

uCosminexus Application Server

Operation, Monitoring, and Linkage Guide

3021-3-J10-10(E)

Notices

■ Relevant program products

See the *Release Notes*.

■ Export restrictions

If you export this product, please check all restrictions (for example, Japan's Foreign Exchange and Foreign Trade Law, and USA export control laws and regulations), and carry out all required procedures.

If you require more information or clarification, please contact your Hitachi sales representative.

■ Trademarks

HITACHI, Cosminexus, DABroker, HA Monitor, HiRDB, JP1, OpenTP1, TPBroker, uCosminexus are either trademarks or registered trademarks of Hitachi, Ltd. in Japan and other countries.

AIX is a trademark of International Business Machines Corporation, registered in many jurisdictions worldwide.

DB2 is a trademark of International Business Machines Corporation, registered in many jurisdictions worldwide.

IBM is a trademark of International Business Machines Corporation, registered in many jurisdictions worldwide.

Intel is a trademark of Intel Corporation or its subsidiaries.

Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

Microsoft, Active Directory are trademarks of the Microsoft group of companies.

Microsoft, Excel are trademarks of the Microsoft group of companies.

Microsoft, Internet Explorer are trademarks of the Microsoft group of companies.

Microsoft, SQL Server are trademarks of the Microsoft group of companies.

Microsoft, Windows are trademarks of the Microsoft group of companies.

Microsoft, Windows Server are trademarks of the Microsoft group of companies.

Microsoft is a trademark of the Microsoft group of companies.

Oracle, Java, and MySQL are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Red Hat is a registered trademark of Red Hat, Inc. in the United States and other countries.

Red Hat Enterprise Linux is a registered trademark of Red Hat, Inc. in the United States and other countries.

SPARC is a registered trademark of SPARC International, Inc. Products bearing SPARC trademarks are based on an architecture developed by Sun Microsystems, Inc

UNIX is a trademark of The Open Group.

Other company and product names mentioned in this document may be the trademarks of their respective owners.

Eclipse is an open development platform for tools integration provided by Eclipse Foundation, Inc., an open source community for development tool providers.

This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>).

■ Microsoft product screen shots

Microsoft product screen shots reprinted with permission from Microsoft Corporation.



■ Issued

Aug. 2022: 3021-3-J10-10(E)

■ Copyright

All Rights Reserved. Copyright (C) 2022, Hitachi, Ltd.

Preface

For details on the prerequisites before reading this manual, see the *Release Notes*.

■ Non-supported functionality

Some functionality described in this manual is not supported. Non-supported functionality includes:

- Audit log functionality
- Compatibility functionality
- Cosminexus Component Transaction Monitor
- Cosminexus Reliable Messaging
- Cosminexus TPBroker and VisiBroker
- Cosminexus Web Service - Security
- Cosminexus XML Security - Core functionality
- JP1 linkage functionality
- Management Server management portal
- Remote installation functionality for the UNIX edition
- SOAP applications complying with specifications other than JAX-WS 2.1
- uCosminexus OpenTP1 linkage functionality
- Virtualized system functionality
- XML Processor high-speed parse support functionality

■ Non-supported compatibility functionality

"Compatibility functionality" in the above list refers to the following functionality:

- Basic mode
- Check of JSP source compliance (cjjsp2java) with JSP1.1 and JSP1.2 specifications
- Database connection using Cosminexus DABroker Library
- EJB client application log subdirectory exclusive mode
- J2EE application test functionality
- Memory session failover functionality
- Servlet engine mode
- Simple Web server functionality
- Switching multiple existing execution environments
- Using EJB 2.1 and Servlet 2.4 annotation

Contents

Notices 2

Preface 4

Part 1: Overview

1	Application Server Functionality	14
1.1	Classification of functionality	15
1.1.1	Functionality that serves as the application execution platform	17
1.1.2	Functionality for operating and maintaining the application execution platform	18
1.1.3	Correspondence between functionality and manuals	19
1.2	Functionality corresponding to the purpose of the system	22
1.2.1	Functionality to support daily system operations	22
1.2.2	Functionality that support system maintenance	23
1.2.3	Operation functionality of a J2EE application	23
1.2.4	Functionality to support system audit (INTENTIONALLY DELETED)	23
1.2.5	Management functionality based on JP1 integration (INTENTIONALLY DELETED)	23
1.2.6	Node switching functionality based on cluster software linkage	24
1.3	Description of functionality mentioned in this manual	25
1.3.1	Meaning of classification	25
1.3.2	Examples of tables describing the classification	25
1.4	System operations by the Component Container Administrator (In UNIX)	27
1.5	Main updates in the functionality of Application Server 11-00	28
1.5.1	Simplifying implementation and setup	28
1.5.2	Supporting standard and existing functionality	28
1.5.3	Maintaining and improving reliability	29
1.5.4	Other purpose	29

Part 2: Operation and Monitoring Functionality

2	Starting and Stopping the System	30
2.1	Organization of this chapter	31
2.2	Starting and Stopping the System during Daily Operations	32
2.3	Mechanism of Starting and Stopping the Logical Server	33
2.3.1	Starting the Logical Server and Checking the Operations	33
2.3.2	Stopping the Logical Server	35
2.4	Automatically Restart When a Failure Occurs	36
2.4.1	Automatic restart of Management Server	36

2.4.2	Automatic restart of Administration Agent	36
2.4.3	Automatic restart of logical server	36
2.5	Monitoring System Operations	39
2.5.1	Monitoring the system operations	39
2.5.2	Monitoring the status	40
2.6	Settings for starting and stopping the system	44
2.6.1	Starting the system	44
2.6.2	Stopping the system	46
2.6.3	Settings for automatic start	48
2.6.4	Settings for automatic restart	50
2.6.5	Settings for automatic stop	53
3	Monitoring the Statistics (Statistics Collection Functionality)	59
3.1	Organization of this chapter	60
3.2	Overview of statistics collection functionality	61
3.3	Statistics File Output Functionality	62
3.3.1	Information Types that You Can Collect in Statistics Files	65
3.3.2	Information that You Can Collect in Statistics Files	67
3.3.3	Output Destination and Number of Statistics Files	68
3.3.4	Settings for the execution environment (J2EE server settings)	69
3.3.5	Output Format and Output Contents of Statistics Files	70
3.4	Event issuing functionality	86
3.4.1	Monitoring target for which threshold value can be set	86
3.4.2	How to issue an event	87
3.4.3	Definition in cosminexus.xml	89
3.4.4	Settings for the execution environment (J2EE server settings)	90
4	Monitoring Resource Depletion	92
4.1	Organization of this chapter	93
4.2	Overview of resource depletion monitoring functionality	94
4.3	Resource depletion monitoring functionality and output of resource depletion monitoring information	95
4.3.1	Types of resources that can be monitored	95
4.3.2	Definition in cosminexus.xml	97
4.3.3	Settings for the execution environment	97
4.3.4	Location of the resource depletion monitoring information file and types of information output to the file	100
4.3.5	Output Format and Contents	102
5	Operations of J2EE Applications	109
5.1	Organization of this chapter	110
5.2	Overview of J2EE Application Operations	111
5.2.1	Canceling a timeout request	111

5.2.2	Stopping a J2EE application	111
5.2.3	Forced termination of J2EE applications	111
5.2.4	Replacing J2EE applications	111
5.2.5	Accessing network resources from J2EE applications	112
5.2.6	Operations and references for J2EE applications	112
5.3	Monitoring and Canceling a J2EE Application During Runtime	114
5.3.1	Overview of monitoring and canceling the execution time of J2EE application	114
5.3.2	Monitoring the execution time of J2EE application	115
5.3.3	Method Timeout	116
5.3.4	Method Cancellation	118
5.3.5	Example of setting timeout value and the range of setting values	121
5.3.6	Thread count to be used	126
5.3.7	Definition in cosminexus.xml	126
5.3.8	Precautions When Implementing	127
5.3.9	Settings for the execution environment	130
5.3.10	Flow of monitoring and canceling the execution time of J2EE application	131
5.3.11	Confirming the execution status of a J2EE application	132
5.3.12	Canceling the request for which a timeout has occurred	133
5.3.13	Log information that is output while monitoring the execution time J2EE applications	134
5.4	Locking the J2EE application service	138
5.4.1	Locking the Web Application Service and Releasing the Lock	138
5.4.2	Methods of locking a service and stopping a J2EE application that can be executed for each system	141
5.4.3	Service Lock Using a Load Balancer	143
5.5	Stopping a J2EE application	145
5.5.1	Terminating the J2EE application	145
5.5.2	Lock process	147
5.5.3	Stop process	150
5.5.4	Forced termination process	152
5.5.5	Definition in cosminexus.xml	152
5.5.6	Settings for execution environment	152
5.5.7	Stopping a J2EE application	153
5.6	Switching the J2EE Application	159
5.6.1	Replacing the J2EE application	159
5.6.2	Switching the Web application by partially locking the service	162
5.6.3	Replacing and Maintaining a J2EE Application	163
5.7	Accessing network resources from J2EE applications	170
5.7.1	Overview of accessing network resources	170
5.7.2	Settings for accessing network resources	170
5.7.3	Problems during operations	173
5.8	Precautions when a J2EE application is operating	174

6	Audit Log Output Functionality (INTENTIONALLY DELETED)	175
6.1	INTENTIONALLY DELETED	176
7	Database Audit Trail Linkage Functionality (INTENTIONALLY DELETED)	177
7.1	INTENTIONALLY DELETED	178
8	Statistical Output using Management Commands	179
8.1	Organization of this chapter	180
8.2	Overview of statistics output using management commands	181
8.3	Methods to output server statistics	182
8.4	Items that can be checked by statistics monitoring	183
9	Automatic Execution of Processing Using Management Event Notification and Management Action	191
9.1	Organization of this Chapter	192
9.2	Overview of Management Event Notification and Management Action	193
9.3	Controlling the Execution of Management Action	195
9.4	Settings for automatic execution of processes by management events	198
9.4.1	Procedure to Set Automatic Execution of Processes by Management Events	198
9.4.2	Settings for execution environment	199
9.4.3	Setting a Message ID List File for Issuing Management Events	200
9.4.4	Setting Each Function Using the Management Event Issuing Function	201
9.4.5	Setting a Property File for Executing Management Actions	201
9.4.6	Settings of Management Action Execution Commands	203
10	Collecting CTM Statistics (INTENTIONALLY DELETED)	209
10.1	INTENTIONALLY DELETED	210
11	Output of the Console Log	211
11.1	Organization of this chapter	212
11.2	Output target of the console log	213
11.2.1	Target operations of the console log output	213
11.2.2	Target processes of the console log output	213
11.3	Settings for Acquiring the Console Log	215

Part 3: Linkage Functionality

12	Operating a JP1 Integrated System (INTENTIONALLY DELETED)	216
12.1	INTENTIONALLY DELETED	217

13	Centralized Monitoring of the System (Integrating with JP1/IM) (INTENTIONALLY DELETED) 218
13.1	INTENTIONALLY DELETED 219
14	Job-based Automatic Operations of a System (Integrating with JP1/AJS) (INTENTIONALLY DELETED) 220
14.1	INTENTIONALLY DELETED 221
15	Linking with Cluster Software 222
15.1	Organization of this chapter 223
15.2	Operations that can be implemented by linking with cluster software 224
15.2.1	Systems that operate the executing node and the standby node in the 1-to-1 ratio (1-to-1 node switching system) 225
15.2.2	System operating on mutual standby (mutual node switching systems) 226
15.2.3	System that operates the standby node as the recovery server (N-to-1 recovery system) 226
15.2.4	System in which Application Server machines (host) of the executing node and the standby node operate in a N-to-1 ratio (Node switching system for the host unit management model) 226
15.3	Prerequisites for linking with the cluster software 228
15.3.1	Servers for which nodes are to be switched 228
15.3.2	Usage of light transaction functionality 228
15.3.3	Operating methods of the systems 228
15.3.4	Prerequisites for the executing node 229
15.3.5	Prerequisites for the standby node 229
16	Node Switching System for the Host Management Model (Linked with Cluster Software) 230
16.1	Organization of this chapter 231
16.2	System configuration and operations of the node switching system for the host management model 232
16.2.1	Example of system configuration of the node switching system for the host management model 232
16.2.2	Node switching timing 233
16.2.3	Node switching procedure 234
16.3	Setting up the node switching system for the host management model 235
16.3.1	Procedure for setting up the node switching system for the host management model 235
16.3.2	Cluster server environment settings 236
16.3.3	Editing the setup file 237
16.3.4	Cluster settings 240
16.3.5	Application Server settings 241
16.4	Starting and stopping the node switching system for the host management model 244
17	1-to-1 Node Switching System (Linked with Cluster Software) 245
17.1	Organization of this chapter 246
17.2	Overview of 1-to-1 node switching system 247

17.3	System configuration and operations of 1-to-1 node switching system	248
17.3.1	Example system configuration for a 1-to-1 node switching system	248
17.3.2	Timing for node switching	249
17.3.3	Flow of the node switching process in the 1-to-1 node switching system	249
17.3.4	System operations during node switching	252
17.3.5	Inheriting information during node switching	253
17.4	Settings of the 1-to-1 node switching system of Application Server (In Windows)	254
17.4.1	Procedure for setting the 1-to-1 node switching system of Application Server	254
17.4.2	Setting the environment of the cluster server	256
17.4.3	Editing the configuration files	258
17.4.4	Setting a cluster	259
17.4.5	Setting Application Server	259
17.5	Settings of the 1-to-1 node switching system in the Management Server (In Windows)	261
17.5.1	Procedure for setting the 1-to-1 node switching system in the Management Server	261
17.5.2	Environment settings in the cluster server	263
17.5.3	Editing the setup files	264
17.5.4	Copying the Management Server settings from the active node to the spare node	265
17.5.5	Setting a cluster	266
17.6	Settings of the 1-to-1 node switching system of Application Server (In UNIX)	268
17.6.1	Procedure for setting the 1-to-1 node switching system of Application Server	268
17.6.2	Setting the environment of the cluster server	271
17.6.3	Editing the configuration files	272
17.6.4	Setting the environment of the HA monitor	273
17.6.5	Creating a shell script file	273
17.6.6	Setting the server-compliant environment	276
17.6.7	Setting the LAN status	277
17.6.8	Setting Application Server	278
17.7	Settings of the 1-to-1 node switching system for the Management Server (In UNIX)	280
17.7.1	Procedure for setting the 1-to-1 node switching system in the Management Server	280
17.7.2	Environment settings in the cluster server	283
17.7.3	Editing the setup files	283
17.7.4	Copying the Management Server settings from the active node to the spare node	284
17.7.5	HA monitor environment settings	285
17.7.6	Creating the shell script file	286
17.7.7	Server-compliant environment settings	287
17.7.8	LAN status setup	289
17.8	Starting and stopping the 1-to-1 node switching system in Application Server (In Windows)	290
17.8.1	Starting the 1-to-1 node switching system of Application Server	290
17.8.2	Stopping the 1-to-1 node switching system of Application Server	291
17.8.3	Starting and stopping during planned node switching in the 1-to-1 node switching system of Application Server	291

17.8.4	Starting and stopping the system during maintenance of the 1-to-1 node switching system of Application Server	291
17.9	Starting and stopping a 1-to-1 node switching system for the Management Server (In Windows)	295
17.9.1	Starting the 1-to-1 node switching system for the Management Server	295
17.9.2	Stopping the 1-to-1 node switching system for the Management Server	295
17.9.3	Starting and stopping a system when there is planned node switching in the 1-to-1 node switching system for the Management Server	296
17.10	Starting and stopping a 1-to-1 node switching system for the Management Server (In UNIX)	297
17.10.1	Starting the 1-to-1 node switching system	297
17.10.2	Stopping the 1-to-1 node switching system	298
17.10.3	Starting and stopping a system when there is a planned node switching in the 1-to-1 node switching system	300
17.10.4	Starting and stopping the system during maintenance of the 1-to-1 node switching system of Application Server	300
17.10.5	Starting and stopping the 1-to-1 node switching system of the Management Server for maintenance	303
17.11	Confirming the settings of a node switching system	305
17.11.1	J2EE server settings	305
17.11.2	HTTP Server settings	305
17.11.3	Management server settings	305
18	Mutual Node Switching System (Linked with Cluster Software)	306
18.1	Organization of this chapter	307
18.2	Overview of the mutual node switching system	308
18.3	System configuration and operations of mutual node switching systems	309
18.3.1	Example of system configuration of the mutual node switching system	309
18.3.2	Flow of the node switching process in the mutual node switching system	311
18.4	Settings for the mutual node switching system (In Windows)	314
18.4.1	Procedure for setting up the mutual node switching systems	314
18.4.2	Environment settings of the cluster server	317
18.4.3	Editing the setup file	318
18.4.4	Setting a cluster	319
18.4.5	Setting Application Server	320
18.5	Settings for the mutual node switching system (In UNIX)	323
18.5.1	Procedure for setting up the mutual node switching system	323
18.5.2	Environment settings in the cluster server	326
18.5.3	Editing the setup file	327
18.5.4	HA monitor environment settings	329
18.5.5	Creating the shell script file	329
18.5.6	Server-compliant environment settings	331
18.5.7	LAN status setup	333
18.5.8	Setting Application Server	333
18.6	Starting and stopping the mutual node switching system (In Windows)	337

18.6.1	Starting the mutual node switching system	337
18.6.2	Stopping the mutual node switching system	338
18.6.3	Starting and stopping a system for planned node switching in the mutual node switching system	339
18.7	Starting and stopping the mutual node switching system (In UNIX)	340
18.7.1	Starting the mutual node switching system	341
18.7.2	Stopping the mutual node switching system	343
18.7.3	Starting and stopping a system for planned node switching in the mutual node switching system	345
18.7.4	Starting and stopping the mutual node switching system for maintenance	347

19 N-to-1 Recovery System (Linked with Cluster Software) 351

19.1	Organization of this chapter	352
19.2	Overview of the N-to-1 recovery system	353
19.3	System configuration and operations of the N-to-1 recovery system	354
19.3.1	Example of system configuration of the N-to-1 recovery system	354
19.3.2	Flow of the recovery process	355
19.3.3	Inheriting information during node switching	356
19.4	Settings for the N-to-1 recovery system (In Windows)	358
19.4.1	Procedure for setting the N-to-1 recovery system	358
19.4.2	Setting the environment of the cluster server	361
19.4.3	Editing the configuration files	362
19.4.4	Setting a cluster	362
19.4.5	Setting Application Server	367
19.5	Settings for the N-to-1 recovery system (In UNIX)	369
19.5.1	Procedure for setting the N-to-1 recovery system	369
19.5.2	Setting the environment of the cluster server	374
19.5.3	Editing the configuration files	374
19.5.4	Setting the environment of the HA monitor	375
19.5.5	Creating a shell script	376
19.5.6	Setting the server-compliant environment	380
19.5.7	Setting the LAN status	383
19.5.8	Setting Application Server	384
19.6	Starting and stopping the N-to-1 recovery system (In Windows)	386
19.6.1	Starting the N-to-1 recovery system	386
19.6.2	Stopping the N-to-1 recovery system	387
19.7	Starting and stopping the N-to-1 recovery system (In UNIX)	388
19.7.1	Starting the N-to-1 recovery system	388
19.7.2	Stopping the N-to-1 recovery system	389

Appendixes 391

A	Settings of Cosminexus TPBroker for Integrating Cluster Software (in Windows)	392
A.1	Starting and stopping a system	392
A.2	Settings for using the ORB functionality	392

B	Settings of Cosminexus TPBroker for Integrating Cluster Software (In UNIX)	398
B.1	Starting and stopping a system	398
B.2	Settings for using the ORB functionality	399
C	Contents of the Protected Area List	404
D	Main Functionality Changes in Each Version	445
D.1	Main functional changes in version 09-87	445
D.2	Main functional changes in version 09-80	445
D.3	Main functional changes in version 09-70	446
D.4	Main functional changes in version 09-60	447
D.5	Main functional changes in version 09-50	448
D.6	Main updates in the functionality of 09-00	451
D.7	Main updates in the functionality of 08-70	454
D.8	Main updates in the functionality of 08-53	456
D.9	Main updates in the functionality of 08-50	458
D.10	Main functionality changes in 08-00	460
E	Glossary	464

Index 465

1

Application Server Functionality

This chapter describes the classification and purpose of the Application Server functionality, and the manuals corresponding to the functionality. This chapter also describes the functionality that has changed in this version.

1.1 Classification of functionality

Application Server is a product for building an application execution environment based on a J2EE server that supports Java EE 7 and for developing applications that can operate in the execution environment. You can use a variety of functionality, such as functionality compliant with Java EE standard specifications and functionality independently extended on Application Server. By selecting and using functionality according to the purpose and intended use, you can build and operate a highly reliable system having an excellent processing performance.

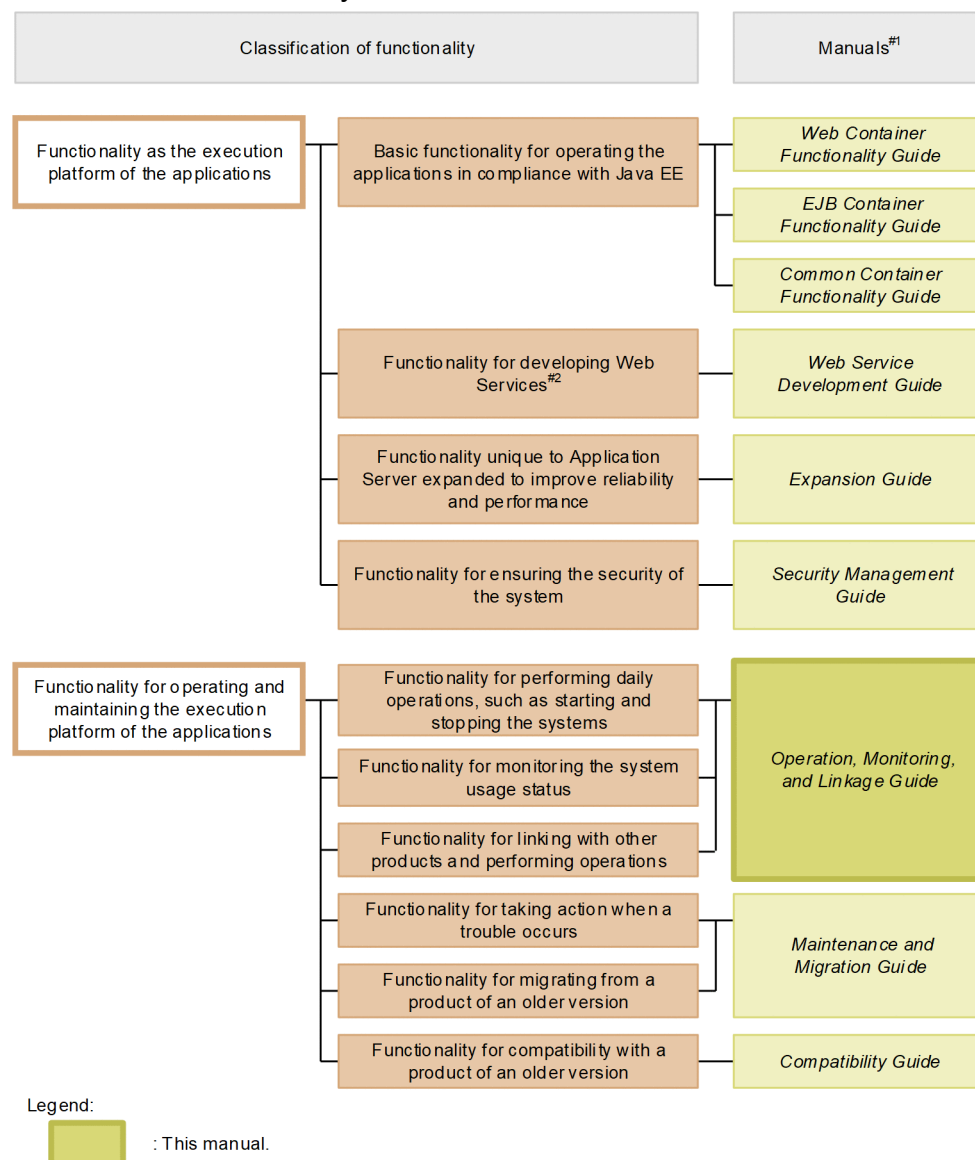
The following are the broad classifications of the Application Server functionality:

- Functionality that serves as an execution platform for the applications
- Functionality that is used for operating and maintaining the execution platform for the applications

The above-mentioned functionality can be further classified according to the positioning and the intended usage of functionality. The Application Server manuals are provided according to the classification of functionality.

The following figure shows the classification of Application Server functionality and the manuals corresponding to the functionality.

Figure 1–1: Classification of Application Server functionality and the manuals corresponding to the functionality



#1

uCosminexus Application Server has been omitted from the manual names.

#2

With Application Server, you can execute SOAP Web Services and RESTful Web Services. Depending on the purpose, reference the following manuals apart from the *uCosminexus Application Server Web Service Development Guide*.

When developing and executing SOAP applications

- *uCosminexus Application Server SOAP Application Development Guide*

When ensuring the security of SOAP Web Services and SOAP applications

- *uCosminexus Application Server XML Security - Core User Guide*
- *uCosminexus Application Server Web Service Security Users Guide*

For details about the XML processing

- *uCosminexus Application Server XML Processor User Guide*

The following subsections describe the classification of functionality and the manuals corresponding to the functionality.

1.1.1 Functionality that serves as the application execution platform

This functionality works as a platform for executing online businesses and batch businesses implemented as the applications. You choose functionality that you want to use according to the intended use of a system and your requirements.

You must determine whether you want to use functionality that serves as the execution platform for the applications, even before you perform the system building or application development.

The following are the classification-wise descriptions of functionality that serve as the application execution platform:

(1) Basic functionality to operate applications (basic development functionality)

This functionality includes the basic functionality for operating applications (J2EE applications). The *J2EE server functionality* is mainly included.

Application Server provides a J2EE server that supports Java EE 7. The J2EE server provides functions that comply with the standard specifications and functions that are specific to Application Server.

The basic development functionality can be further classified into three types according to the types of the J2EE applications for which you use functionality. The manuals for Application Server function guide have been divided according to this classification.

The following is an overview of each classification:

- **Functionality for executing the Web applications (Web containers)**

This classification includes the Web container functionality that serves as the execution platform for Web applications and functionality executed by linking Web containers and Web servers.

- **Functionality for executing the Enterprise Beans (EJB containers)**

This classification includes the EJB container functionality that serves as the execution platform of the Enterprise Bean. This classification also includes the EJB client functionality for invoking the Enterprise Bean.

- **Functionality used in both Web applications and Enterprise Beans (Container common function)**

This classification includes functionality that can be used in the Web applications and the Enterprise Beans operating Web containers and EJB containers respectively.

(2) Functionality for developing Web Services

This includes the functionality as the execution and development environments of Web Services.

Application Server provides the following engines:

- The JAX-WS engine to implement the binding of SOAP Messages according to the JAX-WS specifications
- The JAX-RS engine to implement the binding of RESTful HTTP messages according to the JAX-RS specifications

(3) Application Server independent functionality extended for improving the reliability and performance (extended functionality)

This includes functionality extended independently on Application Server. This also includes functionality implemented by using non-J2EE server processes such as batch server, CTM, and database.

On Application Server, various functionality are extended to improve the reliability of the system, and to implement stable operations. Furthermore, functionality is also extended to operate applications other than J2EE applications (batch applications) in the Java environment.

(4) Functionality for ensuring the security of a system (security management functionality)

This functionality is used for ensuring the security of an Application Server-based system. This includes functionality such as the authentication functionality used for preventing unauthorized access and the encryption functionality used for preventing information leakage in communication channels.

1.1.2 Functionality for operating and maintaining the application execution platform

This functionality is used for effectively operating and maintaining the application execution platform. You use this functionality, after starting the system operations, as and when required. However, depending on functionality, you must implement the settings and applications in advance.

The following are the classification-wise descriptions of functionality used for operating and maintaining the application execution platform:

(1) Functionality used for daily operations, such as starting and stopping the systems (operation functionality)

This classification includes functionality used in daily operations such as starting or stopping systems, starting or stopping applications, and replacing the applications.

(2) Functionality for monitoring system usage (watch functionality)

This classification includes functionality used for monitoring the system usage and resource depletion. This classification also includes functionality to output information used in monitoring the system operation history.

(3) Functionality for operating the system by linking with other products (linkage functionality)

This classification includes functionality to be linked and implemented with other products such as JP1 and the cluster software.

(4) Functionality for troubleshooting (maintenance functionality)

This classification includes functionality used for troubleshooting. This classification also includes functionality used for displaying the information that will be referenced during the troubleshooting.

(5) Functionality for migrating from products of older versions (migration functionality)

This classification includes functionality used for migrating from an older Application Server to a new Application Server.

(6) Functionality for compatibility with products of older version (compatibility functionality)

This includes functionality used for the compatibility with older versions of Application Server. For the compatibility functionality, we recommend the migration to the corresponding recommended functionality.

1.1.3 Correspondence between functionality and manuals

The function guides for Application Server have been divided according to the classifications of functionality.

The following table describes the classifications of functionality and the manuals corresponding to the functionality.

Table 1–1: Classification of functionality and correspondence with the manuals explaining the functionality

Category	Functionality	Reference manuals#1
Basic development functionality	Web container	<i>Web Container Functionality Guide</i>
	Using JSF and JSTL	
	Using JAX-RS 2.0	
	WebSocket	
	NIO HTTP server	
	Servlet and JSP implementation	
	EJB container	<i>EJB Container Functionality Guide</i>
	EJB client	
	Precautions during Enterprise Bean implementation	
	Naming management	<i>Common Container Functionality Guide</i>
	Resource connections and transaction management	
	Invoking Application Server from OpenTP1 (TP1 inbound integrated function)	
	Using JPA 2.1	
	Cosminexus JMS Provider	
	Using JavaMail	
	Using CDI with Application Server	
	Using Bean Validation with Application Server	
	Java Batch	
	JSON-P	

Category	Functionality	Reference manuals ^{#1}
	Concurrency Utilities	
	Application property management	
	Using annotations	
	Formatting and deploying J2EE applications	
	Container extension library	
Extended functionality	Executing applications using the batch server	<i>Expansion Guide</i>
	Scheduling and load balancing requests using CTM	
	Scheduling the batch applications	
	Inheriting the session information between the J2EE servers (Session failover functionality)	
	Database session failover functionality	
	Suppressing Full GC execution by using the Explicit Memory Management functionality	
	Output of the application user log	
Security management functionality	Authentication using the integrated user management	<i>Security Management Guide</i>
	Authentication using application settings	
	Using TLSv1.2 for the SSL/TLS communication	
	Controlling with the management functionality of the load balancers that use API-based direct connections	
Operation functionality	Starting and stopping the system	<i>Operation, Monitoring, and Linkage Guide</i> ^{#2}
	Managing J2EE applications	
Watch functionality	Monitoring the operation information (Statistics collection functionality)	
	Monitoring resource depletion	
	Output of operation information using the management commands	
	Automatic execution of processing by using management event notification and management action	
	Output of the console log	
Linkage functionality	Linking with cluster software	
	1-to-1 node switching system (Linking with cluster software)	
	Mutual node switching system (Linking with cluster software)	
	Node switching system for host unit management models (Integrating with cluster software)	
Maintenance functionality	Troubleshooting related functionality	<i>Maintenance and Migration Guide</i>
	Analyzing performance using the performance analysis trace	
	JavaVM functionality of products (hereafter, might be abbreviated as JavaVM)	
Migration functionality	Migrating from an older version of Application Server	

Category	Functionality	Reference manuals ^{#1}
	Migrating to a recommended functionality	
Compatibility functionality	Functionality for compatibility with the basic development functionality	<i>Compatibility Guide</i>
	Functionality for compatibility with the extended functionality	

#1

uCosminexus Application Server is omitted from the above manual names.

#2

This manual.

1.2 Functionality corresponding to the purpose of the system

On Application Server, you must choose the applicable functionality according to the purpose of the system to be built and operated.

This subsection describes how and where to use the following functionality mentioned in this manual. Note that the functionality is independently extended by Application Server.

- Functionality to support daily system operations
- Functionality that support system maintenance
- J2EE application operation functionality
- Functionality to support system audit
- Management functionality based on JP1 integration
- Node switching functionality based on cluster software linkage

The functionality-wise support for the following items are described here:

- **Reliability**

This functionality works well for performance-focused systems.

Includes functionality used in the performance tuning of the system.

- **Operation and maintenance**

This functionality works well when you want efficient operations and maintenance.

1.2.1 Functionality to support daily system operations

The following table lists the purpose of functionality to support daily system operations. Choose the functionality in accordance with the system purpose. For details on the functionality, see the *Reference* column in the following table.

Table 1–2: Purpose of functionality to support daily system operations

Functionality name	Purpose		Reference
	Reliability	Operations/ Maintenance	
Starting and stopping the system	--	Y	Chapter 2
Monitoring the statistics (Statistics collection functionality)	--	Y	Chapter 3
Monitoring resource depletion	--	Y	Chapter 4
Statistical output using management commands	--	Y	Chapter 8
Automatic execution of processing using management event notification and management action	--	Y	Chapter 9

Legend:

Y: Applicable

--: Not applicable

1.2.2 Functionality that support system maintenance

The following table lists the purpose of functionality to support system maintenance. Choose the functionality in accordance with the system purpose. For details on the functionality, see the *Reference* column in the following table.

Table 1–3: Purpose of functionality to support system maintenance

Functionality name	Purpose		Reference
	Reliability	Operation/ Maintenance	
Output of the console log	--	Y	Chapter 11

Legend:

Y: Applicable

--: Not applicable

1.2.3 Operation functionality of a J2EE application

The following table lists the purpose of the operation functionality of a J2EE application. Choose the functionality in accordance with the system purpose. For details on the functionality, see the *Reference* column in the following table.

Table 1–4: Purpose of the operation functionality of a J2EE application

Functionality name	Purpose		Reference
	Reliability	Operation/ Maintenance	
Monitoring and canceling a J2EE application during runtime	--	Y	5.3
Locking the J2EE application service	--	Y	5.4
Stopping a J2EE application	--	Y	5.5
Switching the J2EE application	--	Y	5.6
Accessing network resources from J2EE applications	--	Y	5.7

Legend:

Y: Applicable

--: Not applicable

1.2.4 Functionality to support system audit (INTENTIONALLY DELETED)

INTENTIONALLY DELETED.

1.2.5 Management functionality based on JP1 integration (INTENTIONALLY DELETED)

INTENTIONALLY DELETED.

1.2.6 Node switching functionality based on cluster software linkage

The following table lists the purpose of the node switching functionality based on cluster software linkage. Choose the functionality in accordance with the system purpose. For details on the functionality, see the *Reference* column in the following table.

Table 1–5: Purpose of the node switching functionality based on cluster software linkage

Functionality name	Purpose		Reference
	Reliability	Operation/ Maintenance	
Linking with cluster software	Y	Y	Chapter 15
Node switching system for a host management model	Y	Y	Chapter 16
1-to-1 node switching system	Y	Y	Chapter 17
Mutual node switching system	Y	Y	Chapter 18
N-to-1 recovery system	Y	Y	Chapter 19

Legend:

Y: Applicable

1.3 Description of functionality mentioned in this manual

This subsection describes the meaning of the classification used in the description of functionality in this manual and the example tables describing the classification.

1.3.1 Meaning of classification

The description of functionality in this manual is classified into the following five points. You can select and read the required location depending on the purpose of referencing the manual.

- **Explanation**
This is the description about functionality. This section describes the purpose, features, and mechanism of functionality. Read this section when you want an overview of functionality.
- **Implementation**
This section describes the methods, such as the coding method and the DD writing method. Read this section when you develop applications.
- **Settings**
This section describes the required property settings for building systems. Read this section when you build a system.
- **Operations**
This section describes the operation method. This section describes the operating procedures and the execution examples of commands to be used. Read this section when you operate the system.
- **Notes**
This section describes the general precautions for using functionality. Make sure that you read the notes.

1.3.2 Examples of tables describing the classification

The following table lists the classification of functionality. The title of the table is "Organization of this chapter" or "Organization of this section".

The following is an example table describing the classification of functionality:

Example table describing the classification of functionality

Table X-1 Organization of this chapter (XX functionality)

Category	Title	Reference
Explanation	What is the XX functionality	X.1
Implementation	Implementation of applications	X.2
	Definitions in DD and <code>cosminexus.xml</code> [#]	X.3
Settings	Settings in the execution environment	X.4
Operations	Operations using the XX functionality	X.5
Notes	Precautions when using the XX functionality	X.6

[#]
For details on `cosminexus.xml`, see *13. Managing Application Attributes* in the *uCosminexus Application Server Common Container Functionality Guide*.

Tip

Property settings for applications that do not contain `cosminexus.xml`

In applications that do not contain `cosminexus.xml`, you set or change properties after importing the properties into the execution environment. You can also change the set properties in the execution environment.

You specify the application settings in the execution environment using the server management commands and property files. For application settings with the server management commands and property files, see *3.5.2 Procedure for setting the properties of a J2EE application* in the *uCosminexus Application Server Application Setup Guide*.

The tags specified in the property files correspond to DD or `cosminexus.xml`. For correspondence of DD or `cosminexus.xml` and the property file tags, see *3. Property Files Used for Setting J2EE Applications* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

Note that the properties specified in each property file can also be specified in the HITACHI Application Integrated Property File.

1.4 System operations by the Component Container Administrator (In UNIX)

In UNIX, a superuser executes the tasks of setting up, starting and stopping the J2EE server or batch server, but even a general user, who is not the superuser, can be granted the permissions that will enable him to execute these tasks. This user is referenced to as a *Component Container administrator*. You can assign any user other than a superuser as the administrator by setting the Component Container administrator.

Set up the Component Container administrator after installing Cosminexus Component Container that is the configuration software of Application Server. After the setup of Component Container administrator is complete, the superuser will not be able to execute the operations that can now be executed by the Component Container administrator, with the exception of a few operations.

The following table describes the operations that the superuser and the Component Container administrator can execute when the Component Container administrator is setup.

Table 1–6: Operations that the superuser and the Component Container administrator can execute

Operations	Superuser	Component Container administrator
Installing Cosminexus Component Container	Y	--
Setting the Component Container administrator (cjenvsetup command)	Y	--
Migrating the working directory and the user definition file (cjenvupdate command)	--	Y
Setting up and removing the setup of the J2EE server (cjsetup command)	--	Y
Setting up a reverse proxy	--	Y
Starting and stopping the J2EE server or batch server (cjstartsv command, cjstopsv command)	--	Y
Executing the server management commands	Y	Y
Acquiring the thread dump (cjdumpsv command)	Y	Y
Acquiring the OS status information (cjgetsysinfo command)	Y	Y
Batch execution command (cjexecjob command)#	Y	Y
Batch forced termination command (cjkilljob command)#	Y	Y
Batch list display command (cjlistjob command)#	Y	Y

Legend:

Y: Available

--: Not available

#

These commands can be used in batch applications.



Reference note

If the Component Container administrator is not setup, the superuser is the administrator, and hence, he executes all the operations of the Component Container administrator mentioned in the table above.

1.5 Main updates in the functionality of Application Server 11-00

This subsection describes the main updates in the functionality of Application Server 11-00 and the purpose of each change.

The contents described in this section are as follows:

- This section gives an overview and describes the main updates in the functionality of Application Server 11-00. For details on the functionality, check the description in the *Reference location* column. The *Reference manual* column and *Reference location* column specify the main sections with the description of a particular functionality.
- *uCosminexus Application Server* is omitted from the manual names mentioned in the *Reference manual* column.

1.5.1 Simplifying implementation and setup

The following table describes the items that are changed to simplify implementation and setup.

Table 1–7: Changes made for simplifying implementation and setup

Item	Overview of changes	Reference manual	Reference location
Windows Server support in the development environment	uCosminexus Developer now supports the Windows Server OS so that an application development environment can be built in a cloud environment.	--	--

1.5.2 Supporting standard and existing functionality

The following table describes the items that are changed to support standard and existing functionality.

