sections are not absolute. Treat them as reference points to work against as you fine-tune the performance of the Sun ONE Application Server.

## Estimating Throughput

Throughput has different implications for application server instances and for the HADB.

A good measure of the throughput for application server instances is the requests per minute processed.

Similarly, a good measure of the throughput for the HADB is the requests per minute processed by the HADB and the session size per request. The session size per request is important because the amount of session data stored varies from request to request.

## Estimating Load on Application Server Instances

To estimate the load on application server instances, consider the following factors:

- [Maximum Number of Concurrent Users](#)
- [Think Time](#)
- [Average Response Time](#)
- [Requests Per Minute](#)

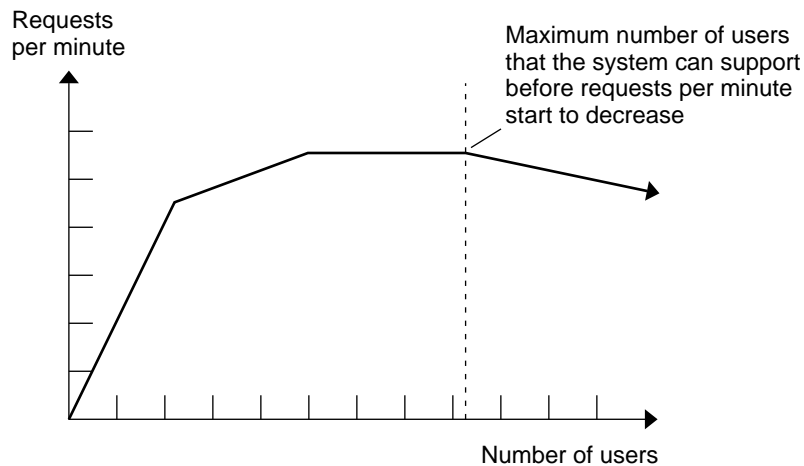
### Maximum Number of Concurrent Users

A user runs a process (for example through a web-browser) that periodically sends requests from a client machine to the Sun ONE Application Server 7, Enterprise Edition. When estimating the number of concurrent users, include all users currently active. A user is considered active as long as the session that user is running is active (for example, the session has not expired or been terminated).

A user is concurrent for as long as the user is on the system as a running process submitting requests, receiving results of requests from the server, and viewing the results of the requests.

Eventually, as the number of concurrent users submitting requests increases, requests processed per minute begin to decline (and the response time begins to increase). The following diagram illustrates this situation.

**Figure 2-2** Performance Pattern with Increasing Number of Users.



You want to identify the point at which adding more concurrent users reduces the number of requests that can be processed per minute, as this indicates when performance starts to degrade.

### Think Time

A user does not submit requests continuously. A user submits a request, the server receives the request, processes it and then returns a result, at which point the user spends some time analyzing the result before submitting a new request. This time spent reviewing the result of a request is called *think time*.

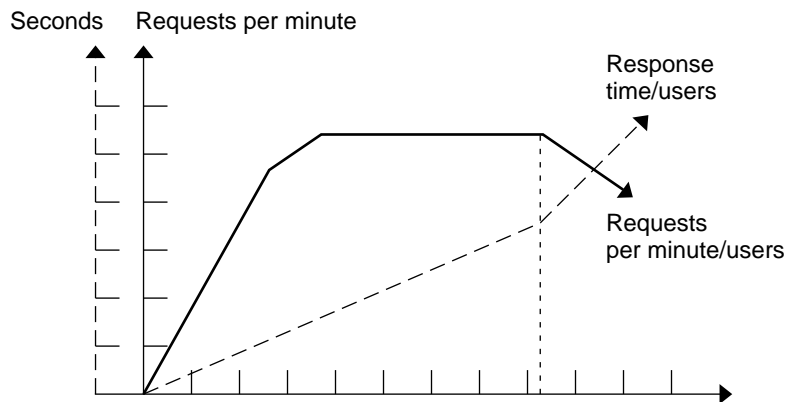
Determining typical think time length is important because you can use it to calculate more accurately the number of requests per minute and the number of concurrent users your system can support. Essentially, when a user is on the system but not submitting a request, a gap opens for another user to submit a request without altering system load. This also means that you can support more concurrent users.

## Average Response Time

Response time refers to the amount of time it takes for request results to be returned to the user. The response time is affected by a number of factors including network bandwidth, number of users, number and type of requests submitted, and average think time. In this section, response time refers to mean, or average, response time. Each type of request has its own minimal response time, but when evaluating system performance, analyze based on the average response time of all requests.

The faster the response time, the more requests per minute are being processed. However, as the number of users on your system increases, response time starts to increase as well, even though the number of requests per minute declines, as the following diagram illustrates:

**Figure 2-3** Response Time with Increasing Number of Users



A system performance graph like this indicates that after a certain point (point A in this diagram), requests per minute are inversely proportional to response time: the sharper the decline in requests per minute, the steeper the increase in response time (represented by the dotted line arrow).

In the diagram, point A represents peak load, the point at which requests per minute start to decline. Prior to this, response time calculations are not necessarily accurate because they are not using peak numbers in the formula. After this point, because of the inversely proportional relationship between requests per minute and response time, you can more accurately calculate response time using the two criteria: maximum number of users and requests per minute.

To determine response time at peak load, use the following formula:

$$\text{Response time} = (\text{concurrent users} / \text{requests per second}) - \text{think time in seconds}$$

To obtain an accurate response time result, you must always include think time in the equation.

For example, if the following conditions exist:

- Maximum number of concurrent users that your system can support at peak load is 5,000
- Maximum number of requests the system can process at peak load is 1,000 per second
- Average think time equals 3 seconds per request

$$\text{Response time} = (5000 / 1000) - 3 \text{ seconds think time}$$

The response time is, therefore, 2 seconds.

After you have calculated your system's response time, particularly at peak load, decide what is an acceptable response time for your enterprise. Response time, along with throughput, is one of the factors critical to Sun ONE Application Server performance and improving it should be one of your goals. If there is a response time beyond which you do not want to wait, and performance is such that you get response times over that level, then work towards improving your response time or redefine your response time threshold.

## Requests Per Minute

If you know the number of concurrent users at any given time, the response time of their requests and the average user think time at that time, you can determine requests per minute. Typically, you start by knowing how many concurrent users are on your system.

For example, after running some web site performance software, suppose you have calculated that the average number of concurrent users submitting requests on your online banking web site is 3,000. This is dependent on the number of users who have signed up to be members of your online bank, their banking transaction behavior, the times of the day or week they choose to submit requests, and so on. Therefore, knowing this information means you can use the requests per minute formula described in this section to calculate how many requests per minute your system can handle for this user base. Then, because requests per minute and response time become inversely proportional at peak load, decide if fewer requests per minute are acceptable as a trade-off for better response time, or alternately, if a slower response time is acceptable as a trade-off for more requests per minute.



Essentially, you start experimenting with the requests per minute and response time thresholds that you will accept as a starting point for fine-tuning system performance. Then you decide which areas of your system you want to adjust.