Table 1–8: Changes made for supporting standard and existing functionality

Item	Overview of changes	Reference manual	Reference location
Servlet 3.0 and 3.1 support	Asynchronous servlets in Servlet 3.0 and the asynchronous I/O API in Servlet 3.1 are now supported.	<i>Web Container Functionality Guide</i>	7.1
EL 3.0 support	EL 3.0 is now supported.	<i>Web Container Functionality Guide</i>	2.3.3
JSF 2.2 support	JSF 2.2 is now supported.	<i>Web Container Functionality Guide</i>	Chapter 3
JAX-RS 2.0 support	JAX-RS 2.0 is now supported.	<i>Web Container Functionality Guide</i>	Chapter 4
WebSocket 1.0 support	WebSocket 1.0 is now supported.	<i>Web Container Functionality Guide</i>	Chapter 5
Addition of the NIO HTTP server functionality	The NIO HTTP server functionality was added as an in-process HTTP server that supports asynchronous servlets and non-blocking I/O processing such as WebSocket, instead of the conventional redirector and in-process HTTP server functionality.	<i>Web Container Functionality Guide</i>	Chapter 6

Item	Overview of changes	Reference manual	Reference location
JPA 2.1 support	JPA 2.1 is now supported so that a JPA provider supporting JPA 2.1 can be used.	<i>Common Container Functionality Guide</i>	<i>Chapter 5</i>
CDI 1.2 support	CDI 1.2 is now supported.	<i>Common Container Functionality Guide</i>	<i>Chapter 8</i>
BV 1.1 support	Bean Validation 1.1 is now supported.	<i>Common Container Functionality Guide</i>	<i>Chapter 9</i>
Java Batch 1.0 support	Batch Applications for the Java Platform (Java Batch) 1.0 is now supported.	<i>Common Container Functionality Guide</i>	<i>Chapter 10</i>
JSON-P 1.0 support	Java API for JSON Processing (JSON-P) 1.0 is now supported.	<i>Common Container Functionality Guide</i>	<i>Chapter 11</i>
Concurrency Utilities 1.0 support	Concurrency Utilities for Java EE 1.0 is now supported.	<i>Common Container Functionality Guide</i>	<i>Chapter 12</i>
WebSocket communication support	A function that relays WebSocket communication from an HTTP server to a J2EE server was added.	<i>HTTP Server</i>	<i>4.15</i>

1.5.3 Maintaining and improving reliability

The following table describes the items that are changed for maintaining and improving reliability.

Table 1–9: Changes made for maintaining and improving reliability

Item	Overview of changes	Reference manual	Reference location
Change of the encrypted-communication module	The <code>mod_ssl</code> module was adopted as an encrypted-communication module for an HTTP server.	<i>HTTP Server</i>	<i>Chapter 5, Appendix H</i>

1.5.4 Other purpose

The following table describes the items that are changed for other purposes.

Table 1–10: Changes made for other purposes

Item	Overview of changes	Reference manual	Reference location
Addition of V9 compatibility mode	V9 compatibility mode was added to maintain the compatibility with version 9 of Application Server for users of Application Server in which the J2EE server has been upgraded from version 9 or earlier.	<i>Maintenance and Migration Guide</i>	<i>10.3.3</i>

2

Starting and Stopping the System

This chapter describes the normal operations that are executed such as starting and stopping the system and automatically restarting the system when a failure occurs.

2.1 Organization of this chapter

This chapter describes the daily operations such as starting and stopping the systems, mechanism of automatic restart when a failure occurs, and monitoring of the system operations.

This chapter is organized as the following table.

Table 2–1: Organization of this chapter (starting and stopping a system)

Category	Title	Reference
Description	Starting and stopping the system during daily operations	2.2
	Mechanism of starting and stopping the logical server	2.3
	Automatically restart when a failure occurs	2.4
	Monitoring system operations	2.5
Settings	Settings for starting and stopping the system	2.6

Note:

The function-specific explanation is not available for "Implementation", "Operations", and "Precautions".

2.2 Starting and Stopping the System during Daily Operations

To operate a system built with the application server, you need to start multiple server processes of the application server configuration software in an appropriate order. You may even start multiple server processes of the same type depending on the system configuration. Further, to start services, you also need to start J2EE applications.

In the systems on the Application Server systems, a closed partial system that provides a business service configured by J2EE server and Web server is managed as a *service unit*. Using the Smart Composer functionality commands you can start and stop the logical servers for each service unit, or you can start and stop all the service units in a batch. You can also collectively start and stop all the service units in the Web system.

As the Smart Composer functionality uses Management Server for management, start the system in the order of the steps described below. To stop the system, reverse the order of the steps. For more information on how to start and stop the logical server, see [2.3 Mechanism of Starting and Stopping the Logical Server](#). Furthermore, for details on the settings to start and stop the systems, see [2.6 Settings for starting and stopping the system](#).

1. Start the Administration Agent.
2. Start the Management Server.
3. Start the service unit (logical server) (Smart Composer functionality commands)
4. Start the resource adapters (server management commands)
5. Start the J2EE applications (server management commands)

For starting and stopping the systems using the Smart Composer functionality, see the following subsections in the *uCosminexus Application Server System Setup and Operation Guide*.

- [4.1.24 Start the system \(when using CUI\)](#)
- [4.1.34 Stop the system \(when using CUI\)](#)

2.3 Mechanism of Starting and Stopping the Logical Server

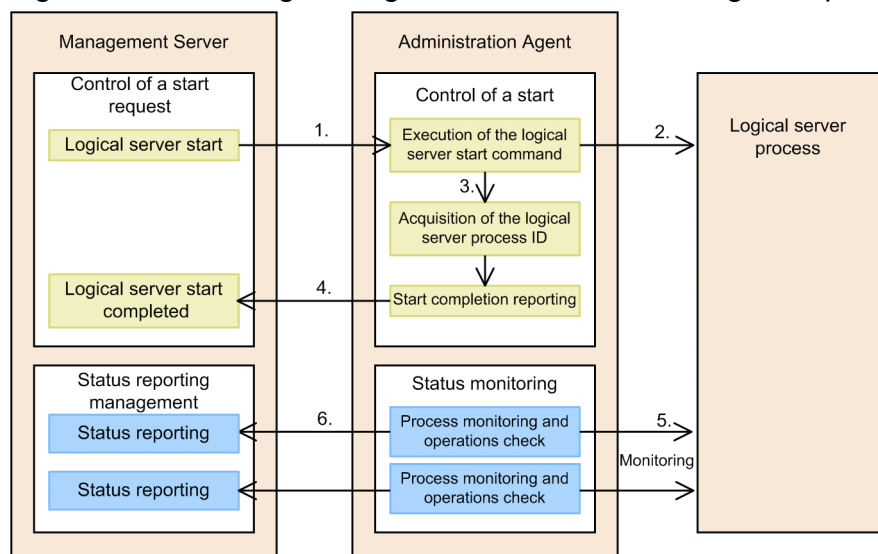
On the Application Server, you use the Management Server to start and stop logical servers and to check operations.

This subsection describes the starting, operation checking mechanism, and the stop mechanism of the logical server using the Management Server.

2.3.1 Starting the Logical Server and Checking the Operations

The following figure illustrates starting the logical server and checking the operations:

Figure 2–1: Starting the logical server and checking the operations



1. The Management Server sends a request to the Administration Agent to start the logical server.
2. The administration agent executes the start command of the logical server to start the requested logical server.
3. The process ID of the logical server process is acquired in the Administration Agent.
4. The administration agent notifies the Management Server that the logical server is now running.
5. The administration agent monitors the logical server process and checks the operation of the logical server.
The administration agent uses the process ID of the logical server process, checks if the process exists, and if it is able to verify the existence of the process, it checks the operation of the logical server.
6. The administration agent notifies the status of the logical server to the Management Server.

The following table describes the method of checking the process and the operation for each type of logical server. Note that for the logical servers in case of which the operation checking method is not mentioned in the table, only the check for verifying whether the process exists is carried out.

Table 2–2: Methods for checking the operations of logical servers

Type of logical server		Method of starting the process [#]	Method of checking the operation of the logical server	
			Check that the process exists	Check the operation
Logical performance tracer		Indirect startup	Check that the process exists by using the process ID obtained by using a command provided by the performance tracer.	--
Logical Smart Agent		Direct startup	Check that the process exists by using the process ID of the command that started the process.	--
Logical Naming Service		Direct startup	Check that the process exists by using the process ID of the command that started the process.	Check that the root context can be obtained.
Logical CTM domain manager		Indirect startup	Check that the process exists by using the process ID obtained by using a command provided by the CTM domain manager.	--
Logical CTM	Naming service	Direct startup	Check that the process exists by using the process ID of the command that started the process.	Check that the root context can be obtained.
	CTM daemon	Indirect startup	Check that the process exists by using the process ID obtained by using a command provided by the CTM daemon.	--
Logical J2EE server		Direct startup	Check that the process exists by using the process ID of the command that started the process.	Check that there is a response to the call by RMI.
Logical Web server		Indirect startup	Check that the process exists by using the process ID of the control process obtained from the <code>httpd.pid</code> file generated by the web server.	Check that a correct response can be received through the HTTP access of the URL for checking the Cosminexus HTTP Server operations. The response is sent by the server process of the Web server.
Logical user server		Direct startup	Check that the process exists by using the process ID of the command that started the process.	When the <code>isAlive</code> command is defined in the logical user server definition file, check that the processing is executed by using the <code>isAlive</code> command. When not defined, it is assumed that the operation check was successful.
		Indirect startup	Check that the process exists by using the process ID obtained by using the <code>getProcessID</code> command.	Use the <code>isAlive</code> command when it is defined in the logical user server definition file. When not defined, it is assumed that the operation check was successful.

Legend:

--: None

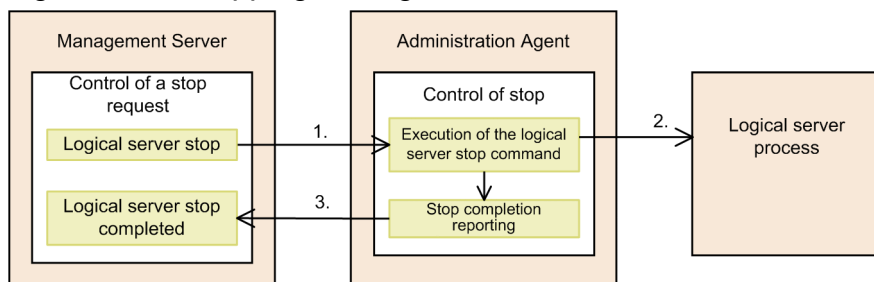
[#]

In the case of direct setup, the command that is executed is also monitored. In the case of indirect setup, the services or the processes invoked by the command are monitored. The command is also used to stop the services and processes that are invoked using other commands.

2.3.2 Stopping the Logical Server

The following figure shows the process of stopping the logical server:

Figure 2–2: Stopping the logical server



1. The Management Server sends a request to the Administration Agent server to stop the logical server.
2. The Administration Agent executes the stop command of the logical server to stop the requested logical server.
3. The Administration Agent notifies the Management Server that the logical server is now stopped.

2.4 Automatically Restart When a Failure Occurs

If a failure occurs, Management Server, Administration Agent, and logical server can restart automatically. This section describes the automatic restart operation of each process.

2.4.1 Automatic restart of Management Server

This subsection describes the automatic restart of Management Server, when a failure occurs.

When a failure occurs, a stopped Management Server can restart automatically. If the processes of Management Server are down due to a failure, Management Server is automatically started, and therefore, operations can be continued. Also, if you specify automatic restart, Management Server also automatically starts simultaneously with the host. You use the `mngautorun` command to specify automatic restart. For details on how to specify the automatic restart, see [2.6.4 Settings for automatic restart](#).

Important note

When Management Server is stopped, if the operating logical server process detects an abnormal termination or a hang-up, and after that even if Management Server is restarted, even then the automatic stop processing and automatic restart processing are not executed.

2.4.2 Automatic restart of Administration Agent

This subsection describes the automatic restart of Administration Agent, when a failure occurs.

When a failure occurs, a stopped Administration Agent can restart automatically. If the processes of Administration Agent are down due to a failure, Administration Agent is automatically restarted, and operations can be continued. Also, if you specify automatic restart, Administration Agent also automatically starts simultaneously with the host. You use the `mngautorun` command to specify automatic restart. For details on how to specify the automatic restart, see [2.6.4 Settings for automatic restart](#).

2.4.3 Automatic restart of logical server

This subsection describes the automatic restart of logical servers, when a failure occurs.

When a failure occurs, a stopped logical server can restart automatically using the Management Server.

Automatic restart is executed when the status after startup is detected as 'failure', from the logical server that was started normally by the Management Server. 'Failure' is a state when the logical server stops without receiving any stop request.

Administration Agent monitors the processes of the logical server and checks the operations of the logical server. When a logical server failure, such as a process is down or has hung up is found, Administration Agent detects the failure and notifies Management Server. When a failure is detected, the Management Server executes the commands and collects the snapshot logs, after that the Management Server collects the data used for troubleshooting, and then the logical server restarts automatically.

When a failure occurs during the startup process requested by the user, instead of automatically restarting the logical server, the user is notified of the failure to start the logical server.

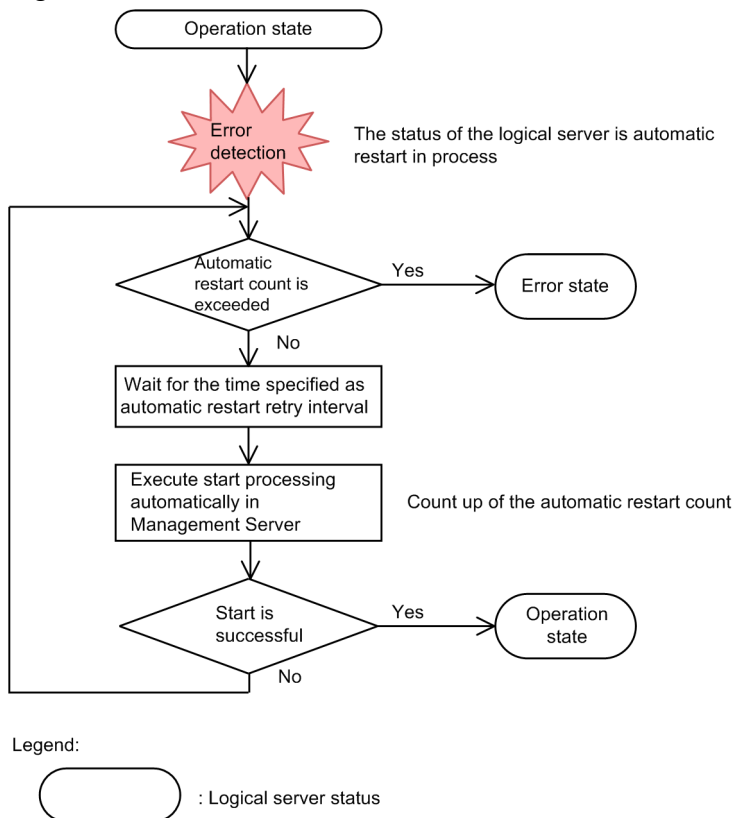
Automatic restart is executed according to the automatic restart frequency and automatic restart retry interval that is specified when building the system. Note that automatic restart is not executed when the specified automatic restart frequency is '0'.

When automatic restart exceeds the specified frequency, a message is output to *Manager-log-output-directory/mngsvr-number-of-files.log* and the logical server status changes to 'failure'.

If `true` is specified in the `com.cosminexus.mngsvr.logical_server_abnormal_stop.exit` key of the `mserver.properties` file (Management Server environment configuration file), Management Server stops when the logical server status changes to 'abnormal termination'. 'Abnormal termination' is a state wherein the logical server stops when the automatic restart frequency is exceeded or when 0 is specified as the automatic restart frequency and a failure is detected.

The following figure shows the flow of automatic restart when a failure occurs and the status of the logical server:

Figure 2–3: Automatic restart when failure occurs



Note that when an automatic restart is executed in the requisite logical server, then after the requisite logical server has automatically restarted, the logical server for which the corresponding logical server is specified as a pre-requisite is also restarted.



Tip

Detection of failure

Check the status of the logical server by monitoring the process of the logical server, and checking the operation of the logical server in the Administration Agent. For example, process monitoring checks the existence of the process ID of the logical server process. If the process ID does not exist, the Administration Agent detects that the process is down and notifies the error to the Management Server.

The contents of process monitoring and checking the operations differ depending upon the type of the logical server. For details, see [2.3 Mechanism of Starting and Stopping the Logical Server](#).

Important note

If the logical server process ends immediately after successfully starting the logical server, the error detection and automatic restart might be executed again. In such cases, you can avoid this process by setting a value more than the time from when the process of the logical server starts until the process ends in the `adminagent.server-type.watch.start_time` key (the time from executing the start command of the logical server until the starting of operation confirmation) of `adminagent.properties`.

When you set the J2EE server start monitoring time for a logical J2EE server, consider the time required for the J2EE application to start. If the J2EE application takes too long to start up during an automatic restart, the automatic restart might fail. Even if the settings are specified so that the J2EE application starts after the J2EE server has started, if the J2EE application starts before an automatic restart, the J2EE application and J2EE server are simultaneously restarted when the automatic restart occurs. Note, however, that this simultaneous restart does not occur during an automatic restart if `-nostartapp` is specified for **Add command options**[#] in the logical J2EE server environment settings.

[#]: If you specify settings from Smart Composer, use the `additional.startcmd` parameter.

2.5 Monitoring System Operations

This subsection describes the status monitoring of the logical servers and the statistics monitoring of systems that are required during the system operations.

2.5.1 Monitoring the system operations

You need to maintain stable operating conditions by checking that the status of a logical server and the statistics of a running system are appropriate. You also need to determine whether tuning is required based on the system performance output in the statistics.

This subsection gives an overview of operations that monitor the system operations. The following table describes the monitoring of system operation:

Table 2–3: Monitoring system operations

Operation contents	Measure	Operation overview	Reference manual	Section
Monitoring the status of service units	Smart Composer functionality commands (cmx_list_status command)	You can check the status of service units in the Web system and can monitor that the system is running normally.	This manual	2.5.2(1)
Monitoring the status of a logical server	Management command (mngsvrutil)	<p>You can check the status of hosts and servers that configure the system and monitor whether the system is running normally. Based on this information, you can start, stop, and restart the system, as required.</p> <p>You can confirm the status of a logical server in the following units:</p> <ul style="list-style-type: none">• Management domain unit• Host unit• Logical server unit		2.5.2(2)
Monitoring system operations	Statistics collection function ^{#1 #2}	<p>The operational statuses of the monitored J2EE server or batch server can be output to statistics files.</p> <p>By monitoring the system performance, you can find the locations in the system where errors occur and this can help for performance tuning.</p>		3.2
	Management command (mngsvrutil)	<p>The operational statuses of the monitored J2EE server or batch server can be output to a CSV file or an SNMP integration format file.</p> <p>By monitoring the system performance, you can find the locations in the system where errors occur and this can help for performance tuning.</p>		8.2
	Function for monitoring resource depletion ^{#1 #3}	By monitoring the transition of usage rate and the number of resources, the system can output a message when the usage rate or the number of used resources exceeds the threshold value. The system can periodically output the		4.2

Operation contents	Measure	Operation overview	Reference manual	Section
		transition of usage rate and number of used resources as the resource depletion monitoring information. When the threshold value is exceeded, you can prevent the occurrence of an error by analyzing the cause based on the output resource depletion monitoring information and take an appropriate action. Further, the resource depletion monitoring information is useful in analyzing the cause of an error.		
	Management command (mngsvrutil)	You can check the statistics for CTM ^{#4} that exist in the management domain for the following units: <ul style="list-style-type: none"> All CTM of the management domain Specified CTM from the management domain or the CTM that exists on a specified host 		<i>Chapter 10</i>
Checking the transaction information	Server management commands (cjlisttrn)	You check information, such as the transaction status on running J2EE servers or the existence of a pending transaction on the stopped J2EE servers.	<i>Common Container Functionality Guide</i>	3.15.9

#1

Since the file is output at a regular interval that is set when building a system, there is no need to use tools such as server management commands and the management command to output the file.

#2

The number of Full GC occurrences and the number of pending requests for each URL group are output to statistics files. When either of these numbers exceeds their thresholds, a message is output (threshold event). Using this message, you can issue a *management event* and automate the corresponding action as the *management action*. For details on the threshold event settings, see [3.4.4 Settings for the execution environment \(J2EE server settings\)](#). For details on the settings for the automatic execution of processes using Management events, see [9.4 Settings for automatic execution of processes by management events](#).

#3

When the resource usage status exceeds the threshold, a *management event* can be issued. If a management event is used, the action can be automated as a *management action*. For details on how to set up the automatic execution of processes with Management events, see the section [9.4 Settings for automatic execution of processes by management events](#).

#4

You can use the CTM only in the products in which Cosminexus Component Transaction Monitor is included in the component software. For products that you can use, see [2.2.1 Relationship of products and component software](#) in the manual *uCosminexus Application Server Overview*.

2.5.2 Monitoring the status

In Application Server, you can monitor that the operations of the system are functioning normally by monitoring the status. The status can be monitored in the following two ways:

- Monitoring the status of the service unit
You can monitor the status of the service unit.
- Monitoring the status of the logical server
You can monitor the status of a logical server.

Reference note

Server management commands can be used to monitor the status of the application that is running on the server and display the transaction information of the running J2EE server or batch server.

(1) Monitoring the status of service units

You can use the Smart Composer functionality command (`cmx_list_status` command) to monitor the operational status of service units. For using the `cmx_list_status` commands, see *cmx_list_status (Display a status of service unit)* in the *uCosminexus Application Server Command Reference Guide*. In the case of monitoring the status of the service unit, you can output the status of a specific service unit or the status of all the service units in the Web system, to a file in CSV format.

(2) Monitoring the status of a logical server

In the case of monitoring the status of the logical server, you can use the management commands (`mngsvrutil`) to monitor the status of the logical server. The status information can be output to a standard output, or a CSV format file.

This section describes how to monitor the statistics (status) of a logical server, and the items that you can monitor.

Reference note

You can use the following methods to check the status of a Management Server and Administration Agent:

- **check subcommand of the management command** (`mngsvrutil`)
- **adminagentcheck command**

For details on the management commands and the `adminagentcheck` commands, see *mngsvrutil (Management Server management command)* and *adminagentcheck (check Administration Agent activation)* in the *uCosminexus Application Server Command Reference Guide*.

(a) How to monitor the status

You can use the management command (`mngsvrutil`) to monitor the status of a logical server.

To monitor the status of logical servers, specify the subcommand `list` in the management command (`mngsvrutil`). As a result, you can output the status information of the logical server in the standard output, or a CSV format or an SNMP integration format file.

You can specify a logical server for which you want to output the status by specifying the value in arguments of the list.

For details on the management commands and there subcommands, see *7.3 Details of subcommands of the mngsvrutil command* in the *uCosminexus Application Server Command Reference Guide*.

The execution format and example of execution are described below:

- **When the logical server name and status information in a management domain are output**

Execution format

```
mngsvrutil -m host-name-of-the-Management-Server[:port-number] -u management-user-ID -p management-password list status
```

Execution example

```
mngsvrutil -m mnghost -u user01 -p pw1 list status
```

- **When the J2EE application name and status information imported in a specified logical server are output**

Execution format

```
mngsvrutil -m host-name-of-the-management-server[:port-number] -u management-user-ID -p Management-password -t logical-server-name list appStatus
```

Execution example

```
mngsvrutil -m mnghost -u user01 -p pw1 -t J2Eeserver1 list appStatus
```

(b) Items that you can check by monitoring the status

You can confirm the start and stop status of a logical server by monitoring the status. You can also confirm the status of all logical servers.

You can confirm the status for each of the following:

- **Management domain unit**

You can confirm the status of all logical servers included in a management domain at the same time.

- **Host unit**

You can confirm the status of all logical servers in a selected host.

- **Logical server unit**

You can check the status of selected logical servers. You can check the status for each type and for individual logical server.

The following table describes types and meanings of a status that you can confirm:

Table 2–4: Types and meaning of operational status

Operational status	Description
Stop	An initial state of a logical server or a state in which a logical server is terminated after reception of a termination request, and completing the termination process. In this state, it is confirmed that the management agent is terminated after restoring the communication failure.
Start	A state where a logical server is being started, after reception of a start request until the logical server is operational.
Run	A state where a logical server is operational, after reception of a start request and completing the start process. Alternatively, this is a state in which it is confirmed that the management agent is operational after restoring the communication failure.
Termination in-process	A state where a logical server is being terminated, after reception of a termination request until the logical server is terminated.
Forced termination in-process	A state where a logical server is being terminated forcefully, after reception of a forced termination request until the logical server is terminated.
Abnormal termination	A state where it is detected that a logical server is terminated without reception of a termination request.
Recovery in-process	A state where a logical server is being started, after reception of a start request in the abnormal termination state until the logical server is operational.
Communication failure	A state where there is a communication failure with a management agent, and the status cannot be displayed.
Automatic termination in-process	A state where a logical server is being terminated forcefully, after reception of a notification of abnormality (notification that a process exists but is not running) of the logical server from a management agent.

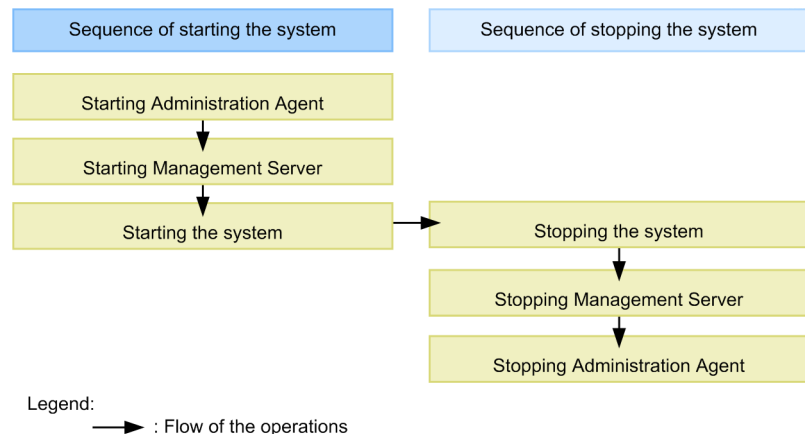
Operational status	Description
Automatic restart in-process	A state in which a running logical server with auto restart specifications, is being automatically restarted on reception of a notification for termination from a management agent.
Planned termination in-process	A planned termination state of a logical Web server until the actual termination, after receipt of a planned termination request. Only a logical Web server goes into this state.

2.6 Settings for starting and stopping the system

This subsection describes the procedure for starting and stopping a built system, and the settings to be specified for automatically starting and stopping the system.

The following figure shows the procedure for starting and stopping of the programs and servers in a built system:

Figure 2–4: Procedure for starting and stopping systems



You can start and stop the programs and servers in a Web system automatically or manually. Determine the starting and stopping method of the system in conformity with the operating method, and specify the required settings.

2.6.1 Starting the system

For the normal operations, we recommend that you use the automatic start method for starting Administration Agent and Management Server. If you specify automatic restart of Administration Agent and Management Server, the series of operations from the starting of Administration Agent up to the starting of Management Server can be executed simultaneously with the starting of the host. For details on how to specify the automatic start, see [2.6.3 Settings for automatic start](#).

The following are the start procedures for programs and servers:

1. Start Administration Agent
2. Start Management Server
3. Start the system

To start programs and servers:

(1) Start Administration Agent

If you start Administration Agent manually, you start it by using the `adminagentctl` command for each host of Application Server.

We recommend that you synchronize the state of the command processing and start processing of Administration Agent. In such cases, you specify the synchronous execution option (`-sync` option) when executing the command. If you specify this option, when the command processing ends, Administration Agent becomes in the operational state. Also, during the synchronous execution, you can specify the waiting time from when the command is

executed until Administration Agent is in the operational state (the timeout time of synchronous execution), with the `-timeout` option.

The start method is shown for each OS.

In Windows

```
Installation-directory-of-Cosminexus(1)\manager\bin\adminagentctl start -sync -timeout timeout-time-of-synchronous-execution
```

Reference note

Note that in Windows, you can also use services to start Administration Agent.

In UNIX

```
# /opt/Cosminexus/manager/bin/adminagentctl start -sync -timeout timeout-time-of-synchronous-execution
```

Reference note

Note that in UNIX, if you specify the `-daemon` option with the `adminagentctl` command or if you use the `daemon` command, you can start Administration Agent as the daemon process. When you use the `daemon` command, the state of the command processing and start processing of Administration Agent cannot be synchronized.

(2) Start Management Server

If you start Management Server manually, you start it by specifying the `start` argument in the `mngsvrctl` command on the host on which Management Server is set up.

Important note

When you define a host by specifying a host name that cannot be resolved or an IP address that does not exist, the starting of Management Server takes time.

We recommend that you synchronize the state of the command processing and start processing of Management Server. In such cases, you specify the synchronous execution option (`-sync` option) when executing the command. If you specify this option, when the command processing ends Management Server is in the operational state. Also, during the synchronous execution, you can specify the waiting time from when the command is executed until Management Server is in the operational state (the timeout time of synchronous execution), with the `-timeout` option.

The start method is shown for each OS.

In Windows

```
Installation-directory-of-Cosminexus\manager\bin\mngsvrctl start -sync -timeout timeout-time-of-synchronous-execution
```

Reference note

Note that in Windows, you can also use services to start Management Server.

In UNIX

```
# /opt/Cosminexus/manager/bin/mngsvrctl start -sync -timeout timeout-time
-of-synchronous-execution
```

Reference note

Note that in UNIX, if you specify the `-daemon` option using the `mngsvrctl` command or if you use the `daemon` command, you can start Management Server as the daemon process. When you use the `daemon` command, the state of the command processing and start processing of Management Server cannot be synchronized.

(3) Start the system

You can start the Web system or service units in a batch using the Smart Composer functionality commands. If the system is started in each Web system, all the logical servers in all the service units that exist in the Web system will start.

Reference note

We recommend that you use the default settings for the starting order of logical servers. You can specify the starting order of logical servers with the `mstartup.no` parameter of the `<configuration>` tag for each logical server, in the Easy Setup definition file.

2.6.2 Stopping the system

The following are the stop procedures and the stop methods for the programs and servers. Note that in UNIX, you cannot specify automatic stop for Administration Agent and Management Server. For the automatic stop, specify the setting in such a way so that Administration Agent and Management Server are stopped simultaneously with the host. For details about the automatic stop settings, see [2.6.5 Settings for automatic stop](#).

The following are the stop procedures and the stop methods for the programs and servers. You use the following procedure to stop programs and servers:

1. Stop the system
2. Stop Management Server
3. Stop Administration Agent

To stop programs and servers:

(1) Stop the system

Use the Smart Composer functionality commands to stop Web systems or service units in a batch.

(2) Stop Management Server

If you stop Management Server manually, you stop it by specifying the `stop` argument in the `mngsvrctl` command on the host on which Management Server is set up.

We recommend that you synchronize the state of the command processing and stop processing of Management Server. In such cases, you specify the synchronous execution option (`-sync` option) when executing the command. If you specify this option, when the command processing ends, Management Server is in the stopped state. Also, during the synchronous execution, you can specify the waiting time from when the command is executed until Management Server is in the stopped state (the timeout time of synchronous execution), with the `-timeout` option.

The stop method is shown for each OS.

In Windows

```
Installation-directory-of-Cosminexus(1)\manager\bin\mngsvrctl stop -sync  
-timeout timeout-time-of-synchronous-execution
```

Reference note

Note that in Windows, you can also use services to stop Management Server.

In UNIX

```
# /opt/Cosminexus/manager/bin/mngsvrctl stop -sync -timeout the timeout-ti  
me-of-synchronous-execution
```

(3) Stop Administration Agent

If you stop Administration Agent manually, you stop it by using the `adminagentctl` command on each host of Application Server.

We recommend that you synchronize the state of the command processing and stop processing of Administration Agent. In such cases, you specify the synchronous execution option (`-sync` option) when executing the command. If you specify this option, when the command processing ends, Administration Agent is in the stopped state. Also, during the synchronous execution, you can specify the waiting time from when the command is executed until Administration Agent is in the stopped state (the timeout time of synchronous execution), with the `-timeout` option.

The stop method is shown for each OS.

In Windows

```
Installation-directory-of-Cosminexus\manager\bin\adminagentctl stop -sync  
-timeout timeout-time-of-synchronous-execution
```

Reference note

Note that in Windows, you can also use services to stop Administration Agent.

In UNIX

```
# /opt/Cosminexus/manager/bin/adminagentctl stop -sync -timeout timeout-ti  
me-of-synchronous-execution
```

2.6.3 Settings for automatic start

Management Server and Administration Agent can automatically start simultaneously with the host. You use the `mngautorun` command to specify the automatic start of Management Server and Administration Agent. For details on the `mngautorun` command, see *mngautorun (Set up/canceling the set up of autostart and autorestart)* in the *uCosminexus Application Server Command Reference Guide*.

We recommend that you make the automatic start processing of Management Server and Administration Agent that is set up with the `mngautorun` command to execute synchronously. In such cases, you specify the synchronous execution option (`-sync` option) when executing the command. If you specify this option, during the automatic start the starting of Management Server and Administration Agent is executed synchronously. Also, during the automatic start, you can specify the waiting time until the starting of Management Server and Administration Agent is executed synchronously (the timeout time of synchronous execution), with the `-timeout` option.

Important note

If you are executing operations by specifying the automatic start settings of a version earlier than 09-00, when enabling the settings described in this subsection, you are required to cancel the automatic start settings of the version earlier than 09-00. For details on how to change the settings, see *Chapter 10* in the *uCosminexus Application Server Maintenance and Migration Guide*.

In UNIX, if you are executing operations by specifying the automatic start settings of a version earlier than 09-00, you cannot enable the synchronous execution. When canceling the automatic start settings of the version earlier than 09-00 and specifying the automatic start settings by using the `mngautorun` command, enable the synchronous execution.

The execution format and the execution example of the `mngautorun` command are as follows:

Execution format

```
mngautorun [once] {server|agent|both} [-e run-level(1)]# [-sync [-timeout timeout-time-of-synchronous-execution]]
```

#: The run level can be specified in UNIX. In Windows, you cannot specify *run-level*.

Execution example

- If Management Server automatically starts, when the host starts
`mngautorun once server`
- If Administration Agent automatically starts, when the host starts
`mngautorun once agent`
- If Management Server and Administration Agent automatically start, when the host starts
`mngautorun once both`
- If Management Server and Administration Agent automatically start with synchronous execution, when the host starts
`mngautorun once both -sync`

The following subsections describe the contents of automatic start, set up in Management Server and Administration Agent, when executing the `mngautorun` command.

(1) Contents of automatic start specified in Management Server

The contents of automatic start specified in Management Server when executing the `mngautorun` command are described here for each OS.

- In Windows

In the service of Management Server, the following contents are specified such that the service starts when the host starts:

- Type of startup: Automatic

Also, if you start both Management Server and Administration Agent on the same host, a dependency relationship is set up such that Administration Agent starts first.

- In Unix

The settings[#] that automatically start Management Server when the OS starts or restarts can be specified by using the `mngautorun` command with the `once` option specified (or with both the `once` and `respawn` options omitted).

[#]: In the `/etc/inittab` file, specify the Management Server start command with the `once` option specified. In Linux, register `CoMS.service` (service that automatically starts or stops Management Server) in a subdirectory of the `/etc/systemd/system` directory.

(2) Contents of automatic start setup in Administration Agent

The contents of automatic start specified in Administration Agent when executing the `mngautorun` command are described here for each OS.

Reference note

With the automatic start process, before starting Administration Agent, the Administration Agent automatic start configuration file (`/opt/Cosminexus/manager/config/AdminAgentrc`) is read. You can code the following settings in this file, so that the logical server which starts from Administration Agent inherits the settings:

- Control settings of resources (`ulimit`)
- Permissions at the time of file creation (`umask`)
- Environment variables

Do not code the settings other than the settings mentioned in this subsection, in the Administration Agent automatic start configuration file. If you code other settings in the Administration Agent automatic start configuration file, operations are not guaranteed.

The Administration Agent automatic start configuration file is executed as Shell Script. Therefore, if you change the contents coded in the Administration Agent automatic start configuration file, adequately check the operations.

- In Windows

In the Administration Agent service, the following contents are set up to start the service, when the host starts:

- Type of startup: Automatic

Also, if you start both Management Server and Administration Agent on the same host, a dependency relationship is set up such that Administration Agent starts first.

- In UNIX

The settings[#] that automatically start Administration Agent when the OS starts or restarts can be specified by using the `mngautorun` command with the `once` option specified (or with both the `once` and `respawn` options omitted).

#: In the `/etc/inittab` file, specify the Administration Agent start command with the `once` option specified. In Linux, register `CoAA.service` (service that automatically starts or stops Administration Agent) in a subdirectory of the `/etc/systemd/system` directory.

2.6.4 Settings for automatic restart

When a failure occurs, the stopped Management Server and Administration Agent can restart automatically. If you set up automatic restart, the settings for automatic start are also valid. You use the `mngautorun` command to specify the automatic restart settings of Management Server and Administration Agent. For details on the `mngautorun` command, see *mngautorun (Set up/canceling the set up of autostart and autorestart)* in the *uCosminexus Application Server Command Reference Guide*.

We recommend that in Windows, you make the automatic restart processing of Management Server and Administration Agent set up with the `mngautorun` command to execute synchronously. In such cases, you specify the synchronous execution option (`-sync` option) when executing the command. If you specify this option, during the automatic restart, the starting of Management Server and Administration Agent is executed synchronously. Also, during the automatic restart, you can specify the waiting time until the starting of Management Server and Administration Agent is executed synchronously (the timeout time of synchronous execution), with the `-timeout` option.

In UNIX, the synchronous execution option cannot be specified for the automatic restart processing.

Important note

When executing operations by specifying the automatic start settings of a version earlier than 09-00

If you are executing operations by specifying the automatic start settings of a version earlier than 09-00, when enabling the settings described in this subsection, you are required to cancel the automatic start settings of the version earlier than 09-00. For details on how to change the settings, see *Chapter 10* in the *uCosminexus Application Server Maintenance and Migration Guide*.

Settings for automatic restart when JavaVM is terminated abnormally

We recommend that when JavaVM terminates abnormally, after you acquire the user dump, for the correct operations of automatic restart of Application Server, you specify the registry value (`DontShowUI : 1`) in one of the following registry keys, and control response requests when acquiring the user dump:

\\HKEY_LOCAL_MACHINE\\SOFTWARE\\Microsoft\\Windows\\Windows
Error Reporting

\\HKEY_CURRENT_USER\\SOFTWARE\\Microsoft\\Windows\\Windows Error Reporting
(to be used when the above mentioned registry key is not present)

The execution format and execution example of the `mngautorun` command are as follows:

Execution format

```
mngautorun respawn {server|agent|both} [-runlevel run-level]#1 [-sync#2 [-t  
imeout timeout-time-of-synchronous-execution#2]]
```

#1: The run level can be specified in UNIX. In Windows, the run level cannot be specified.

#2: The synchronous execution option and the timeout time of the synchronous execution can be specified in Windows. In UNIX, synchronous execution option and the timeout time of the synchronous execution cannot be specified.

Example of execution

- When restarting Management Server automatically
`mngautorun respawn server`
- When restarting Administration Agent automatically
`mngautorun respawn agent`
- When restarting Management Server and Administration Agent automatically
`mngautorun respawn both`
- When synchronously executing the automatic restart of Management Server and Administration Agent[#]
`mngautorun respawn both -sync`

#: The synchronous execution of the automatic restart can be specified in Windows. In UNIX, the synchronous execution of the automatic restart cannot be specified.

The following subsection describes the contents of an automatic restart, set up in Management Server and Administration Agent, when executing the `mngautorun` command:

(1) Contents of automatic restart setup in Management Server

This subsection describes the OS-wise settings specified in Management Server for automatic restart, when executing the `mngautorun` command.

- In Windows
The following contents are set up in the recovery functionality of a service so that if an error occurs, the service of Management Server restarts:
 - Initial error: Restart the service
 - Reset of an error count: 0 (error count is not reset)
 - Restart of the service: 0 (restart is executed immediately after an abnormal termination)

Note that you can manually change the settings for the values of *reset of an error count* and *restart of the service*.

- In UNIX
The settings[#] that automatically restart Management Server when the OS starts or restarts can be specified by using the `mngautorun` command with the `respawn` option specified.

If these settings are specified, the Management Server service is monitored and, when the service terminates, it is restarted.

#: In the `/etc/inittab` file, specify the Management Server start command with the `respawn` option specified. In Linux, register `CoMS.service` (service that automatically starts, stops, or restarts Management Server) in a subdirectory of the `/etc/systemd/system` directory.

Important note

In UNIX, irrespective of whether any failure occurs in the services of Management Server, if a service of Management Server is stopped by using the `mngsvrctl` command, the service is restarted.

(2) Contents of automatic restart setup in Administration Agent

This subsection describes the OS-wise settings specified in Administration Agent for automatic restart, when executing the `mngautorun` command.

Reference note

With the automatic restart process, the Administration Agent automatic start configuration file (`/opt/Cosminexus/manager/config/AdminAgentrc`) is read before starting Administration Agent. You can code the following settings in this file, so that the logical server which starts from Administration Agent inherits the settings:

- Control settings of resources (`ulimit`)
- Permissions at the time of file creation (`umask`)
- Environment variables

Do not code settings other than the settings mentioned in this subsection, in the Administration Agent automatic start configuration file. If you code other settings in the Administration Agent automatic start configuration file, operations are not guaranteed.

The Administration Agent automatic start configuration file is executed as Shell Script. Therefore, if you change the contents coded in the Administration Agent automatic start configuration file, adequately check the operations.

- In Windows

The following contents are set up in the recovery functionality of a service so that if an error occurs, the service of Administration Agent restarts:

- Initial error: Restart the service
- Reset of an error count: 0 (error count is not reset)
- Restart of the service: 0 (restart is executed immediately after an abnormal termination)

Note that you can manually change the settings for the values of *reset of an error count* and *restart of the service*.

Important note

After the `adminagent` process is down and then restarts, even if you use a service or a command to stop Administration Agent, a logical server is not stopped, because the following key is not valid:

- `adminagent.finalization.stop_servers` key of *Installation-directory-of-Cosminexus\manager\config\adminagent.properties*

- In UNIX

The settings[#] that automatically restart Administration Agent when the OS starts or restarts can be specified by using the `mngautorun` command with the `respawn` option specified.

If these settings are specified, the Administration Agent service is monitored and, when the service terminates, it is restarted.

#: In the `/etc/inittab` file, specify the Administration Agent start command with the `respawn` option specified. In Linux, register `CoAA.service` (service that automatically starts, or stops, or restarts Administration Agent) in a subdirectory of the `/etc/systemd/system` directory.

Important note

In UNIX, irrespective of whether any failure occurs in the services of Administration Agent, if a service of Administration Agent is stopped by using the `adminagentctl` command, the service is restarted.

2.6.5 Settings for automatic stop

In UNIX, Management Server and Administration Agent can be stopped automatically when the host stops.

(1) Settings for automatically stopping Management Server

The procedures for specifying the settings for automatically stopping Management Server are described here for each OS.

Important note

If you specify settings to automatically restart Management Server, Management Server cannot be stopped by these procedures. Stop the host after stopping each logical server. However, in Red Hat Enterprise Linux Server 7 or later, even if the settings are specified so that the Management Server automatically restarts, you can use these procedures to stop the Management Server.

(a) Stopping procedures for AIX

The following actions are required to make Management Server stop automatically when the host stops:

1. Create a script file to make Management Server stop.
2. Add the processing to make Management Server stop in the `/etc/rc.shutdown` script.

The following are the methods to create a script file for stopping Management Server, and the methods of adding the stop processing of Management Server to the `/etc/rc.shutdown` script:

Creating a script file to make Management Server stop

Create a script file for making Management Server stop with any file name under `/etc/` (example: `/etc/MngSvrStop`). Furthermore, set up the authority of the script file to 755.

The following is an example of a script file:

```
#!/bin/sh
BIN_PATH=/opt/Cosminexus/manager/bin
# Stopping Management Server
if [ -x $BIN_PATH/mngsvrctl ] ; then
    $BIN_PATH/mngsvrctl stop
fi
exit 0
```

In this example, even if the stop processing fails, the shutdown processing is not interrupted because of the last `exit 0`. If the shutdown is interrupted when an error occurs, check the return codes after the execution of respective commands and return a return code other than 0.

Note that to stop a logical server when the host stops, add the script that stops the logical server above the script that stops the Management Server.

The following is an example of a script file:

```
#!/bin/sh
BIN_PATH=/opt/Cosminexus/manager/bin
#Stopping logical server
if [ -x $BIN_PATH/mngsvrutil ] ; then
    $BIN_PATH/mngsvrutil -m mnghost:28080 -u user1 -p user1 -t mnghost -k ho
st -s stop server
fi
# Stopping Management Server
if [ -x $BIN_PATH/mngsvrctl ] ; then
    $BIN_PATH/mngsvrctl stop
fi
exit 0
```

Specify the various options of the `mngsvrutil` command in accordance with the operating environment.

Adding the stop processing of Management Server to the `/etc/rc.shutdown` script

Add the script file that stops Management Server in the `/etc/rc.shutdown` script as the processing executed using the `/etc/rc.shutdown` script. When stopping Management Server and Administration Agent on the same host, set up the order so that Management Server stops before Administration Agent stops.

The following is an example of adding the stop processing of Management Server to the `/etc/rc.shutdown` script. Note that this is an example when the script file that stops Management Server is saved in `/etc/MngSvrStop`.

```
if [ -x /etc/MngSvrStop ]; then
    /etc/MngSvrStop
fi
```

(b) Stopping procedures in Linux

The procedures to specify the settings for Management Server to stop automatically when the host stops are described here. Also, the procedures for specifying the settings for a logical server to stop automatically when the host stops are described.

Procedures for specifying the settings that stop Management Server automatically when the host stops

If automatic start of Management Server is enabled (by using the `mngautorun` command), automatic stop is also enabled.

If the `-sync` option is specified in the `mngautorun` command, the timeout value for automatic stop of Management Server is set to 120 seconds by default. If you want to change the timeout value (by using the `-timeout` option) for automatic stop of Management Server, use the following procedure:

1. In the `/etc/systemd/system` directory, open the `CoMS.service` file, and then add the timeout value specification to the `ExecStop=` option.

The following shows an example of adding the timeout value specification.

Example:

Before changing the `CoMS.service` file:

```
ExecStop= /opt/Cosminexus/manager/bin/mngsvrctl stop -sync
```

After changing the `CoMS.service` file:

```
ExecStop= /opt/Cosminexus/manager/bin/mngsvrctl stop -sync -timeout 120
```

Note: The portion that is in bold type and has a background color has been added. You can change the italicized value according to the operating environment.

2. Execute the following command to apply the change made to the `CoMS.service` file in step 1:

```
# systemctl reenable CoMS.service
```

Important note

The settings specified in the `CoMS.service` file in this procedure are reset when the `mngautorun` command is executed again. To change the settings again, perform the procedure again.

If you want both Administration Agent and Management Server to stop on the same host, enable automatic start for both components (by specifying the `both` option in the `mngautorun` command). In this case, if you set synchronous start (by specifying the `-sync` option in the `mngautorun` command), Management Server stops and then Administration Agent stops when the host stops or restarts.

Procedures for specifying the settings that stop a logical server automatically when the host stops

The following is the procedure for specifying the settings for a logical server to stop automatically when the host stops. Note that in this procedure, it is a prerequisite that the management domain is configured only on the host on which Management Server is stopped.

1. In the `/etc/systemd/system` directory, open the `CoMS.service` file with a text editor, and then copy and record the command that is set for the `ExecStop=` option.
2. Create a script file that stops the logical server and Management Server. You can create this file in any location and with any name of your choice (in this example, create the `/opt/Cosminexus/manager/bin/LS_MNG_stop` file). The following is an example to create a script file:

```
#!/bin/sh

ret=0

/opt/Cosminexus/manager/bin/mngsvrutil -m mnghost:28080 -u user1 -p use
r1 -t mnghost -k host -s stop server
ERROR=$?
if [ $ERROR -ne 0 ] ; then
    ret=1
fi

/opt/Cosminexus/manager/bin/mngsvrctl stop -sync
ERROR=$?
if [ $ERROR -ne 0 ] ; then
    ret=1
fi

exit $ret
```

For the `mngsvrutil` command, specify the appropriate options according to the operating environment. Note that the command statement you recorded in step 1 is specified in this file to stop Management Server.

3. Change the access permission settings for the created script file:

```
# chmod 755 /opt/Cosminexus/manager/bin/LS_MNG_stop
```


4. In the `/etc/systemd/system` directory, open the `CoMS.service` file with a text editor again, and then change the path in the `ExecStop=` option to the path of the script file you created in step 2.

The following shows an example of changing the settings in the `CoMS.service` file.

Before the change (in the case of `mngautorun` with `-sync` specified):

```
ExecStop=/opt/Cosminexus/manager/bin/mngsvrctl stop -sync
```

After the change (in the case of the script created in step 2):

```
ExecStop=/opt/Cosminexus/manager/bin/LS_MNG_stop
```

5. If asynchronous start is set (`mngautorun` with both specified and `-sync` omitted) for Administration Agent and Management Server, add `CoAA.service` to the `After=` option in the `CoMS.service` file.

The following shows an example of changing the settings in the `CoMS.service` file.

Before changing `CoMS.service` (in the case of `mngautorun` with `-sync` omitted):

```
After=network.target multi-user.target
```

After the change:

```
After=CoAA.service network.target multi-user.target
```

6. Execute the following command to apply the changes to the `CoMS.service` file:

```
# systemctl reenable CoMS.service
```



Important note

The settings specified in the `CoMS.service` file in this procedure are reset when the `mngautorun` command is executed again. To change the settings again, perform the procedure again.

Procedure for removing the settings that stop the logical server automatically when the host stops:

After you have specified the settings that stop the logical server automatically when the host stops, if you want to remove the settings, use the following procedure:

1. In the procedure shown earlier, you created `/opt/Cosminexus/manager/bin/LS_MNG_stop`, a script file that stops the logical server and Management Server. In this step, open this file with a text editor, and then copy and record the Management Server stop command.
2. In the `/etc/systemd/system` directory, open the `CoMS.service` file, and then change the command in the `ExecStop=` option to the command you recorded in step 1.

The following shows an example of changing the settings in the `CoMS.service` file.

Before the change:

```
ExecStop=/opt/Cosminexus/manager/bin/LS_MNG_stop
```

After the change:

```
ExecStop=/opt/Cosminexus/manager/bin/mngsvrctl stop -sync
```

3. If asynchronous start is set (`mngautorun` with both specified and `-sync` omitted) for Administration Agent and Management Server, delete `CoAA.service` from the `After=` option in the `CoMS.service` file.

The following shows an example of changing the settings in the `CoMS.service` file.

Before the change:


```
After=CoAA.service network.target multi-user.target
```

After the change:

```
After=network.target multi-user.target
```

4. Execute the following command to apply the changes to the `CoMS.service` file:

```
# systemctl reenable CoMS.service
```

5. Delete the script file that stops the logical server and Management Server (`/opt/Cosminexus/manager/bin/LS_MNG_stop` in this example).

(2) Settings for automatically stopping Administration Agent

The procedure for specifying the settings for automatically stopping Administration Agent are described here for each OS.

Important note

If you specify settings to automatically restart Administration Agent, Administration Agent cannot be stopped by these procedures. Stop the host after stopping each logical server. However, in Red Hat Enterprise Linux Server 7 or later, even if the settings are specified so that the Administration Agent automatically restarts, you can use these procedures to stop the Administration Agent.

(a) Stopping procedures for AIX

The following actions are required for Administration Agent to stop automatically when the host stops:

1. Create a script file for Administration Agent to stop.
2. Add a processing for Administration Agent to stop in the `/etc/rc.shutdown` script.

The following are the procedures to create a script file for Administration Agent to stop, and the procedures of adding the stop processing of Administration Agent to the `/etc/rc.shutdown` script:

Creating a script file for Administration Agent stop

Create the script file that stops Administration Agent with any file name under `/etc/` (example: `/etc/AdminAgentStop`). Furthermore, set the authority of the script file to 755.

The following is an example of a script file:

```
#!/bin/sh
BIN_PATH=/opt/Cosminexus/manager/bin
# Stopping Administration Agent
if [ -x $BIN_PATH/adminagentctl ] ; then
    $BIN_PATH/adminagentctl stop
fi
exit 0
```

In this example, even if the stop processing fails, the shutdown processing is not interrupted because of the last `exit 0`. If the shutdown is interrupted when an error occurs, check the return codes after the execution of respective commands and return a return code other than 0.

Adding the stop processing for Administration Agent to the /etc/rc.shutdown script

Add the script file that makes Administration Agent to stop in the `/etc/rc.shutdown` script as the processing executed using the `/etc/rc.shutdown` script.

The following is an example of adding the stop processing of Administration Agent to the `/etc/rc.shutdown` script. Note that this is an example when the script file for Administration Agent to stop is saved in `/etc/AdminAgentStop`.

```
if [ -x /etc/AdminAgentStop ]; then
    /etc/AdminAgentStop
fi
```

(b) Stopping procedures of Linux

If automatic start of Administration Agent is enabled (by using the `mngautorun` command), automatic stop is also enabled.

If the `-sync` option is specified in the `mngautorun` command, the timeout value for automatic stop of Administration Agent is set to 120 seconds by default. If you want to change the timeout value (by using the `-timeout` option) for automatic stop of Administration Agent, use the following procedure:

1. In the `/etc/systemd/system` directory, open the `CoAA.service` file, and then add the timeout value specification to the `ExecStop=` option.

The following shows an example of adding the timeout value specification.

Example:

Before changing the `CoAA.service` file:

```
ExecStop=/opt/Cosminexus/manager/bin/adminagentctl stop -sync
```

After changing the `CoAA.service` file:

```
ExecStop=/opt/Cosminexus/manager/bin/adminagentctl stop -sync -timeout 120
```

Note: The portion that is in bold type and has a background color has been added. You can change the italicized value according to the operating environment.

2. Execute the following command to apply the change made to the `CoAA.service` file in step 1:

```
# systemctl reenable CoAA.service
```

Important note

The settings specified in the `CoAA.service` file in this procedure are reset when the `mngautorun` command is executed again. To change the settings again, perform the procedure again.

If you want both Administration Agent and Management Server to stop on the same host, enable automatic start for both components (by specifying the `both` option in the `mngautorun` command). In this case, if you set synchronous start (by specifying the `-sync` option in the `mngautorun` command), Management Server stops and then Administration Agent stops when the host stops or restarts.

3

Monitoring the Statistics (Statistics Collection Functionality)

This chapter describes the functionality for collecting statistics such as server performance and resource information. When you use the statistics collection functionality to monitor the statistics, you can output the statistics in a statistics file. Furthermore, using the statistics collection functionality and the event issuing functionality, you can detect errors in the operating status that you want to monitor.

3.1 Organization of this chapter

The following table lists the statistics collection functionality and the reference section for the functionality:

Table 3–1: Statistics collection functionality and the reference section for each functionality

Functionality	Reference
Overview of statistics collection functionality	3.2
Statistics file output functionality	3.3
Event issuing functionality	3.4

3.2 Overview of statistics collection functionality

This section provides an overview of monitoring the operating information about the J2EE server and batch server by using *statistics files*.

Using the *statistics collection functionality*, you can monitor the operating status of a J2EE server and a batch server periodically and can obtain the statistics such as server performance and resource information in a statistics file. Operating information about the functions executed on the J2EE server and batch server is periodically output to statistics files. A statistics file is created for each function of the J2EE server. The information output to statistics files can be used to check the operational statuses of the J2EE server and batch server, look at the operating results statistically, and tune the server configuration parameters.

The information output to statistics files includes the number of Full GC occurrences and the number of pending requests for each URL group. By monitoring either of these numbers, a message (threshold event) can be output when the number exceeds the threshold. You can issue a management event using the output message, and can specify settings to automatically execute the required processes corresponding to this alert. For details on setting the automatic execution of processes with Management events, see the section [9.4 Settings for automatic execution of processes by management events](#).

You can use the following functionality by monitoring the statistics:

- **Output the statistics**

You can output the obtained statistics in a statistics file. For details on the output of statistics files, see [3.3 Statistics File Output Functionality](#).

- **Issue an event**

You can set a threshold value for the monitoring targets and issue an event when the monitoring targets exceed this threshold value. For details on issuing the events, see [3.4 Event issuing functionality](#).

When monitoring the statistics, you use the statistics file described in this chapter for normal operations. When you want to obtain more detailed information than the information obtained in the statistics file, only in that case you use the management commands to monitor the statistics. For details on the monitoring statistics using the management commands, see [8.2 Overview of statistics output using management commands](#).

3.3 Statistics File Output Functionality

This subsection describes the functionality to output the obtained statistics as a statistics file.

The following table describes the organization of this section:

Table 3–2: Organization of this section (Statistics file output functionality)

Category	Title	Reference
Description	Information types that you can collect in statistics files	3.3.1
	Information that you can collect in statistics files	3.3.2
	Output destination and number of statistics files	3.3.3
Settings	Settings for the execution environment (J2EE server settings)	3.3.4
Operations	Output format and output contents of statistics files	3.3.5

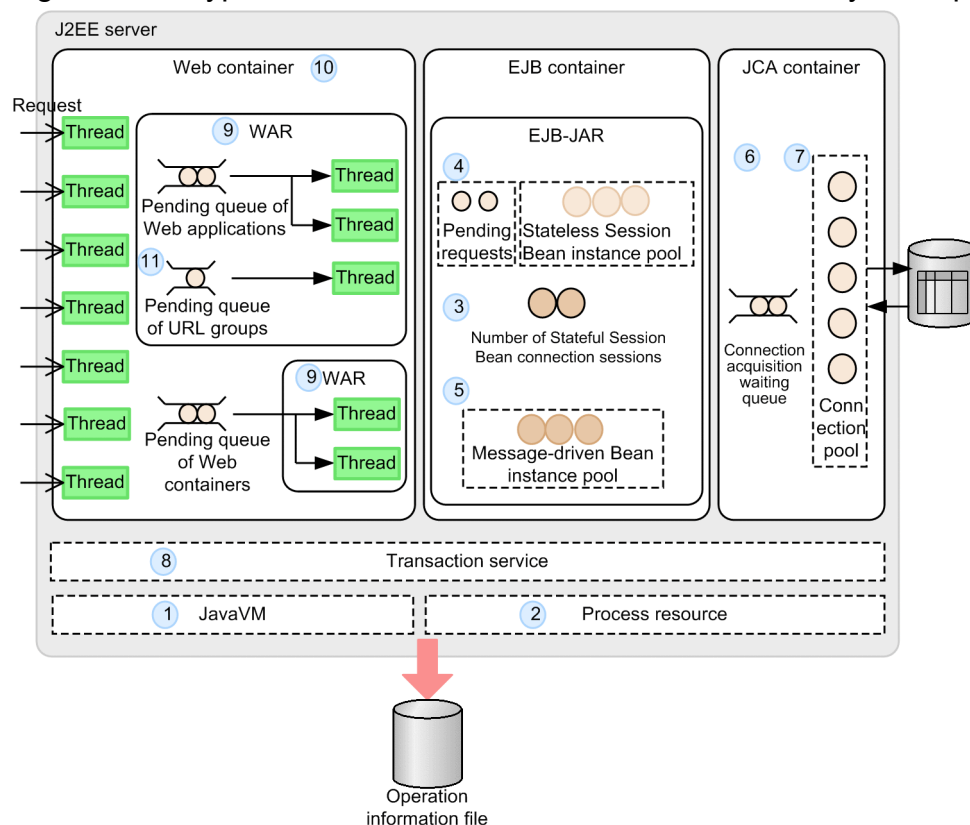
Note:

Function-specific explanation is not available for "Implementation" and "Precautions".

The statistics file output functionality collects the statistics that are output by the functionality of the J2EE server or batch server at regular interval and outputs the statistics in a text file.

The following figure illustrates the statistics output functionality:

Figure 3–1: Types of statistics and J2EE server functionality to output the statistics



The functionality that outputs the statistics is described below. The numbering of each functionality listed below corresponds to the number in the figure.

1. JavaVM
2. Process resource
3. Stateful Session Bean[#]
4. Stateless Session Bean[#]
5. Message-driven Bean[#]
6. DB Connector
7. JCA resource
8. Transaction service
9. Web Application[#]
10. Web container[#]
11. URL group[#]

[#]

Not applicable to the batch server.

The following table describes the statistics that are output:

Table 3–3: Output statistics

Statistics			Functionality that outputs the statistics
Category	Type	Information	
Server performance	Execution result information	Number of received requests [#]	Web Application
			URL group
		Number of responded requests [#]	Web Application
			URL group
		Number of sessions [#]	Web Application
		Number of requests exceeding the upper limit of the number of requests pending for execution [#]	Web container
			Web Application
			URL group
		Number of messages received [#]	Message-driven Bean
		Number of transactions resolved	Transaction service
		Number of transaction rollbacks	Transaction service
		Number of connection failures	JCA resource
		Number of pooled PreparedStatement	DBConnector
		Number of pooled CallableStatement	DBConnector
		Frequency of invoking the PrepareStatement method	DBConnector
		Frequency of invoking the PrepareCall method	DBConnector

Statistics			Functionality that outputs the statistics
Category	Type	Information	
		Hit frequency of PreparedStatement in the pool	DBConnector
		Hit frequency of CallableStatement in the pool	DBConnector
Server resource	Flow control resource information	Number of synchronous threads [#]	Web container
			Web Application
			URL group
		Number of pending requests [#]	Web container
			Web Application
			URL group
		Number of entire pending requests [#]	Web container
			Web Application
		Number of pooled instances [#]	Stateless Session Bean
		Number of used instances in the pool [#]	Stateless Session Bean
		Number of pending requests [#]	Stateless Session Bean
		Number of connection sessions [#]	Stateful Session Bean
		Number of pooled instances [#]	Message-driven Bean
		Number of used instances in the pool [#]	Message-driven Bean
		Number of pooled connections	JCA resource
		Number of used connections in the pool	JCA resource
		Number of threads waiting for connection	JCA resource
	OS resource information	JavaVM heap size	JavaVM
		Number of copy GC occurrences	JavaVM
		Number of Full GC occurrences	JavaVM
		Number of classes loaded	JavaVM
		Number of running JavaVM threads	JavaVM
		Number of threads blocked due to monitor lock	JavaVM
		Explicit heap size	JavaVM
		Number of Explicit memory blocks in the Explicit heap area	JavaVM
		Maximum size of Explicit memory block	JavaVM
		Maximum size of Explicit memory block acquired in an HTTP session	JavaVM

Statistics			Functionality that outputs the statistics
Category	Type	Information	
		Number of Explicit memory blocks acquired in an HTTP session	JavaVM
		Explicit heap size managed by containers excluding the Explicit heap area acquired in an HTTP session	JavaVM
		Explicit heap size managed by user applications and JavaVM	JavaVM
		Number of threads	Process resource
		Number of file descriptors	Process resource

#

Not applicable to batch servers.

3.3.1 Information Types that You Can Collect in Statistics Files

The statistics file output functionality outputs the operating information to a statistics file for each function of the J2EE server and batch server. The information related to server performance and server resources is output to a statistics file as the statistics. The following table shows the target server functions and the types of statistics that can be acquired.

Table 3–4: Types of the statistics that you can acquire with each function

Functionality	Description of functionality	Type of the statistics
JavaVM	Java VM used by the J2EE server	<ul style="list-style-type: none"> • JavaVM heap size • Number of copy GC occurrences • Number of Full GC occurrences • Number of classes loaded • Number of operational threads • Number of threads blocked due to monitor lock • Explicit heap size • Number of Explicit memory blocks in the Explicit heap area • Maximum size of Explicit memory block • Maximum size of Explicit memory block acquired in an HTTP session • Number of Explicit memory blocks acquired in an HTTP session • Explicit heap size managed by containers excluding the Explicit heap area acquired in an HTTP session • Explicit heap size managed by user applications and JavaVM
Process resource	Process resource used by the J2EE server	<ul style="list-style-type: none"> • Number of threads • Number of file descriptors
Stateful Session Bean	Stateful Session Bean running on a J2EE server	<ul style="list-style-type: none"> • Number of connection sessions
Stateless Session Bean	Stateless Session Bean running on a J2EE server	<ul style="list-style-type: none"> • Number of pooled instances • Number of used instances in the pool • Number of pending requests
Message-driven Bean	Message-driven Bean running on a J2EE server	<ul style="list-style-type: none"> • Number of pooled instances • Number of used instances in the pool

Functionality	Description of functionality	Type of the statistics
		<ul style="list-style-type: none"> Number of messages received
DB Connector	DB Connector running on a J2EE server	<ul style="list-style-type: none"> Number of pooled PreparedStatement Number of pooled CallableStatement Frequency of invoking the PrepareStatement method Frequency of invoking the PrepareCall method Hit frequency of PreparedStatement in the pool Hit frequency of CallableStatement in the pool
JCA resource	JCA resources that are used by resource adapters	<ul style="list-style-type: none"> Number of pooled connections Number of used connections in the pool Number of threads waiting for connection Number of connection failures
Transaction service	Transaction services used by a J2EE server	<ul style="list-style-type: none"> Number of transactions resolved Number of transaction rollbacks
Web Application	Web applications running on a J2EE server	<ul style="list-style-type: none"> Number of synchronous threads Number of pending requests for Web applications Number of entire pending requests for Web applications Number of requests exceeding the upper limit of the number of requests pending for Web applications Number of received requests Number of responded requests Number of sessions
Web container	Web containers running on a J2EE server	<ul style="list-style-type: none"> Number of synchronous threads Number of pending requests for Web containers or number of default pending requests Number of entire pending requests for Web containers Number of requests exceeding the upper limit of the number of requests pending for Web containers
URL group	Control of number of concurrently executing threads for each URL group defined in Web applications	<ul style="list-style-type: none"> Number of synchronous threads Number of pending requests for URL groups Number of requests exceeding the upper limit of the number of requests pending for URL groups Number of received requests Number of responded requests

Tip

The following table describes the relationship of the number of pending requests for Web applications, Web containers, and URL groups.

Target	Items	Explanation
Web application	Number of pending requests for Web applications	Among the requests executed in a Web container, the number of pending requests for a Web application that controls the number of synchronous threads for Web applications. The number of pending requests for URL groups is not included.
	Number of entire pending requests for Web applications	Among the requests executed in a Web container, the number of pending requests for a Web application that controls the number of synchronous threads for Web applications. The number of pending requests for URL groups is also included.

Target	Items	Explanation
Web container	Number of pending requests for Web containers or number of default pending requests	Among the requests executed in a Web container, the number of pending requests for a Web application that does not control the number of synchronous threads for Web applications.
	Number of entire pending requests for Web containers	The sum of the number of pending requests for Web containers and the number of the entire pending requests for all the Web applications.
URL group	Number of pending requests for URL groups	Among the requests executed in a Web container, the number of pending requests for a specific URL group that controls the number of synchronous threads for URL groups in a Web application controlling the number of synchronous threads for Web applications.

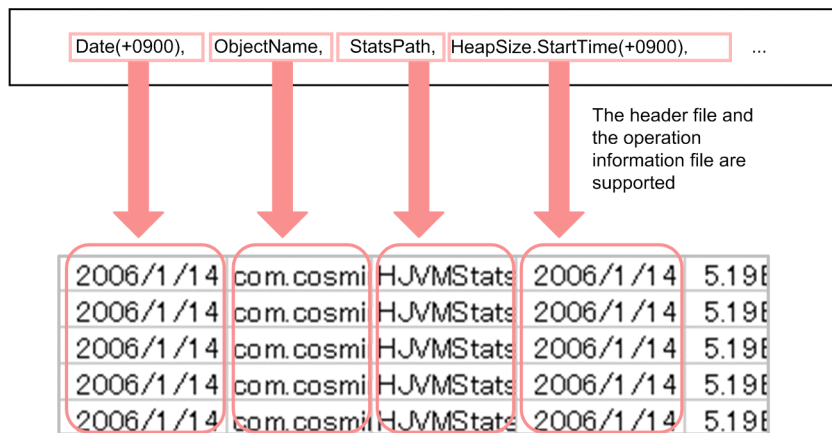
3.3.2 Information that You Can Collect in Statistics Files

To investigate statistics files, you need to reference them with *header files*. There is a header file for each J2EE server function whose operating information is collected in a statistics file. Each header file contains the names of information items that are output to a statistics file. As a result, you need to investigate the statistics while mapping the header files with the statistics files.

The following figure shows an example of mapping header files with statistics files:

Figure 3–2: Mapping header files with statistics files

Header file (items output in the operation information file are displayed)



Operation information file (the value of the items displayed in the header file is displayed)

While investigating statistics files, reference the details such as date described in the first file name, and then select the statistics file to be investigated. Edit the selected statistics file and corresponding header file together in one file, and reference that file in an application program such as Excel. In such cases, Hitachi recommends you to reference the edited file by converting in a graph or table format. When referencing the items describing time in an application program such as Excel, Hitachi recommends the display settings of the cell to be in a format in which time is described. For example, when using Excel, set the display format of the cell as Time.

The header file is created in the following folders when the J2EE server is running:

In Windows

Work-directory#\ejb\J2EE-server-name\stats

In UNIX

Work-directory[#]/ejb/J2EE-server-name/stats

#

The *work-directory* indicates a directory specified in the user definition of a J2EE server (ejb.public.directory in the usrconf.cfg file). The default value in Windows is *Cosminexus-installation-directory*\CC\server\public, and in UNIX the default value is /opt/Cosminexus/CC/server/public.

The configuration files and header files consist of items that are common for all statistics files and individual items. An individual item differs for each target function. For details on the output format and output contents of the statistics file, see [3.3.5 Output Format and Output Contents of Statistics Files](#).

3.3.3 Output Destination and Number of Statistics Files

This subsection describes the output destination of statistics files, number of files, switching intervals, and file names.

(1) Output destination of statistics files

In the case of default settings, statistics files are created in the following folders when the J2EE server is running:

In Windows

Work-directory[#]\ejb\J2EE-server-name\stats

In UNIX

Work-directory[#]/ejb/J2EE-server-name/stats

#

The *work-directory* indicates a directory specified in the user definition of a J2EE server (ejb.public.directory in the usrconf.cfg file). The default value in Windows is *Cosminexus-installation-directory* \CC\server\public, and in UNIX the default value is /opt/Cosminexus/CC/server/public.

You can also change the output destination of statistics files.

For output destination settings of statistics files, see [3.3.4 Settings for the execution environment \(J2EE server settings\)](#).

(2) Number and switching intervals of statistics files

You can set the number of statistics files to be stored in an output destination and switching intervals of statistics files. For example, if you set the number of files to be stored as 7 and the switching interval of files as 24 (hours), the statistics files are switched once daily and you can save the files of the recent week (7 days).

For details on the settings of the number of statistics files and the file switching interval, see [3.3.4 Settings for the execution environment \(J2EE server settings\)](#).

(3) File names of statistics files

The file name of a statistics file differs for each statistics collection function. The following table describes the file names of statistics files for each function:

Table 3–5: File names of statistics files

Functionality	File name
JavaVM	HJVMStats_YYYYMMDDhhmmTZ.csv
Process resource	HOSStats_YYYYMMDDhhmmTZ.csv
Stateful Session Bean	HStatefulSessionBeanStats_YYYYMMDDhhmmTZ.csv
Stateless Session Bean	HStatelessSessionBeanStats_YYYYMMDDhhmmTZ.csv
Message-driven Bean	HMessageDrivenBeanStats_YYYYMMDDhhmmTZ.csv
DB Connector	HDBConnectorStats_YYYYMMDDhhmmTZ.csv
JCA resource	HJCAConnectionPoolStats_YYYYMMDDhhmmTZ.csv
Transaction service	HJTASStats_YYYYMMDDhhmmTZ.csv
Web Application	HWebModuleStats_YYYYMMDDhhmmTZ.csv
Web container	HWebContainerStats_YYYYMMDDhhmmTZ.csv
URL group	HWebURLGroupStats_YYYYMMDDhhmmTZ.csv

Notes:

- The following are the values showing the time when a file is output in YYYYMMDDhhmmTZ:
YYYY: Christian year, MM: month, DD: day, hh: hour, mm: minute
- In TZ, the time zone is displayed with a time difference from GMT (Greenwich Mean Time). In Japan, +0900 is displayed.

3.3.4 Settings for the execution environment (J2EE server settings)

To collect the statistics files, you must specify settings for the J2EE server. You implement the J2EE server settings in the Easy Setup definition file. Specify the definition for statistics file collection in the <configuration> tag of the logical J2EE server (j2ee-server) in the Easy Setup definition file.

The statistics file is output at the following locations by default. You specify the working directory in the `ejb.public.directory` parameter of the <configuration> tag for the logical J2EE server (j2ee-server), in the Easy Setup definition file.

- In Windows
Work-directory\ejb\server-name\stats
- In UNIX
Work-directory/ejb/server-name/stats

By default, the statistics output by the J2EE server or batch server is collected and output into a statistics file. When the statistics monitoring target exceeds the threshold value, an event is issued. If you want to change the output destination or number of statistics files or if you want to change the statistics threshold value and monitoring interval, change the settings in the Easy Setup definition file.

The following table describes the statistics file collection definition specified in the Easy Setup definition file. For batch server, the statistics file collection is specified in the user properties for the batch server in the Easy Setup definition file.

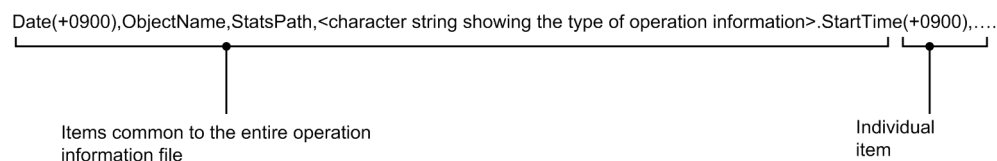
Table 3–6: Definition required for statistics file collection in the Easy Setup definition file

Items	Parameters to be specified	Setting contents
Statistics collection interval	<code>ejbserver.management.statistics.interval</code>	Specify the interval (seconds) to acquire the statistics.
Output the statistics	<code>ejbserver.management.stats_file.enabled</code>	Specify whether to output the statistics files. The default value is <code>true</code> .
	<code>ejbserver.management.stats_file.dir</code>	Specify the output directory of the statistics file.
	<code>ejbserver.management.stats_file.num</code>	Specify the number of statistics files.
	<code>ejbserver.management.stats_file.period</code>	Specify the file switching interval (hour) for the statistics files.
	<code>ejbserver.management.stats_file.base_time</code>	Specify the file switching base time for the statistics file.

For details on the Easy Setup definition file and the parameters to be specified, see 4.3 *Easy Setup definition file* in the *uCosminexus Application Server Definition Reference Guide*.

3.3.5 Output Format and Output Contents of Statistics Files

The configuration files and header files consist of items that are common for all statistics files and individual items. An individual item differs for each target function. The following figure shows the configuration of a header file:



The following table describes the viewpoint of items that are common to all statistics files:

Table 3–7: Viewpoint of items that are common to all statistics files

Character string (character string output in a header file)	Description
<code>Date[#]</code>	The collection time of statistics files is displayed as <code>YYYY/MM/DD hh:mm:ss.nnn</code> . <code>YYYY</code> : Christian year, <code>MM</code> : month, <code>DD</code> : day, <code>hh</code> : hour, <code>mm</code> : minute, <code>ss</code> : seconds, <code>nnn</code> : milliseconds
<code>ObjectName</code>	Information is displayed for each statistics file in the same format.
<code>StatsPath</code>	Information that is specific for each statistics file is output.
<code>Character-string-showing-the-type-of-statistics.StartTime[#]</code>	For each type of statistics, the time when the collection target became operational is output. The time is displayed in milliseconds from 1st January 1970, 00.00 hours onwards.

#

In the header file, in addition to this information, the time is displayed with a time difference from GMT (Greenwich Mean Time).

For example, (for Japan) :`Date(+0900)`, `HeapSize.StartTime(+0900)`

Note that if a J2EE application is replaced by reloading, the time of reloading is displayed.

The information is output in the following format for each statistics file in the ObjectName:

Table 3–8: ObjectName format

Statistics file (file name)	Format of ObjectName
JavaVM (HJVMStats_YYYYMMDDhhmmTZ.csv)	com.cosminexus.management.j2ee:J2EEServer=J2EE-server-name,j2eeType=JVM,name=jvm
Process resource (HOSStats_YYYYMMDDhhmmTZ.csv)	com.cosminexus.management.j2ee:J2EEServer=J2EE-server-name,j2eeType=OSResource,name=os
Stateful Session Bean (HStatefulSessionBeanStats_YYYYMMDDhhmmTZ.csv)	com.cosminexus.management.j2ee:EJBModule=EJB-application-display-name,J2EEApplication=J2EE-application-display-name,J2EEServer=J2EE-server-name,j2eeType=StatefulSessionBean,mode=Operation-mode-of-application ^{#1} ,name=Stateful-Session-Bean-name
Stateless Session Bean (HstatelessSessionBeanStats_YYYYMMDDhhmmTZ.csv)	com.cosminexus.management.j2ee:EJBModule=EJB-application-display-name,J2EEApplication=J2EE-application-display-name,J2EEServer=J2EE-server-name,j2eeType=StatelessSessionBean,mode=Operation-mode-of-an-application ^{#1} ,name=Stateless-Session-Bean-name
Message-driven Bean (HmessageDrivenBeanStats_YYYYMMDDhhmmTZ.csv)	com.cosminexus.management.j2ee:EJBModule=EJB-application-display-name,J2EEApplication=J2EE-application-display-name,J2EEServer=J2EE-server-name,j2eeType=MessageDrivenBean,mode=Operation-mode-of-an-application ^{#1} ,name=Message-driven-Bean-name
DB Connector (HDBConnectorStats_YYYYMMDDhhmmTZ.csv)	com.cosminexus.management.j2ee:J2EEApplication=J2EE-application-display-name ^{#2} ,J2EEServer=J2EE-server-name,ResourceAdapterModule=resource-adapter-display-name,j2eeType=ResourceAdapter,mode=Operation-mode-of-an-application ^{#1} ,name=resource-adapter-display-name
JCA resource (HJCAConnectionPoolStats_YYYYMMDDhhmmTZ.csv)	com.cosminexus.management.j2ee:J2EEServer=J2EE-server-name,ResourceAdapter=resource-adapter-display-name,j2eeType=JCAResource,mode=Operation-mode-of-an-application ^{#1} ,app=J2EE-application-display-name ^{#3} ,name=resource-adapter-display-name
Transaction service (HJTASStats_YYYYMMDDhhmmTZ.csv)	com.cosminexus.management.j2ee:J2EEServer=J2EE-server-name,j2eeType=JTAResource,name=JTAResource
Web application (HWebModuleStats_YYYYMMDDhhmmTZ.csv)	com.cosminexus.management.j2ee:J2EEApplication=J2EE-application-display-name,J2EEServer=J2EE-server-name,j2eeType=WebModule,mode=Operation-mode-of-an-application ^{#1} ,name=Web-application-display-name
Web container (HWebContainerStats_YYYYMMDDhhmmTZ.csv)	com.cosminexus.management.j2ee:J2EEServer=J2EE-server-name,j2eeType=WebContainer,name=WebContainer
URL group (HWebURLGroupStats_YYYYMMDDhhmmTZ.csv)	com.cosminexus.management.j2ee:J2EEApplication=J2EE-application-display-name,J2EEServer=J2EE-server-name,WebModule=Web-application-name,j2eeType=WebURLGroup,mode=Operation-mode-of-application ^{#1} ,name=Definition-name-for-controlling-the-number-of-concurrently-executing-threads-in-a-URL-group-unit

#1: The following values are output in an operation mode of an application:

- test: When operation mode of an application is test mode
- normal: When operation mode of an application is normal mode

#2: *J2EE-application-display-name* is output only when the resource adapter is included and deployed in a J2EE application. If the resource adapter is deployed on the J2EE server directly, *null* is output.

#3: *app=J2EE-application-display-name* is output only when the resource adapter is included and deployed in a J2EE application.