The requests per second formula is as follows:

$$\text{requests per second} = \text{concurrent users} / (\text{response time in seconds} + \text{think time in seconds})$$

For example, if the following conditions exists:

- Concurrent users equals 2,800
- Average response time equals 1 second per request
- Average think time equals 3 seconds

$$\text{Requests per second} = 2800 / (1+3)$$

Therefore, the number of requests per second is 700 and the number of requests per minute is 42000.

## Estimating Load on HADB

To calculate the load on HADB, consider the following factors:

- Requests per minute received by the HADB
- Session size per request

The session persistence settings that you specify affect the load to the HADB. For more information on configuring session persistence, see the *Sun ONE Application Server Administrator's Guide*.

## Understanding Session Persistence

As an application session proceeds, there is often data that is part of the session that is not stored in a traditional database. An example of such data is the content of your shopping cart. Sun ONE Application Server provides the capability to save, or persist, this session state in a repository, so that if an application server instance experiences a failure, the session state can be recovered and the session can continue without information loss.

Apart from the number of requests being served by the Sun ONE Application Server, the session persistence configuration settings that you specify affect the number of requests received per minute by the HADB and the session information in each request.

The persistence settings can be defined for each application server instance. However, all application server instances in a particular cluster must have the same persistence configuration. If you are using more than one cluster, it is not necessary for all clusters to have the same persistence configuration settings.

---

**NOTE** You should use the `cladmin` command to ensure that the session persistence settings are homogeneous for all instances in the cluster. For more information on using the `cladmin` command, see the *Sun ONE Application Server Administrator's Guide*.

---

### Number of Requests per Minute Received by the HADB

The number of requests per minute received by the HADB depends on the persistence frequency, which is the frequency at which the session information is stored in the HADB. This is defined through the persistence frequency settings. The persistence frequency options are:

- `web-method`: In the `web-method` persistence frequency mode, the session is stored after every web request.
- `time-based`: In the `time-based` persistence frequency, the session is stored at configurable time intervals.

The `web-method` persistency frequency provides the highest guarantee that the persisted session information will be up to date, but it increases the traffic to the HADB. This is because information is being stored to the HADB at the end of every web request. The `time-based` frequency reduces the traffic to the HADB but it provides less of a guarantee as compared to the `web-method` persistence frequency that the session information will be up to date.

### Comparison of Persistence Frequency Options

The table here summarizes the advantages and disadvantages of the persistence frequency options. The left column lists the persistence frequency option, the middle column lists the advantage(s) of the option, and the right column lists the disadvantage(s) of the option.

**Table 2-1** Comparison of Persistence Frequency Options

Persistence Frequency Option	Advantage(s)	Disadvantage(s)
web-method	Guarantees that the most up-to-date session information is available.	Potentially increased response time and reduced throughput.

**Table 2-1** Comparison of Persistence Frequency Options (*Continued*)

Persistence Frequency Option	Advantage(s)	Disadvantage(s)
time-based	Better response time and potentially better throughput.	Less guarantee than the web-method persistence frequency that the most updated session information is available after the failure of an application server instance.

## Session Size Per Request

The session size per request depends on how much session information is stored in the session.

---

**NOTE** To improve overall performance, reduce the amount of information in the session as much as possible.

---

You can further fine-tune the session size per request through the persistence scope settings. You can choose from the following options for persistence scope:

- `session`: The entire session is saved every time session information is saved to the HADB database.
- `modified-session`: The session is saved only if it has been modified.
- `modified-attribute`: Only those attributes are stored that have been modified (inserted, updated, or deleted) since the last time the session was stored.

---

**NOTE** The persistence scope `modified-attribute` is not certified as a full production-quality feature. If you use this persistence scope, evaluate the performance and stability of the Sun ONE Application Server in expected peak load conditions. If exceptions are logged or the response time is more than that is acceptable, do not use this persistence scope for your production environment.

---

# Comparison of Persistence Scope Options

The table here summarizes the advantages and disadvantages of the persistence scope options. The left column lists the persistence scope option, the middle column lists the advantage(s) of the option, and the right column lists the disadvantage(s) of the option.

**Table 2-2**    Comparison of Persistence Scope Options

Persistence Scope Option	Advantage(s)	Disadvantage(s)
modified-session	Provides improved response time for requests that do not modify session state.	Your application must call the <code>setAttribute</code> method (if the attribute was changed) or the <code>removeAttribute</code> method (if the attribute was removed) on the session during the execution of a web method (typically <code>doGet</code> or <code>doPost</code> ).
session	No constraint on applications.	Potentially poorer throughput and response time as compared to the <code>modified-session</code> and the <code>modified-attribute</code> options.

**Table 2-2**    Comparison of Persistence Scope Options (*Continued*)

Persistence Scope Option	Advantage(s)	Disadvantage(s)
modified-attribute	Better throughput and response time for requests in which the percentage of session state modified is low.	<ol style="list-style-type: none"><li>1. As the percentage of session state that gets modified for a given request grows to around 60%, the throughput and the response time degrade. In such cases, the performance gets worse than the session or modified-session persistence scope because of the overhead of splitting the attributes into separate records.</li><li>2. Your application must be written to meet the following constraints required by this mode:<ul style="list-style-type: none"><li>• Call <code>setAttribute</code> or <code>removeAttribute</code> every time you modify the session state.</li><li>• Make sure there are no cross references between attributes.</li><li>• Distribute the session state across multiple attributes, or at least between a read-only attribute and a modifiable attribute.</li></ul></li></ol>

## Designing for Peak Load or Steady State Load

In a typical deployment, there is a difference between steady state and peak workloads.

If you design for peak load, you must deploy a system that can sustain the expected maximum load of users and requests without a degradation in response time. This means that your system can handle the extreme cases of expected system load. If the difference between peak load and steady state load is substantial, designing for peak loads may mean that you are spending on resources that will be idle for a significant amount of time.

If you design for steady state load, then you don't have to deploy a system with all the resources required to handle the server's expected peak load. However a system designed to support upto steady load will have slower response time when peak load occurs.

## Frequency and Duration of Peak Load

A factor that may affect whether or not you want to design for peak load or for steady state is how often your system is expected to handle the peak load. If peak load occurs several times a day or even per week, you may decide that this is enough time to warrant expanding capacity to handle this load. If the system operates at steady state 90 percent of the time, and at peak only 10 percent of the time, then you may decide that you prefer deploying a system designed around steady state load. This means that 10 percent of the time your system's response time will be slower than the other 90 percent of the time. You decide if the frequency or duration of time that the system operates at peak justifies the need to add resources to your system, should this be required to handle peak load.

## Design Decisions to Make

Depending on the load on the application server instances, the load on the HADB, and the failover requirements, here are the design decisions that you should make at this stage:

- Number of applications server instances needed
- [Number of HADB Nodes Required](#)
- [HADB Storage Capacity Required](#)
- [Whether You Want to Design for Peak Load or for Steady State Load](#)

## Number of Applications Server Instances Needed

To determine the number of applications server instances needed, evaluate your environment on the basis of the factors explained in [“Estimating Load on Application Server Instances” on page 19](#). Each application server instance can use more than one CPU and should have at least one CPU allocated to it.

## Number of HADB Nodes Required

As a general guideline, you should plan to have one HADB node for each Central Processing Unit (CPU) in your system. For example, use two HADB nodes for a machine that has two CPUs.