The overview of individual items in each statistics file is given below. The individual items are displayed in the following format in a header file:

```
character-string-showing-type-of-the-statistics.item-name
```

Each item is output by delimiting with a comma (,) after a character string that shows type of the statistics. The following table describes the character strings that show types of the statistics:

Table 3–9: Character strings showing type of the statistics

Type of the statistics	Character string showing type of the statistics
JavaVM heap size	HeapSize
Number of copy GC occurrences	CopyGCCount
Number of Full GC occurrences	FullGCCount
Number of classes loaded	LoadedClassCount
Number of threads blocked due to monitor lock	ThreadBlockedCount
Explicit heap size	EHeapSize
Number of Explicit memory blocks in the Explicit heap area	EMemoryBlockCount
Maximum size of Explicit memory block	EMemoryBlockMaxSize
Maximum size of Explicit memory block acquired in an HTTP session	HTTPSessionEMemoryBlockMaxSize
Number of Explicit memory blocks acquired in an HTTP session	HTTPSessionEMemoryBlockCount
Explicit heap size managed by containers excluding the Explicit heap area acquired in an HTTP session	ContainerEHeapSize
Explicit heap size managed by user applications and JavaVM	ApplicationEHeapSize
Number of threads	ThreadCount
Number of file descriptors	FileDescriptorCount
Number of connection sessions	ActiveSessionCount
Number of pooled instances	PooledInstanceCount
Number of used instances in the pool	ActivePooledInstanceCount
Number of messages received	MessageCount
Number of pending requests	WaitingRequestCount
Number of pooled PreparedStatement	PooledPreparedStatementCount
Number of pooled CallableStatement	PooledCallableStatementCount
Frequency of invoking the PrepareStatement method	InvokedPrepareStatementMethodCount
Frequency of invoking the prepareCall method	InvokedPrepareCallMethodCount
Hit frequency of PreparedStatement in the pool	PooledPreparedStatementHitCount
Hit frequency of CallableStatement in the pool	PooledCallableStatementHitCount

Type of the statistics	Character string showing type of the statistics
Number of pooled connections	PoolSize
Number of used connections in the pool	ActivePoolSize
Number of threads waiting for connection	WaitingThreadCount
Number of connection failures	FailedRequestCount
Number of transactions resolved	CompletionCount
Number of transaction rollbacks	RolledbackCount
Number of responded requests	ResponseCount
Number of synchronous threads	ActiveThreadCount
Number of requests exceeding the upper limit of the number of requests pending for execution	OverflowRequestCount
Number of received requests	RequestCount
Number of sessions	SessionCount

Table 3–10: Item names of statistics files

Item name (a character string output in a header file)	Description
Count	Accumulation count or number of accumulations from the point at which the information, such as starting of the object, was acquired.
HighWaterMark	Maximum value after the previous output of statistics files
LowWaterMark	Minimum value after the previous output of statistics files
Current	Current value
UpperBound	Upper limit
LowerBound	Lower limit

The following sections describe the items that are output separately for each statistics file:

(1) Information output to a statistics file of JavaVM

This section describes the information output to a statistics file of JavaVM. In the statistics file of JavaVM, you can investigate the statistics of JavaVM that is used by a J2EE server.

Name of the statistics file

HJVMStats_YYYYMMDDhhmmTZ.csv

Name of the corresponding header file

HJVMStats.txt

Output of the statistics

The following table describes the contents that are output to a statistics file of JavaVM:

Table 3–11: Contents output to a JavaVM statistics file

Type of the statistics	Character string (character string output in a header file)
JavaVM heap size	HeapSize.UpperBound
	HeapSize.LowerBound

Type of the statistics	Character string (character string output in a header file)
	HeapSize.HighWaterMark
	HeapSize.LowWaterMark
	HeapSize.Current
Number of copy GC occurrences ^{#1#2}	CopyGCCount.Count
Number of Full GC occurrences ^{#2#3}	FullGCCount.Count
Number of classes loaded	LoadedClassCount.HighWaterMark
	LoadedClassCount.LowWaterMark
	LoadedClassCount.Current
Number of operational threads	ThreadCount.HighWaterMark
	ThreadCount.LowWaterMark
	ThreadCount.Current
Number of threads blocked due to monitor lock	ThreadBlockedCount.HighWaterMark
	ThreadBlockedCount.LowWaterMark
	ThreadBlockedCount.Current
Explicit heap size	EHeapSize UpperBound
	EHeapSize LowerBound
	EHeapSize HighWaterMark
	EHeapSize LowWaterMark
	EHeapSize Current
Number of Explicit memory blocks in the Explicit heap area	EMemoryBlockCount HighWaterMark
	EMemoryBlockCount LowWaterMark
	EMemoryBlockCount Current
Maximum size of Explicit memory block	EMemoryBlockMaxSize HighWaterMark
	EMemoryBlockMaxSize LowWaterMark
	EMemoryBlockMaxSize Current
Maximum size of Explicit memory block acquired in an HTTP session ^{#4}	HTTPSessionEMemoryBlockMaxSize.HighWaterMark
	HTTPSessionEMemoryBlockMaxSize.LowWaterMark
	HTTPSessionEMemoryBlockMaxSize.Current
Number of Explicit memory blocks acquired in an HTTP session	HTTPSessionEMemoryBlockCount.HighWaterMark
	HTTPSessionEMemoryBlockCount.LowWaterMark
	HTTPSessionEMemoryBlockCount.Current
Explicit heap size managed by containers excluding the Explicit heap area acquired in an HTTP session	ContainerEHeapSize.HighWaterMark
	ContainerEHeapSize.LowWaterMark
	ContainerEHeapSize.Current
Explicit heap size managed by user applications and JavaVM ^{#4}	ApplicationEHeapSize.HighWaterMark

Type of the statistics	Character string (character string output in a header file)
	ApplicationEHeapSize.LowWaterMark
	ApplicationEHeapSize.Current

#1

The number of copy GC occurrences is output if serial GC is enabled.
The total number of young GC and mixed GC occurrences is output if G1 GC is enabled.

#2

The number of copy GC occurrences and the number of Full GC occurrences in Java VM statistics include the number of times GC was explicitly performed at the following times:

- When only Web applications are reloaded with the reload of J2EE applications
- When EJB applications are reloaded with the reload of J2EE applications
- When J2EE applications are stopped
- When a stub is generated with the starting of J2EE applications
- When an attempt to start J2EE applications fails
- When J2EE applications are deleted

If you enable the monitoring of the threshold for the number of Full GC occurrences, you must take into account the number of occurrences counted at the preceding times when determining the threshold value.

#3

The number of Full GC occurrences in Java VM statistics includes the number of times GC was explicitly performed by batch server processes at either of the following times:

- When the amount of memory used by the GC control functionality exceeds the threshold
- When batch applications are stopped

If you monitor the number of Full GC occurrences, you must take into account the number of occurrences counted at the preceding times when determining the threshold value.

#4

The output contents are different if you are using the memory usage reduction functionality of the Explicit heap used in an HTTP session. For details, see *7.11.3 Points to be considered when using the memory saving functionality of the Explicit heap that is used in an HTTP session* in the *uCosminexus Application Server Expansion Guide*.

(2) Information output to a statistics file of a process resource

This section describes the information output to a statistics file of a process resource. In the statistics file of a process resource, you can investigate the statistics of OS resources using J2EE server processes.

Name of the statistics file

HOSStats_YYYYMMDDhhmmTZ.csv

Name of the corresponding header file

HOSStats.txt

Output of the statistics

The following table describes the contents that are output to the statistics file of a process resource:

Table 3–12: Contents output to the statistics file of a process resource

Type of the statistics	Character string (character string output in a header file)
Number of threads created by J2EE server processes	ThreadCount.UpperBound ^{#1}
	ThreadCount.LowerBound ^{#2}
	ThreadCount.HighWaterMark ^{#2}
	ThreadCount.LowWaterMark ^{#2}
	ThreadCount.Current ^{#2}
Number of file descriptors that use J2EE server processes	FileDescriptorCount.UpperBound ^{#3}
	FileDescriptorCount.LowerBound ^{#3}
	FileDescriptorCount.HighWaterMark ^{#4}
	FileDescriptorCount.LowWaterMark ^{#4}
	FileDescriptorCount.Current ^{#4}

#1

In Windows and Linux, this item is not applicable, and therefore the string -1 is output.

#2

This is invalid in Linux, and -1 is output.

#3

This is invalid in Windows, and -1 is output.

#4

This is invalid in Windows or AIX, and -1 is output.

(3) Information output to a statistics file of a Stateful Session Bean

This section describes the information output to a statistics file of a Stateful Session Bean. In the statistics file of Stateful Session Bean, you can investigate the statistics of Stateful Session Bean.

Name of the statistics file

HStatefulSessionBeanStats_YYYYMMDDhhmmTZ.csv

Name of the corresponding header file

HStatefulSessionBeanStats.txt

Output of the statistics

The following table describes the contents that are output to a statistics file of a Stateful Session Bean:

Table 3–13: Contents output to the statistics file of a Stateful Session Bean

Type of the statistics	Character string (character string output in a header file)	Remarks
Number of connected sessions	ActiveSessionCount.UpperBound	0 is output when the maximum number of active sessions in a Stateful Session Bean is unlimited.
	ActiveSessionCount.LowerBound	This string will become invalid and -1 is output, since you cannot set a lower limit.
	ActiveSessionCount.HighWaterMark	--
	ActiveSessionCount.LowWaterMark	--

Type of the statistics	Character string (character string output in a header file)	Remarks
	ActiveSessionCount.Current	--

Legend:

--: Not applicable

(4) Information output to a statistics file of a Stateless Session Bean

This section describes the information output to a statistics file of a Stateless Session Bean. In the statistics file of Stateless Session Bean, you can investigate the statistics of Stateless Session Bean.

Name of the statistics file

HStatelessSessionBeanStats_YYYYMMDDhhmmTZ.csv

Name of the corresponding header file

HStatelessSessionBeanStats.txt

Output of the statistics

The following table describes the contents that are output to a statistics file of a Stateless Session Bean:

Table 3–14: Contents output to the statistics file of a Stateless Session Bean

Type of the statistics	Character string (character string output in a header file)	Remarks
Number of pooled instances	PooledInstanceCount.UpperBound	0 is output when the maximum number of instances in a pool of Stateless Session Bean is unlimited.
	PooledInstanceCount.LowerBound	--
	PooledInstanceCount.HighWaterMark	--
	PooledInstanceCount.LowWaterMark	--
	PooledInstanceCount.Current	--
Number of used instances in the pool	ActivePooledInstanceCount.UpperBound	0 is output when the maximum number of instances in a pool of Stateless Session Bean is unlimited.
	ActivePooledInstanceCount.LowerBound	This string will become invalid and -1 is output, since you cannot set a lower limit.
	ActivePooledInstanceCount.HighWaterMark	--
	ActivePooledInstanceCount.LowWaterMark	--
	ActivePooledInstanceCount.Current	--
Number of pending requests	WaitingRequestCount.HighWaterMark	--
	WaitingRequestCount.LowWaterMark	--
	WaitingRequestCount.Current	--

Legend:

--: Not applicable

(5) Information output to a statistics file of a Message-driven Bean

This section describes the information output to a statistics file of a Message-driven Bean. In a statistics file of Message-driven Bean, you can investigate the statistics of Message-driven Bean.

Name of the statistics file

HMessageDrivenBeanStats_YYYYMMDDhhmmTZ.csv

Name of the corresponding header file

HMessageDrivenBeanStats.txt

Output of the statistics

The following table describes the contents that are output to a statistics file of a Message-driven Bean:

Table 3–15: Contents output to a statistics file of a Message-driven Bean

Type of the statistics	Character string (character string output in a header file)	Remarks
Number of pooled instances	PooledInstanceCount.UpperBound	0 is output when the maximum number of instances in a pool of Message-driven Bean is unlimited.
	PooledInstanceCount.LowerBound	--
	PooledInstanceCount.HighWaterMark	--
	PooledInstanceCount.LowWaterMark	--
	PooledInstanceCount.Current	--
Number of used instances in the pool	ActivePooledInstanceCount.UpperBound	0 is output when the maximum number of instances in a pool of Message-driven Bean is unlimited.
	ActivePooledInstanceCount.LowerBound	This string will become invalid and -1 is output, since you cannot set a lower limit.
	ActivePooledInstanceCount.HighWaterMark	--
	ActivePooledInstanceCount.LowWaterMark	--
	ActivePooledInstanceCount.Current	--
Number of messages received	MessageCount.Count	--

Legend:

--: Not applicable

(6) Information output to a statistics file of a DB Connector

This section describes the information output to a statistics file of a DB Connector. In the statistics file of DB Connector, you can investigate the statistics of DB Connector.

When you are not using the connection pooling function, values other than the frequency of invoking the prepareStatement method (InvokedPrepareStatementMethodCount.Count) and frequency of invoking the PrepareCall method (InvokedPrepareCallMethodCount.Count) will be invalid, and -1 will be output.

Name of the statistics file

HDBConnectorStats_YYYYMMDDhhmmTZ.csv

Name of the corresponding header file

HDBConnectorStats.txt

Output of the statistics

The following table describes the contents that are output to a statistics file of a DB Connector:

Table 3–16: Contents output to the statistics file of a DB Connector

Type of the statistics	Character string (character string output in a header file)	Remarks
Number of PreparedStatement cached by the statement pooling function	PooledPreparedStatementCount.UpperBound ^{#1}	The following values are output: Output value=Size of the PreparedStatement for each connection set in the DB Connector * Upper limit for a connection pool If the upper limit of a connection pool is infinite, the character string is invalid and -1 is output.
	PooledPreparedStatementCount.LowerBound	This string will become invalid and -1 is output, since you cannot set a lower limit.
	PooledPreparedStatementCount.HighWaterMark ^{#1#2}	--
	PooledPreparedStatementCount.LowWaterMark ^{#1#2}	--
	PooledPreparedStatementCount.Current ^{#1#2}	--
	PooledPreparedStatementCount.Current ^{#1#2#4}	--
Number of CallableStatement cached by the statement pooling function	PooledCallableStatementCount.UpperBound ^{#3}	The following values are output: Output value=Size of PooledCallableStatement for each connection set in the DB Connector * Upper limit for a connection pool If the upper limit of a connection pool is infinite, the character string is invalid and -1 is output.
	PooledCallableStatementCount.LowerBound	This string will become invalid and -1 is output, since you cannot set a lower limit.
	PooledCallableStatementCount.HighWaterMark ^{#2#3}	--
	PooledCallableStatementCount.LowWaterMark ^{#2#3}	--
	PooledCallableStatementCount.Current ^{#2#3}	--
	PooledCallableStatementCount.Current ^{#2#3#4}	--
Frequency of invoking the PrepareStatement method	InvokedPrepareStatementMethodCount.Count ^{#2}	--
Frequency of invoking the PrepareCall method	InvokedPrepareCallMethodCount.Count ^{#2}	--
Frequency of PreparedStatement hits cached by the statement pooling function	PooledPreparedStatementHitCount.Count ^{#1#2}	--
Frequency of CallableStatement hits cached by the statement pooling function	PooledCallableStatementHitCount.Count ^{#2#3}	--

Legend:

--: Not applicable

#1

If you specify 0 for a pool size of PreparedStatement for each connection that is the DB Connector setting, the setting will become invalid and -1 will be output as the PreparedStatement will not be pooled.

#2

If you are using Oracle as a database to be connected to, or if you detect an error in the connection that is not using HiRDB as a database to be connected to, 1 or a value more than 1 is output even if the statement object is not created.

#3

If you specify 0 for a pool size of CallableStatement for each connection that is the DB Connector setting, the setting will become invalid and -1 is output since the CallableStatement will not be pooled.

#4

If used concurrently with the connection count adjustment function or connection failure detection, the unused connections removed from the connection pool are not counted as the number of connections in the connection pool; therefore, the number of pooled PreparedStatement and CallableStatement might temporarily exceed the following values:

- Number of PreparedStatement
 $\text{maximum-value-of-connection-pool} \times \text{PreparedStatementPoolSize}$
- Number of CallableStatement
 $\text{maximum-value-of-connection-pool} \times \text{CallableStatementPoolSize}$

(7) Information output to a statistics file of a JCA resource

This section describes the information output to a statistics file of a JCA resource. In the statistics file of a JCA resource, you can investigate the statistics of the JCA resource. Note, however, that if the connection pool is disabled, only the number of connection failures is output. If you are using the DB Connector for Cosminexus RM, the statistics is output to Cosminexus RM. The statistics is not output to the DB Connector for Cosminexus RM.

If the used resource adapter is compliant with the Connector 1.5 specifications, the statistics corresponding to the first connection definition (the first definition in `ra.xml`) is output. If the Outbound connection definition does not exist, the HJCAConnectionPoolStats statistics is not output.

Name of the statistics file

HJCAConnectionPoolStats_YYYYMMDDhhmmTZ.csv

Name of the corresponding header file

HJCAConnectionPoolStats.txt

Output of the statistics

The following table describes the contents that are output to a statistics file of a JCA resource:

Table 3–17: Contents output to the statistics file of a JCA resource

Type of the statistics	Character string (character string output in a header file)	Remarks
Number of pooled connections	PoolSize.UpperBound	--
	PoolSize.LowerBound	--
	PoolSize.HighWaterMark	--
	PoolSize.LowWaterMark	--
	PoolSize.Current	--
Number of used connections in the pool	ActivePoolSize.UpperBound	--
	ActivePoolSize.LowerBound	This string will become invalid and -1 is output, since you cannot set a lower limit.
	ActivePoolSize.HighWaterMark	A value might change temporarily using the command execution and the

Type of the statistics	Character string (character string output in a header file)	Remarks
	ActivePoolSize.LowWaterMark	automatic regulation of number of connections functionality.
	ActivePoolSize.Current	
Number of threads waiting for connection	WaitingThreadCount.UpperBound	This string will become invalid and -1 will be output, since you cannot set a upper limit.
	WaitingThreadCount.LowerBound	This string will become invalid and -1 is output, since you cannot set a lower limit.
	WaitingThreadCount.HighWaterMark	--
	WaitingThreadCount.LowWaterMark	--
	WaitingThreadCount.Current	--
Number of connection failures	FailedRequestCount.Count	When you are using DB Connector for Cosminexus RM, this character string is output to the Cosminexus RM. Normally, 0 is output to the DB Connector for Cosminexus RM.

Legend:

--: Not applicable

(8) Information output to a statistics file of a transaction service

This section describes the information output to the statistics file of a transaction service. In the statistics file of a transaction service, you can investigate the statistics of the transaction service.

Name of the statistics file

HJTASStats_YYYYMMDDhhmmTZ.csv

Name of the corresponding header file

HJTASStats.txt

Output of the statistics

The following table describes the contents output to a statistics file of a transaction service:

Table 3–18: Contents output to the statistics file of a transaction service

Type of the statistics	Character string (character string output in a header file)
Number of settled transactions	CompletionCount.Count
Number of rolled back transactions	RolledbackCount.Count

(9) Information output to a statistics file of a Web application

This section describes the information output to the statistics file of a Web application. You can investigate the statistics of the entire target Web application in a Web application statistics file.

Name of the statistics file

HWebModuleStats_YYYYMMDDhhmmTZ.csv

Name of the corresponding header file

HWebModuleStats.txt

Output of the statistics

The following table describes the contents output to a statistics file of a Web application:

Table 3–19: Contents output to the statistics file of a Web application

Type of the statistics	Character string (character string output in a header file)	Remarks
Maximum number of concurrently executing threads	ActiveThreadCount.UpperBound ^{#1#2}	--
	ActiveThreadCount.LowerBound ^{#3}	--
	ActiveThreadCount.HighWaterMark ^{#1}	--
	ActiveThreadCount.LowWaterMark ^{#1}	--
	ActiveThreadCount.Current ^{#1}	--
Number of pending requests for Web applications	WaitingRequestCount.UpperBound ^{#1#2#4}	--
	WaitingRequestCount.LowerBound ^{#3#4}	--
	WaitingRequestCount.HighWaterMark ^{#1 #4}	--
	WaitingRequestCount.LowWaterMark ^{#1 #4}	--
	WaitingRequestCount.Current ^{#1 #4}	--
Number of entire pending requests for Web applications	WholeWaitingRequestCount.UpperBound ^{#1 #4}	--
	WholeWaitingRequestCount.LowerBound ^{#1 #4}	--
	WholeWaitingRequestCount.HighWaterMark ^{#1 #4}	--
	WholeWaitingRequestCount.LowWaterMark ^{#1 #4}	--
	WholeWaitingRequestCount.Current ^{#1 #4}	--
Number of requests exceeding the upper limit of the number of requests pending for Web applications	OverflowRequestCount.Count ^{#1}	--
Number of received requests	RequestCount.Count ^{#4}	The number of requests received by a Web container for a Web application is output. The number of pending requests for Web applications is also counted. The number of requests received by a Web server but not transferred to the Web container is not counted.
Number of responded requests	ResponseCount.Count	--
Number of sessions	SessionCount.UpperBound	If upper limit of the number of sessions is not set, this string becomes invalid and -1 is output.
	SessionCount.LowerBound ^{#3}	--
	SessionCount.HighWaterMark	--
	SessionCount.LowWaterMark	--
	SessionCount.Current	--

Legend:

--: Not applicable

#1

If you are not using the function to control the number of concurrently executing threads for each Web application or the function to control the number of concurrently executing threads in the target Web application, this string becomes invalid and -1 is output.

#2

If the control for the maximum number of concurrently executing threads in a target Web application is set, the maximum number of concurrently executing threads for each Web application is output. If you change the maximum number of concurrently executing threads dynamically, the value after the change is output.

#3

This string becomes invalid and -1 is output.

#4

The number of pending execution requests received by a Web application is not counted during the switch over processing using the reload function. The number of pending execution requests is counted after reloading is complete.

(10) Information output to a statistics files of a Web container

This section describes the information output to the statistics file of a Web container. You can investigate the statistics of a running Web application in a statistics file of a Web container.

Name of the statistics file

HWebContainerStats_YYYYMMDDhhmmTZ.csv

Name of the corresponding header file

HWebContainerStats.txt

Output of the statistics

The following table describes the contents output to a statistics file of a Web container:

Table 3–20: Contents output to the statistics file of a Web container

Type of the statistics	Character string (character string output in a header file)
Maximum number of concurrently executing threads (total number of synchronous, asynchronous, and WebSocket threads)	WorkerThreadCount.UpperBound
	WorkerThreadCount.LowerBound ^{#1}
	WorkerThreadCount.HighWaterMark ^{#2}
	WorkerThreadCount.LowWaterMark ^{#2}
	WorkerThreadCount.Current ^{#2}
Maximum number of concurrently executing threads (the number of only synchronous threads)	ActiveThreadCount.UpperBound
	ActiveThreadCount.LowerBound ^{#1}
	ActiveThreadCount.HighWaterMark ^{#2}
	ActiveThreadCount.LowWaterMark ^{#2}
	ActiveThreadCount.Current ^{#2}
Number of pending requests for web containers or default pending requests (total number of synchronous, asynchronous, and WebSocket threads)	WaitingWorkerThreadCount.UpperBound ^{#3}
	WaitingWorkerThreadCount.LowerBound ^{#1#3}
	WaitingWorkerThreadCount.HighWaterMark ^{#4}
	WaitingWorkerThreadCount.LowWaterMark ^{#4}

Type of the statistics	Character string (character string output in a header file)
	WaitingWorkerThreadCount.Current ^{#4}
Number of pending requests for Web containers or default pending requests (the number of only synchronous threads)	WaitingRequestCount.UpperBound ^{#3}
	WaitingRequestCount.LowerBound ^{#1#3}
	WaitingRequestCount.HighWaterMark ^{#4}
	WaitingRequestCount.LowWaterMark ^{#4}
	WaitingRequestCount.Current ^{#4}
Number of entire pending requests for Web containers	WholeWaitingRequestCount.UpperBound ^{#5}
	WholeWaitingRequestCount.LowerBound ^{#5}
	WholeWaitingRequestCount.HighWaterMark ^{#5}
	WholeWaitingRequestCount.LowWaterMark ^{#5}
	WholeWaitingRequestCount.Current ^{#5}
Number of requests exceeding the upper limit of the number of requests pending for Web containers	OverflowRequestCount.Count ^{#3}

#1

This string becomes invalid and -1 is output.

#2

If the setting to control the maximum number of concurrently executing threads for each Web application is disabled, a new request received and pending for execution during reloading of the J2EE application is counted as a running thread.

#3

If the control for the maximum number of concurrently executing threads for each Web application is not set, this string becomes invalid and -1 is output. If the control is set, default information of the queue of pending executions is output.

#4

If the control for the maximum number of concurrently executing threads for each Web application is not set, the information of the queue of pending executions for each Web container is output. If the control is set, default information of the queue of pending executions is output.

#5

During the replacement processing using the reload functionality, the number of pending requests accepted by the Web container is not counted. The number of pending requests accepted by the Web container is counted after the reload processing ends.

(11) Information output to a statistics file of a URL group

This section describes the information output to a statistics file of a URL group. You can investigate the statistics of a URL group in a statistics file of a URL group.

Name of the statistics file

HWebURLGroupStats_YYYYMMDDhhmmTZ.csv

Name of the corresponding header file

HWebURLGroupStats.txt

Output of the statistics

The following table describes the contents output to a statistics file of a URL groups:

Table 3–21: Contents output to the statistics file of a URL group

Type of the statistics	Character string (character string output in a header file)
Maximum number of concurrently executing threads	ActiveThreadCount.UpperBound ^{#1}

Type of the statistics	Character string (character string output in a header file)
	ActiveThreadCount.LowerBound ^{#2}
	ActiveThreadCount.HighWaterMark
	ActiveThreadCount.LowWaterMark
	ActiveThreadCount.Current
Number of pending requests for URL groups	WaitingRequestCount.UpperBound ^{#3}
	WaitingRequestCount.LowerBound ^{#2#3}
	WaitingRequestCount.HighWaterMark ^{#3}
	WaitingRequestCount.LowWaterMark ^{#3}
	WaitingRequestCount.Current ^{#3}
Number of requests exceeding the upper limit of the number of requests pending for URL groups	OverflowRequestCount.Count
Number of received requests	RequestCount.Count ^{#4}
Number of responded requests	ResponseCount.Count

#1

If you dynamically change the maximum number of concurrently executing threads for each Web application, the value after change and not the set value, is output.

#2

This string becomes invalid and -1 is output.

#3

The number of pending requests for URL groups received by a Web application is not counted during the switch over processing using the reload function. The number of pending execution requests is counted after reloading is complete.

#4

The number of requests for URL groups received by a Web container is output. The number of pending requests for URL groups is also counted. The number of requests received by a Web server but not transferred to the Web container is not counted.

3.4 Event issuing functionality

This subsection describes the functionality for issuing an event when a threshold value is set for a monitoring target and the monitoring target exceeds the threshold value.

The following table describes the organization of this section:

Table 3–22: Organization of this section (event issuing functionality)

Category	Title	Reference
Description	Monitoring target for which threshold value can be set	3.4.1
	How to issue an event	3.4.2
Implementation	Definition in cosminexus.xml	3.4.3
Settings	Settings for the execution environment (J2EE server settings)	3.4.4

Note: Function-specific explanations are not available for "Operations" and "Precautions".

This functionality issues an event when the items whose statistics are to be monitored exceed the specified threshold value. You can detect any error in the status when an event is issued.

A message is output, when an event is issued. You can use the message to issue a Management Event. For details on the management events and management actions, see the chapter [9. Automatic Execution of Processing Using Management Event Notification and Management Action](#).

3.4.1 Monitoring target for which threshold value can be set

This subsection describes the two types of targets whose statistics are to be monitored and for which you can set a threshold value.

- **Number of Full GC occurrences**

Monitors the number of times Full GC occurs. With this item, an abnormal state in which Full GC frequently occurs can be detected. A threshold value can be set for this item.

You can monitor the frequency of Full GC in the execution environments of the J2EE applications and batch applications.

The message ID reported as a Management event is KDJE53850-W.

- **Number of entire pending requests for Web containers**

Monitors the number of entire pending requests for Web containers. Before the number of entire pending requests for Web containers exceeds the request queue size, it can be detected that the request queue space has decreased. For a threshold value, specify the ratio of the number of entire pending requests for Web containers to the pending request queue size of the number of entire pending requests for Web containers.

You can monitor the number of entire pending requests for Web containers in the execution environment of J2EE applications.

The message IDs reported as a Management event are KDJE53862-W (High Threshold value), and KDJE53863-I (Low Threshold value).

- **Number of pending requests for Web containers**

Monitors the number of pending requests for Web containers. Before the number of pending requests for Web containers exceeds the request queue size, it can be detected that the request queue space has decreased. For a

threshold value, specify the ratio of the number of pending requests for Web containers to the pending request queue size of the number of pending requests for Web containers.

You can monitor the number of pending requests for Web containers in the execution environment of J2EE applications.

The message IDs reported as a Management event are KDJE53864-W (High Threshold value), and KDJE53865-I (Low Threshold value).

- **Number of entire pending requests for Web applications**

Monitors the number of entire pending requests for Web applications. Before the number of entire pending requests for Web applications exceeds the request queue size, it can be detected that the request queue space has decreased. For a threshold value, specify the ratio of the number of entire pending requests for Web applications to the pending request queue size of the number of entire pending requests for Web applications.

You can monitor the number of entire pending requests for Web applications in the execution environment of J2EE applications.

The message IDs reported as a Management event are KDJE53866-W (High Threshold value), and KDJE53867-I (Low Threshold value).

- **Number of pending requests for Web applications**

Monitors the number of pending requests for Web applications. Before the number of pending requests for Web applications exceeds the request queue size, it can be detected that the request queue space has decreased. For a threshold value, specify the ratio of the number of pending requests for Web applications to the pending request queue size of the number of pending requests for Web applications.

You can monitor the number of pending requests for Web applications in the execution environment of J2EE applications.

The message IDs reported as a Management event are KDJE53868-W (High Threshold value), and KDJE53869-I (Low Threshold value).

- **Number of pending requests in URL group**

Monitors the number of requests present in the pending requests queue of a URL group. This type of monitoring can detect the reduced number of free slots in the request queue, before the number of pending requests for URL groups exceeds the size of the request queue. For the threshold value, specify the ratio of number of pending requests for URL groups to the pending request queue size for URL groups.

The number of pending requests in a URL group can be monitored with the execution environment of J2EE applications.

The message IDs reported as a Management event are KDJE53860-W (High Threshold value), and KDJE53861-I (Low Threshold value).

Use the resource depletion monitoring functionality to set up and monitor a threshold value for the number of requests in the default pending queue for Web containers and the pending queue for Web applications. For the resource depletion monitoring functionality, see [4.3 Resource depletion monitoring functionality and output of resource depletion monitoring information](#).

3.4.2 How to issue an event

The two ways of issuing an event include the frequency type and the incremental type, depending upon the monitoring target for which the threshold value is set. The following table describes the mapping of event issuing methods and monitoring targets.

Table 3–23: Mapping of event issuing methods and monitoring targets

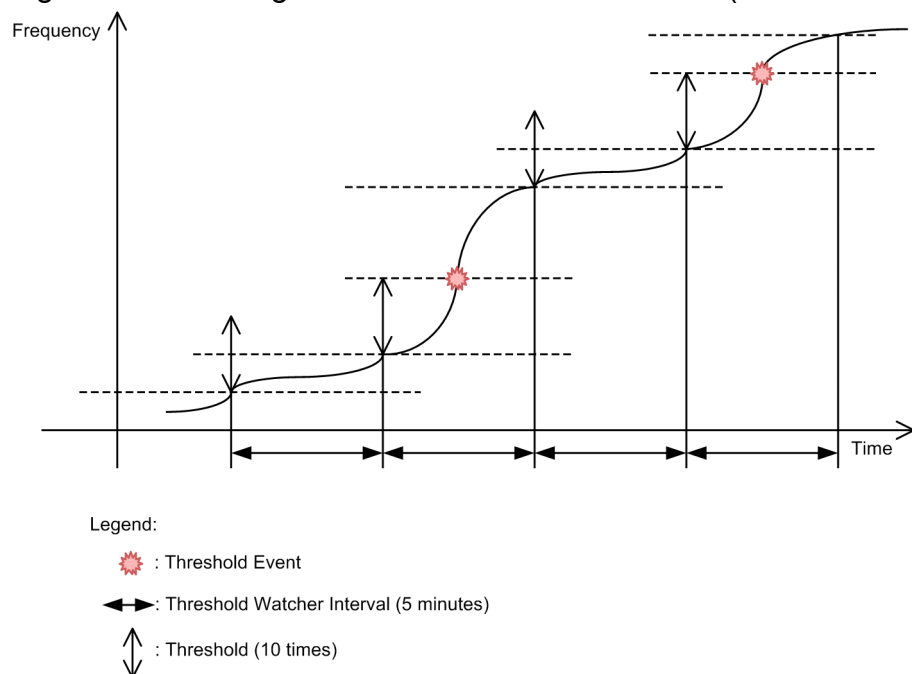
Event issuing method	Monitoring target
Frequency type	Number of Full GC occurrences
Incremental type	<ul style="list-style-type: none"> • Number of entire pending requests for Web containers • Number of pending requests for Web containers • Number of entire pending requests for Web applications • Number of pending requests for Web applications • Number of pending requests for URL groups

(1) For the number of Full GC occurrences (frequency type)

The number of times Full GC occurs within a preset length of time (threshold watcher interval) is counted, and when the counter value reaches the threshold, an event is issued.

The following figure shows how a threshold event is issued for the number of Full GC occurrences.

Figure 3–3: Issuing of an event of threshold value (for the number of Full GC occurrences)



In the preceding figure, the threshold for the number of Full GC occurrences is 10 and the threshold watcher interval is 5 minutes. From when the monitoring of operating information starts, the number of Full GC occurrences is counted in each threshold watcher interval. If the counter value in a threshold watcher interval exceeds the threshold (10), an event is issued.

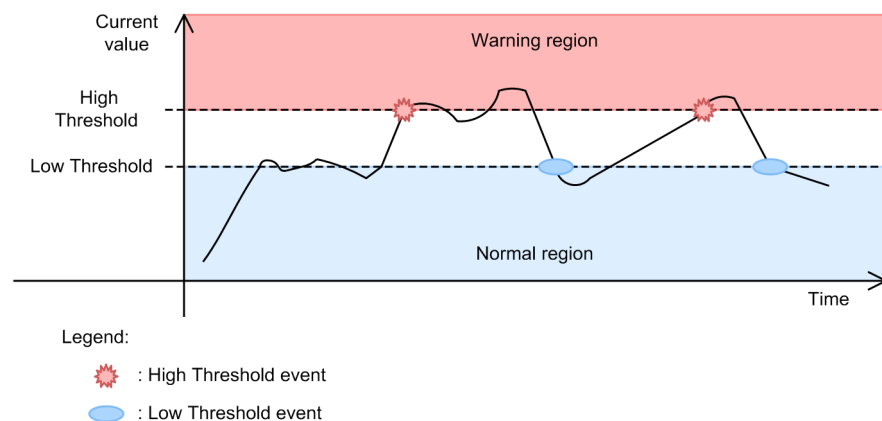
(2) In the case of number of pending requests in a URL group (incremental type)

Specify the maximum and minimum threshold values and issue an event when the number of pending requests in a URL group reaches the respective threshold value. The event of maximum threshold value and the event of minimum threshold value are issued at the timing mentioned below. The maximum threshold value needs to be greater than the minimum threshold value.

- A maximum threshold value event is issued when the number of pending requests in a URL group reaches the maximum threshold value (warning zone).
- A minimum threshold value event is issued to indicate that normal state is restored, when the number of pending requests in a URL group reaches the minimum threshold value after a maximum threshold value event was issued.
- A maximum threshold value event is issued when the maximum threshold value is reached once again after the number of pending requests in a URL group has fallen below the minimum threshold value once after the maximum threshold value event was issued. A maximum threshold value event is not issued when the maximum threshold value is reached again without reaching the minimum threshold value, after the maximum threshold value event was issued.
- A minimum threshold value event is issued when the minimum threshold value is reached again after the number of pending requests in a URL group exceeds the maximum threshold value since a minimum threshold value event was issued. A minimum threshold value event is not issued when the minimum threshold value is reached again without reaching the maximum threshold value, after the minimum threshold value event is issued.

The following figure illustrates the issuing of a threshold value event when the number of pending requests in a URL group is used:

Figure 3–4: Issuing of a threshold value event (when the number of pending requests in a URL group is used)



In the above figure, a maximum threshold value event is issued when the threshold value reaches the maximum value from the normal zone. Next, the maximum threshold value is reached once again after falling below the maximum threshold value, but a maximum threshold value event is not issued at this time. From this point onwards, a minimum threshold value event is issued when the minimum threshold value is reached and a maximum threshold value event is issued when the maximum threshold value is reached.

3.4.3 Definition in cosminexus.xml

This subsection describes definition of `cosminexus.xml` required in the application development environment.

Specify the definition for using the event issuing functionality in the `<war>` tag of `cosminexus.xml`.

The following table describes the definition of the event issuing functionality in `cosminexus.xml`:

Table 3–24: Definition of the event issuing functionality in cosminexus.xml

Monitoring target	Tags to be specified	Setting contents
Number of entire pending requests for Web applications	<code><thread-control>-<thread-control-stats-monitor>-<whole-waiting-request-count>-<enabled></code>	Specify whether to enable the monitoring of the entire pending request count of a Web application. The default value is <code>true</code> .
	<code><thread-control>-<thread-control-stats-monitor>-<whole-waiting-request-count>-<high-threshold></code>	Specify the upper limit threshold value of the entire pending request count for a Web application. The value is specified with the percentage (%) stored for the number of entire pending requests for Web applications.
	<code><thread-control>-<thread-control-stats-monitor>-<whole-waiting-request-count>-<low-threshold></code>	Specify the lower limit threshold value of the entire pending request count for a Web application. The value is specified with the percentage (%) stored for the number of entire pending requests for Web applications.
Number of pending requests for Web applications	<code><thread-control>-<thread-control-stats-monitor>-<waiting-request-count>-<enabled></code>	Specify whether to enable the monitoring of the pending request count of a Web application. The default value is <code>true</code> .
	<code><thread-control>-<thread-control-stats-monitor>-<waiting-request-count>-<high-threshold></code>	Specify the upper limit threshold value for the pending request count for a Web application. The value is specified with the percentage (%) stored for the number of pending requests for Web applications.
	<code><thread-control>-<thread-control-stats-monitor>-<waiting-request-count>-<low-threshold></code>	Specify the lower limit threshold value of the pending request count for a Web application. The value is specified with the percentage (%) stored for the number of pending requests for Web applications.
Number of pending requests for URL groups	<code><thread-control>-<urlgroup-thread-control>-<stats-monitor>-<waiting-request-count>-<enabled></code>	Specify whether to enable the monitoring of the pending request count of the URL group. The default value is <code>true</code> .
	<code><thread-control>-<urlgroup-thread-control>-<stats-monitor>-<waiting-request-count>-<high-threshold></code>	Specify the upper limit threshold value for the pending request count for a URL group. The value is specified with the percentage (%) stored for the number of pending requests for URL groups.
	<code><thread-control>-<urlgroup-thread-control>-<stats-monitor>-<waiting-request-count>-<low-threshold></code>	Specify the lower limit threshold value for the pending request count for a URL group. The value is specified with the percentage (%) stored for the number of pending requests for URL groups.

For details on the tags to be specified, see 2.2.6 *Details of the WAR property* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

3.4.4 Settings for the execution environment (J2EE server settings)

To issue events using the statistics file, you must specify the J2EE server and J2EE application settings. You reference the J2EE application settings only when you want to set or change the properties of a J2EE application that does not contain `cosminexus.xml`.

(1) Setting J2EE servers

Implement the J2EE server settings in the Easy Setup definition file. Specify the definition for issuing events of the `<configuration>` tag for the logical J2EE server (`j2ee-server`), in the Easy Setup definition file.

The following table describes the definition for event issuing specified in the Easy Setup definition file.