---

<b>NOTE</b>	If you have more than one HADB node per machine (for example if you are using bigger machines), then you must ensure that there is enough redundancy and scalability on the machines—for example multiple uninterruptible power supplies, independent disk controllers, and so on.
-------------	--

---

## HADB Storage Capacity Required

The HADB provides near-linear scaling by adding more nodes until network capacity is exceeded. Each node must be configured with storage devices on a dedicated disk or disks. All nodes must have equal space allocated on the storage devices. Make sure that the storage devices are allocated on local disks.

Suppose the expected session data is  $X$  MB. HADB replicates the data on mirror nodes, and, therefore,  $2X$  MB of storage is needed.

Further, HADB uses indexes to enable fast access to data. An additional  $2X$  MB is required (for both nodes together) for indexes and assuming a less than 100% fillings rate. This means that a storage capacity of  $4X$  is required.

Therefore, the expected storage capacity needed by the HADB is four times the expected data volume.

If the system has to be designed for future expansion (by adding bigger disks to nodes or adding new nodes to the system) without loss of data from HADB, the expected storage capacity is eight times the expected data volume. This is because for online upgrades, you might want to refragment the data after adding new nodes. In that case, you will need a similar amount ( $4X$ ) of additional space on the data devices, thus increasing the total storage capacity to  $8X$ .

Additionally, HADB uses disk space for internal use as follows:

- Space for temporary storage of log buffer. This space is four times the `logBufferSize`. The `logBufferSize` is the size of the log buffer, which keeps track of operations related to data.

---

<b>NOTE</b>	The default value of <code>logBufferSize</code> is 48 MB.
-------------	---

---

- Space for internal administration purpose. This space is one percent of the storage device size.

For more information, see the *Sun ONE Application Server Administrator's Guide* and the *Sun ONE Application Server Performance Tuning Guide*.

The following table summarizes the HADB storage space requirements for a session data of X MB. The left column lists the condition (whether addition or removal of HADB nodes while online is required), and the right column lists the HADB storage space required.

Table 2-3 HADB Storage Space Requirement for Session Size of X MB	
Condition	HADB Storage Space Required
Addition or removal of HADB nodes while online is <i>not</i> required.	(4X MB) + (4*logBufferSize) + (1% of Device Size)
Addition or removal of HADB nodes while online is required.	(8X MB) + (4*logBufferSize) + (1% of Device Size)

If the HADB runs out of device space, error codes 4593 or 4592 are returned and error messages are written to the history file(s). For more information on these messages, see the *Sun ONE Application Server Error Message Reference*.

If the HADB runs out of device space, any client requests to insert or update data are not accepted. Delete operations are, however, accepted.

*Setting Data Device Size*

To set the size of the data device(s) of HADB, use the following command:

```
hadbm set TotalDatadeviceSizePerNode
```

The `hadbm` command restarts all the nodes, one by one, for the change to take effect. For more information, see the *Sun ONE Application Server Administrator's Guide*.

<b>NOTE</b>	The current version of the <code>hadbm</code> command does not add data devices to a running HADB database.
-------------	---

Whether You Want to Design for Peak Load or for Steady State Load

While making a decision about whether to design for peak or steady state load, refer to the information provided in [“Designing for Peak Load or Steady State Load” on page 27](#).



# Planning Network Configuration to Meet Your Performance Goals

When planning how to integrate Sun ONE Application Server into your network for optimal performance, you should estimate the bandwidth requirements and plan your network in a way that it can meet the performance requirements.

## Estimating Bandwidth Requirements

As you decide on the desired size and bandwidth of your network, first determine your network traffic and identify its peak. See whether there is a particular hour, day of the week, or day of the month in which overall volume peaks, and then determine the duration of that peak.

At all times consult network experts at your site about the size and type of all network components you are considering adding.

### Peak Load Times

During peak load times, the number of packets that are being sent is at its highest level. In general, if you design for peak load, scale your system with the goal of handling 100 percent of peak volume. Bear in mind, however, that any network behaves unpredictably and that despite your scaling efforts, 100 percent of peak volume might not always be handled.

For example, assume that at peak load, five percent of your users occasionally do not have immediate Internet access when accessing applications deployed on Sun ONE Application Server 7, Enterprise Edition. Of that five percent, determine how many users retry access after the first attempt. Again, not all of those users may get through, and of that unsuccessful portion, another percentage will retry. As a result, the peak appears longer because peak use is spread out over time as users continue to attempt access.

To ensure optimal access during the peak, start by verifying that your Internet service provider (ISP) has a backbone network connection that can reach an Internet hub without degradation.

### Calculating Bandwidth Required

Depending on the calculations you made in [“Establishing Performance Goals” on page 18](#), you should determine the additional bandwidth required for the Sun ONE Application Server deployment on your site.

Depending on your method of access (T-1 lines, ISDN, and so on), you can calculate the amount of increased bandwidth you require to handle your estimated load. For example, suppose your site uses T-1 or the higher-speed T-3 links for Internet access. Given their bandwidth, you can estimate how many lines you will need on your network based on the average number of requests generated per second at your site and the maximum peak load. You can calculate these figures using a web site analysis- and monitoring-tool.

A single T-1 line can handle 1.544 Mbps. So a network of four T-1 lines carrying 1.544 Mbps each can handle approximately 6 Mbps of data. Assuming that the average HTML page sent back to a client is 30 kilobytes (KB), this network of four T-1 lines can handle the following traffic per second:

$6,176,000 \text{ bits} / 8 \text{ bits} = 772,000 \text{ bytes per second}$

$772,000 \text{ bytes per second} / 30 \text{ KB} = \text{approximately } 25 \text{ concurrent client requests for pages per second.}$

At traffic of 25 pages per second, this system can handle 90,000 pages per hour (25 x 60 seconds x 60 minutes), and therefore 2,160,000 pages per day maximum, assuming an even load throughout the day. If the maximum peak load is greater than this, you will want to increase the bandwidth accordingly.

## Peak Load

Having an even load throughout the day is probably not realistic. You need to determine when peak load occurs, how long it lasts, and what percentage of the total load it is. For example, in the scenario outlined here, if peak load lasts for two hours and takes up 30 percent of the total load of 2,160,000 pages, this means that 648,000 pages must be carried over the T-1 lines during two hours of the day.

Therefore, to accommodate peak load during those two hours, you should increase the number of T-1 lines according to the following calculations:

$648,000 \text{ pages} / 120 \text{ minutes} = 5,400 \text{ pages per minute}$

$5,400 \text{ pages per minute} / 60 \text{ seconds} = 90 \text{ pages per second}$

If four lines can handle 25 pages per second, then approximately four times that many pages requires four times that many lines, in this case 16 lines. The 16 lines are meant for handling the realistic maximum of a 30 percent peak load.

Obviously, the other 70 percent of your load can be handled throughout the rest of the day by these many lines.

## Subnets

If you use the separate tier topology, in which the application server instances and HADB nodes are on separate tiers, you can achieve a performance improvement by keeping HADB nodes on a separate subnet. This is because HADB uses the User Datagram Protocol (UDP), and using a separate subnet reduces the UDP traffic on the machines outside of the subnet.