Table 3–25: Definition for event issuing in the Easy Setup definition file

Parameters to be specified	Setting contents
<code>ejbserver.management.JVM.stats_monitor.FullGCCount.enabled</code>	Specifies whether to enable the monitoring of the number of Full GC occurrences. The default value is <code>true</code> .
<code>ejbserver.management.JVM.stats_monitor.FullGCCount.threshold</code>	Specifies the threshold for event issuance during the monitoring of the number of Full GC occurrences.
<code>ejbserver.management.JVM.stats_monitor.FullGCCount.interval</code>	Specifies the threshold watcher interval for the number of Full GC occurrences.
<code>ejbserver.management.webcontainer.stats_monitor.whole_waiting_request_count.enabled</code>	Specify whether to enable the monitoring of the entire pending request count of a Web container. The default value is <code>true</code> .
<code>ejbserver.management.webcontainer.stats_monitor.whole_waiting_request_count.high_threshold</code>	Specify the upper limit threshold value of the entire pending request count for a Web container.
<code>ejbserver.management.webcontainer.stats_monitor.whole_waiting_request_count.low_threshold</code>	Specify the lower limit threshold value of the entire pending request count for a Web container.
<code>ejbserver.management.webcontainer.stats_monitor.waiting_request_count.enabled</code>	Specify whether to enable the monitoring of the pending request count of a Web container. The default value is <code>true</code> .
<code>ejbserver.management.webcontainer.stats_monitor.waiting_request_count.high_threshold</code>	Specify the upper limit threshold value of the pending request count for a Web container.
<code>ejbserver.management.webcontainer.stats_monitor.waiting_request_count.low_threshold</code>	Specify the lower limit threshold value of the pending request count for a Web container.

For details on the Easy Setup definition file and the parameters to be specified, see *4.3 Easy Setup definition file* in the *uCosminexus Application Server Definition Reference Guide*.

(2) Setting J2EE applications

You implement the J2EE application settings for the execution environment using the server management commands and the property files. You use the WAR property file to define the issuing of events.

The tags specified in the WAR property file correspond to `cosminexus.xml`. For details on the definitions of `cosminexus.xml`, see *3.4.3 Definition in cosminexus.xml*.

4

Monitoring Resource Depletion

This chapter describes the resource depletion monitoring functionality provided in the Application Server system. You can also use this functionality in combination with management events.

4.1 Organization of this chapter

The following table describes the resource depletion monitoring functionality and the reference section for each functionality:

Table 4–1: Resource depletion monitoring functionality and the reference section for each functionality

Functionality	Reference
Overview of resource depletion monitoring functionality	4.2
Resource depletion monitoring functionality and output of resource depletion monitoring information	4.3

4.2 Overview of resource depletion monitoring functionality

This section describes how to monitor resource depletion on the J2EE server based on the changes in resource usage percentage or quantity.

You can use the resource depletion monitoring functionality to monitor seven types of resources. You can monitor all resources, or select the resources for monitoring. In the resource depletion monitoring functionality, specify the resources to be monitored and the threshold values for each resource. The resources of the J2EE server or batch server are subject to resource depletion monitoring.

If resource depletion monitoring for the J2EE server or batch server is enabled, information about monitoring-target resources is periodically output to a file. This information is called *resource depletion monitoring information*. You can choose whether to output the resource depletion monitoring information for each type of resource to be monitored. The file to which the resource depletion monitoring information is output is called *resource depletion monitoring log file*.

When the resource depletion monitoring information is used, you can check how the usage rate of resources or the number of resources used change. Moreover, this information can also be used to investigate the reason due to which the usage rate of resources or the number of used resources exceeds the threshold value.

An alert occurs when the threshold value specified for the usage rate of resources or the number of used resources is exceeded. If an alert occurs when using the Management Server, you can output a message and notify an event to the Management Server. This event is called a *Management event*. At the Management Server side, you can define the operation to be executed when a management event is notified, so that an action can be automatically executed when the management event occurs. This action is called a *Management action*. You can combine and use the resource depletion monitoring functionality and the Management event to monitor the resources effectively and reliably. For details on the management events and management actions, see the section [9.3 Controlling the Execution of Management Action](#).

For details on setting the automatic execution of processes with Management events, see the section [9.4 Settings for automatic execution of processes by management events](#).

For output destination of the resource depletion monitoring log file, see [4.3.1 Acquiring the Cosminexus Component Container Logs](#) in the *uCosminexus Application Server Maintenance and Migration Guide*.

For settings for monitoring the resource depletion, see [4.3.3 Settings for the execution environment](#).



Reference note

You check the output destination file for thread dumps by monitoring the resource depletion. When there are multiple files in the thread dump output destination (default is *work-directory/obj/J2EE-server-name*), multiple system resources (such as the CPU) are consumed in the check processing.

4.3 Resource depletion monitoring functionality and output of resource depletion monitoring information

This subsection describes the types of resources that you can monitor using the resource depletion monitoring functionality, and the information output into the log file.

The following table describes the organization of this section:

Table 4–2: Organization of this section (functionality that output resource depletion monitoring log file)

Category	Title	Reference
Description	Types of resources that can be monitored	4.3.1
Description	Definition in cosminexus.xml	4.3.2
Settings	Settings for the execution environment	4.3.3
Operations	Location of the resource depletion monitoring information file and types of information output to the file	4.3.4
	Output format and contents	4.3.5

Note:

The function-specific explanation is not available for "Precautions".

4.3.1 Types of resources that can be monitored

The resource depletion monitoring functionality can monitor seven types of resources. You can choose to monitor all resources or the necessary resources only. However, there are resources that cannot be monitored depend on the OS used. Furthermore, in the batch server, you cannot monitor the HTTP request pending queue and the number of sessions.

The details of each resource are explained below:

- Monitoring the memory

Monitoring the memory involves monitoring the heap memory area in the JavaVM. Symptoms that trigger Full GC can be detected by monitoring memory usage. The monitoring results can also be used to tune the heap and memory sizes.

KDJE34500-W is output when the set threshold value is exceeded.

Tip

Although this functionality monitors memory usage, Full GC does not always occur when the percentage of memory usage has become close to 100%. The monitoring results are used to detect symptoms that are highly likely to trigger Full GC.

- Monitoring the file descriptor

Monitoring the file descriptor involves monitoring the file descriptors that are open in the server processes. Monitoring the number of file descriptors helps to detect the depletion of file descriptors that can be used in the processing of server allocated to the system, and detect the depletion before the number of file descriptors that are requested in the estimate, is reached.

KDJE34520-W is output when the set threshold value is reached.

File descriptors cannot be monitored in Windows and AIX.

- Monitoring the threads

Monitoring the threads involves monitoring the number of threads generated during the processing of server. By monitoring the number of threads that are generated, you can detect whether the number of threads that can be generated during the process of the server allocated to the system has exceeded before the number of threads requested in the estimate is reached.

KDJE34540-W is output when the set threshold value is reached.

Thread monitoring cannot be used in Linux.

- Monitoring the thread dump

Monitoring the thread dump involves monitoring the number of thread dump files that are output when monitoring the `cjdumpsv` command, or the J2EE application execution time, in a server. By monitoring the number of thread dump files; you can detect before the maximum number of files is reached.

Note that if you specify the environment variable `JAVACOREDIR`, this functionality monitors the total number of files of the thread dump of both the directories; the directory specified in the environment variable `JAVACOREDIR` and the default output destination directory (in Windows: `work-directory\ejb\server-name` and in UNIX: `work-directory/ejb/server-name`). If the path specified in the environment variable `JAVACOREDIR` is not in the directory and the file list cannot be acquired, only the number of files of the default output destination are monitored. KDJE34580-W is output when the set threshold value is reached. KDJE34581-E is output when the maximum value is reached.

For details on monitoring the J2EE application during runtime, see [5.3 Monitoring and Canceling a J2EE Application During Runtime](#).

Notes

The process of monitoring the thread dump has a load in proportion to the number of files and directories of the directory specified in the environment variable `JAVACOREDIR` and the default output destination directory. Therefore, set up the monitoring interval of the thread dump for a longer time or do not allocate unnecessary files to the directories.

- Monitoring the HTTP requests queue

Monitoring the HTTP requests queue involves monitoring the number of requests in the pending queue in the synchronous thread control of a Web application. The pending queues to be monitored are the pending queue of each Web application and the default pending queue. You can monitor the queues on the J2EE server or the Web application. Monitoring the pending queue for each Web application and the default pending queue of the HTTP requests helps to detect the queue status before the requests exceed the size of these pending queues.

However, when the specified size of the pending queue for each Web application and the default pending queue are '0', the HTTP requests queue is not monitored.

The initial value specified for the threshold value does not change even when the size of the pending queue for each Web application changes dynamically.

KDJE34621-W is output when the set threshold value is reached.

For the number of concurrently executed threads, see [2.15 Controlling the number of concurrently executing threads in the Web application](#) in the *uCosminexus Application Server Web Container Functionality Guide*.

Use the statistics collection functionality to monitor requests in the pending queue for URL groups. For how to set up the threshold value using the statistics collection functionality, see [3.4.1 Monitoring target for which threshold value can be set](#).

- Monitoring the number of sessions

Monitoring the number of sessions involves monitoring the number of sessions that are generated in a Web application. Monitoring the number of sessions can detect any increase in the number of sessions.

The number of sessions, however, is not monitored when '0' is specified for the number of sessions that can be created during the setup of number of sessions.

KDJE34640-W is output when the set threshold value is reached.

- Monitoring the connection pool

Monitoring the connection pool involves monitoring the usage of the connection pool. This can detect the depletion of connections before it occurs and hence, can be used to tune the connection pool.

The usage of connection cannot be monitored when connection pool is inactive. Moreover, when the maximum value of the connection pool is infinite, the usage of the connection pool can be monitored but the management event is not notified.

KDJE34660-W is output when the set threshold value is exceeded. KDJE34661-W is output when the maximum number is reached.

For connection pool, see *3.14.1 Connection pooling* in the *uCosminexus Application Server Common Container Functionality Guide*.

4.3.2 Definition in cosminexus.xml

This subsection describes the definitions required in `cosminexus.xml` in the application development environment.

Specify the definition for using the resource depletion monitoring functionality in the `<war>` tag of `cosminexus.xml`.

The following table describes the definitions of the resource depletion monitoring functionality in `cosminexus.xml`:

Table 4–3: Definition of the resource depletion monitoring functionality in `cosminexus.xml`

Type of resource	Specified tags	Setting contents
HTTP requests pending queue	Specify the following tags under the <code><war></code> - <code><thread-control></code> - <code><resource-watcher></code> tag <ul style="list-style-type: none"> • <code><watcher-enabled></code> tag • <code><watcher-threshold></code> tag • <code><watcher-interval></code> tag • <code><watcher-writefile-enabled></code> tag 	Make the settings for monitoring the HTTP requests pending queue for the pending queue of each Web application in each J2EE application. <ul style="list-style-type: none"> • Issuing of alert by monitoring the HTTP requests pending queue • Monitoring the threshold value • Monitoring interval • Existence of output of resource depletion monitoring log file
Number of sessions	Specify the following tags under the <code><http-session></code> - <code><resource-watcher></code> tag <ul style="list-style-type: none"> • <code><watcher-enabled></code> tag • <code><watcher-threshold></code> tag • <code><watcher-interval></code> tag • <code><watcher-writefile-enabled></code> tag 	Make the settings for monitoring the number of sessions for each J2EE application. <ul style="list-style-type: none"> • Issuing of alert by monitoring the number of sessions • Monitoring the threshold value • Monitoring interval • Existence of output of resource depletion monitoring log file

For details on the specified tags, see *2.2.6 Details of the WAR property* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

4.3.3 Settings for the execution environment

The resource depletion monitoring functionality can be used on the J2EE server or batch server. In J2EE servers, the resources that can be monitored include memory, file descriptors, number of threads, thread dumps, HTTP requests pending queue, number of sessions, and connection pools. In batch servers, the resources that can be monitored include

memory, file descriptors, number of threads, thread dumps, and connection pools. Note that file descriptors cannot be monitored in Windows and AIX. Furthermore, the number of threads cannot be monitored in Linux.

The following settings are required when using the resource depletion monitoring functionality:

- J2EE server
- Hitachi Connector Property file
- J2EE application

You reference the above settings only for setting or changing the properties of J2EE applications that do not include `cosminexus.xml`.

(1) Setting J2EE servers

You implement the J2EE server settings in the Easy Setup definition file. Specify the definition for resource depletion monitoring functionality in the `<configuration>` tag of the logical J2EE server (`j2ee-server`) in the Easy Setup definition file. You specify the settings for each batch server in the user properties for the batch server, in the Easy Setup definition file.

The following two definitions are specified for the resource depletion monitoring functionality in the Easy Setup definition file:

- Application of the resource depletion monitoring function
- Settings for monitoring for each resource type

(a) Enabling the resource depletion monitoring function

Specify whether you want to use the resource depletion monitoring functionality in the `ejbserver.watch.enabled` parameter of the `<configuration>` tag that exists in the logical J2EE server (`j2ee-server`) in the Easy Setup definition file. By default, the resource depletion monitoring function is applied and all the resources are monitored. To disable the resource depletion monitoring functionality, you specify `false` (disable the resource depletion monitoring functionality) in the `ejbserver.watch.enabled` parameter.

(b) Setting for monitoring for each resource type

The following table describes the definitions for monitoring each resource type exists in the Easy Setup definition file:

Table 4–4: Definition of the resource depletion monitoring functionality in the Easy Setup definition file

Type of resource	Parameters to be specified	Setting contents
Memory	<ul style="list-style-type: none">• <code>ejbserver.watch.memory.enabled</code>• <code>ejbserver.watch.memory.threshold</code>• <code>ejbserver.watch.memory.interval</code>• <code>ejbserver.watch.memory.writefile.enabled</code>	<p>These parameters can be used to specify the memory monitoring settings for each J2EE server or batch server.</p> <p>Specify the alert issue existence, monitoring threshold value, monitoring interval, and existence of output of resource depletion monitoring log file based on memory monitoring.</p> <p>Also, specify the number of files and the size for the resource depletion monitoring log files.^{#1#2}</p>
File descriptors	<ul style="list-style-type: none">• <code>ejbserver.watch.fileDescriptor.enabled</code>• <code>ejbserver.watch.fileDescriptor.threshold</code>	<p>These parameters can be used to specify the file descriptor monitoring settings for each J2EE server or batch server.</p>

Type of resource	Parameters to be specified	Setting contents
	<ul style="list-style-type: none"> <code>ejbserver.watch.fileDescriptor.interval</code> <code>ejbserver.watch.fileDescriptor.writefile.enabled</code> 	<p>Specify the alert issue existence, monitoring threshold value, monitoring interval, and existence of output of resource depletion monitoring log file based on file descriptor monitoring.</p> <p>Also, specify the number of files and the size for the resource depletion monitoring log files.^{#2}</p>
Number of threads	<ul style="list-style-type: none"> <code>ejbserver.watch.thread.enabled</code> <code>ejbserver.watch.thread.threshold</code> <code>ejbserver.watch.thread.interval</code> <code>ejbserver.watch.thread.writefile.enabled</code> 	<p>These parameters can be used to specify the thread monitoring settings for each J2EE server.</p> <p>Specify the alert issue existence, monitoring threshold value, monitoring interval, and existence of output of resource depletion monitoring log file based on the monitoring of the number of threads.</p> <p>Also, specify the number of files and the size for the resource depletion monitoring log files.^{#2}</p>
Thread dump	<ul style="list-style-type: none"> <code>ejbserver.watch.threaddump.enabled</code> <code>ejbserver.watch.threaddump.threshold</code> <code>ejbserver.watch.threaddump.interval</code> <code>ejbserver.watch.threaddump.writefile.enabled</code> <code>ejbserver.server.threaddump.filename</code> 	<p>These parameters can be used to specify the thread dump monitoring settings for each J2EE server or batch server.</p> <p>Specify the alert issue existence, monitoring threshold value, monitoring interval, and existence of output of resource depletion monitoring log file based on thread dump monitoring.</p> <p>Also, specify the number of files and the size for the resource depletion monitoring log files.^{#2}</p>
HTTP requests pending queue	<ul style="list-style-type: none"> <code>ejbserver.watch.defaultRequestQueue.enabled</code> <code>ejbserver.watch.defaultRequestQueue.threshold</code> <code>ejbserver.watch.defaultRequestQueue.interval</code> <code>ejbserver.watch.defaultRequestQueue.writefile.enabled</code> 	<p>Make the settings for monitoring the default HTTP requests pending queue for each J2EE server.</p> <p>Specify the alert issue existence, monitoring threshold value, monitoring interval, and existence of output of resource depletion monitoring log file based on the monitoring of the HTTP requests pending queue.</p> <p>Also, specify the number of files and the size for the resource depletion monitoring log files.^{#2}</p>
Number of sessions	--	Specify the number and size of resource depletion monitoring log files in each J2EE server. ^{#2}
Connection pool	--	Specify the number and size of resource depletion monitoring log files in each J2EE server or batch server. ^{#2}

Legend:

--: None

#1 To perform memory monitoring, define the Java VM start parameters under the `<configuration>` tag for the logical J2EE server (j2ee-server). Specify `-XX:MetaspaceSize` and `-XX:MaxMetaspaceSize` as the Java VM start parameters and set the same memory size for these parameters. If you set different memory sizes, no alert might be generated when Full GC occurs. The following contents are to be specified for the JavaVM start parameter:

`<param-name>` tag

`add.jvm.arg`

`<param-value>` tag

`-XX:MetaspaceSize=setup-value`

`-XX:MaxMetaspaceSize=setup-value`

To detect symptoms that trigger Full GC, you must tune the sizes of the Old and New areas of Java VM.

#2 Specify the number of files and the size for the resource depletion monitoring log file in the following parameters of the Easy Setup definition file:

- `ejbserver.logger.channels.define.name-of-resource-to-be-monitoredWatchLogFile.filename`

Specifies the number of resource depletion monitoring log files for each resource type.

- `ejbserver.logger.channels.define.name-of-resource-to-be-monitoredWatchLogFile.filesize`

Specifies the size of the resource depletion monitoring log file for each resource type.

In these parameters, specify the channel name for each resource type as the name of the resource to be monitored.

- For memory: `MemoryWatchLogFile`
- For file descriptor: `FileDescriptorWatchLogFile`
- For the number of threads: `ThreadWatchLogFile`
- For thread dump: `ThreaddumpWatchLogFile`
- For HTTP requests pending queue: `RequestQueueWatchLogFile`
- For the number of sessions: `HttpSessionWatchLogFile`
- For connection pools: `ConnectionPoolWatchLogFile`

For details on the Easy Setup definition file and the parameters to be specified, see *4.3 Easy Setup definition file* in the *uCosminexus Application Server Definition Reference Guide*.

(2) Settings for Hitachi Connector Property file

Make the settings for monitoring the connection pool in each resource adapter. You implement the resource adapter property settings using the server management commands and the Hitachi Connector Property file.

Specify the issuing of alert, monitoring threshold values, monitoring intervals, and application of resource depletion monitoring log file output based on monitoring the connection pool in the `<WatchEnabled>`, `<WatchThreshold>`, `<WatchInterval>`, and `<WatchWriteFileEnabled>` tag specified as the `<property-name>` tag of the Hitachi Connector Property file.

For details on the tags to be specified, see *4.1 HITACHI Connector Property file* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

For resource adapter property settings in the server management commands and the Hitachi Connector Property file, see *3.5.1 Procedure for setting the properties of a J2EE resource adapter* in the *uCosminexus Application Server Application Setup Guide*.

(3) Setting J2EE applications

You implement the settings for the J2EE applications in the execution environment using the server management commands and the property files. Use the WAR property file to define the resource depletion monitoring functionality.

The tags specified in the WAR property file correspond to `cosminexus.xml`. For details on the definitions of `cosminexus.xml`, see *4.3.2 Definition in cosminexus.xml*.

4.3.4 Location of the resource depletion monitoring information file and types of information output to the file

(1) Output destination of resource depletion monitoring information

By default, the resource depletion monitoring information file is created in the following folder when the J2EE server starts:

In Windows:

`ejb.server.log.directory\watch\`

In UNIX:

ejb.server.log.directory/watch/

Important note

In the preceding paths, *ejb.server.log.directory* refers to the directory specified for the *ejb.server.log.directory* key in the user definition (*usrconf.cfg* file). In Windows, the default directory is *Cosminexus-installation-directory\CC\server\public\ejb\server-name\logs*. In UNIX, the default directory is */opt/Cosminexus/CC/server/public/ejb/server-name/logs*.

If the value of the *ejb.server.log.directory* key is changed, the output destination of resource depletion monitoring information is also changed accordingly.

(2) Types of information that are output

The following table describes the information output in resource depletion monitoring information. You cannot monitor the number of file descriptors in Windows or AIX. Thread monitoring cannot be used in Linux:

Table 4–5: Information output in resource depletion monitoring information

Information type	File name	Information overview
Monitoring information of memory depletion	Cjmemorywatch	The result of monitoring the usage of heap and memory area of JavaVM is output. This information is output if true is set in the <i>ejbserver.watch.memory.writefile.enabled</i> key of the <i>usrconf.properties</i> .
Monitoring information of depletion of file descriptors	cjfiledescriptorwatch	The result of monitoring the number of file descriptors using a J2EE server process is output. This information is output if true is set in the <i>ejbserver.watch.fileDescriptor.writefile.enabled</i> key of the <i>usrconf.properties</i> .
Monitoring information of thread depletion	Cjthreadwatch	The result of monitoring the number of threads created by a J2EE server process is output. This information is output if true is set in the <i>ejbserver.watch.thread.writefile.enabled</i> key of the <i>usrconf.properties</i> .
Monitoring information of thread dump depletion	cjthreaddumpwatch	The result of monitoring the number of output files of thread dump is output. This information is output if true is set in the <i>ejbserver.watch.threaddump.writefile.enabled</i> key of the <i>usrconf.properties</i> .
Monitoring information of depletion of queue of pending execution of HTTP requests	cjrequestqueuewatch	The following two types of queues of requests that are pending for execution are monitored, and results are output as results of monitoring the queue of HTTP requests pending execution: <ul style="list-style-type: none">• The queue of pending execution of Web applications, for which the maximum number of concurrently executing threads for each Web application is not set, is monitored and the result of monitoring the stored number of requests is output. This information is output if true is set in the <i>ejbserver.watch.defaultRequestQueue.writefile.enabled</i> key of the <i>usrconf.properties</i>.• The queue of pending execution of Web applications, for which the maximum number of concurrently executing threads for each Web

Information type	File name	Information overview
		<p>application is set, is monitored and the result of monitoring the stored number of requests is output.</p> <p>This information is output as application attribute if true is set in the <watcher-writefile-enabled> tag below the <war><hitachi-war-property><thread-control><resource-watcher> tag.</p> <p>Note that the preceding queues are monitored if control of the maximum number of concurrently executing threads for each Web application is enabled.</p> <p>The information about the pending queue for URL groups is not output to the depletion monitoring information of the HTTP request pending queue. Use the statistics collection functionality to monitor the information of the pending queue for URL groups. For the statistics collection functionality, see 3. Monitoring the Statistics (Statistics Collection Functionality).</p>
Monitoring information of depletion of number of sessions	cjhttpsessionwatch	<p>The result of monitoring the number of sessions for each Web application is output.</p> <p>This information is output as application attribute if true is set in the <watcher-writefile-enabled> tag below the <war><hitachi-war-property><http-session><resource-watcher> tag.</p>
Monitoring information of depletion of connection pool	cjconnectionpoolwatch	<p>The result of monitoring the number of connection pools for each resource adapter is output.</p> <p>This information is output if true is set in the WatchWriteFileEnabled property as resource adapter property.</p>

4.3.5 Output Format and Contents

The records of resource depletion monitoring information are output in CSV format files.

The following figure shows the output format:

Figure 4–1: Output format of resource depletion monitoring information

yyyy/mm/dd hh:mm:ss.sss	pid	tid	message-id	message(LANG=ja)
0000 2004/99/99 99:99:99.999	HEJB	00000000	00000000	
,"Title1","Title2","Title3",...				
0000 2004/99/99 99:99:99.999	HEJB	00000000	00000000	,"Data1","Data2","Data3",...
0000 2004/99/99 99:99:99.999	HEJB	00000000	00000000	,"Data1","Data2","Data3",...

The output format is described below:

- The title line is output after every 100 lines so that contents of each column are understood clearly.
- The following items are output in the title as common information of resources. These items are output in the place of "Title1", "Title2", "Title3"... in the figure.
 - Rate: Rate (usage)
 - Current: Current value
 - Max: Maximum value
 - Threshold: Threshold value

Apart from this, required information is output for each resource. For details, see the output contents of each resource.

- Result of determining the threshold value is output in the order of current value followed by the set threshold value. If multiple threshold values are determined multiple values are output.

- If threshold value is set in %, the calculation result of threshold value is also output in %. For example, if the calculation result is 0.752, and you have specified the threshold value in %, the calculation result is output as 75.2.
- If threshold value is an absolute value, the calculation result of the threshold value is not output.
- If an upper limit is not set for the threshold value, the calculation result of threshold value is output as -.
- If the value is in decimals, the second place after the decimal is rounded off and the value with one place after the decimal point is output. Even when determining a threshold value, the second place after the decimal is rounded off and the threshold value with one place after the decimal is used.

The file output contents depend on the values set in `usrconf.properties`, application attributes, or resource adapter attributes.

The output information for each is described in the following sections.

(1) Monitoring information of memory depletion

The following table describes the information output in monitoring information of memory depletion:

Table 4–6: Contents output in monitoring information for memory depletion

Character string of output title	Output contents
Rate1	<p>If serial GC is enabled: Memory usage of the Old area is output. The unit is %. The memory usage of the Old area is calculated using the following formula:</p> <ul style="list-style-type: none"> • Memory usage of the Old area = Size of occupied Old area / Total size of Old area × 100 (Rate1 = (Total [Old] - Free [Old]) / Total [Old] × 100) <p>If G1 GC is enabled: The Java heap space usage is output. The unit is %. The memory usage in the Java heap space is calculated using the following formula:</p> <ul style="list-style-type: none"> • Memory usage in the Java heap space [%] = Size of the Java heap space that is used / size of the entire Java heap space × 100
Rate2	<p>If serial GC is enabled: Total memory usage of the New area is output for maximum free memory in the Old area. The unit is %. Total memory usage of the New area for the maximum free memory in the Old area is calculated using the following formula:</p> <ul style="list-style-type: none"> • Total memory usage of the New area for the maximum free memory in the Old area = Total size of New area / Maximum free size in the Old area × 100 (Rate2 = Total [New] / (Max [Old] - (Total [Old] - Free [Old])) × 100) <p>Note that this value may exceed 100.</p> <p>If G1 GC is enabled: No information is output for this item. The value -1 is always output.</p>
Rate3	<p>Memory usage of the Metaspace area is output. The unit is %. The memory usage of the Metaspace area is calculated using the following formula:</p> <ul style="list-style-type: none"> • Memory usage of the Metaspace area = Occupied size of Metaspace area / Total size of Metaspace area × 100 (Rate3 = (Total [Permanent] - Free [Permanent]) / Total [Permanent] × 100)
Free [New]	<p>The free memory size of the New area is output. The unit is bytes.</p>

Character string of output title	Output contents
Total [New]	The total memory size of the New area is output. The unit is bytes.
Max [New]	If serial GC is enabled: The maximum memory size of the New area is output. The unit is bytes. If G1 GC is enabled: No information is output for this item. The value -1 is always output.
Free [Old]	The free memory size of the Old area is output. The unit is bytes.
Total [Old]	The total memory size of the Old area is output. The unit is bytes.
Max [Old]	If serial GC is enabled: The maximum memory size of the Old area is output. The unit is bytes. If G1 GC is enabled: The maximum size of memory available in the Java heap space is output.
Free [Permanent]	The free memory size of the Metaspace area is output. The unit is bytes.
Total [Permanent]	The total memory size of the Metaspace area is output. The unit is bytes.
Max [Permanent]	The maximum memory size of the Metaspace area is output. The unit is bytes.
Threshold	Threshold value is output. The unit is %.

Important note

- The memory depletion monitoring functionality assumes that it is used in a system in which memory requirements have been estimated properly and the settings are tuned so that Full GC is unlikely to occur. This functionality is aimed at detecting symptoms that trigger Full GC, resulting in a temporary stop of the application. Note that if the memory tuning is insufficient, unnecessary alerts may be output.

To monitor memory depletion, you must set the same value for the maximum size and initial size of the Metaspace area. To be more precise, set the same value for the `-XX:MaxMetaspaceSize` and `-XX:MetaspaceSize` of `usrconf.cfg`. If different values are specified, alerts might be issued even during the expansion of the Metaspace area. For details on other tuning parameters, see the information on the Oracle website.

Although this functionality monitors memory usage, Full GC does not always occur when the percentage of memory usage has become close to 100%. The monitoring results are used to detect symptoms that are highly likely to trigger Full GC.

- From Application Server 09-00 onwards, the Java heap is efficiently utilized so that Full GC is unlikely to occur. For this reason, an unnecessary alert might be generated if the system continues to operate at high usage of Java heap space. To stop alert generation for Rate1, adjust the threshold. To stop alert generation for Rate2, set `ejbserver.watch.memory.rate2alert.enabled=false` in the `usrconf.properties` file.

Reference note

The heap and memory of JavaVM contain the following three types of areas:

- **New area**

This area is applicable to Eden and Survivor. This area stores new objects.

- **Old area**

This area is applicable to Tenured. This area stores objects existing for a long time.

- **Metaspace area**

This area stores classes loaded in JavaVM.

The term *Metaspace area* has been adopted in Application Server 09-70 onwards.

In earlier versions of Application Server than 09-70, this area was called *Permanent area*. Therefore, the string `Permanent` is used, for compatibility with old versions, in the header strings for the output information items that indicate Metaspace area sizes.

The Information of free area, occupied area, and maximum area size for each of these areas at that point of time is output in memory monitoring result.

If memory monitoring is set, messages are output in the following cases:

- **If the occupied size of the Old area corresponding to the total size of the Old area exceeds the threshold value** (when serial GC is enabled)

The message is output if " $\text{Occupied size of Old area} / \text{Total size of Old area} \times 100 \geq \text{threshold value}(\%)$ ".

- **If the total size of the New area corresponding to the maximum free size of the Old area exceeds the threshold value** (when serial GC is enabled)

The message is output if " $\text{Total size of New area} / \text{Maximum free size of Old area} \times 100 \geq \text{threshold value}(\%)$ ".

- **If the percentage of the used Java heap area size in the entire Java heap area size reaches or exceeds the threshold value** (when G1 GC is enabled)

A message is output if " $\text{Used Java heap area size} / \text{Total size of Java heap area} \times 100 \geq \text{Threshold value}(\%)$ ".

- **If the percentage of the used Metaspace area size in the entire Metaspace area size reaches or exceeds the threshold value**

A message is output if " $\text{Used Metaspace area size} / \text{Maximum Metaspace area size} \times 100 \geq \text{Threshold value}(\%)$ ".

(2) Monitoring information of depletion of file descriptors

The following table describes the information output in the result of monitoring file descriptors information. You cannot monitor the number of file descriptors in Windows or AIX.

Table 4–7: Contents output in the monitoring information for depletion of file descriptors

Character string of output title	Output contents
Current	The number of file descriptors used by J2EE server processes is output.
Max	The number of file descriptors that can be allocated to a process is output.
Threshold	Threshold value is output.

(3) Monitoring information of thread depletion

The following table describes the information output in monitoring information of thread depletion:

Table 4–8: Contents output in the monitoring information for thread depletion

Character string of output title	Output contents
Current	The number of threads using self-process is output.
Max	<ul style="list-style-type: none">• In Windows Normally "-" is output• In UNIX Number of threads that can be generated by a process is output.
Threshold	Threshold value is output.

(4) Monitoring information of depletion of thread dump

The following table describes the information output in the monitoring information of depletion of thread dump:

Table 4–9: Contents output in the monitoring information for thread dump depletion

Character string of output title	Output contents
Rate	The percentage of number of files of the current thread dump corresponding to its maximum value is output. The unit is %.
Current	The current value of the number of thread dump files is output.
Max	The upper limit of the number of thread dump files is output.
Threshold	Threshold value is output. The unit is %.

(5) Monitoring information of depletion of queue of HTTP requests pending execution

The queue of HTTP requests pending execution is the pending queue for Web applications and the default pending queue corresponding to the concurrently executing threads of the Web application.

The following table describes the information output in the monitoring information depletion of queue of HTTP request pending execution. This depletion monitoring information includes the following two types of information:

- Result of monitoring the number of default stored requests of the pending queues for each J2EE server.
- Result of monitoring the number of stored queues of requests pending execution for each Web application.

Table 4–10: Contents output in the monitoring information for depletion of HTTP requests pending queue

Character string of output title	Output contents
J2eeApplicationName	The J2EE application name is output. For monitoring results of default queue of requests pending execution, "-" is output.
ContextRootName	Context root name is output. For monitoring results of default queue of requests pending execution, "-" is output.
Rate	The percentage of the number of currently stored queues corresponding to its maximum value is output.

Character string of output title	Output contents
	The unit is %.
Current	The current value of the number of stored queues is output.
Max	The maximum value of the number of requests stored in the pending queue for Web applications and the default pending queue is output.
Threshold	Threshold value is output. The unit is %.

(6) Monitoring information of the number of depleted sessions

The following table describes the information output in monitoring information of the number of depleted sessions:

Table 4–11: Contents output in the monitoring information for depletion of the number of sessions

Character string of output title	Output contents
J2eeApplicationName	The J2EE application name is output.
ContextRootName	Context root name is output.
Rate	The percentage of the number of current sessions created corresponding to its maximum value is output. The unit is %. If the maximum value is not set, "-" is output.
Current	The current value of the number of sessions is output.
Max	The maximum value of the number of sessions is output. If the maximum value is not set, "-" is output.
Threshold	Threshold value is output. The unit is %.

(7) Monitoring information of the depletion of connection pool

The following table describes the information output in monitoring information of depletion of a connection pool:

Table 4–12: Contents output in the monitoring information for connection pool depletion

Character string of output title	Output contents
ResourceName	The resource adapter name given by the user is output in the following format: When the resource adapter is directly deployed on a J2EE server <i>Resource-adapter-name</i> When the resource adapter is deployed by including in a J2EE application <i>J2EE-application-name : Resource-adapter-name</i> When the resource adapter is deployed by including in a J2EE application, and then starting in the test mode <i>TEST#J2EE-application-name : Resource-adapter-name</i>
Rate	The usage of a connection pool is output. The unit is %. If the maximum value of the number of connections is infinite, "-" is output.
Active	The number of connections in use is output.
Free	The number of unused connections is output.
Current	The current value of the number of connections is output. #

Character string of output title	Output contents
Min	The minimum value of the number of connections is output.
Max	The maximum value of the number of connections is output. If a connection pool is infinite, "-1" is output.
Threshold	Threshold value is output. The unit is %.
All	The total number of connections (total number of connections managed in connection pool and not managed in connection pool) is output. #

#

Depending on the timing when the connection is established, a value greater than All might be output to Current temporarily.

5

Operations of J2EE Applications

This chapter describes the operations of the J2EE applications.

In the J2EE application operations, you can monitor the execution time of a J2EE application, perform service lock operations on a J2EE application, stop the J2EE application, switch the services, and access network resources from J2EE applications.

5.1 Organization of this chapter

The following table lists the operation and reference of a J2EE application:

Table 5–1: Operations of J2EE applications and the corresponding reference sections

Functionality	Reference
Overview of J2EE application operations	5.2
Monitoring and canceling a J2EE application during runtime	5.3
Locking the J2EE application service	5.4
Stopping a J2EE application	5.5
Switching the J2EE application	5.6
Accessing network resources from J2EE applications	5.7

5.2 Overview of J2EE Application Operations

J2EE application operations are the standard operations in daily business, such as appropriately changing the settings in accordance with the changes in business conditions to tune the J2EE applications deployed in the J2EE server.

This subsection provides an overview of the operations of J2EE applications.

5.2.1 Canceling a timeout request

When you cannot control a request that is being executed in a J2EE application because of causes such as infinite loop, you need to cancel the request forcefully and release the resources. You can cancel the incomplete request at the scheduled time by monitoring the execution status of methods in the J2EE application, in advance.

5.2.2 Stopping a J2EE application

For a J2EE application that provides services to a client only for a specific time in a day, you need to stop the J2EE application every day at a fixed time and restart it on the following day. Further, when maintaining services that are continuously operated for 24 hours, you need to determine the operations that can provide continuous services by stopping only some operations instead of stopping all J2EE applications at the same time. When stopping a J2EE application, the method of first stopping the receipt of requests, processing the requests that are already received and then stopping the J2EE application is called *service lock*. When you stop a J2EE application during daily operations, you can stop the services safely by executing an appropriate service lock. The way of service lock that you can execute differs based on the configuration of J2EE applications and the system configuration.

5.2.3 Forced termination of J2EE applications

In the case of terminating a J2EE application by normal operations or when you want to terminate a J2EE application immediately when an error occurs, the processing of a request in the J2EE application may not be finished, as a result, the termination processing also may not be finished. In such cases, you need to forcefully terminate the J2EE application.

In forceful termination of J2EE applications, the incomplete requests are forcefully terminated. As a result, you can terminate the J2EE application.

5.2.4 Replacing J2EE applications

You replace J2EE applications in the case of version up and maintenance of running J2EE applications.

When you do not want to stop the services when replacing J2EE applications, you need to execute an appropriate service lock. The CTM will be used to control the request from the client to the J2EE server, and the J2EE application is replaced without stopping the entire system. As a result, the J2EE application is replaced in the online status while accepting the requests from the client. Note that CTM can control only the requests for remote interface invocation of the Stateless Session Bean that uses the RMI-IIOP communication.

The CTM can only be used in the products that contain Cosminexus Component Transaction Monitor in the component software. For available products, see *2.2.1 Relationship of products and component software* in the manual *uCosminexus Application Server Overview*.

5.2.5 Accessing network resources from J2EE applications

You can access the path of network resources from J2EE applications using a UNC or a network drive. Note that this functionality is used in Windows. In UNIX, you can access network resources from J2EE applications without any specific settings.

5.2.6 Operations and references for J2EE applications

Tables 5-2 to 5-5 describe the operations of J2EE applications and the corresponding reference section of each operation. Note that the network resources or network drives can be accessed from J2EE applications only if the settings are specified beforehand; therefore, no tasks are performed during operations.

Table 5–2: Monitoring and canceling the execution time of J2EE application

Operation contents	Measure	Operation overview	Reference manual	Reference
Confirming the execution status of a J2EE application	Server management command (cjlistthread)	Confirms the execution status of a running J2EE application.	This manual	5.3.11
Canceling the request for which a timeout has occurred	Server management command (cjstopthread)	Cancels the request that is not finished from amongst the requests running in a J2EE application.		5.3.12

Table 5–3: Locking a J2EE application

Operation contents	Measure	Operation overview	Reference manual	Section
Service lock using a load balancer	Functions of load balancer	Locks a service with the following methods, when a load balancer is used in a Web front: <ul style="list-style-type: none">• Changes the request distribution destination of a load balancer• Stops the load balancer• Stops only a part of the Web application that configures the service	This manual	5.4.3
Locking a service that uses CTM	Management command (mngsvrutil)	In a back-end system that uses CTM, when you want to stop the J2EE applications in a host at one time or you want to stop the J2EE applications sharing a queue at one time, directly lock the schedule queue of the J2EE applications.		3.7.4

Table 5–4: Stopping a J2EE application

Operation contents	Measure	Operation overview	Reference manual	Section
Stopping a J2EE application	Server management command (cjstopapp)	Performs normal termination or forced termination of a J2EE application. You can set a timeout for normal termination, and can also execute forced termination automatically.	This manual	5.5.7

Table 5–5: Replacing and maintaining a J2EE application

Operation contents	Measure	Operation overview	Reference manual	Section
Replacing a J2EE application	Server management commands (cjstopapp, cjimportapp#)	Stops a J2EE application, and replaces with a new application. After replacing, restarts the J2EE application.	This manual	5.6.3(1)
Replacing a J2EE application by re-deploying	Server management command (cjreplaceapp)	Replaces a J2EE application by re-deploying. You can replace a J2EE application in archive format with fewer procedures.		5.6.3(2)
Replacing a J2EE application by reloading	Server management command (cjreloadapp)	Replaces a J2EE application by re-loading. You can replace a J2EE application in exploded archive format with fewer procedures.		5.6.3(3)
Replacing a J2EE application to be executed after pre-compiling JSPs	Server management command (cjstartapp) or cjjspc command	Uses the JSP pre-compile function, and replaces a J2EE application after pre-compiling JSPs.		5.6.3(4)
Renaming a J2EE application	Server management command (cjrenameapp)	Renames an application, and simplifies the management of a J2EE application before and after replacing.		5.6.3(5)
Replacing a J2EE application that uses CTM in an online status	Management command (mngsvrutil)	Closes the outlet of the schedule queue and then replaces the J2EE application.	Expansion Guide	3.7.2

#

In the case of a WAR application, the cjimportwar command is used.

5.3 Monitoring and Canceling a J2EE Application During Runtime

This subsection describes the functionality for monitoring a J2EE application during runtime, and the method timeout and method cancellation functionality that is provided by the J2EE runtime monitoring functionality.

Important note

Due to the constraints in Java specifications and JavaVM implementation, in some cases you cannot cancel a method. In such cases, the area for which method cancellation cannot be performed is set as a special *protected area* that is not a class unit. If a method cannot be cancelled due to the constraints in Java specifications and JavaVM implementation, the following class name is displayed with detailed information for the message KDJE52718-W:

```
JP.co.Hitachi.soft.jvm.CriticalClass.dummy (Native Method)
```

For details on the protected area, see [5.3.4\(1\) Protected area and non-protected area](#).

The following table describes the organization of this section:

Table 5–6: Organization of this section (Monitoring and canceling the execution time of the J2EE application during runtime)

Category	Title	Reference
Description	Overview of monitoring and canceling the execution time of J2EE application	5.3.1
	Monitoring the execution time of J2EE application	5.3.2
	Method timeout	5.3.3
	Method cancellation	5.3.4
	Example of setting timeout value and the range of setting values	5.3.5
	Thread count to be used	5.3.6
Implementation	Definition in cosminexus.xml	5.3.7
Settings	Settings for the execution environment	5.3.9
Operations	Flow of monitoring and canceling the execution time of J2EE application	5.3.10
	Confirming the execution status of a J2EE application	5.3.11
	Canceling the request for which a timeout has occurred	5.3.12
	The log information that is output by monitoring the execution time of J2EE applications	5.3.13
Notes	Precautions when implementing	5.3.8

5.3.1 Overview of monitoring and canceling the execution time of J2EE application

If an infinite loop occurs internally in a J2EE application, you will not be able to control the operations of the J2EE application. For troubleshooting, set a timeout value in the runtime of the J2EE application such that the control is returned even if the process is not completed after a definite period.

You can use the following two types of functionality to specify such settings:

- Method timeout functionality
- Method cancellation functionality

With the function, a timeout period can be set and monitored in each method of a J2EE application, and the timeout of requests that are not stopped within a fixed period can be detected (the method timeout function). Furthermore, if a timeout occurs, a method can be canceled forcibly (the method cancellation function). At that time, transaction is also forcibly concluded. You can set the method cancellation function such that it is executed automatically.

5.3.2 Monitoring the execution time of J2EE application

The functionality for monitoring the J2EE application execution time monitors the request execution time of the EJB and Web applications. If an error such as, infinite loop, is detected in the J2EE application through this functionality, the user can be notified by timing out the method process that did not terminate within the fixed time. You can also use the method cancellation functionality to terminate the transactions forcibly and cancel the methods that are in use.

The functionality for monitoring the J2EE application execution time includes method timeout and method cancellation functionality.

Method timeout functionality

This functionality notifies the user by performing timeout of the method processing that did not terminate within a fixed time, from among the requests existing in the monitoring base. The *monitoring base* provides the interface required for implementing the functionality for monitoring the J2EE application execution time and manages the monitoring information.

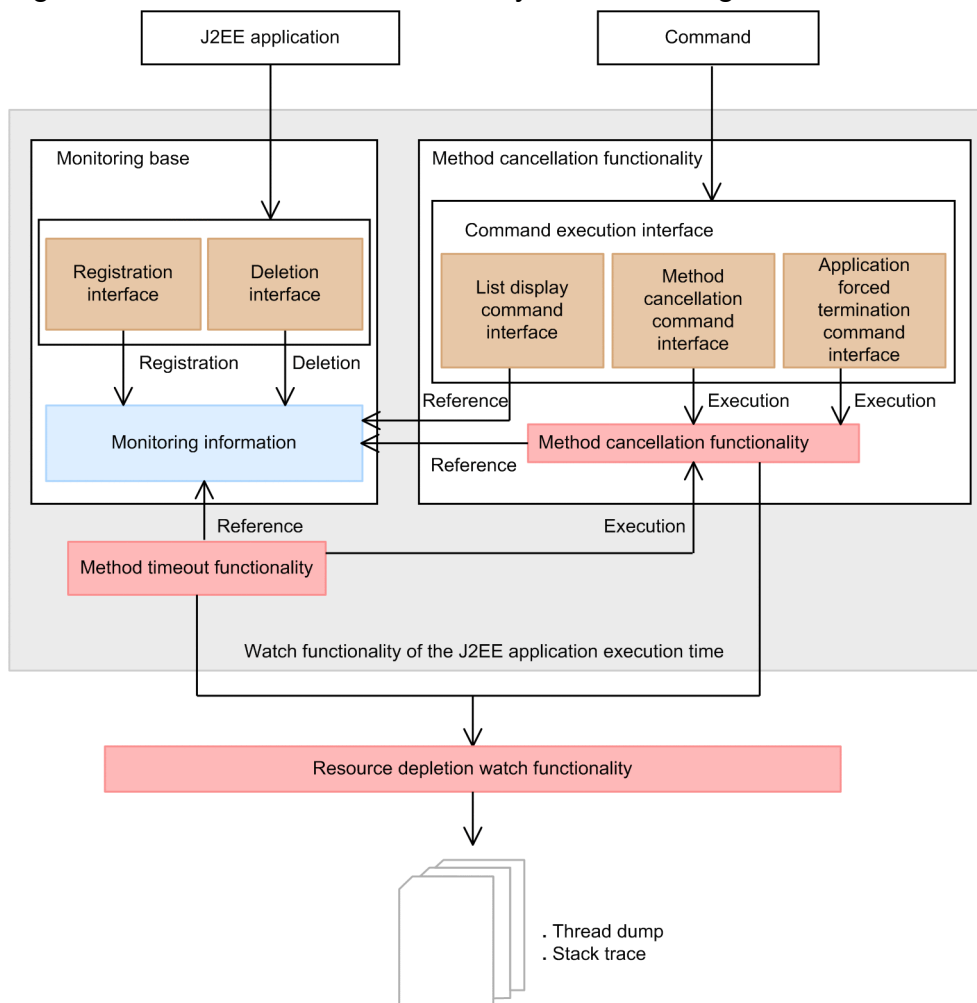
Method cancellation functionality

This function cancels the method after timeout has been notified by the method timeout functionality. The transaction is also concluded forcibly when the database is being accessed at the time of cancellation.

Note that in the method timeout and the method cancellation functionality, the thread dump is output as the log information. The number of thread dump files that are generated can be monitored by the resource depletion monitoring functionality. For the resource monitoring functionality, see [4. Monitoring Resource Depletion](#).

The following figure shows the functionality for monitoring the J2EE application execution time:

Figure 5–1: Overview of functionality for monitoring the execution time of J2EE application



The following points describe the commands in the above figure:

- **List display command**
This command is used to confirm whether it is possible to execute the method cancellation command, before actually implementing it. For the list display command, see *cjlistthread (display thread information)* in the *uCosminexus Application Server Command Reference Guide*.
- **Method cancellation command**
This command is used when timeout is detected or when the thread status is invalid. For the method cancellation command, see *cjstopthread (stop threads)* in the *uCosminexus Application Server Command Reference Guide*.
- **Command for forced termination of application**
This command is used to terminate the application forcibly. For forced termination of applications, see [5.5 Stopping a J2EE application](#). For the command for forced termination of applications, see *cjstopapp (stop J2EE application)* in the *uCosminexus Application Server Command Reference Guide*.

5.3.3 Method Timeout

The method timeout functionality performs timeout of the methods that do not terminate within a fixed time and notifies the same to the user.

(1) Determining a timeout and operations after the timeout

The timeout occurs when the invocation of the method monitored in the monitoring base satisfies the following formula:

$$\textit{Time-when-the-timeout-is-determined} - \textit{Method-start-time} > \textit{Timeout-time}$$

You can select either of the following operations (Method cancellation mode) after the occurrence of timeout.

- Output the KDJE52703-W message
- Output the KDJE52703-W message and execute the method cancellation functionality

In both the cases, KDJE52716-I message is displayed when the timed out invocation terminates.

(2) Target processes of the method timeout functionality

Specify the timeout settings in the target processes of method timeout. The locations set are as follows:

- Request process of the Web application
- EJB method invocation process

Note that the following processes are not included in the scope of method timeout:

- Processes operating when the J2EE applications are deployed and undeployed
- Enterprise Bean processes operating in the status management functionality and the active session timeout functionality of the EJB container

Apart from the method invocation processes of the Enterprise Bean, the processes that the EJB container performs automatically as per the specified settings are not included in the scope of method timeout.

There are times when you cannot perform method timeout depending upon the data structure of the J2EE application. For details, see [5.3.8 Precautions When Implementing](#).

(3) Timeout detection interval and method timeout time

The following settings are required when using the method timeout functionality:

- *Time interval for detecting timeout*

Specify the time interval for detecting the timeout.

The J2EE server checks after every fixed interval of time to determine whether timeout has occurred. This time is set for each J2EE server.

Moreover, as timeout detection is performed at fixed intervals, the maximum amount of time required for detection is as follows:

Formula for calculating the time required for detecting the timeout

$$\textit{Time-taken-for-timeout-detection} = \textit{Timeout-time} + \textit{Interval-of-time-to-detect-timeout}$$

- *Method timeout time for each J2EE application*

Specify the time to be set as method timeout in the J2EE application.

The settings are made for each Enterprise Bean in the J2EE application or each Web application.

For details on the settings for monitoring the execution time of J2EE application, see [5.3.9 Settings for the execution environment](#).

5.3.4 Method Cancellation

The method cancellation functionality cancels the running process that is the cause of timeout. When canceling the method, determine whether the process being currently executed can be cancelled.

Tip

When using a resource adapter compliant with the Connector 1.5 specifications, the following processes cannot be specified for method cancellation:

- Resource adapter specific methods
- Work executed in work management

However, the processes in the Message-driven Bean invoked by Work can be specified for method cancellation.

(1) Protected area and non-protected area

The area where a method can be cancelled is known as a non-protected area and the area where a method cannot be cancelled is known as a protected area.

Protected area is the area where a method cannot be cancelled. The protected area maintains data and area shared by J2EE server operations and guarantees the process executed in JavaVM, and hence, that method cannot be cancelled. The J2EE service, Web container, and EJB container are the protected areas. For details on the contents defined as protected area, see [Appendix C. Contents of the Protected Area List](#). In addition to the classes relevant to the protected area described here, if there are classes you want to handle as protected area, you describe the classes in the *Protected area list file*.

On the other hand, the non-protected area is where the methods can be cancelled. A J2EE application is a non-protected area.

The judgment of protected area and non-protected area is performed for each class. However, there are times when non-protected area is handled as a protected area depending upon the runtime conditions. Even a J2EE application is determined as a protected area, when a native method is invoked or when a static initializer is executed. Furthermore, from the wait method of the `java.lang.Object` class is determined as an exceptional non-protected area, only in the cases where the method is not invoked in the extension of the static initializer.[#]

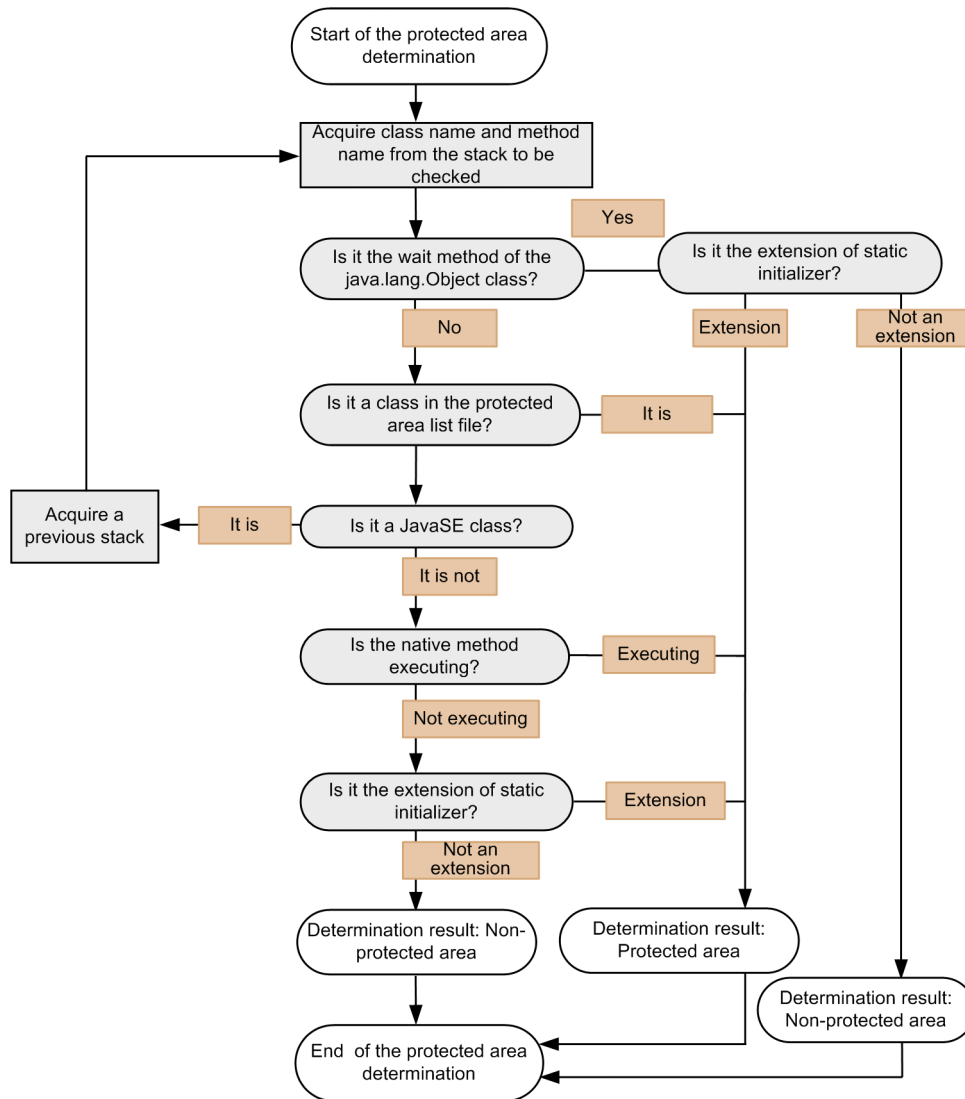
[#] In versions subsequent to the wait method of the `java.lang.Object` class acquires the thread monitor (lock), when the process is restarted due to the lapse of the specified time or `notify/notifyAll`. If the protected area is determined while waiting for acquiring the monitor, the area is determined as a non-protected area, but cancellation is actually executed immediately after the thread monitor is acquired.

(2) Method cancellation processing

A method can be cancelled only when the process currently executed is in the non-protected area. To confirm whether a method can be cancelled, when determining the protected area, determine whether the method that is to be cancelled is being executed in the non-protected area. Method is cancelled when the process being executed is in the non-protected area. In case of a protected area, the process of determining the protected area is retried at fixed intervals. Whenever the area is determined as a protected area in the determining process, the KDJE52718-W message is displayed. When the control does not move to non-protected area in a fixed time, it is considered that method cancellation has failed and the method cancellation process terminates.

The following figure shows the flow of the process for determining the protected area:

Figure 5–2: Process for determining the protected area



Note that the following processing is executed during the method cancellation:

- When a transaction is started, the transaction is timed out forcibly and is marked for rollback.
- When an SQL statement is executing and the statement cancellation functionality is enabled, the SQL statement is cancelled.

The above mentioned processing is executed even when the method cancellation is not performed because the protected area is executing.

For details on the transaction timeout, see *3.15.8 Transaction timeout and statement cancellation* in the *uCosminexus Application Server Common Container Functionality Guide*.

(3) Operations during method cancellation in Web container and EJB container

If a method is cancelled, `ThreadDeath` occurs in the threads executing the methods that are to be cancelled. In the Web container and EJB container, catch this `ThreadDeath` and implement the required processes.

The following section describes the operations when canceling a method in the Web container and EJB container:

(a) Operations in the Web container

The operations differ depending upon the timing when `ThreadDeath` is thrown.

When `ThreadDeath` is thrown during the request process in Web application filter, servlet, or JSP

The `javax.servlet.ServletException` is thrown in the calling filter, servlet, or JSP.

For example, when the servlet is executed in the `doFilter` method invocation extension of `javax.servlet.FilterChain` from filter and `ThreadDeath` occurs during the execution of that servlet, `javax.servlet.ServletException` is thrown in the invocation of the `doFilter` method.

Similarly, when the `forward` method or the `include` method of `javax.servlet.RequestDispatcher` is invoked to forward a request to the servlet or JSP, `ThreadDeath` object is thrown during the `getRootCause` method of `javax.servlet.ServletException`.

When `ThreadDeath` is thrown during the listener process of the Web application

The Web container catches `ThreadDeath`, but an exception is not thrown for the user program because of which the event that led to the listener is invoked.

For example, when `HttpSession` is created by invoking the `getSession` method of `javax.servlet.http.HttpServletRequest`, the `sessionCreated` method of `javax.servlet.http.HttpSessionListener` is invoked. When `ThreadDeath` is thrown during the execution of this `sessionCreated` method, the Web container catches the thrown `ThreadDeath`, but exception is not thrown when invoking the `getSession` method where the event of `HttpSession` generation has occurred.

(b) Operations in EJB container

When `ThreadDeath` is thrown during the EJB method invocation, operations are the same as those when system exception defined in EJB specifications occurs. In the `getCause` method of the exception returned to calling source, the `ThreadDeath` object is returned.

(4) Timing of Method cancellation execution

This section explains the timing for canceling the method and the maximum time taken until the method is cancelled.

- **When timeout occurs**

In the method cancellation mode for method timeout, when stop thread is set, the method is cancelled when timeout occurs. In the case of settings for message output only, method is not cancelled even when timeout occurs.

The cancellation process of the method where timeout occurs is performed at fixed intervals. Moreover, the method cancellation process is executed asynchronously in a thread that is different from the thread that examines (monitors the timeout) whether the request process in operation has timed out. Hence, the maximum time required from the detection of timeout until the cancellation of method will be equal to the interval of method cancellation process.

The time interval of the method cancellation process is the same as the time interval for monitoring the timeout. For the time interval settings of the method cancellation process, see [5.3.9 Settings for the execution environment](#).

- **When the method cancellation command is executed**

The method is cancelled when method cancellation command is executed during operation.

You check the status of the thread where the timed out method is running, confirm that the method can be cancelled, and then execute the method cancellation command, as and when required.

For details on the procedure for executing method cancellation, see [5.3.10 Flow of monitoring and canceling the execution time of J2EE application](#).

The method cancellation process is executed asynchronously after command execution. Moreover, the method cancellation process does not influence the time interval of the method cancellation process.

- **When forced termination of J2EE applications is executed**

When a J2EE application is terminated forcibly, method cancellation is executed when there are running threads in the J2EE application that is to be stopped.

For details on the forced termination of an application, see [5.5.4 Forced termination process](#).

The method cancellation process is executed asynchronously after command execution. Moreover, the method cancellation process does not influence the time interval of the method cancellation process.

(5) Notes on method cancellation

During method cancellation, the local variable information is output to the stack trace by default.

Output of the local variable information to the stack trace might affect the performance, so we recommend that you specify `-XX:-HitachiLocalsInStackTrace` in the JavaVM extension option so that the local variable information is not output to the stack trace during method cancellation.

5.3.5 Example of setting timeout value and the range of setting values

This section provides an example to set the timeout value and the range of the settings value when using the functionality for monitoring the J2EE application execution time for Web applications and for EJB.

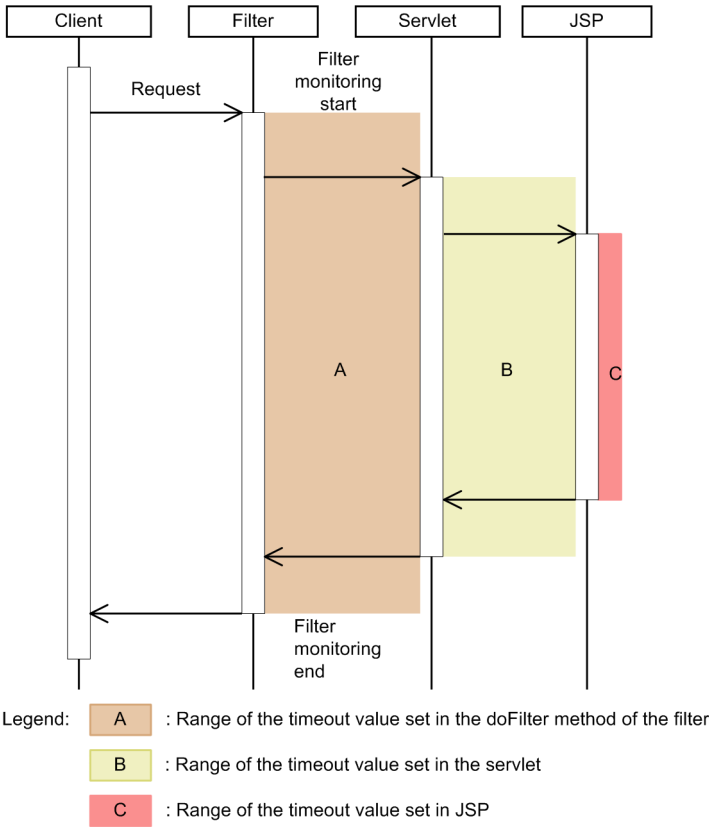
(1) In the case of Web application

The following sections explain the settings of the timeout value for the Web applications:

(a) Invocation of filter, servlet, or JSP

The following figure shows the timeout settings for filter, servlet, or JSP invocation:

Figure 5–3: Timeout settings for filter, servlet, or JSP invocation



You need to set the timeout value for each location appropriately after considering the invocation sequence. For example, the timeout value of the `doFilter` method of the range A in the figure is applied to the monitoring of the process time from start of method until its completion. For this reason, when the process time of B or C reaches the time set in timeout value of A, timeout occurs at that time. As shown in the following example:

Example where timeout occurs during processing

It is assumed that the time shown in the following table is set as the timeout value of each of A, B, C in the figure:

Table 5–7: Example of settings 1 for timeout value

Location	Timeout value that has been set
A in the figure (<code>doFilter</code> method of filter)	240 seconds
B in the figure (Servlet)	180 seconds
C in the figure (JSP)	120 seconds

In the case of such settings, the following time is taken for the processing from A to B and the processing from B to C:

- Process from A to B: 120 seconds
- Process from B to C: 60 seconds

Here, if the process in C takes more than 60 seconds, timeout occurs before process completion as the timeout value of A in the figure is reached in 240 seconds.

■ Precautions when specifying the settings

In the extension of processes within the servlets or JSPs performing uploads and downloads, there are times when communication with the client occurs. In such cases, since uploading and downloading are handled as processing of servlets and JSP, the execution time of the method is monitored.

For this reason, you need to set the timeout for the servlets and JSPs performing uploads and downloads after considering the communication delays with the client. Especially for downloads, a dialog is displayed in the browser to confirm the download with the client. In such cases, note that the download is not complete (Servlet or JSP process does not complete) until the user on the client performs operations in the dialog for download confirmation, depending upon the data quantity to be downloaded.

Note that when using a filter, the same precaution is required for the corresponding filter.

(b) Initial request to servlet

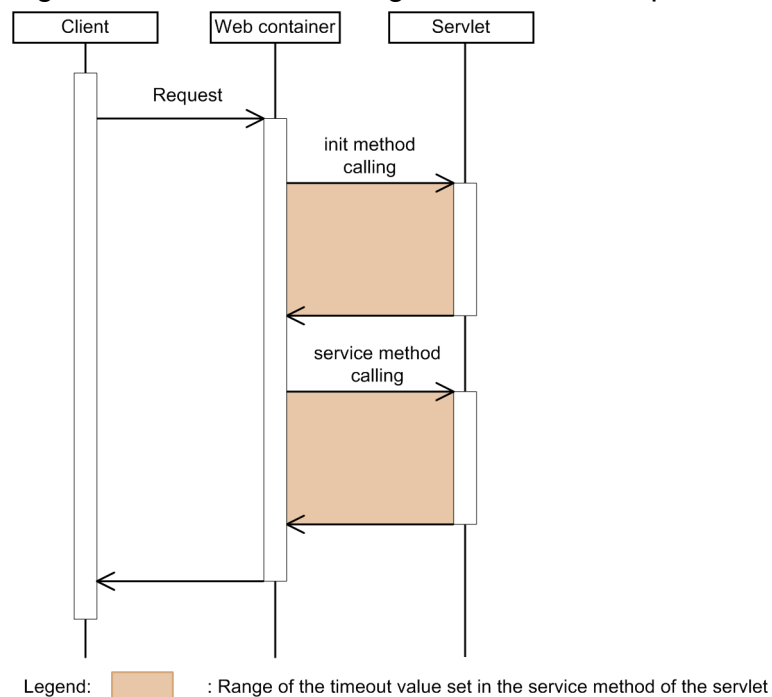
During initial access to a servlet, the timeout value set in the service method of the servlet becomes valid.

However, in the case of a servlet or JSP where the `<load-on-startup>` tag of `DD (web.xml)` is not set, `init` method is executed during initial request. At this time, the timeout value set in the service method of corresponding servlet or JSP becomes valid in the `init` method.

Moreover, even in the destroy method executed when `javax.servlet.UnavailableException` is thrown, indicating that the servlet or JSP cannot be used in the service method, the timeout value set in the service method of the corresponding servlet or JSP is applied.

The following figure shows the timeout settings for the initial request to a servlet:

Figure 5–4: Timeout settings for the initial request to a servlet



(c) Initial request of servlets or JSPs passing through a filter

The following figure shows the timeout settings for the initial request of servlets and JSPs passing through a filter:

Figure 5–5: Timeout settings for the initial request of servlets passing through a filter

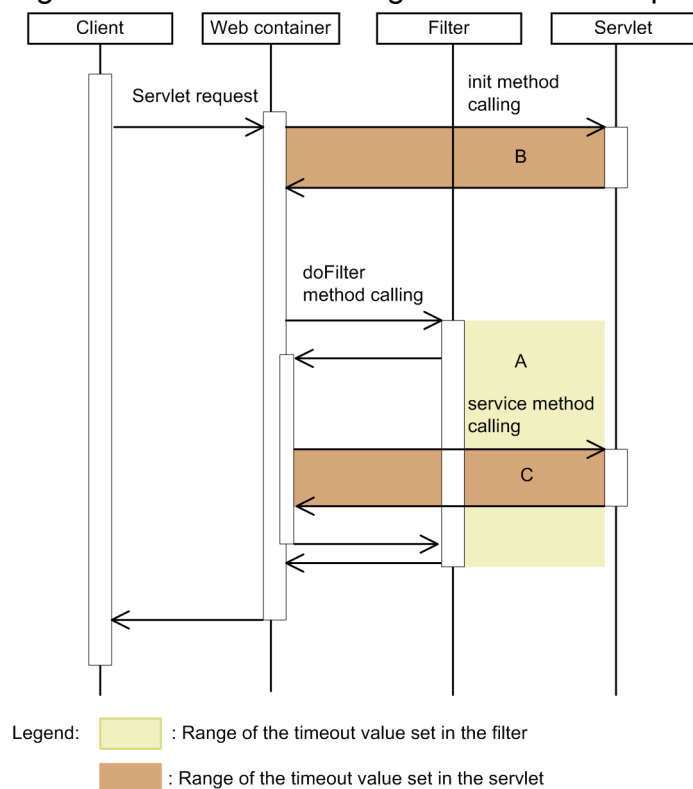
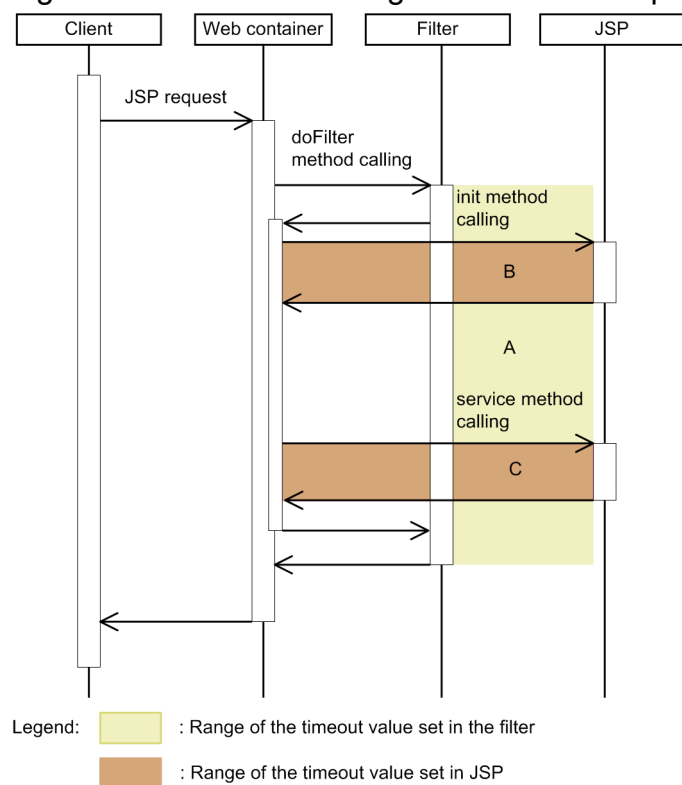


Figure 5–6: Timeout settings for the initial request of JSPs passing through a filter



In the filter process extensions, the filter timeout value when a servlet or JSP is invoked needs to be set after considering the timeout value of servlet or JSP invoked in filter extension.

For example, in case of Figure 5-6, the timeout value of the filter of range A in the figure is applied to filter process time. For this reason, when the process time of B or C reaches the time set in timeout value of A, timeout occurs at that time. The following example shows the case of Figure 5-6:

Example where timeout occurs during processing

It is assumed that the time shown in the following table is set as the timeout value of each of A, B, C in the figure:

Table 5–8: Example of settings 2 for timeout value

Location	Timeout value that has been set
A in the figure (filter)	240 seconds
B and C in the figure (servlet)	180 seconds

For such settings, the following time is taken for the processing in the filter and in the servlet:

- Processing in the filter: 60 seconds
- Processing in the init method of the servlet: 120 seconds

At this time, if the process in the service method of the servlet takes more than 60 seconds, as the timeout value of A in the figure is reached in 240 seconds, timeout occurs before process completion of the service method of the servlet.

Moreover, in the case of servlet or JSP where the `<servlet>` tag of DD (`web.xml`) is defined but the `<load-on-startup>` tag is not set, the init method of servlet or JSP is executed when executing the initial request in the extension of the `doFilter` method of the filter. The following table shows the timing when the `init` method is executed.

Table 5–9: Timing when the 'init' method is executed

Target	Definition of <code><servlet></code> tag	Definition of <code><load-on-startup></code> tag	Timing when the 'init' method is executed
Servlet	Defined	None	Before the <code>doFilter</code> method is called
	None	None	Extension of the <code>doFilter</code> method
JSP	Defined	None	Extension of the <code>doFilter</code> method
	None	None	Extension of the <code>doFilter</code> method

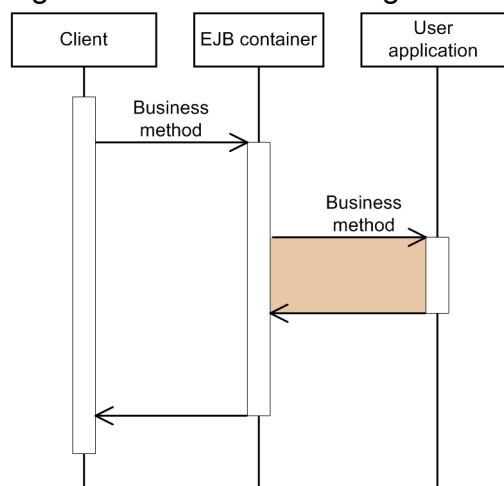
At this time, one of the following values is applied to the timeout value of the `init` method:

- When the *Timeout-value-of-filter* \geq *Timeout-value-of-Web-application*
The timeout value set in the service method of the corresponding servlet or JSP becomes valid.
- When the *Timeout-value-of-the-filter* $<$ *Timeout-value-of-Web-application*
The timeout value set in the filter becomes valid.

(2) In the case of EJB

The following figure shows the timeout settings for business method invocation of Stateless Session Bean. Note that the same settings can also be performed for other Bean types (Stateful Session Bean, Entity Bean, and Message-driven Bean).

Figure 5–7: Timeout settings for business method invocation of Stateless Session Bean



Legend: : Range of the timeout value set in the business method

Timeout occurs if the time set in the timeout value of the business method is exceeded.

Further, since the callback invoked in the EJB method invocation extension is a part of the business process, consider the execution time of the `callback` method when setting the timeout value. For details on the callback invoked in each method, see [5.3.8\(2\) Setting the timeout value in the method invocation process of the Enterprise Bean](#).

5.3.6 Thread count to be used

The following table describes the generated threads and the thread count, when using the functionality for monitoring the J2EE application execution time:

Table 5–10: Generated threads and the thread count

Threads	Number of threads
Threads for method timeout monitoring [#]	1
Threads for canceling the method by using timeout	4
Threads for canceling the method by using commands	4
Threads for fetching the current time	1

#

These threads examine whether the request process in operation has timed out.

Further, the threads for canceling the method using commands are generated when the method cancellation command and the application forced termination command are executed, and are deleted when the command execution is completed. Other threads are generated with the start of the J2EE server and run until the J2EE server is stopped.

5.3.7 Definition in `cosminexus.xml`

Specify the definition for using the functionality for monitoring the execution time of J2EE applications in the `<war>` tag of `cosminexus.xml`.

The following table describes the functionality for monitoring the execution time of J2EE application defined in `cosminexus.xml`:

Table 5–11: Definition of the functionality for monitoring the execution time of J2EE application in `cosminexus.xml`

Items		Tag to be specified	Setting contents
Setting the method timeout	Settings in each EJB	<code><ejb-method-observation-timeout><method-observation-timeout></code> tag under <code><session></code> , <code><entity></code> , or <code><message></code> tag	Specify the value for method timeout.
	Settings in each Web application	<code><war><servlet><method-observation-timeout></code> tag or <code><war><filter><method-observation-timeout></code> tag	Specify the method timeout value common to the methods in the servlet.
Setting the mode of method cancellation		<code><cosminexus-app><method-observation-recovery-mode></code> tag	Specify the mode of method cancellation.

For details on the tags to be specified, see *2.2 Details of each property specified in the Cosminexus application property file (`cosminexus.xml`)* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

5.3.8 Precautions When Implementing

This subsection describes the settings and precautions when implementing the method timeout functionality and method cancellation functionality.

(1) Setting the timeout value in the processing of the Web application requests

In the processing of the Web application requests, you can specify timeout values for the methods listed in the following table. For the range of the timeout values for each method, see *5.3.5 Example of setting timeout value and the range of setting values*.

Define the servlets and JSPs that uses timeout in the `<servlet>` tag of DD (`web.xml`).

Table 5–12: Targets applicable for method timeout (Processing of the Web application requests)

Case	Method	Conditions
Servlets or JSPs	<code>service</code>	No conditions.
	<code>init</code>	Method timeout is applied to servlets (JSP when JSP is used) where the <code><load-on-startup></code> tag is not defined in DD(<code>web.xml</code>), only when the method timeout is executed for the initial request.
	<code>destroy</code>	Method timeout is applied only when the method throws <code>javax.servlet.UnavailableException</code> . This exception indicates that the servlet is permanently unavailable when the service method is invoked.
Filter	<code>doFilter</code>	No conditions.

When the servlet is implemented as a sub class of the `javax.servlet.http.HttpServlet` class and the `doXXX` method (`doGet` method or `doPost` method) invoked from the service method of the `javax.servlet.http.HttpServlet` class is overridden, a timeout is applicable to the overridden `doXXX` method.

The JSP service method indicates the service method of the class where the `javax.servlet.jsp.JspPage` interface generated from JSP is implemented.

The `_jspService` method defined in JSP is executed with the `JspPage` service method extension from the implementation class of the `JspPage` interface. If `JspPage` is not specified in JSP, the implementation class of the `JspPage` interface provided by the Web container is used.

(2) Setting the timeout value in the method invocation process of the Enterprise Bean

In the method invocation process of the Enterprise Bean, you can specify the timeout value for each method. You cannot specify the timeout value for each Enterprise Bean.

[Table 5-13](#) lists the methods to which method timeout is applied in the methods for invoking the J2EE applications, and [Table 5-14](#) lists the methods to which method timeout is applied in the callback methods implemented in the J2EE application.

As in the case of the `find` method, the `select` method in CMP2.0, and the `find` method in CMP1.0, the method timeout is not applied to the methods that are not implemented in Java programs. Also, the method timeout is not applied to the callback methods (such as `ejbActivate` or `ejbLoad`) that determine whether to call a method depending on the status when the method was invoked.

Table 5–13: Targets applicable for method timeout (Enterprise Bean invocation)

Interface	Method	Stateless Session Bean	Stateful Session Bean	Singleton Session Bean	Entity Bean (BMP)	Entity Bean (CMP2.0)	Entity Bean (CMP1.1)	Message-driven Bean
Home interface	<code>create</code>	N	Y	--	Y	Y	Y	N
	<code>finder</code>	--	--	--	Y	N	N	--
	<code>select</code>	--	--	--	--	N	--	--
	<code>home</code>	--	--	--	Y	Y	--	--
Component interface	<code>remove</code>	N	Y	--	Y	Y	Y	N
	Business method	Y	Y	--	Y	Y	Y	--
Business interface	Business method	Y	Y	A	--	--	--	--
	Business method (asynchronous)	A	N	A	--	--	--	--
<code>javax.jms.MessageListener</code>	<code>onMessage</code>	--	--	--	--	--	--	Y
Any message listener interface (EJB 2.1 or later)	Any message	--	--	--	--	--	--	Y

Interface	Method	Stateless Session Bean	Stateful Session Bean	Singleton Session Bean	Entity Bean (BMP)	Entity Bean (CMP2.0)	Entity Bean (CMP1.1)	Message-driven Bean
	listener method							

Legend:

Y: Applied.

N: Not applied.

A: A method timeout is applied, but method cancellation is not applied.

--: Not applicable.

Table 5–14: Targets applicable for method timeout (Callback method)

Method	Stateless Session Bean	Stateful Session Bean	Singleton Session Bean	Entity Bean (BMP)	Entity Bean (CMP2.0)	Entity Bean (CMP1.1)	Message-driven Bean
Constructor	N	N	N	N	N	N	N
setSessionContext/ setEntityContext/ setMessageDrivenContext	N	N	N	N	N	N	N
unsetSessionContext/ unsetEntityContext	N	N	N	N	N	N	--
ejbCreate<Method>	N	Y	--	Y	Y	Y	N
ejbPostCreate<Method>	--	--	--	N	N	N	--
ejbRemove [#]	N	Y	--	Y	Y	Y	N
ejbActivate	N	N	--	N	N	N	--
ejbPassivate	N	N	--	N	N	N	--
ejbLoad	--	--	--	N	N	N	--
ejbStore	--	--	--	N	N	N	--
ejbfind<Method>	--	--	--	Y	--	--	--
ejbSelect<Method>	--	--	--	--	--	--	--
@PostConstruct	N	Y	A	--	--	--	--
@PreDestroy	N	Y	A	--	--	--	--
home method	--	--	--	Y	Y	--	--
Business method	Y	Y	A	Y	Y	Y	--
onMessage	--	--	--	--	--	--	Y
Any message listener method (EJB 2.1 or later)	--	--	--	--	--	--	Y

Legend:

Y: Applied

N: Not applied

A: A method timeout is applied, but method cancellation is not applied.

--: Not applicable

#

Method timeout is not applied when the method is invoked by the container processes such as the timeout functionality.