## Network Cards

For greater bandwidth and optimal network performance, use at least 100 Mbps Ethernet cards or, preferably, 1 Gbps Ethernet cards between servers hosting the Sun ONE Application Server and the HADB nodes and also among any other resources such as HADB databases that are hosted on other machines.

## Network Settings for HADB

Here are requirements and suggestions for HADB to work optimally in the network:

- Use switched routers so that each network interface has a dedicated 100 Mbps or better Ethernet channel.
- If you are running HADB on a multi-CPU machine hosting four or more HADB nodes, use 1 Gbps Ethernet cards.

---

**NOTE** If the average session size is greater than 50 KB, use 1 Gbps Ethernet cards even with less than four HADB nodes per machine.

---

- If you suspect network bottlenecks within HADB:
  - Run network monitoring software on your HADB computers to diagnose the problem.
  - Consider replacing any 100 Mbps Ethernet cards in the network with 1 Gbps Ethernet cards.
- The current release of HADB is not generally capable of running on computers with multiple network interface cards. If you need network bandwidth beyond what can be offered with a single network interface card per computer, consult Sun customer support for alternative solutions.

# Planning Availability

Availability must be planned according to the application and customer requirements.

There are two ways to achieve high availability:

- [Adding Redundancy to the System](#)
- [Using Multiple Clusters to Improve Availability](#)

## Adding Redundancy to the System

One way to achieve high availability is to add redundancy to the system—redundancy of hardware and software. When one unit fails, the redundant unit takes over. This is also referred to as fault tolerance.

In general, to achieve high availability, you should determine and remove every possible point of failure in the system.

### Failure Classes

The level of redundancy is determined by the failure classes (types of failure) that the system needs to tolerate. Some examples of failure classes are: system process, machine, power supply, disk, network failures, and building fires and catastrophes.

Duplicated system processes tolerate single system process failures. Duplicated machines tolerate single machine failures. Attaching the duplicated mirrored (paired) machines to different power supplies tolerates single power failures. By keeping the mirrored machines in separate buildings, a single building fire can be tolerated and by keeping them in separate geographical locations, natural catastrophes like earth quake in a location can be tolerated.

When planning availability, you should determine the failure classes covered by the system.

### Using Redundancy Units to Improve Availability

To improve availability, HADB nodes are always used in Data Redundancy Units (DRUs) as explained in [“Data Redundancy Units” on page 16](#).

## Using Spare Nodes to Improve Fault Tolerance

The use of spare nodes as explained in [“Spare Nodes” on page 16](#) improves fault tolerance. Although spare nodes are not mandatory, their use is recommended for maximum availability.

## Planning Failover Capacity

Failover capacity planning means deciding how many additional servers and processes to add to your Sun ONE Application Server installation so that in the event of a server or process failure, the system can seamlessly recover data and continue processing. If your system gets overloaded, a process or server failure might result, causing response time degradation or even total loss of service. Preparing for such an occurrence is critical to a successful deployment.

To maintain capacity, especially at peak loads, it is recommended that you add spare machines that run application server instances to your Sun ONE Application Server installation. For example, assume you have a system with two machines running one applications server instance each. Together, these machines can handle a peak load of 300 requests per second. If one of these machines becomes unavailable, the system is able to handle only 150 requests, assuming an even load distribution between the machines. This means that half the requests during peak load are not being served.

## Using Multiple Clusters to Improve Availability

To improve availability, instead of using a single cluster, you should group the application server instances into multiple clusters. This way, you can perform online upgrades for clusters (one by one) without loss of service.

For more information on setting up multiple clusters and using multiple clusters to perform online upgrades without loss of service, see the *Sun ONE Application Server Administrator's Guide*.



# Selecting a Topology

Once you have estimated the factors related to performance as explained in [Chapter 2, “Planning your Environment”](#), you should decide the *topology* that you will use to deploy the Sun<sup>TM</sup> Open Net Environment (Sun ONE) Application Server 7, Enterprise Edition.

A topology is the schematic arrangement of Sun ONE Application Server components (machines, application server instances, and HADB nodes) and the communication flow between these components.

This chapter describes two recommended topologies. These topologies are:

- Co-located: The application server instance and the HADB node are on the same machine.
- Separate Tier: The application server instance and the HADB node are on different machines.

Both the topologies have common building blocks—multiple application server instances that form a cluster, a mirrored set of HADB nodes, and HADB spare nodes. Both of them require a set of common configuration settings to function properly.

This chapter contains the following topics:

- [Common Requirements](#)
- [Co-located Topology](#)
- [Separate Tier Topology](#)
- [Comparison of the Topologies](#)

# Common Requirements

The following topics in this section describe the requirements that are common to both the topologies:

- [General Requirements](#)
- [HADB Nodes and Machines](#)
- [Load Balancer Configuration](#)

## General Requirements

- Machines that host HADB nodes must be provided in pairs.
- Each DRU must have the same number of machines. You must create the HADB database in such a way that the mirrored (paired) nodes are on a different DRU than the primary nodes.
- Each machine that hosts HADB nodes must have local disk storage, which is used to store all persisted information in the HADB.
- Machines that host the HADB nodes must run the same operating system. These machines should be as identical as possible in terms of configuration and performance.
- For session information to be persisted to the HADB, the application server instances must be in a cluster and satisfy all related requirements. For more information on using clusters, see the *Sun ONE Application Server Administrator's Guide*.
- The machines hosting the application server instances should be as identical as possible in terms of configuration and performance. This is because the load balancer plug-in uses a round-robin policy for load balancing, and if you have machines of different classes hosting instances, then the load will not be balanced in the most optimum way across these machines.
- Each DRU should preferably have a separate Uninterruptible Power Supply (UPS).



## HADB Nodes and Machines

Each DRU contains a complete copy of your data and can continue servicing requests in a degraded (that is, non-fault-tolerant) mode if the other DRU becomes unavailable. However, if a node in one DRU and its mirror in another DRU fail at the same time, some portion of your data is lost. For this reason, it is important that you do not set up your system so that both DRUs can be impacted by a single failure, such as a power failure, disk failure, and so on.

---

**NOTE** Each DRU must run on a completely independent, redundant system.

---

To increase capacity and throughput, add nodes in pairs with one node for each DRU.

Assume that each machine in your configuration runs  $N$  data nodes. The failure of a single machine brings down  $N$  nodes. It is therefore recommended that for each DRU you have  $N$  spare nodes.

It is recommended that you run the same number of HADB nodes on all machines and thereby balance load as evenly as possible.

---

**CAUTION** It is highly recommended that you do not run nodes from different DRUs on the same machine. If you do run nodes from different DRUs on the same machine, ensure that the machine can handle any single point of failure (for failures related to disk, memory, CPU, power, operating system crashes, and so on)

---

## Load Balancer Configuration

Both the topologies comprise application server instances in a cluster. These instances persist session information to the HADB. You must configure the load balancer to include configuration information for all the application server instances in the cluster.

For more information on setting up a cluster and adding application server instances to it, see the *Sun ONE Application Server Administrator's Guide*.

# Co-located Topology

In the co-located topology, the application server instance and the HADB nodes are on the same machine (hence the name *co-located*).

This topology requires lesser numbers of machines as compared to the separate tier topology explained later in the chapter. The co-located topology also offers improved effectiveness of CPU utilization—an application server instance and an HADB node share one machine and the processing is distributed evenly among them.

A minimum of two machines are required for this topology. To improve throughput, more machines can be added in pairs.

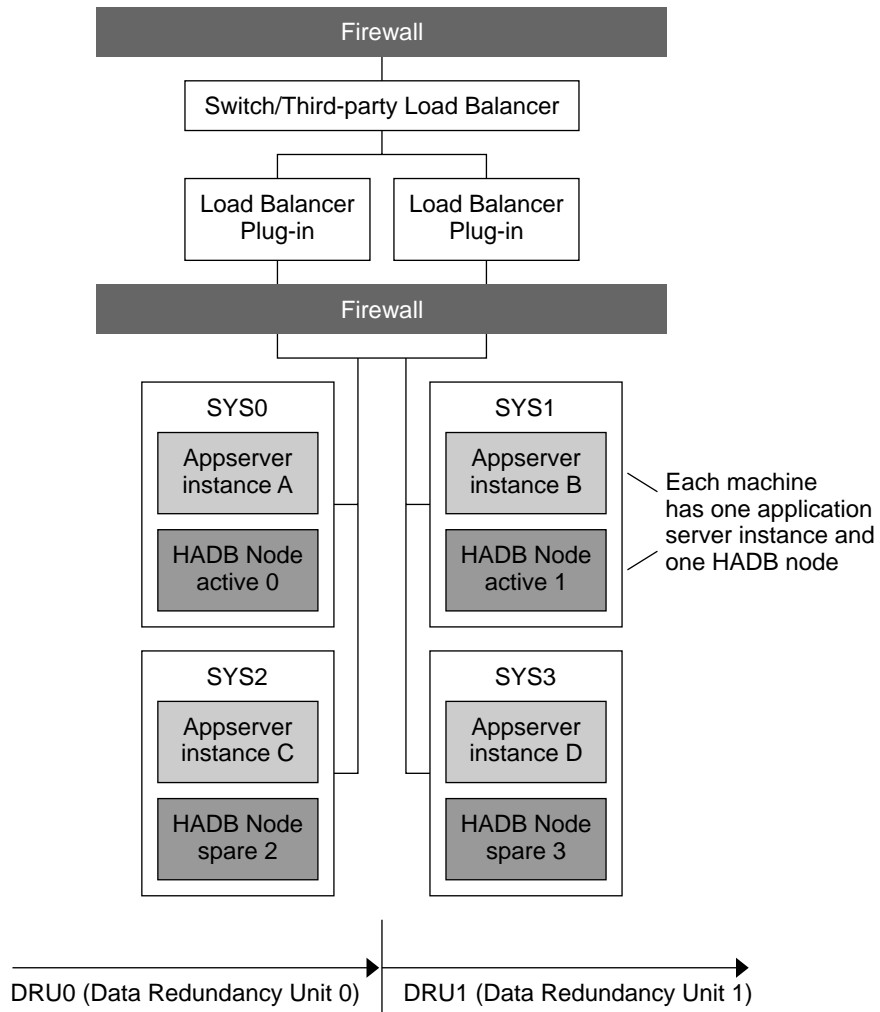
---

<b>NOTE</b>	The co-located topology is a good fit for large, Symmetric Multiprocessing (SMP) machines as you can take full advantage of the processing power of these machines.
-------------	---

---

## Reference Co-located Topology

The figure here shows the reference co-located topology.

**Figure 3-1** Reference Co-located Topology

Application server instance A is on machine SYS0, application server instance B is on machine SYS1, application server instance C is on machine SYS2, and application server instance D is on machine SYS3.

These four instances form a cluster that persists information to the two HADB Data Redundancy Units: DRU0 and DRU1.

DRU0 comprises two machines: SYS0 and SYS2. HADB Node active 0 is on the machine SYS0. HADB Node spare 2 is on the machine SYS2.

DRU1 comprises two machines: SYS1 and SYS3. HADB Node active 1 is on the machine SYS1. HADB Node spare 3 is on the machine SYS3.

---

**NOTE** The configuration settings described in this document assume that the host names for the machines correspond to the machine names as described in the topologies. For example, for the reference co-located topology, the host names are SYS0, SYS1, SYS2, and SYS3. This applies to both the topologies (and their variations).

---

## Configuration Settings for Reference Co-located Topology

Use the `clsetup` command for configuring the cluster (as part of the cluster configuration, the `clsetup` command creates an HADB database and sets to the JDBC connection pool and the JDBC resource for the HADB). The `clsetup` command is described in the *Sun ONE Application Server Installation Guide*.

The `clsetup` command uses the following input files:

- `clresource.conf`: This is the resource configuration file for the application server instances and the HADB.
- `clinstance.conf`: This file contains information about application server instances.

Make the changes as described in the subsequent sections to these input files before you run the `clsetup` command.

### *Changes to `clresource.conf` File*

For configuration related to the topologies described in this guide, the following properties should be changed in the `clresource.conf` file:

- `hosts`: A comma separated list of host names for the machines that host HADB active nodes. For each HADB active node, include the host name of the machine. Therefore, if a machine hosts for example two HADB nodes, the host name of the machine must appear twice.
- `steadypoolsize`: The value of the `steadypoolsize` property is calculated using the following formula:  

$$8 * (\text{number of HADB nodes}) / (\text{number of application server instances})$$

If the resulting number is a decimal, round it off to the next even number.
- `maxpoolsize`: The value of the `maxpoolsize` property is calculated using the following formula:

```
16*(number of HADB nodes)/(number of application server
instances)
```

If the resulting number is a decimal, round it off to the next even number.

---

**NOTE**

- The HADB nodes include both active and spare nodes.
  - The description and the calculation of values described here apply to both the topologies (and their variations).
- 

The following table describes the changes needed to the `clresource.conf` file for the reference co-located topology. The left column lists the section in the file in which the property to be changed is listed, the middle column lists the property name, and the right column lists the value of the property

**Table 3-1** Changes Needed to the `clresource.conf` File for Reference Co-located Topology

Section of <code>clresource.conf</code> File in Which the Property Appears	Property Name	Value
HADBINFO	hosts	SYS0,SYS1,SYS2,SYS3
JDBC_CONNECTION_POOL	steadypoolsize	8
JDBC_CONNECTION_POOL	maxpoolsize	16