(3) Option message settings when the method timeout occurs

When method timeout occurs, you can add an optional character string in the detailed part of the message that notifies the timeout by setting an option in advance. The following is the execution example.

```
// User processing
...
// Addition of a message
RequestMonitorMessage.setMessage("Add a message.");
...
// User processing
```

You cannot use this method in applications that are not monitored such as the followings:

- EJB client applications
- Applications operating in threads generated and started by J2EE applications

(4) Precautions during application development using method cancellation

When using method cancellation, you must be careful about the data structure of the application when you develop the application.

If you execute method cancellation, the execution of the thread is interrupted at an unexpected location. As a result, you must not use method cancellation for an application with the shared data that is updated or deleted using the threads other than the threads targeted for method cancellation. The shared data is destroyed and might affect the processing of other requests.

For example, suppose a shared area (variables A=1, B=2) is set for an Enterprise Bean and is accessed by multiple threads (thread 1 and thread 2), if the `ThreadDeath` exception occurs while thread 1 is updating (A=10, B=20) the shared area, thread 1 will end with the half-finished (A=10, B=2) update. At this time, if thread 1 references the half-updated data with thread 2 and calculates A*B, the result of the calculation will be invalid (Right: 200, Wrong: 20). As a result, you cannot use method cancellation in such examples.

Furthermore, if the optimize local invocation functionality is used, you must check that the data has such a structure so that the update and delete operations are executed using multiple threads within or across applications.

5.3.9 Settings for the execution environment

To use the functionality for monitoring the execution time of J2EE application you must specify settings for the J2EE server and J2EE applications.

You reference the J2EE application settings only to set or change the properties of J2EE applications that do not contain `cosminexus.xml`.

(1) Setting J2EE servers

Implement the J2EE server settings in the Easy Setup definition file. Specify whether to use the functionality for monitoring the execution time of J2EE application in `ejbserver.ext.method_observation.interval` parameter in the `<configuration>` tag of the logical J2EE server (`j2ee-server`) in the Easy Setup definition file. If you specify 0, the functionality for monitoring the execution time of J2EE application will be disabled. If you specify a valid value of 1 or more, the functionality for monitoring the execution time of J2EE application will be enabled and that value is used as the following time intervals:

- Time interval to check if the processing of a request has timed out
- Time interval to cancel a timed out request (method)

For details on the Easy Setup definition file and the parameters to be specified, see *4.3 Easy Setup definition file* in the *uCosminexus Application Server Definition Reference Guide*.

(2) Setting for J2EE applications

Implement the settings for J2EE applications in the execution environment using the server management commands and the property files. You use the following property files for defining the functionality for monitoring the execution time of J2EE application.

Setting the method timeout

- Settings in each EJB
SessionBean property file, EntityBean property file, or MessageDrivenBean property file
- Settings in each Web application
Servlet property file or filter property file

Setting the method cancellation mode

Application property file

The tags to be specified in each property file correspond to `cosminexus.xml`. For the definitions of `cosminexus.xml`, see *5.3.7 Definition in cosminexus.xml*.

5.3.10 Flow of monitoring and canceling the execution time of J2EE application

For the settings to monitor the execution time of a J2EE application and to automatically cancel a request for which a timeout has occurred, use server management commands. Specify whether to monitor the execution time of a J2EE application, and specify the monitoring time interval by customizing the operation settings of a J2EE server. For details on the settings for monitoring the execution time of J2EE application, see *5.3.9 Settings for the execution environment*.

When method timeout occurs in a J2EE application, and when a thread for which an attempt of method cancellation has failed, exists, you need to use server management commands and manually execute (re-execute) method cancellation.

The following is the procedure for executing method cancellation with server management commands:

Procedure

For executing the method cancellation, when a message indicating that a timeout has occurred or the thread status is invalid is received:

1. Confirm the execution status of the J2EE application (see subsection *5.3.11*)

Perform this task using server management commands.

2. Cancel the request executed in the J2EE application when a timeout occurred (see subsection [5.3.12](#))

Perform this task using server management commands.

Note that the method is canceled even when the J2EE application is terminated forcibly. For details on the forced termination of J2EE applications, see [5.5.7 Stopping a J2EE application](#).

5.3.11 Confirming the execution status of a J2EE application

When a message indicating that a timeout occurred in a method in a J2EE application or a message indicating that the thread status is incorrect due to the failure in method cancellation is output, check the execution status of the running J2EE application. Determine whether method cancellation can be executed according to the confirmation results.

To check the execution status, use the server management commands (`cjlistthread`).

Note that this command can also be used for confirming the stack trace acquired in the case of transition of the thread status. However, a stack trace is not output when the thread status is running. Note that when the thread status is stopping, the stack trace is output multiple times, so the latest stack trace will be output instead of the stack trace of the time of transition.

Also, the threads exceeding the maximum number of concurrent executions specified in the Web container might be displayed in a list.

The execution format and example are described below. For details on the `cjlistthread` command, see *cjlistthread (display thread information)* in the *uCosminexus Application Server Command Reference Guide*.

Execution format

```
cjlistthread J2EE-server-name
```

If you specify the `-detail` option, you can also output the stack trace as details of the running thread information.

```
cjlistthread J2EE-server-name -detail
```

Execution example

```
cjlistthread MyServer -detail
```

The following is the format of an execution result when the `-detail` option is specified:

Execution result

```
Current Time=HH:MM:SS
ThreadID=11111, RootApInfo=RootAP1, Status=timeout, AppName=AP1, StartTime
=HH:MM:SS, TimeOut=60
  com.hitachi.XXXX
  at com.hitachi.YYYY
  at user.code.UserClass1
  at com.hitachi.ZZZZ
  .
  .
  .
ThreadID=22222, RootApInfo=RootAP2, Status=stopping, AppName=AP2, StartTim
e=HH:MM:SS, TimeOut=60
```

```
com.hitachi.xxxx  
at com.hitachi.yyyy  
at user.code.UserClass2  
at com.hitachi.zzzz
```

The thread status is output after Status = of the execution result.

The following table describes the thread status and the possibility of executing method cancellation:

Table 5–15: Thread status and the possibility of executing method cancellation

Thread status	Meaning	Possibility of executing method cancellation
running	The execution time is being monitored. Running normally.	Can be executed.
timeout	A timeout is detected depending on the method timeout.	Can be executed.
stopping	The method cancellation process is being executed.	Cannot be executed.
stopped	The method cancellation process is yet to be completed.	Cannot be executed.
failed	The method cancellation process has failed.	Can be executed.

However, a method cannot be canceled even when the thread status indicates that the method can be canceled, if that method is executed in the *Protected area*. For details on the protected area, see [5.3.4 Method Cancellation](#).

Tip

The thread information output by this command is the snapshot when the `cjlistthread` command is executed. Therefore, the thread information changes every moment. Hitachi recommends you to execute the command multiple times and confirm the information for obtaining the accurate thread status.

Note that if you specify the `-detail` option, the output quantity of the execution result increases. When you specify the `-detail` option, Hitachi recommends you to set a file as the output destination.

5.3.12 Canceling the request for which a timeout has occurred

In a J2EE application, cancel the request for which a timeout has occurred. This process is called method cancellation. Execute method cancellation when method cancellation is possible (running, timeout, or failed) after you confirm the thread execution status using the `cjlistthread` command, according to the contents of [5.3.11 Confirming the execution status of a J2EE application](#).

For method cancellation, use the server management commands (`cjstopthread`). You can cancel the methods of multiple threads by executing the command once.

The execution format and example are described below. For details on the `cjstopthread` command, see *cjstopthread (stop threads)* in the *uCosminexus Application Server Command Reference Guide*.

Execution format

```
cjstopthread J2EE-server-name -tid thread-ID [-tid thread-ID]
```

Execution example

```
cjstopthread MyServer -tid 11111
```

In the thread-ID of the option argument of the `-tid` option, specify a thread ID output to a message indicating that a timeout occurred, or a message indicating that the thread status is invalid.

Note that when you execute this command, if the method cancellation has already been running, or a thread that is specified as a target of method cancellation does not exist, the method cancellation will not get executed and the command will end successfully. You can confirm the status of the thread after executing the command by using the `cjlistthread` command. Confirm with the procedures described in [5.3.11 Confirming the execution status of a J2EE application](#).

5.3.13 Log information that is output while monitoring the execution time J2EE applications

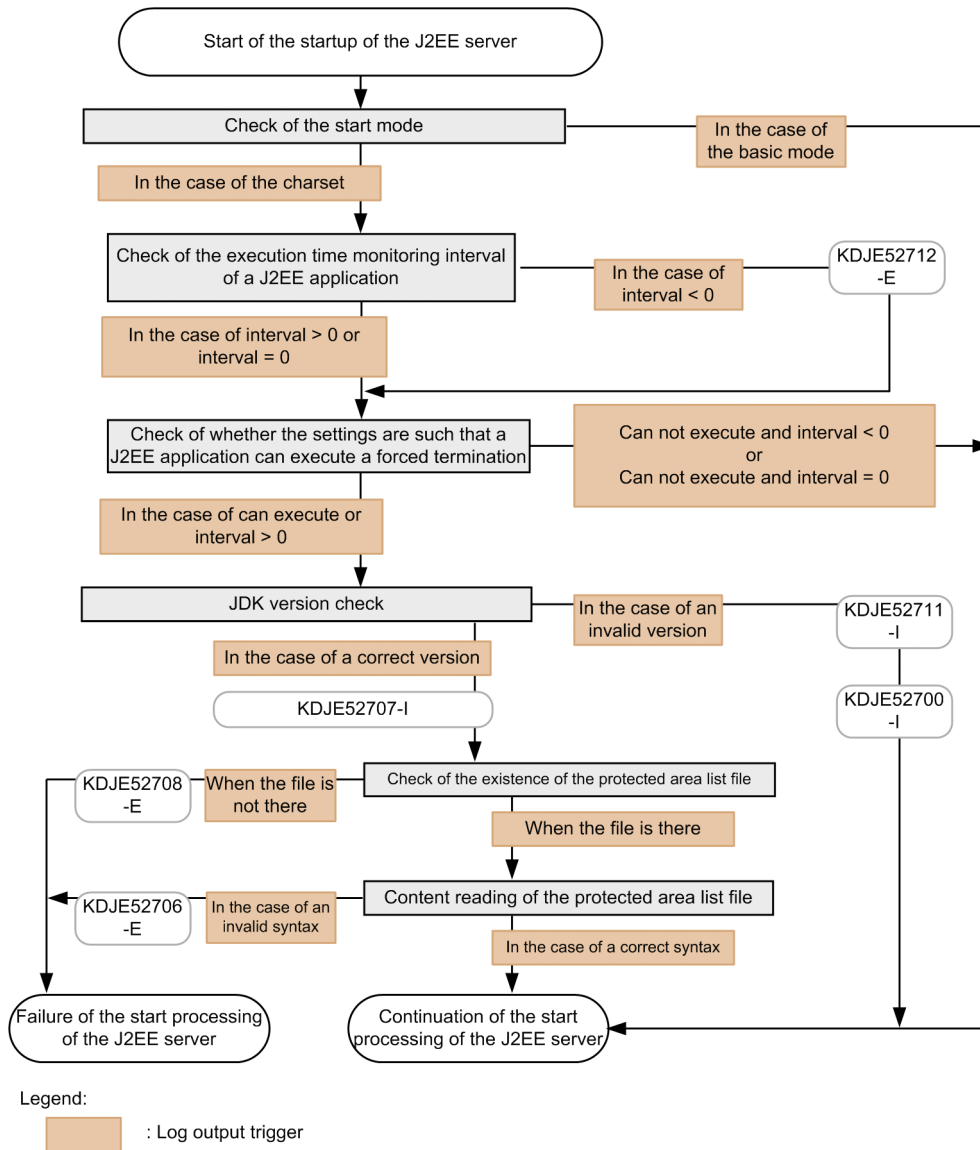
This subsection describes the log information that is output using the execution time monitoring functionality of J2EE applications.

(1) Log information that is output when a J2EE server starts

When a J2EE server starts, reading of properties, checking of JavaVM version, and reading of a protected area list file is executed.

The following figure shows the log information that is output, when a J2EE server starts.

Figure 5–8: Log information that is output when a J2EE server starts



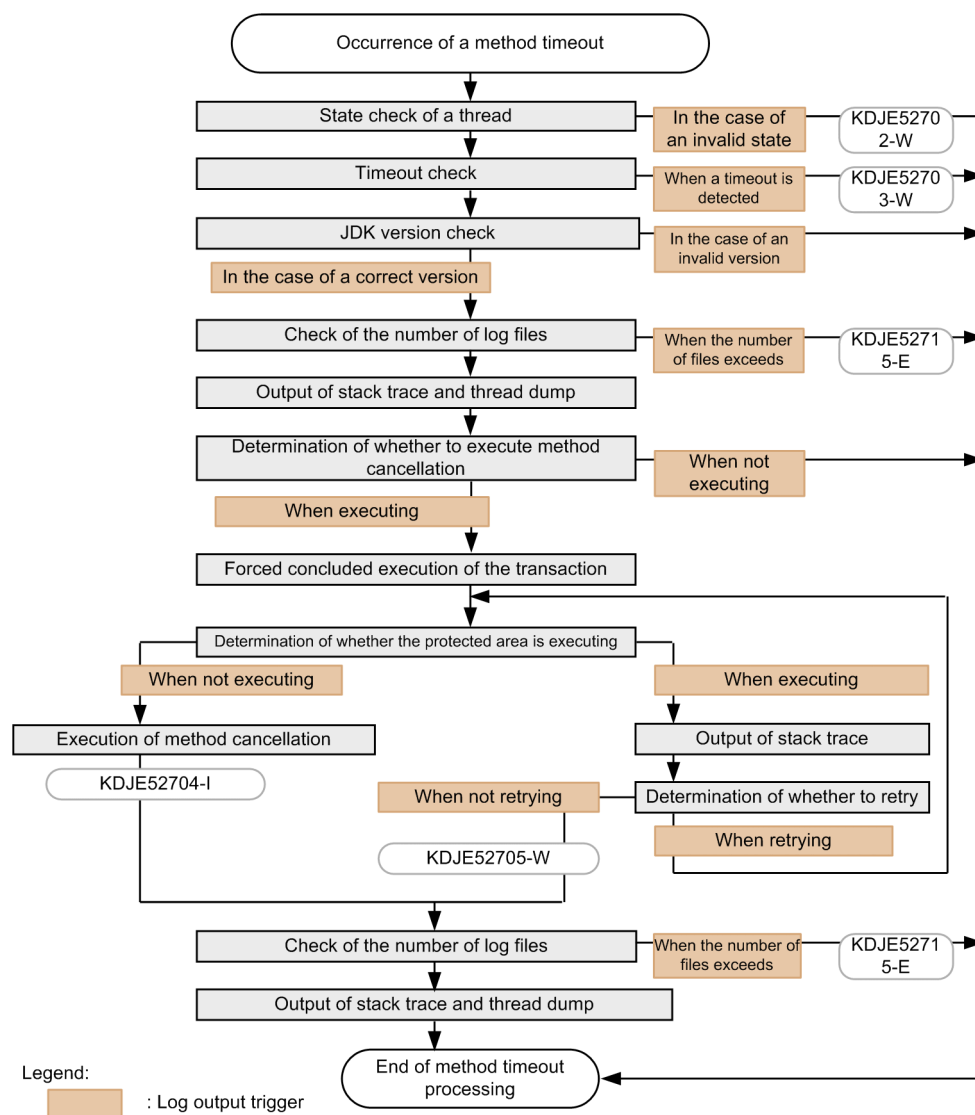
(2) Log information that is output when a method cancellation is executed by the extension of a timeout

The following figure shows the log information that is output when a timeout occurs, and the method cancellation is executed by extending that process. The processing is continued even if any message is output. Note that the output count of the thread dump varies depending on whether the method cancellation mode is valid or invalid.

- If the method cancellation mode is valid
 The log information is output twice; when a timeout occurs and when the method cancellation is successful or failed.
- If the method cancellation mode is invalid
 The log information is output only once, when a timeout occurs.

Also, KDJE52716-I is output, if the method that detects a timeout ends. The KDJE52716-I is output irrespective of whether the method cancellation is executed successfully. The log information is also output even if the method ends without execution. In other words, KDJE52703-W and KDJE52716-I are output in one to one ratio for the execution of one method.

Figure 5–9: Log information that is output when the method cancellation is executed by extending a timeout

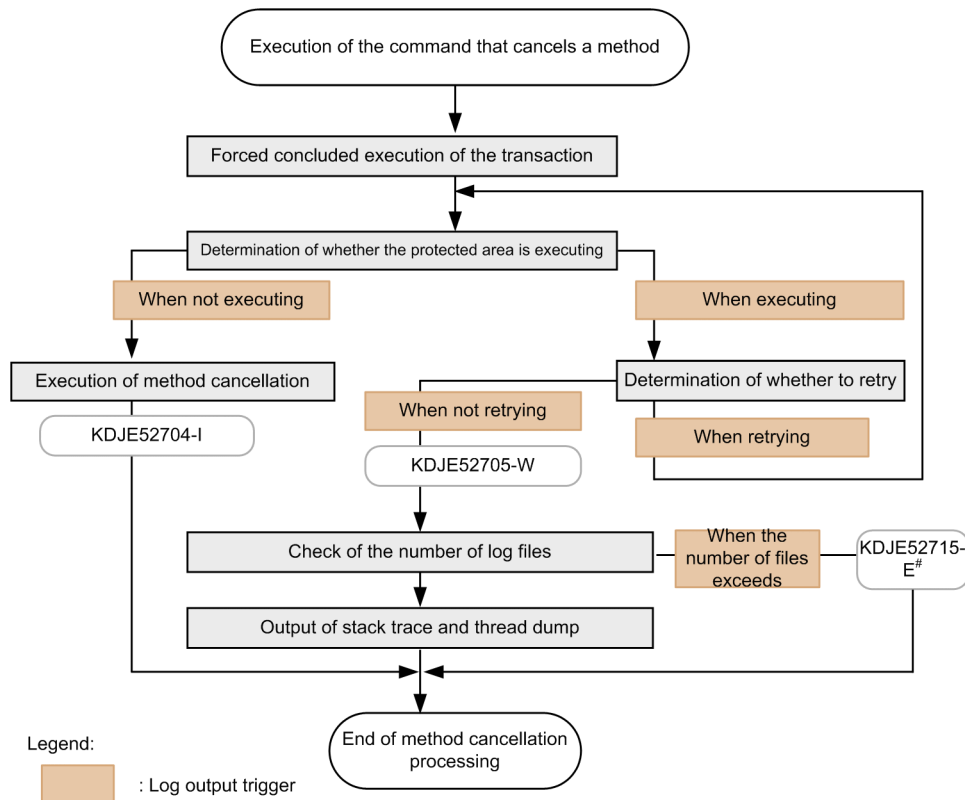


(3) Log information that is output when the method cancellation is executed by a using command

The following figure shows the log information that is output when canceling a method, in which a timeout has occurred, by executing Forced Stop or the method cancellation command (`cj stopthread`) of J2EE applications. The processing is continued, even if any message is output.

Note that if an attempt to cancel the method fails, thread dump is output only once. Also, if the execution of the method cancellation command fails, the KDJE52713-E message is output.

Figure 5–10: Log information that is output when the method cancellation is executed by using a command



#

In the case of the output of more log files than the value specified for `ejbserver.server.threaddump.filenum` in `usrconf.properties`, the "KDJE52715-E" message is output and the stack trace and the thread dump are not output.

(4) Notes

After checking the upper limit of number of the thread dump files of a J2EE server, a time lag occurs because the thread dump output request and the thread dump output processing of JavaVM is executed asynchronously. As a result, thread dump files that are output might exceed the value of upper limit.

The upper limit of the number of thread dumps that are output is `Upper limit of the specified thread dumps + 9`.

Also, with the thread dump output process of JavaVM, if multiple output requests exist simultaneously, you cannot output a thread dump. In such cases, the number of thread dumps is even less than the number that you want to be actually output.

5.4 Locking the J2EE application service

This subsection describes the locking of the Web application service and the locking of the CTM schedule queue.

The following table describes the organization of this section:

Table 5–16: Organization of this section (Locking the J2EE application service)

Category	Title	Reference
Description	Locking the Web application service and releasing the lock	5.4.1
Operations	Methods of locking a service and stopping a J2EE application that can be executed for each system	5.4.2
	Service lock using a load balancer	5.4.3

Note:

Function-specific explanation is not available for "Implementation", "Settings", and "Notes".

A service implies the business provided by the J2EE application. You can lock the services by returning an error for the new request and by not returning an error and continuing the process for a request that is being executed.

5.4.1 Locking the Web Application Service and Releasing the Lock

Locking the Web application service implies locking of services of the J2EE application that fronts the Web application. For example, the Web application service is locked when stopping services during non-business hours and during system maintenance.

(1) Locking the services

The following are the five methods of locking the Web application services:

- Changing the distribution destination of requests in the load balancer^{#1}
- Stopping the load balancer^{#2}
- Stopping the Web server^{#3}
- Stopping the J2EE server
- Stopping the application

#1

When changing the distribution destination of requests in the load balancer, you use the following functionality of the load balancer to continue a request process being executed, without returning an error:

- Functionality that maintains the connection between the load balancer prior to change and the Web server, until the completion of the request process, when changing the distribution destination of request.

#2

When stopping the load balancer, you use the following functionality of the load balancer to continue a request process being executed while the load balancer is stopped, without returning an error:

- Functionality that maintains the connection between the load balancer and the Web server until the completion of the request process, when the load balancer stops

Stopping the Web server is possible only when using Cosminexus HTTP Server as the Web server. When using Microsoft IIS, this method cannot be used since an error is returned to the request being executed.

(2) Service lock in a system configuration using the load balancer

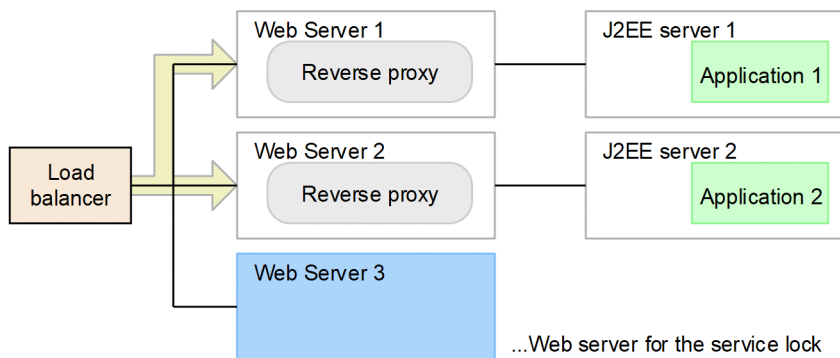
The procedure to lock a service differs depending upon whether a load balancer is used or not. This section explains the locking of a service by changing the distribution destination of the request in the load balancer.

When using the load balancer to change the distribution destination of requests, it is necessary to deploy a Web server for service locks. After locking the running Web applications, set the load balancer to forward requests to the Web server for service locks.

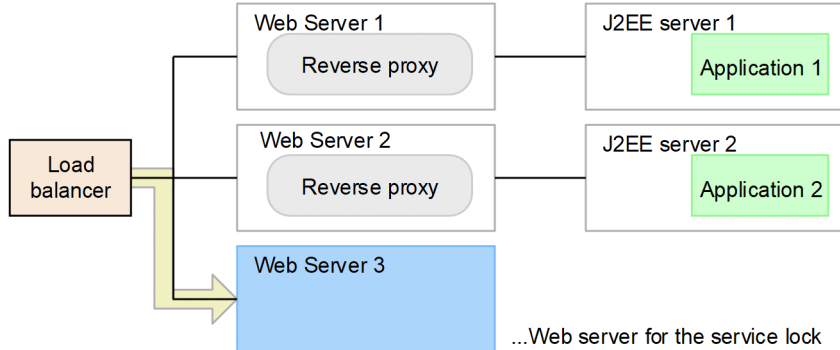
The following figure shows the distribution of requests before and after service lock:


Figure 5–11: Distribution of requests before and after service lock

• Before the service lock



• After the service lock



Legend:  : Request distribution destination

Note that in the Web server for service locks, an error page is returned when a request comes after locking the Web application service.

The displayed error page can also be customized. For the displayed error page, see [\(3\) Displaying the error page](#).

(3) Displaying the error page

An error page is displayed in the client when the services of a locked Web application are accessed. The displayed error page is any one of the following:

- Default error page of the Web server or error page created by the user

- Connection error page[#]
- Error page displaying error status code

#

It is displayed when the HTTP port of the connection destination is closed.

(a) Method of locking the service and the displayed error page

The displayed error page differs depending upon the method of locking the Web application service. The following table shows the method of locking the service and the displayed error page:

Table 5–17: Method of locking the service and the displayed error page

Method of locking the service	Displayed error page
Changing the distribution destination of requests in the load balancer	Error page set in the Web server for service locks
Stopping the load balancer	Connection error page
Stopping the Web server	Connection error page ^{#1}
Stopping the J2EE server	The page displaying any of the following error status codes <ul style="list-style-type: none"> • 404 error ^{#2} • 500 error ^{#3} • 503 error ^{#4}
Stopping the application	The page displaying any of the following error status codes <ul style="list-style-type: none"> • 404 error ^{#5} • 503 error ^{#4}

#1

If the load balancer is used, it depends upon the specifications of the used load balancer.

#2

404 error, displayed when the J2EE server is stopped, is returned for the requests that have reached the J2EE server when the application is stopped, but not reached the pending queue of the Web application.

#3

500 error is returned for the requests that have not yet arrived at the J2EE server when the application stops.

#4

503 error is returned for the requests that are in the pending queue of the Web application when the application is stopped.

#5

404 error, displayed when the application is stopped, is returned for the requests that have not reached the pending queue of the Web application.

(b) Using an error page created by the user

The displayed error page can be customized to a user-created error page. However, whether customization is possible depends on the method of locking the service. The following table indicates the method of locking the service and support for error page customization:

Table 5–18: Method of locking the service and error page customization

Method of locking the service	Customizing the error page
Changing the distribution destination of requests in the load balancer	Y
Stopping the load balancer	--
Stopping the Web server	--

Method of locking the service	Customizing the error page
Stopping the J2EE server	Y [#]
Stopping the application	Y [#]

Legend:

Y: Yes

--: No

#

In the Web server settings for customizing error page, 302 error is returned when full URL (a URL starting with `http://` and specifying the contents of another site) is specified in the `ErrorDocument` directive of the `httpsd.conf` file.

For details about the settings for error page customization, see the manual *HTTP Server*. The error page can be customized only if an HTTP server is used as the web server.

(4) Releasing the service lock

The releasing of the service locks in Web applications differs depending upon the method of locking the service. The methods for releasing service locks are described below for each service locking method:

- Changing the distribution destination of requests in the load balancer
Change the request distribution destination of the load balancer so that the request is forwarded to J2EE server.
- Stopping the load balancers
Start the load balancers.
- Stopping the Web servers
Start the Web server.
- Stopping the J2EE servers
Start the J2EE server.
- Stopping the applications
Start the application.

5.4.2 Methods of locking a service and stopping a J2EE application that can be executed for each system

Execute a service lock for an element *Front*) that is the socket of requests from the client perspective. If you lock the front element, you can continue the execution of the request already received in the J2EE application without reception of new requests from clients.

A Web front system or a back-end system that uses CTM has a load balancer, Web server, or CTM at the front-end of the J2EE application. If you lock an element corresponding to the front of the J2EE application, before directly terminating the J2EE application, you can execute a certain service lock.

Stop the J2EE application to be executed after the service lock with the management command (`mngsvrutil`):

For system types such as the Web front-end system and back-end system, see *3.1.1 Purpose and configuration of the system* in the *uCosminexus Application Server System Design Guide*.

(1) Service lock of a Web front system

A Web front system is a system that receives requests sent from a Web browser that is the front-end, and processes those requests.

The target of service lock is a Web application in the J2EE application.

In this system, you can execute a service lock with the methods described in the following table. You can reliably lock a service by executing the service lock with different methods before terminating the J2EE application.

Table 5–19: Service lock that can be executed in a Web front system

Service lock methods	Description	Measure	Reference
Change the request distribution destination of a load balancer	Changes the settings of a load balancer such that no request is distributed to a J2EE server in which the J2EE application is to be terminated. As a result, the J2EE server cannot receive any request.	Functions of load balancer	5.4.3
Stop the load balancer [#]	Stops a load balancer and stops the transfer of any request to J2EE server. As a result, the J2EE server cannot receive any request.	Functions of load balancer	5.4.3
Terminate a J2EE application normally	If you normally terminate a J2EE application, that J2EE application cannot receive any new request. However, termination is not complete until the running process finishes.	Server management commands	5.5.7

#

You can execute this method for a system using a load balancer.

(2) Service lock of a back-end system (for a system using CTM)

Back-end system is a system for executing common business services in multiple business systems operating at the back of a Web front system.

In the system using CTM, you can execute a service lock for the CTM schedule queue.

In this system, you can execute a service lock with the methods described in the following table. You can reliably lock a service by executing the service lock with different methods before terminating the J2EE application.

Table 5–20: Service lock that can be executed in a back-end system (when CTM is used)

Service lock methods	Description	Measure	Reference manual	Section
Execute a service lock using the CTM functionality	The schedule queue will be locked using the CTM functionality, and forwarding of the requests to the J2EE server will be stopped. As a result, the J2EE server cannot receive any request.	Management commands	<i>Expansion Guide</i>	3.7.4
Terminate a J2EE application normally	If you normally terminate a J2EE application, that J2EE application cannot receive any new request. However, termination is not complete until the running process finishes.	Server management commands	This manual	5.5.7

(3) Service lock of a back-end system (for a system not using CTM)

For a back-end system without CMT, the target of a service lock is a front Enterprise Bean (*Front EJB*) of a J2EE application.

You can execute a service lock with the following methods in this system:

Table 5–21: Service lock that can be executed in a back-end system (when CTM is not used)

Service lock methods	Description	Measure	Reference
Terminate a J2EE application normally	If you normally terminate a J2EE application, that J2EE application cannot receive any new request. However, termination is not complete until the running process finishes.	Server management commands	5.5.7

Note that when building a system, you need to define the front EJB in advance.

5.4.3 Service Lock Using a Load Balancer

This subsection describes how to execute a service lock using a load balancer. You can execute a service lock using a load balancer for a Web front system. There are two methods of service lock using load balancer as follows:

- Normal service lock
- Partial service lock

The following is the description for how to execute each method:

Tip

Execute a service lock using a load balancer when the load balancer is not managed by the Smart Composer functionality. You use the `cmx_stop_target` command, to execute a service lock when the load balancer is managed by the Smart Composer functionality. For `cmx_stop_target` commands, see *cmx_stop_target (Stop a Web system or a service unit)* in the *uCosminexus Application Server Command Reference Guide*.

(1) Normal service lock

Execute a normal service lock using load balancer by either of the following methods:

- Changes the request distribution destination of a load balancer
Prepare a Web server other than the normally used Web server for service lock and switch the request distribution destination to a Web server used for locking the service at the service lock time. In the Web server used for service lock, set such that to display an error page.
- Stops the load balancer
This method stops the load balancer, and then stops the reception of requests. The requests already sent to the Web server are executed as it is.

Note that for lifting the service lock, switch the request distribution destination once again or start the load balancer.

For the procedure, follow the usage methods of the load balancer in use.

(2) Partial service lock

When a Web server that receives a request and a J2EE server that executes the request are made redundant using a load balancer, you can stop only a part of the Web application configuring the service, without stopping the entire service. This locking method is called *Partial service lock*.

Note that a system with the following configuration enables Partial service lock of a Web application:

- System using a load balancer
- Web server and J2EE server are redundant, and Web server and J2EE server are in the ratio of 1:1 or 1:N (N is an integer greater than 1)

You can use the Partial service lock of a Web application by changing the request distribution destination using a load balancer, and not distributing the requests to the node locking the service. For execution methods, confirm the usage methods of the load balancer.

Tip

When executing Partial service lock using a load balancer, the session is disconnected for the request using the session created for a Web server to be stopped. In this case, the session needs to be reconnected for a running Web server. However, when the session failover function is used, part of the information of the session is inherited. For the session failover function, see 5. *Inheriting Session Information Between J2EE Servers* in the *uCosminexus Application Server Expansion Guide*.

5.5 Stopping a J2EE application

This section provides an overview of normal termination and forced termination and their process flow.

The following table describes the organization of this section:

Table 5–22: Organization of this section (Stopping a J2EE application)

Category	Title	Reference
Explanation	Terminating the J2EE application	5.5.1
	Lock process	5.5.2
	Stop process	5.5.3
	Forced termination process	5.5.4
Implementation	Definition in cosminexus.xml	5.5.5
Settings	Settings for execution environment	5.5.6
Operations	Stopping a J2EE application	5.5.7

Note:

The function-specific explanation is not available for "Notes".

5.5.1 Terminating the J2EE application

Stopping a J2EE application means terminating the Web applications and Enterprise Beans that configure a J2EE application, and changing the status of the J2EE application so that the J2EE application does not accept requests from the client. This state is called termination. After this, the J2EE application stops on its own.

This section describes the types of J2EE application termination and the method of termination.

(1) Types of J2EE application termination

The two types of J2EE application termination are as follows:

Normal termination

This is a method for terminating a J2EE application in daily operations. This method stops a Web application and Enterprise Bean in a J2EE application sequentially and stops the service safely. Note that when a request has been executed in the J2EE application, the J2EE application is not terminated until the request is finished. However, when normal termination is executed, you can set a timeout. If you have set a timeout, the control is returned to the client after a fixed period even if the J2EE application is not terminated.

In normal termination, at first the service is locked, then the reception of requests is stopped, and finally the J2EE application is terminated. *Service lock* means executing a request that has already been received and refusing a new request.

For example, in the following cases, if you want a planned termination for a J2EE application, first lock the service and then terminate the J2EE application:

- When a service is provided only for a specific period throughout the day
- When maintaining a system

Forced termination

This is a method for forcefully terminating a request without waiting for the request running in a J2EE application to finish. The resource that is being used is released by forced termination.

When an error occurs in the J2EE application and the control is not returned to the client, you can promptly terminate the service using forced termination.

Tip

You cannot perform forced termination of a J2EE application in the following cases:

- **When a statement cannot be canceled**

When a database is being operated and the processing of a running SQL is not returned, the statement is canceled. *Statement cancel* is the process of canceling the processing of a running SQL.

However, depending on the resource adapter settings, system configuration, and the types of transactions being used, the statement may not be canceled. When a statement cannot be canceled, you cannot forcefully terminate a J2EE application in which the transaction with running SQL has timed out, and a J2EE application in which the SQL process has not returned.

For statement cancellation, see [3.15.8 Transaction timeout and statement cancellation](#) in the *uCosminexus Application Server Common Container Functionality Guide*.

- **When method cancellation cannot be executed**

On executing forced termination, the executing threads are interrupted at any unexpected place due to method cancellation. As a result, you cannot use method cancellation for a J2EE application with the shared data that is updated or deleted using the threads other than the threads targeted for method cancellation.

For the configuration of J2EE applications for which method cancellation cannot be executed, see [5.3.4 Method Cancellation](#).

- **When 0 (zero) is set as the timeout value in the settings for de-activating the schedule queue used for undeploying a J2EE application in an environment with CTM**

When 0 is set as the timeout value of the process for de-activating the CTM schedule queue, locking by CTM does not finish until the request is complete. Therefore, you cannot terminate the J2EE application.

For details, see [3.7.4 Locking and controlling requests for a schedule queue](#) in the *uCosminexus Application Server Expansion Guide*.

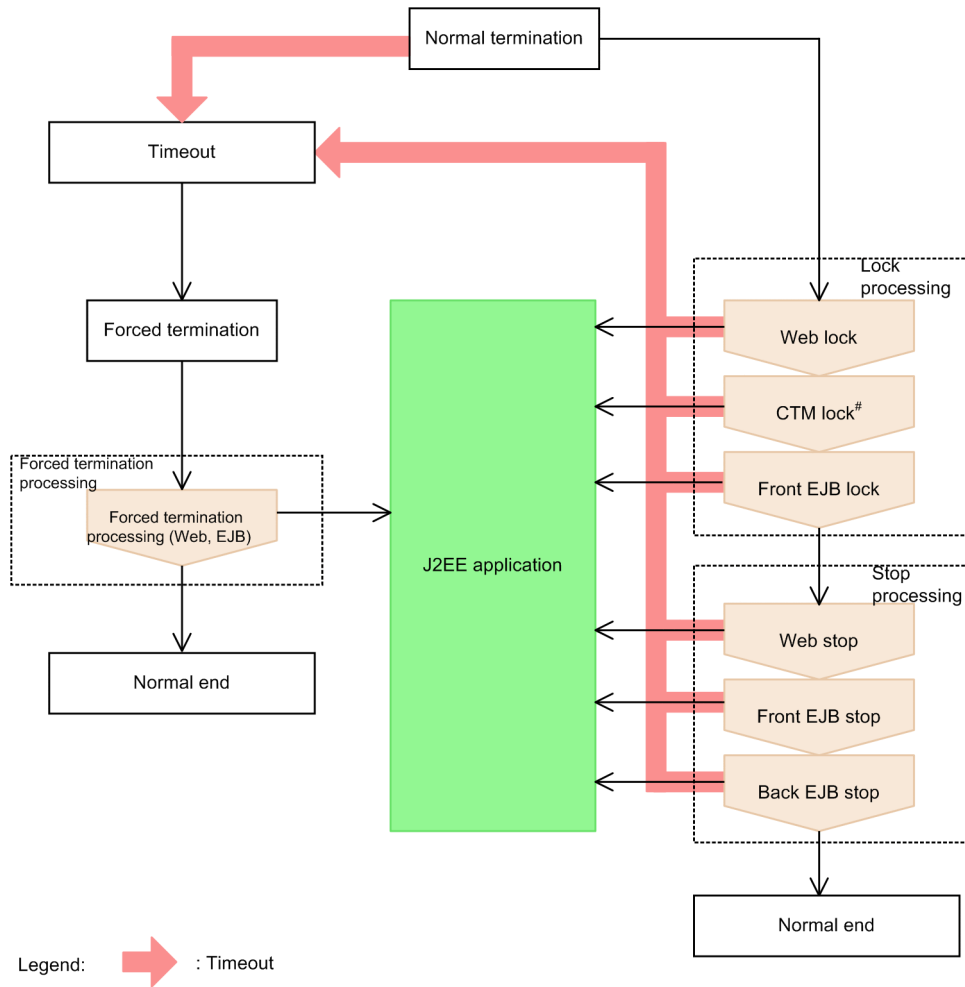
(2) Termination methods

Execute normal termination and forced termination using the server management commands. For details on stopping a J2EE application, see [5.5 Stopping a J2EE application](#).

(3) J2EE application termination process flow

This section explains the J2EE application termination process flow. The following figure shows the process flow of normal termination and forced termination:

Figure 5–12: Flow of J2EE application termination process



When there is no CTM integration, CTM lock is not applied.

Lock process and termination process are implemented in the normal termination process.

The following sections explain lock process, termination process, and forced termination process.

5.5.2 Lock process

In the *lock process*, from among the elements configuring the J2EE applications to be locked, lock the front part that receives requests, thus stopping the receipt of new requests.

Note that the request being processed is processed uninterruptedly.

(1) Lock sequence and process contents

Lock procedure:

1. Lock process of the Web application
2. Lock process of the CTM schedule queue[#]
3. Lock process of the front-end EJB

#

If the system is not integrated with CTM, the lock process of the CTM schedule queue does not exist.

The following sections explain the processing in each lock process:

(a) Lock process of the Web application

The following processes are executed:

- Terminate the receipt of new requests.
- Continue processing the requests that are being processed.
- Among the requests exist in the queue that controls the number of concurrently executed requests of the Web application, the requests for which processing has not started in the Web container are not processed. All the processes return the 503 error to the client.

(b) Lock process of the CTM schedule queue

The following processes are executed:

- Terminate the receipt of new requests.
- Among the requests entered in the CTM schedule queue, only the requests distributed to the J2EE server are processed. The exception `java.rmi.RemoteException` is thrown to the client.
- Among the requests entered in the CTM schedule queue, the processing of the requests continues, for which processing has already been started in the J2EE server.