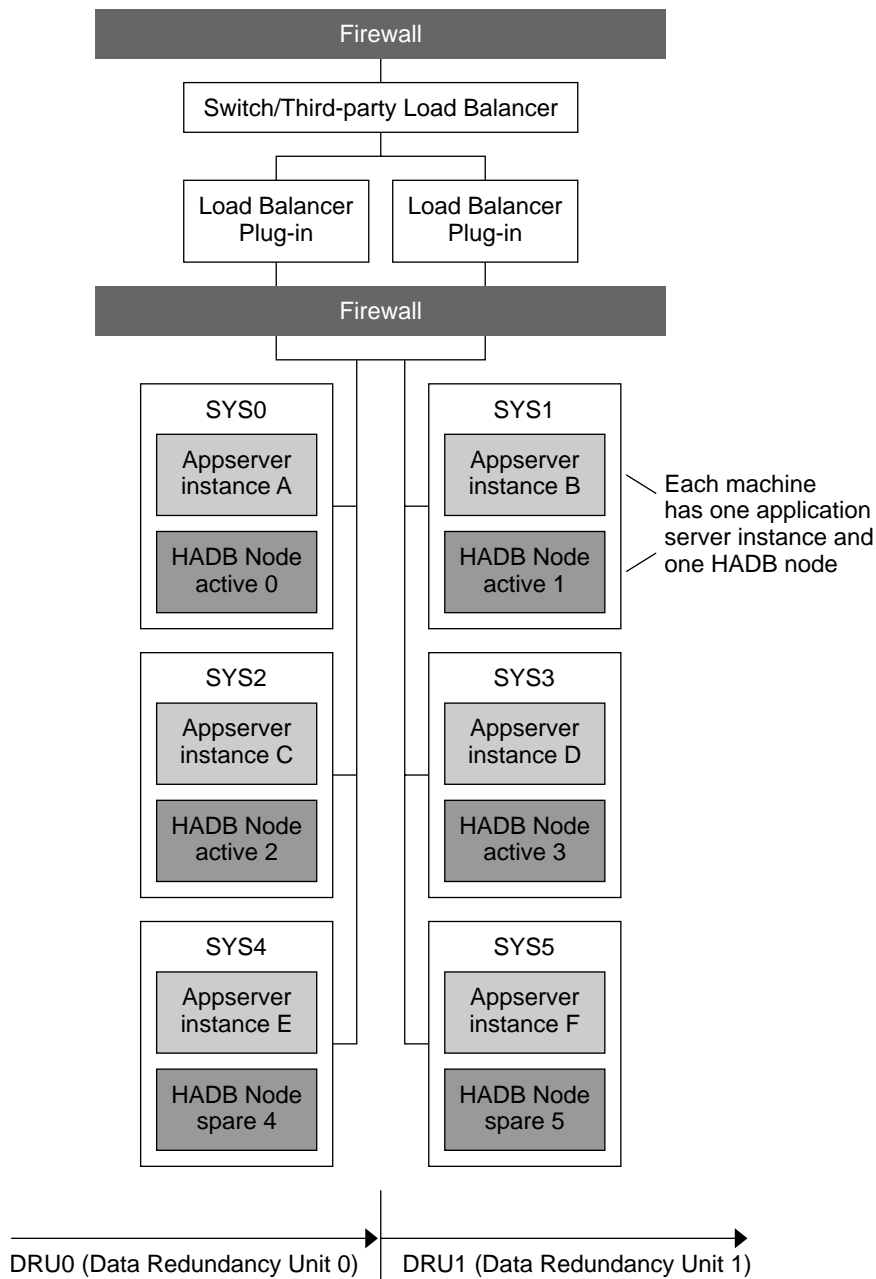
### *Changes to `clinstance.conf` File*

In the `clinstance.conf` file, include the information for each instance. For more information, see the *Sun ONE Application Server Installation Guide*. This applies to both topologies and their variations.

## Variation to Reference Co-located Topology

For better scalability and throughput, you can increase the number of application server instances and HADB nodes by adding more machines.

For example, you can add two machines, each with one application server instance and one HADB node. Make sure that you add the HADB nodes in pairs, assigning one node for each DRU. This configuration is shown in the figure here.

**Figure 3-2** Variation to Reference Co-located Topology

In this variation, the machines SYS4 and SYS5 have been added to the reference co-located topology described in [“Reference Co-located Topology” on page 40](#).

Application server instance A is hosted on machine SYS0, application server instance B is hosted on machine SYS1, application server instance C is hosted on machine SYS2, application server instance D is hosted on machine SYS3, application server instance E is hosted on machine SYS4, and application server instance F is hosted on machine SYS5.

These instances form a cluster that persists information to the HADB Data Redundancy Units DRU0 and DRU1.

Data Redundancy Unit DRU0 comprises the machines SYS0, SYS2, and SYS4. HADB Node active 0 is on the machine SYS0. HADB Node active 2 is on the machine SYS2. HADB Node spare 4 is on the machine SYS4.

Data Redundancy Unit DRU1 comprises the machines SYS1, SYS3, and SYS5. HADB Node active 1 is on the machine SYS1. HADB Node active 3 is on the machine SYS3. HADB Node spare 5 is on the machine SYS5.

## Configuration Settings for Variation to the Reference Co-located Topology

Make the changes as described in the subsequent sections before you run the `clsetup` command.

### *Changes to clresource.conf File*

The following table describes the changes needed to the `clresource.conf` file for the variation to the reference co-located topology as described in this section. The left column lists the section in the file in which the property to be changed is listed, the middle column lists the property name, and the right column lists the value of the property. For more information on the values of these properties, see [“Changes to clresource.conf File” on page 42](#).

**Table 3-2** Changes Needed to `clresource.conf` File for Variation to Reference Co-located Topology

Section of <code>clresource.conf</code> File in Which the Property Appears	Property Name	Value
HADBINFO	hosts	SYS0,SYS1,SYS2,SYS3,SYS4,SYS5
JDBC_CONNECTION_POOL	steadypoolsize	8
JDBC_CONNECTION_POOL	maxpoolsize	16

### *Changes to `clinstance.conf` File*

In the `clinstance.conf` file, include the information for each instance. For more information, see the *Sun ONE Application Server Installation Guide*.

## Separate Tier Topology

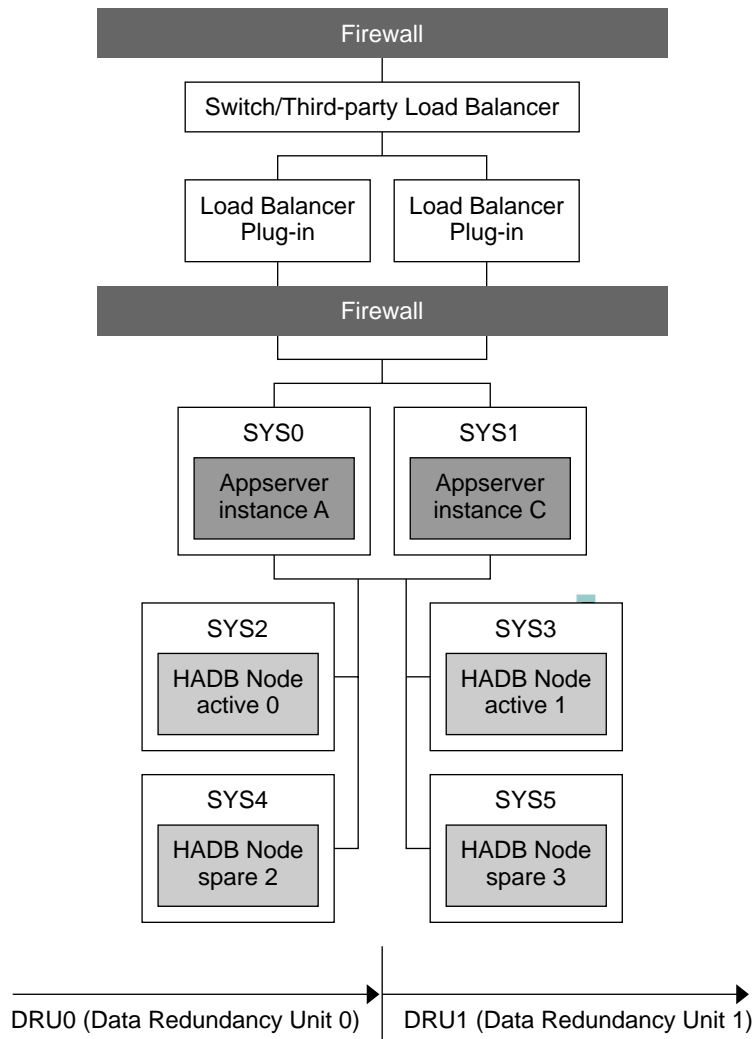
In this topology the application server instances and the HADB nodes are on different machines (therefore the name *separate tier*).

This topology requires more hardware than the co-located topology. This topology may be a good fit if you have different types of machine—you can allocate a different set of machines to the application server instances tier and to the HADB nodes tier. For example, you can use the more powerful machines for the application server instances tier and the less powerful machines for the HADB tier.

## Reference Configuration

The figure here shows the reference separate tier topology.



**Figure 3-3** Reference Separate Tier Topology

In this reference topology, the application server instance A is hosted on machine SYS0 and the application server instance B is hosted on the machine SYS1.

These two instances form a cluster that persists session information to Data Redundancy Units DRU0 and DRU1.

The Data Redundancy Unit DRU0 comprises two machines: SYS2 and SYS4. The HADB Node active 0 is on machine SYS2 and the HADB Node spare 2 is on machine SYS4.

The Data Redundancy Unit DRU1 comprises two machines SYS3 and SYS5. The HADB Node active 1 is on machine SYS3 and the HADB Node spare 3 on machine SYS5.

All the nodes on a DRU are on different machines, so that even if one machine becomes unavailable, the complete data for any DRU continues to be available on other machines.

### Configuration Settings for Reference Separate Tier Topology

Make the changes as described in the subsequent sections to these input files before you run the `clsetup` command.

*Changes to `clresource.conf` File*

The following table describes the changes needed to the `clresource.conf` file for the reference separate tier topology. The left column lists the section in the file in which the property to be changed is listed, the middle column lists the property name, and the right column lists the value of the property. For more information on the values of these properties, see [“Changes to `clresource.conf` File” on page 42](#).

**Table 3-3** Changes Needed to `clresource.conf` File for Reference Separate Tier Topology

Section of <code>clresource.conf</code> File in Which the Property Appears	Property Name	Value
HADBINFO	hosts	SYS2,SYS3,SYS4,SYS5
JDBC_CONNECTION_POOL	steadypoolsize	16
JDBC_CONNECTION_POOL	maxpoolsize	32

*Changes to `clinstance.conf` File*

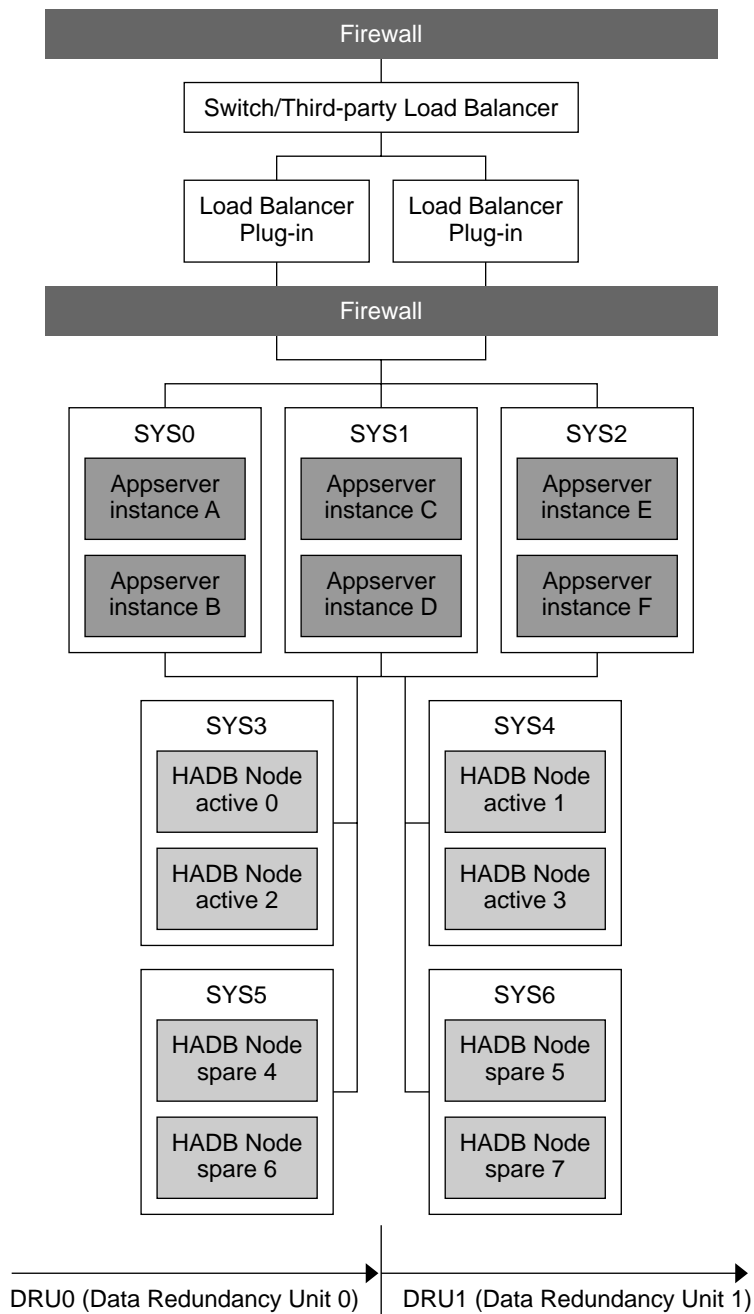
In the `clinstance.conf` file, include the information for each instance. For more information, see the *Sun ONE Application Server Installation Guide*.

## Variation to Reference Separate Tier Topology

You can increase the number of application server instances by adding more machines horizontally to the configuration. For example, you can add another machine to the reference configuration by creating a new application server instance. Similarly, you can increase the number of HADB nodes by adding more machines to host HADB nodes. Make sure that you add the HADB nodes in pairs with one node for each DRU.

This configuration is shown in the figure here.

**Figure 3-4** Variation to Reference Separate Tier Topology



In this configuration, each machine hosting application server instances has two application server instances. There are thus a total of six appserver instances in the cluster.

The HADB nodes are on machines SYS3, SYS4, SYS5, and SYS6.

The Data Redundancy Unit DRU0 comprises two machines: SYS3 and SYS5. HADB Node active 0 and the HADB Node active 2 are on machine SYS3. HADB Node spare 4 and the HADB Node spare 6 are on machine SYS5.

The Data Redundancy Unit DRU1 comprises two machines SYS4 and SYS6. HADB Node active 1 and HADB Node active 3 are on machine SYS4. HADB Node spare 5 and HADB Node spare 7 are on machine SYS6.

Each machine hosting HADB nodes hosts two HADB nodes each. There are thus a total of eight HADB nodes (four active nodes and four spare nodes).

## Configuration Settings for Variation to Reference Separate Tier Topology

Make the changes as described in the subsequent sections before you run the `clsetup` command.

### *Changes to clresource.conf File*

The following table describes the changes needed to the `clresource.conf` file for the variation to the reference separate tier topology. The left column lists the section in the file in which the property to be changed is listed, the middle column lists the property name, and the right column lists the value of the property. For more information on the values of these properties, see [“Changes to clresource.conf File” on page 42](#).

**Table 3-4** Changes Needed to `clresource.conf` File for Variation to Reference Separate Tier Topology

Section of <code>clresource.conf</code> File in Which the Property Appears	Property Name	Value
HADBINFO	hosts	SYS3,SYS4,SYS3,SYS4,SYS5,SYS6,SYS5,SYS6
JDBC_CONNECTION_POOL	steadypoolsize	12
JDBC_CONNECTION_POOL	maxpoolsize	22

*Changes to clinstance.conf File*

In the `clinstance.conf` file, include the information for each instance. For more information, see the *Sun ONE Application Server Installation Guide*.

# Comparison of the Topologies

The table here presents a comparison of the co-located topology and the separate tier topology. The left column lists the name of the topology, the middle column lists the advantages of the topology, and the right column lists the disadvantages of the topology

**Table 3-5**    Comparison of Topologies

Topology	Advantages	Disadvantages
Co-located Topology	<ul style="list-style-type: none"><li>Requires fewer numbers of machines as compared to the separate tier topology. Because the HADB nodes and the application server instances are on the same tier, you can create an application server instance on each spare node to handle additional load.</li><li>Improved effectiveness of CPU utilization. An application server instance and an HADB node share one machine and the processing is distributed evenly among them.</li><li>Useful for large, Symmetric Multiprocessing (SMP) machines as you can take full advantage of the processing power of these machines.</li></ul>	Increased complexity of maintenance. For example, if you want to perform maintenance tasks (that require shutting down of machines) on HADB nodes, the application server instances on the machine hosting HADB nodes also become unavailable while the machine is unavailable.

**Table 3-5**    Comparison of Topologies

Topology	Advantages	Disadvantages
<b>Separate Tier Topology</b>	<ul style="list-style-type: none"><li>• Easier maintenance. For example, you can perform maintenance tasks for the machines that host application server instances without having to bring down HADB nodes.</li><li>• Useful in situations where you have different types of machines. You can allocate a different set of machines to the application server instances tier and to the HADB tier. For example, you can use the more powerful machines for the application server instances tier and the less powerful machines for the HADB tier.</li></ul>	<ul style="list-style-type: none"><li>• Requires more machines as compared to the co-located topology. Because application server instances and HADB nodes are located on separate tiers, application server instances cannot be located on the machines that host the HADB spare nodes.</li><li>• Reduced effectiveness of CPU utilization. The tier consisting of application server instances and the tier consisting of HADB nodes will likely have uneven loads. This is more significant when the number of machines is smaller (four to six).</li></ul>

## Determining Which Topology to Use

You should test the different topologies mentioned in this chapter and experiment with different combinations of machines and CPUs to determine which topology (or its variation) best meets your performance and availability requirements.

Determine what trade offs you want to make to serve your needs the best. For example, if ease of maintenance is a critical requirement for you, the separate tier topology is more suitable. However, you will have to use a higher number of machines as compared to the co-located topology.

An important factor in the choice of topology is the type of machines you have in your setup. If you have large, Symmetric Multiprocessing (SMP) machines in your system, the co-located topology is an attractive option because you can take full advantage of the processing power of these machines. If you have different types

of machines, separate tier topology may be more useful because you can allocate a different set of machines to the application server instances tier and to the HADB tier. For example, you can use the more powerful machines for the application server instances tier and the less powerful machines for the HADB tier.



# Index

## A

- availability 12
  - for Data Redundancy Unit 39
  - improving with multiple clusters 35

## B

- bandwidth requirements, estimating 31
- building blocks, of topology 37

## C

- capacity, using spare machines to maintain 35
- clinstance.conf file 43
  - changes required 43
- clresource.conf file 42
  - changes for reference co-located topology 42
  - changes for reference separate tier topology 48
  - changes for variation to co-located topology 45
  - changes for variation to reference topology 51
- clsetup command 42
- clusters
  - using multiple clusters to improve availability 35
- co-located topology 37, 40

- configuration settings for reference topology 42
- configuration settings for variation 45
- reference topology 40
- using Symmetric Multiprocessing machines 40
- variation 43

- common topology requirements 38

- comparison of topologies 52

- configuration

  - load balancer 39

- configuration settings

  - for reference co-located topology 42

  - for reference separate tier topology 48

  - for variation to co-located topology 45

  - for variation to separate tier topology 51

## D

- Data Redundancy Unit

  - ensuring availability 39

  - improving availability with 34

  - number of machines in 38

  - power supply for 38

- deployment

  - about 11

  - important goals 11

- deployment phases

  - planning your environment 13, 15

- running tests 14
- selecting a topology 13

document, organization 6

## E

environment planning 13, 15

ethernet cards 33

## F

failover capacity, planning 35

failure

- classes 34
- types 34

fault tolerance 34

## G

goals, of deployment 11

## H

HADB 11

- network bottlenecks 33
- network settings for 33
- nodes 39
- spare nodes 35

HADBINFO 43, 48, 51

high availability, achieving 34

host names, specifying for topology 42

hosts 43, 48, 51

hosts, value of property 42

## I

intended audience 5

## J

JDBC 43

JDBC\_CONNECTION\_POOL 43, 48, 51

## L

load balancer configuration 39

local disk storage 38

## M

machines

- in Data Redundancy Unit 38
- maintaining capacity with spare machines 35

maximizing performance

- availability 12
- response time 12
- throughput 12

maxpoolsize 43, 48, 51

- calculating 42

mirrored machines 34

multiple clusters, improving availability with 35

multiple network cards 33

## N

network cards 33

- multiple 33

nodes 39

- spare 35

## P

- peak load 31, 32
- peak load times 31
- planning your environment 15
- pre-requisites, for using this guide 5

## R

- recommended topologies 37
- redundancy 34
- response time 12
- routers 33

## S

- selecting, topology 13
- separate tier topology 37, 46
  - configuration settings for 48
  - configuration settings for variation 51
  - reference configuration 46
  - variation 49
- spare machines, maintaining capacity with 35
- spare nodes 35
  - improving fault tolerance with 35
- steadypoolsize 43, 48, 51
  - calculating 42
- subnets 33
- Sun customer support 10
- Symmetric Multiprocessing machines, for co-located topology 40

## T

- tests, running tests 14
- throughput 12
- topology
  - building blocks of 37

- co-located 37, 40
- common requirements 38
- comparison 52
- determining which to use 53
- recommended topologies 37
- selecting 13, 37
- separate tier 37, 46
- specifying host names for 42
- types, of failure 34

## U

- User Datagram Protocol (UDP) traffic 33