(c) Lock process of the front-end EJB

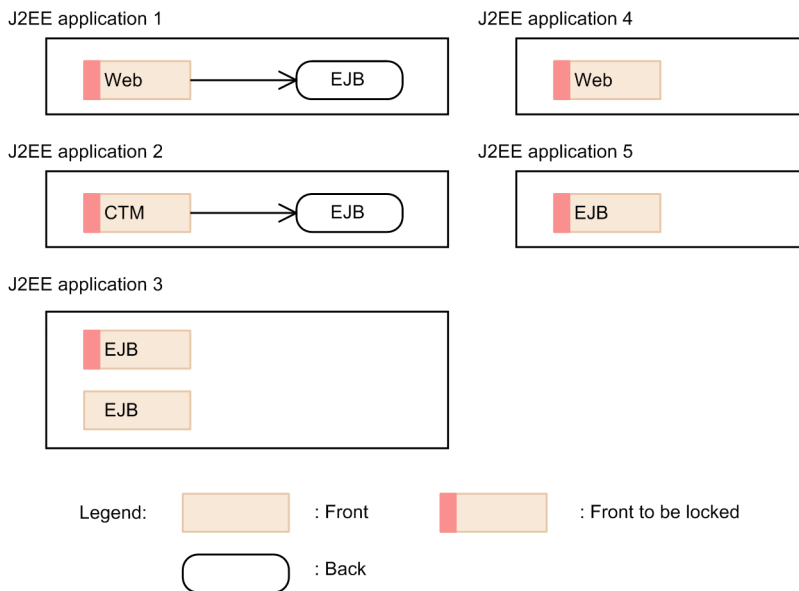
The following processes are executed:

- Terminate the receipt of new requests.
- Continue processing the requests that are being processed.

(2) Configuration pattern and locking methods of J2EE applications

The following figure shows the form of J2EE application for the locking method of each configuration pattern of the J2EE application and describes each locking method:

Figure 5–13: Configuration pattern of a J2EE application



- **Locking method of J2EE application 1**

J2EE application 1 includes Web applications.

In this case, the front-end is normally a Web application. In J2EE application 1, lock the Web applications.

- **Locking method of J2EE application 2**

The J2EE application 2 integrates with CTM to invoke EJB.

In this case, CTM becomes the front-end, as CTM is always invoked from the client. In J2EE application 2, lock the CTM.

- **Locking method of J2EE application 3**

The J2EE application 3 invokes EJB without using CTM.

In this case, it is not possible to determine from the J2EE application configuration as to which EJB is the front-end. The user specifies the front-end EJB. In J2EE application 3, lock the front-end EJB specified by the user.

- **Locking method of J2EE application 4**

J2EE application 4 is a J2EE application where Web application is independent.

In this case, the Web application is the front-end. In J2EE application 4, lock the Web applications.

- **Locking method of J2EE application 5**

J2EE application 5 is a J2EE application where EJB is independent.

In this case, the EJB is the front-end. The user specifies the front-end EJB. In J2EE application 5, lock the front-end EJB specified by the user.

For details on specifying the front-end EJB, see [5.5.6 Settings for execution environment](#).

(3) Precautions when executing the lock process

It is necessary to take care of the following points when executing the lock process:

(a) Precautions when using CTM

You must take the following precautions for executing the lock process, when CTM is used:

- **When the Web application or EJB use CTM to access the EJB present in the same J2EE application**

When the lock process finishes, even the requests that are being processed throw an exception to the client. In such cases, terminate the processing of the client that invokes the applicable Web application or EJB, and then stop the J2EE application.

- **When a request is entered without using CTM for EJB invoked from CTM after the lock process**

New request is received since the EJB lock process is not implemented. In such cases, stop the processing of the client that invokes the applicable EJB, and then stop the J2EE application.

- **When 0 is specified in the `ejbserver.ctm.DeactivateTimeOut` key of `usrconf.properties`**

In `usrconf.properties`, do not specify 0 in the `ejbserver.ctm.DeactivateTimeOut` key.

If 0 is specified and an attempt is made to stop the application when a request is being processed via CTM, the lock process does not end until the request is complete. Therefore, if the request is incomplete due to failure, you cannot execute the forced termination.

(b) Precautions when invoking Web applications across J2EE applications

In a Web application, you can use `getContext` of `javax.servlet.ServletContext` and access a Web application on the same Web container. When the Web application to be accessed is deployed on another J2EE application, first stop the J2EE application that includes the calling front-end Web application.

5.5.3 Stop process

The *stop process* is implemented in the following order:

1. Stop process of the J2EE application
2. Undeploy process of the J2EE application

The following points describe each process:

(1) Stop process

The stop process is implemented in the following order. Note that the J2EE application cannot be executed after the process ends.

1. Processing to stop the Web application
2. Processing to stop EJB
3. Processing to stop the J2EE application

The following points describe each process:

(a) Processing to stop the Web application

The stop process of Web application is implemented in the following order:

1. Processing to end the servlet or JSP
The `destroy` method of servlet and the `jspDestroy` method of JSP is executed.
2. Destroying the HTTP session
The HTTP session attributes are deleted and disabled.
3. Processing to end the filter
The `destroy` method of filter is executed.

4. Destroying the servlet context

The servlet context attributes are destroyed and disabled.

5. Deleting the execution environment of the Web application from the name space

ejb-ref, ejb-local-ref, resource-ref, env-entity, etc., are deleted from the name space.

(b) Processing to stop EJB

The stop process of EJB is implemented in the following order:

1. Deleting the execution environment of EJB from the name space

ejb-ref, ejb-local-ref, resource-ref, env-entity, etc., are deleted from the name space.

2. Deleting the EJB home object or the EJB local home object from the name space

The EJB home object or the EJB local home object is deleted from the name space.

3. Deleting the pool

The method-ready pool is deleted in the case of Session Bean and Message-driven Bean.

The pool and ready pool are deleted in the case of Entity Beans.

(c) Processing to stop the J2EE application

The entries of the J2EE application are deleted from the name space in the stop process of the J2EE application.

(2) Un-deploy process

The un-deploy process is implemented in the following order. Note that the J2EE application is deleted from the J2EE server after the process ends.

1. Processing to un-deploy the Web application

2. Processing to un-deploy EJB

3. Processing to un-deploy the J2EE application

The following points describe each process:

(a) Processing to un-deploy the Web application

Java source files and class files that are the JSP compilation results are deleted during the un-deploy process of the Web application. For JSP compilation results, see *2.5.6 JSP compilation results when JSP pre-compile is not used* in the *uCosminexus Application Server Web Container Functionality Guide*.

(b) Processing to un-deploy EJB

The un-deploy process of EJB is implemented in the following order:

1. Deleting the remote adapter (only when the remote interface is supported)

2. Deleting the table created when CMR is used

(c) Processing to un-deploy the J2EE application

The files downloaded manually by the user are deleted in the un-deploy process of the J2EE application.

5.5.4 Forced termination process

In the *forced termination process*, the lock process and stop process are not executed for the normal termination. The process is cancelled forcibly even when there is a request that is being processed in the J2EE application. Note that a method being executed is cancelled.

Handling the transaction when a J2EE application is terminated forcibly

When a J2EE application is forcibly terminated, the transaction that is about to be performed is timed out forcibly. As a result, an exception occurs and the transaction cannot be started if you try to start the transaction after the J2EE application is terminated forcibly.

However, when the in-process OTS is running across multiple JavaVM, the transaction is not timed out forcibly even when the J2EE application is terminated forcibly.

Furthermore, if an SQL statement is being executed in Statement, CallableStatement, or PreparedStatement when the J2EE application is terminated forcibly, the execution is cancelled and the control tends to return to the J2EE application. The connection used in the J2EE application is destroyed and does not return to the pool.

For details on the transaction timeout, see *3.15.8 Transaction timeout and statement cancellation* in the *uCosminexus Application Server Common Container Functionality Guide*.

5.5.5 Definition in cosminexus.xml

This subsection describes the definitions in `cosminexus.xml` required in the application development environment.

Specify the definition for using the forced termination functionality of a J2EE application in the `<front-ejb>` tag under the `<session>`, `<entity>`, or `<message>` tags of `cosminexus.xml`.

For details on the tags to be specified, see *2.2 Details of each property specified in the Cosminexus application property file (cosminexus.xml)* in the *uCosminexus Application Server Application and Resource Definition Reference Guide*.

5.5.6 Settings for execution environment

When using the functionality for J2EE application operations, you must specify settings for the J2EE server and J2EE applications.

(1) Setting J2EE servers

To use the functionality for J2EE application operations, you must specify settings for the J2EE server. Implement the J2EE server settings in the Easy Setup definition file. Specify the definition of the functionality for J2EE application operations in the `<configuration>` tag of the logical J2EE server (j2ee-server) in the Easy Setup definition file.

Table 5–23: Definitions required for the functionality for J2EE application operations

Items	Parameter to be specified	Setting contents
<i>Forced termination of a J2EE application</i>	<code>ejbserver.deploy.app.stopforcibly.disabled^{#1}</code>	Specifies whether to use the forced termination functionality of the J2EE application. If the front-end (element that forms the request socket from the client perspective) of the J2EE application is an Enterprise Bean, use the server management command to specify the front Enterprise Bean (Front EJB). For details on the settings for using the server

Items	Parameter to be specified	Setting contents
		management commands to forcibly terminate a J2EE application, see (2) Setting the J2EE applications .
<i>Monitoring the execution time of J2EE application</i>	<code>ejbserver.ext.method_observation.interval</code> ^{#2}	<p>Specifies whether to use the functionality for monitoring the execution time of J2EE application , the time interval for checking whether the processing of the request has timed out, and the time interval for canceling a timed out request (method).</p> <p>Specify the value for method timeout and the method cancellation mode. This value will be used in the functionality for monitoring the execution time of J2EE application runtime using the server management commands.</p> <p>For detailed settings, see 5.3.9(2) Setting for J2EE applications.</p>

#1
Specify this parameter in the JavaVM system properties of the J2EE server in the Easy Setup definition file.

#2
Specify this parameter in the user properties of the J2EE server in the Easy Setup definition file.

For details on the Easy Setup definition file and the parameters to be specified, see [4.3 Easy Setup definition file](#) in the *uCosminexus Application Server Definition Reference Guide*.

(2) Setting the J2EE applications

Implement the J2EE application settings in the execution environment using the server management commands and property files. To define the forced termination of a J2EE application, use the SessionBean property file, EntityBean property file, or the MessageDrivenBean property file.

The tags to be specified in each property file correspond to `cosminexus.xml`. For details on the definition of `cosminexus.xml`, see [5.5.5 Definition in cosminexus.xml](#).

5.5.7 Stopping a J2EE application

This subsection describes how to use server management commands to stop a J2EE application.

There are five methods of stopping a J2EE application using server management commands as follows:

1. Executing normal termination by default timeout period
2. Executing normal termination by setting any timeout period
3. Executing forced termination when a J2EE application is not stopped with normal termination by default timeout
4. Executing forced termination when a J2EE application is not stopped by setting any timeout period and executing normal termination
5. Executing forced termination automatically when a J2EE application is not stopped by setting any timeout period and executing normal termination

When the command is executed in an order other than mentioned above, the command is terminated abnormally. For example, if normal termination is not performed, the command cannot be executed in the forced termination format.

Note that you can execute the above-mentioned process 1., 2., or 5. only once for one J2EE application.

However, forced termination can be executed only in a system where the operation settings of the J2EE server are customized and the system is set in such a way that forced termination is applied. For details on the forced termination settings of J2EE servers, see [5.5.6 Settings for execution environment](#).

The following processes are executed in the termination process of a J2EE application:

When a J2EE application front is a Web application

- New requests cannot be received.
- Requests being processed are kept processing.
- Within the requests stored in the control queue of concurrently executing number of the Web applications, HTTP503 error is returned only to the requests that are not processed in the Web container.

When a J2EE application front is Enterprise Bean

- New requests cannot be received.
- Requests being processed are kept processing.

You cannot invoke the method of Enterprise Beans included in an application that has terminated. If you attempt to invoke this method, the message indicating the stop could not lock string is output in the standard error output of the J2EE server. However, this is not a problem.

The following subsections describe the procedures for executing server management commands for each pattern:

(1) Executing normal termination in default timeout period

The execution procedure, format, and example of the command are described below:

1. Execute normal termination of a J2EE application.

Execute the `cjstopapp` command.

Execution format

```
cjstopapp J2EE-server-name -name J2EE-application-name
```

Execution example

```
cjstopapp Myserver -name Appl
```

Note that when the command is executed in default timeout period, the control of the command is returned after 60 seconds even if the stop process is not completed.

2. Confirm the termination status of the J2EE application when the control of the command is returned by timeout.

When the control of the command is returned by timeout, the stop process of the J2EE application may not have been completed. Therefore, check the status of the J2EE application when a timeout occurs.

Check the status of the J2EE application by executing the `cjlistapp` command of server management commands.

Execution format

```
cjlistapp sever-name
```

Execution example

```
cjlistapp MyServer
```

Either of the following status is output as the status of the J2EE application:

Table 5–24: Status of the J2EE application

The output string	Meaning of the status
running	Start
stopped	Terminated
stopFailure	Normal termination failure
forceStopFailure	Forced termination failure
blockadeFailure	Lock failure
blockading	Locking
blockaded	Locked
stopping	Normal termination in-process
forceStopping	Forced termination in-process

After executing the procedure, confirm the status of the J2EE server. You need to restart the J2EE server in the following situations:

- Lock failure
- Normal termination failure

(2) Setting any timeout period, and executing normal termination

The execution procedure, format, and example of the command are described below:

1. Execute normal termination of a J2EE application. At that time, set the timeout period for returning the control of the command with the command option.

Specify the `-t` option in the `cjstopapp` command and perform normal termination.

Execution format

```
cjstopapp J2EE-server-name -name J2EE-application-name -t timeout-time
```

Execution example

```
cjstopapp MyServer -name App1 -t 120
```

In this execution example, control of the command is returned after 120 seconds even if the stop process is not completed.

2. Confirm that the J2EE application is terminated when the control of the command is returned by timeout.
When the control of the command is returned by timeout, the stop process of the J2EE application may not have been completed. Therefore, check the status of the J2EE application when a timeout occurs.

Check the status of the J2EE application by executing the `cjlistapp` command.

Execution format

```
cjlistapp J2EE-server-name
```

Execution example

```
cjlistapp MyServer
```

For details on the contents that are output as the status of J2EE applications, see procedure 2 of the subsection (1) *Executing normal termination in default timeout period*.

After executing the procedure, confirm the status of the J2EE server. You need to restart the J2EE server in the following situations:

- Lock failure
- Normal termination failure

(3) Executing forced termination when a J2EE application is not terminated by executing normal termination in default timeout

The execution procedure, format, and example of the command are described below:

1. Execute normal termination of a J2EE application.

Execute the `cjstopapp` command.

Execution format

```
cjstopapp J2EE-server-name -name J2EE-application-name
```

Execution example

```
cjstopapp MyServer -name App1
```

Note that when the command is executed in default timeout period, the control of the command is returned after 60 seconds even if the stop process is not completed.

2. Confirm that the J2EE application is terminated when the control of the command is returned by timeout.

When the control of the command is returned by timeout, the stop process of the J2EE application may not have been completed. Therefore, check the status of the J2EE application when a timeout occurs.

Check the status of the J2EE application by executing the `cjlistapp` command.

Execution format

```
cjlistapp J2EE-server-name
```

Execution example

```
cjlistapp MyServer
```

For details on the contents that are output as the status of J2EE applications, see procedure 2 of the subsection [\(1\) Executing normal termination in default timeout period](#).

3. Execute forced termination of the J2EE application.

Execute forced termination when the J2EE application is not stopped by normal termination.

Specify the `-cancel` option in the `cjstopapp` command, and forcefully terminate the J2EE application.

Execution format

```
cjstopapp J2EE-server-name -name J2EE-application-name -cancel
```

Execution example

```
cjstopapp MyServer -name App1 -cancel
```

If forced termination is successful, the request being processed in the J2EE application is aborted and normal termination that was being executed earlier finishes.

After executing the procedure, confirm the status of the J2EE server. You need to restart the J2EE server in the following situations:

- Forced termination failure

- Lock failure
- Normal termination failure

(4) Performing forced termination when the J2EE application does not stop even by setting a timeout and executing normal termination

The execution procedure, format, and example of the command are described below:

1. Execute normal termination of a J2EE application. At that time, set the timeout period for returning the control of the command with the command option.

Specify the -t option in the cjstopapp command and perform normal termination.

Execution format

```
cjstopapp J2EE-server-name -name J2EE-application-name -t timeout-time
```

Execution example

```
cjstopapp MyServer -name App1 -t 120
```

In this execution example, control of the command is returned after 120 seconds even if the stop process is not completed.

2. Confirm that the J2EE application is terminated when the control of the command is returned by timeout.

When the control of the command is returned by timeout, the stop process of the J2EE application may not have been completed. Therefore, check the status of the J2EE application when a timeout occurs.

Check the status of the J2EE application by executing the cjlistapp command.

Execution format

```
cjlistapp J2EE-server-name
```

Execution example

```
cjlistapp MyServer
```

For details on the contents that are output as the status of J2EE applications, see procedure 2 of the subsection (1) [Executing normal termination in default timeout period](#).

3. Execute forced termination of the J2EE application.

Execute forced termination when the J2EE application is not stopped by normal termination.

Specify the -cancel option in the cjstopapp command, and forcefully terminate the J2EE application.

Execution format

```
cjstopapp J2EE-server-name -name J2EE-application-name -cancel
```

Execution example

```
cjstopapp Myserver -name App1 -cancel
```

If forced termination is successful, the request being processed in the J2EE application is aborted and normal termination that was being executed earlier finishes.

After executing the procedure, confirm the status of the J2EE server. You need to restart the J2EE server in the following situations:

- Forced termination failure
- Lock failure

- Normal termination failure

(5) Executing forced termination automatically, when a J2EE application does not terminate by setting a timeout period and executing normal termination

The execution procedure, format, and example of the command are described below:

1. Terminate the J2EE application with forced termination after timeout.

In this format, first try to stop the J2EE application normally and if the J2EE application does not stop in the specified timeout, terminate it forcefully.

Specify the `-t` option and `-force` option in the `cjstopapp` command, and then execute forced termination.

Execution format

```
cjstopapp J2EE-server-name--name J2EE-application-name -t timeout-time  
-force
```

Execution example

```
cjstopapp MyServer -name Appl -t 120 -force
```

After executing the procedure, confirm the status of the J2EE server. You need to restart the J2EE server in the following situations:

- Forced termination failure
- Lock failure
- Normal termination failure

5.6 Switching the J2EE Application

This section provides an overview on how to switch a J2EE application.

The following table describes the organization of this section:

Table 5–25: Organization of this section (Replacing a J2EE application)

Category	Title	Reference
Description	Replacing the J2EE application	5.6.1
	Switching the Web application by partially locking the service	5.6.2
Operations	Replacing and maintaining a J2EE application	5.6.3

Note:

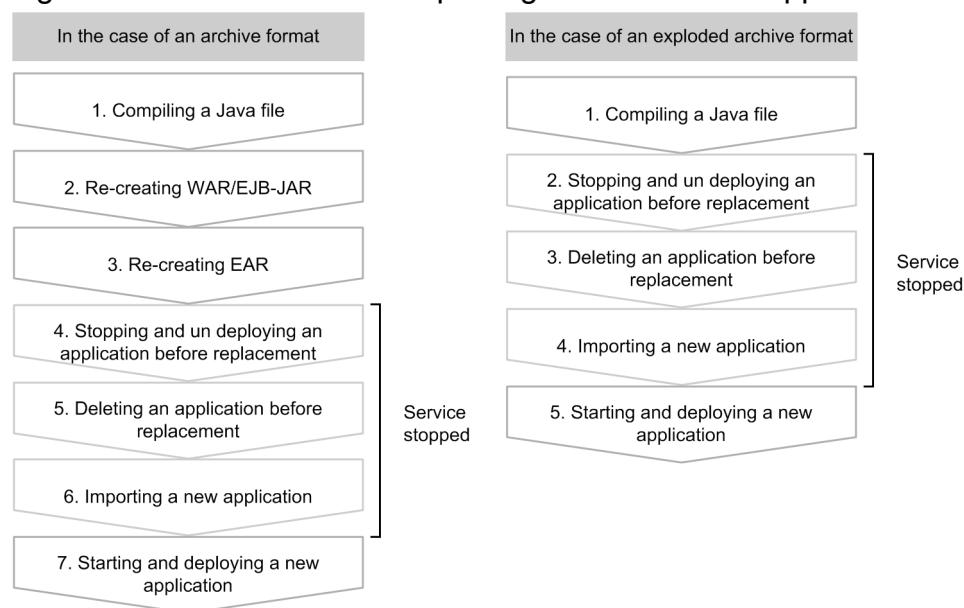
The function-specific explanation is not available for "Implementation", "Settings", and "Notes".

5.6.1 Replacing the J2EE application

A J2EE application may be switched in order to maintain and upgrade the version of the J2EE application, after the system starts operating.

Normally, to switch a J2EE application, it is necessary to stop the J2EE application that is running on the J2EE server, delete it, and then import and deploy a new J2EE application. The following figure shows the procedure to switch a normal J2EE application:

Figure 5–14: Procedure for replacing a normal J2EE application



When normally switching a J2EE application, it is necessary to stop the service for the span when the J2EE application is being switched. However, if partial lock of the service is used, you can switch the J2EE application without stopping the service. Moreover, if re-deploy functionality and reload functionality are used, you can switch a J2EE application in fewer steps as compared to normally switching J2EE applications.

The following table describes the methods for switching J2EE applications:

Table 5–26: Method of switching a J2EE application

Method of switching	Target J2EE applications	
	Archive format	Expanded archive format
Switch by using partial lock of service Y	Y	Y
Switching by redeploy functionality	Y	--
Switching by reload functionality	--	Y

Legend:

Y: Yes

--: No

The following points describe each method of switching the J2EE application. For details on the replacing methods, see [5.6.3 Replacing and Maintaining a J2EE Application](#).

When replacing a J2EE application, you can also manage the generation of the J2EE application by renaming the existing J2EE application.

When replacing a J2EE application that requires a 24-hour service, you can replace the J2EE application without stopping the service by partial locking of the Web application service and using CTM.

(1) Replacing a J2EE application by partial locking of the service

This implies partial locking of the service. This method can be used to maintain the system without stopping the service. You can maintain the system without stopping the service by using partial lock of the service when switching a J2EE application that needs to provide non-stop service for 24 hours.

The execution method of the service partial lock differs depending upon the form of J2EE application. For example, in the case of a Web application, the distribution of requests from the load balancer is stopped for the J2EE server on which the J2EE application that is to be switched is running, and then the J2EE application is switched. You can switch the J2EE application without stopping the service by processing the request in another J2EE server during the span when the J2EE application is being switched.

For details on switching the J2EE application by partially locking the service when the J2EE application is a Web application, see [5.6.2 Switching the Web application by partially locking the service](#).

Reference note

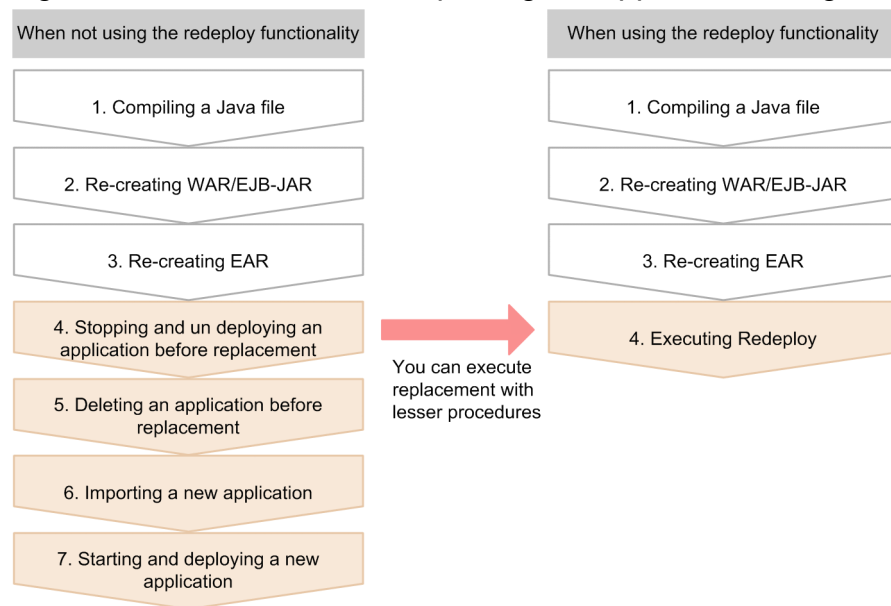
You can also partially lock the service using CTM. For how to switch a J2EE application by locking the CTM schedule queue, see [3.7 Locking and controlling requests](#) in the *uCosminexus Application Server Expansion Guide*.

(2) Replacing a J2EE application using the re-deploy functionality

When testing during application development and during system operations, if you want to switch a running J2EE application with a corrected J2EE application, you can switch using the re-deploy functionality. The *re-deploy* is a deployment method of replacing a J2EE application with fewer procedures and at a high speed. You can use this functionality when switching J2EE applications where only the logic is changed. However, if the conditions are not suitable for executing the redeploy functionality, you need to use the normal procedures when switching the J2EE applications. For the conditions in which you can execute the redeploy functionality, see [15.7 Re-deploying J2EE applications](#) in the *uCosminexus Application Server Common Container Functionality Guide*.

You can use the re-deploy functionality to replace the *archive format* J2EE applications. The procedures for switching the application using the re-deploy functionality are as follows:

Figure 5–15: Procedure for replacing the application using the re-deploy functionality



When the re-deploy functionality is used, you can switch the application with a fewer number of steps as compared to switching a normal archive format J2EE application.

Reference note

Hitachi recommends that you execute the *JSP pre-compile functionality* before executing the re-deploy. The JSP pre-compile functionality is a functionality that compiles the JSP files included in the Web application before the files are deployed and generates the class files. The class files are already generated, so the response time, when the request reaches JSP for the first time and the start-up time of the WEB application can be reduced. For the JSP pre-compile functionality, see 2.5 *Storing the JSP pre-compile functionality and compilation results* in the *uCosminexus Application Server Web Container Functionality Guide*.

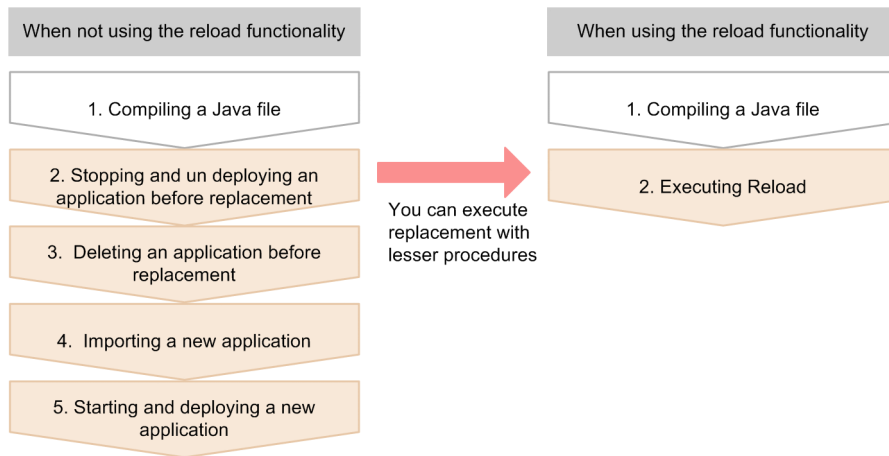
For the redeploy functionality, see 15.7 *Re-deploying J2EE applications* in the *uCosminexus Application Server Common Container Functionality Guide*.

(3) Replacing a J2EE application using the reload functionality

When testing during application development and during system operations, if you want to switch a running J2EE application with a changed J2EE application you can switch them by using the *reload* functionality. You can reload the updated J2EE application using update notification and command execution, when a file in the *expanded archive format* J2EE application is updated. By using the reload functionality, the J2EE application is replaced dynamically using fewer steps.

The process of switching with the reload functionality can be used to switch the expanded archive format J2EE applications. The procedure of switching an application by using the reload functionality is as follows:

Figure 5–16: Switching an application by using the reload functionality



When the reload functionality is used, you can switch an application in less number of steps as compared to switching a normal expanded archive format J2EE application.

For the reload functionality, see *15.8 Detecting updates and reloading J2EE applications* in the *uCosminexus Application Server Common Container Functionality Guide*.

For replacing and maintaining a J2EE application, see *5.6.3 Replacing and Maintaining a J2EE Application*. For the operations of the server management commands, see *3. Basic Operations of Server Management Commands* in the *uCosminexus Application Server Application Setup Guide*.

5.6.2 Switching the Web application by partially locking the service

With service lock of the Web application, you can also lock the services partially. You can lock the Web application service partially to switch the J2EE application and perform system maintenance without stopping the service.

(1) System configuration wherein partially locking the service is possible

The system configurations where partially locking the service is possible are as follows:

- Configuration in which the load balancer is used to distribute requests to multiple Web servers
- Configuration in which the Web server and J2EE server are in a 1:1 or 1:n configuration

Note that in the case of an application that uses a session, it is necessary to process requests of the same session in the same J2EE server. However, if you perform partial lock of the Web application service, the request is processed in another J2EE server that is different from the J2EE server that has the locked Web application. In order to inherit the session information, use the session failover functionality. For the session failover functionality, see *5. Inheriting Session Information Between J2EE Servers* in the *uCosminexus Application Server Expansion Guide*.

(2) Method of partially locking the service

To partially lock a Web application service, change the request distribution destination of the load balancer and ensure that the request is not forwarded to the Web application that you want to lock.

At this time, in order to ensure that an error is not returned for the request, confirm that there is no request executed in the J2EE server. Implement the following depending upon the Web server used:

- When using Cosminexus HTTP Server as the Web server

Terminate Cosminexus HTTP Server. No request will be forwarded to the J2EE server if Cosminexus HTTP Server is terminated.

- When using Microsoft IIS as the Web server

Confirm that there is no request being executed in the corresponding J2EE server. To check the number of requests being executed in a J2EE server, obtain the information by using the `mngsvrutil` command of the Management Server. For details, see *mngsvrutil (Management Server management command)* in the *uCosminexus Application Server Command Reference Guide*.

Note that when changing the request distribution destination, the load balancer requires a function to maintain the connection between the load balancer and the Web server prior to change until the request processing is completed. If a load balancer without this function is used, an error is returned in the request being executed.

5.6.3 Replacing and Maintaining a J2EE Application

This subsection describes the replacing and maintaining of a J2EE application.

A J2EE application may be replaced for version upgrade and maintenance.

When replacing J2EE applications that require 24-hour service, if you use the method of replacing the J2EE applications in an online status using CTM, the J2EE application is replaced without stopping the service.

(1) Replacing a J2EE application

This subsection describes how to replace a J2EE application using server management commands. After terminating the J2EE application, replace with a new application. After replacing, restarts the J2EE application.

This subsection describes the procedure of replacing an application containing `cosminexus.xml`. All the information required for a J2EE application is defined in `cosminexus.xml`.

To replace the J2EE application:

1. Terminate the J2EE application to be replaced.

Execute the `cjstopapp` command. The execution format and example are described below.

Execution format

```
cjstopapp J2EE-server-name -name J2EE-application-name
```

Execution example

```
cjstopapp MyServer -name App1
```

2. Delete the J2EE application.

Execute the `cjdeleteapp` command. The execution format and example are described below.

Execution format

```
cjdeleteapp J2EE-server-name -name J2EE-application-name
```

Execution example

```
cjdeleteapp MyServer -name App1
```

3. Import the J2EE application after replacing.

When you use server management commands, execute the `cjimportapp` command. The execution format and example are described below.

Execution format

```
cjimportapp J2EE-server-name -f EAR-file-path
```

Execution example

```
cjimportapp MyServer -f Appl.ear
```

Note that in the case of a WAR application, the `cjimportwar` command is executed.

4. Start the J2EE application.

When you use server management commands, execute the `cjstartapp` command. The execution format and example are described below.

Execution format

```
cjstartapp J2EE-server-name -name J2EE-application-name
```

Execution example

```
cjstartapp MyServer -name Appl
```

Reference note

For an application that contains `cosminexus.xml` that defines the required information, you need not to acquire and change the property file that is used after the application is imported. For the procedure of acquiring and changing the property file by using the server management commands, see *3.5 Property settings using the property file* in the *uCosminexus Application Server Application Setup Guide*.

(2) Replacing a J2EE application by redeploying

This subsection describes how to replace J2EE applications by redeploying.

Redeploy involves minimal changes, and you can redeploy an archive-type J2EE application with fewer procedures and a high speed. You can use this functionality when switching J2EE applications where only the logic is changed. You can redeploy with server management commands.

Redeploying is possible in following conditions:

Conditions where redeploying is possible

- Only a J2EE application that does not contain runtime information can be replaced. A J2EE application (ZIP file) containing runtime information cannot be redeployed.
- Configuration of the J2EE application before and after replacing must be same. Redeploying is not possible if the number of EJB-JARs, resource adapters, and WARs contained in the J2EE application are different and the file names are different. The names of the J2EE applications must also be the same.
- The method definition and annotation value of the home interface (local and remote) in an EJB-JAR, component interface (local and remote), and business interface (local and remote) contained in the J2EE application must be same before and after redeploying.
- When a J2EE application is set by inheriting only runtime attributes, the definition contents of a DD file (`application.xml`, `ejb-jar.xml`, `ra.xml`, and `web.xml`) that are set in the application development environment needs to be same.

Moreover, when you replace a J2EE application, if you rename and save the J2EE application before replacing, you can manage the generation of the J2EE application using names. This subsection also describes how to rename a J2EE application.

In redeploying, information of a J2EE application before replacing is inherited to the J2EE application after replacing. In default settings, all the attributes of a J2EE application before replacing are inherited by the J2EE application after replacement. You can have the new application inherit only runtime attributes[#] from the old application by specifying an option when executing the `cjreplaceapp` command. For details on the commands, see *cjreplaceapp (replace application)* in the *uCosminexus Application Server Command Reference Guide*.

#

You can set the definition of DD (`application.xml`, `ejb-jar.xml`, `ra.xml`, and `web.xml`) and independent attribute files, in the attribute file. The defining of independent property files is called *runtime attributes*.

When executing redeploy, the J2EE application can be either running or terminated. When a running J2EE application is replaced, the J2EE application starts automatically after replacing. However, J2EE application objects stored in the pool and cache are destroyed. When a terminated J2EE application is replaced, the J2EE application after replacing will also be terminated.

Tip

If a J2EE application is redeployed during initialization, it is terminated while redeploying and restarted after replacing. At this time, when the termination process exceeds the timeout set by server management commands (`cjreplaceapp`), forcefully terminate the J2EE application. If a timeout is not specified and terminating of the J2EE application exceeds 60 seconds that is the default timeout, the J2EE application is forcefully terminated. After the forced termination, if the terminating process further exceeds the timeout period, the command is ended abnormally.

Note that when a J2EE application restarts after replacing, if initialization takes more time than the timeout period specified in the `ejbserver.rmi.request.timeout` key of `usrconf.properties` for server management commands, the command is ended abnormally.

The execution format and example of replacing are described below:

Execution format

```
cjreplaceapp J2EE-server-name -name J2EE-application-name -f Path-of-application-file-to-be-exchanged
```

Execution example

```
cjreplaceapp MyServer -name App1 -f App1.ear
```

Important note

- When redeploying an application containing `cosminexus.xml`, the Application Server-specific information defined in `cosminexus.xml` is overwritten by the information defined in `cosminexus.xml`, after the application is replaced. The other information defined for the elements (such as EJB-JAR file and resource adapter) and DD configuring the J2EE application inherits the information that exists before the application is replaced.

- When redeploying and replacing a WAR application, the `cosminexus.xml` file cannot be re-read. When changing the application properties of a WAR application, use server management commands (`cjgetappprop` and `cjsetappprop` commands).

(3) Replacing a J2EE application by reloading

This subsection describes how to replace a J2EE application by reloading.

When an application containing `cosminexus.xml` is reloaded, the Application Server-specific information defined in `cosminexus.xml` is not reloaded. For details on replacing J2EE applications containing `cosminexus.xml` in exploded archive format, see the procedures in [5.6.3\(1\) Replacing a J2EE application](#).

Reload is a function with which you can replace a J2EE application in exploded archive format with fewer procedures. For replacing a J2EE application using the reload function, operations like terminating and deleting an existing J2EE application, and archiving, importing, and restarting the J2EE application after replacing are not required. This function is especially effective in operations of a system where maintenance occurs frequently since you can update the J2EE application only by updating a class file and reloading the J2EE application.

Note that settings are required in advance for replacing the J2EE application by reloading. For details on the settings, see [15.8.12 Settings for detecting updates and reloading J2EE applications](#) in the *uCosminexus Application Server Common Container Functionality Guide*.

You can use the server management commands (`cjreloadapp` command) to replace a J2EE application by reloading. For details on the `cjreloadapp` command, see *cjreloadapp (reload application)* in the *uCosminexus Application Server Command Reference Guide*.

To reload:

1. Edit or create a Java source file as per the contents of maintenance, and compile in a class file.
2. Reload the J2EE application.

Execute the `cjreloadapp` command. The execution format and example are described below.

Execution format

```
cjreloadapp J2EE-server-name -name J2EE-application-name
```

Execution example

```
cjreloadapp MyServer -name App1
```

Important note

To delete an application for which an attempt to reload has failed, terminate the application and delete it after it is successfully reloaded or delete the application after restarting the J2EE server.

(4) J Replacing a J2EE application to be executed after pre-compiling JSPs

When JSPs are edited during the maintenance of a J2EE application, normally, the JSPs are compiled when the first request is received after a J2EE application is replaced. If you use the *JSP pre-compile* function, you can compile JSPs for the J2EE application before deployment, and can reduce the response time for the first request.

To execute the JSP pre-compile function, use server management commands or `cjjspc` command. This subsection describes the execution timings and methods of the JSP pre-compile function.

When a system is running, you can execute the JSP pre-compile function at the following timings:

- When starting a J2EE application (server management commands)
- When replacing a J2EE application by reloading (the `cjjspc` command)
- When replacing a J2EE application by re-deploying (the `cjjspc` command)

The following section describes how to execute the JSP pre-compile function for each timing:

(a) JSP pre-compile when starting a J2EE application

Execute the JSP pre-compile function when starting a J2EE application. In this case, specify the `-jspc` option and execute the `cjstartapp` command.

The execution format and example are described below.

Execution format

```
cjstartapp server-name -name J2EE-application-name -jspc
```

Execution example

```
cjstartapp MyServer -name account -jspc
```

For details on the `cjstartapp` command, see *cjstartapp (start J2EE application)* in the *uCosminexus Application Server Command Reference Guide*.

(b) JSP pre-compile when replacing a J2EE application by reloading

When you replace a J2EE application by reloading, execute the JSP pre-compile function before executing the `cjreloadapp` command. In this case, execute the `cjjspc` command.

The execution format and example are described below.

Execution format

```
cjjspc -root root-directory-of-Web-application
```

Execution example (In Windows)

```
cjjspc -root d:\app\webapp1
```

Execution example (In UNIX)

```
cjjspc -root /tmp/app/webapp1
```

For details on the `cjjspc` command, see *cjjspc (pre-compile JSP)* in the *uCosminexus Application Server Command Reference Guide*.

(c) JSP pre-compile when replacing a J2EE application by redeploying

When replacing a J2EE application by redeploying, execute the JSP pre-compile function before executing the `cjreplaceapp` command. In this case, execute the `cjjspc` command.

The execution format and example are described below.

Execution format

```
cjjspc -root root-directory-of-Web-application
```

Execution example (In Windows)

```
cjjspc -root d:\app\webapp1
```

Execution example (In UNIX)

```
cjjspc -root /tmp/app/webapp1
```

For details on the `cjjspc` command, see *cjjspc (pre-compile JSP)* in the *uCosminexus Application Server Command Reference Guide*.

Important note

- When you use the pre-compile function for a J2EE application to be replaced by redeploying, the JSP pre-compile function cannot be executed unless the JSP compiled results are included in the J2EE application to be replaced. When using the JSP pre-compile function even after replacing the J2EE application, execute JSP pre-compile of each Web application and set such that the JSP compiled results are included in the J2EE application to be replaced.
- When a tag file, static included file, or TLD file is updated, compile all JSP files that reference the updated files.
- When a JSP file or tag file is added to the application in exploded archive format that uses the JSP pre-compile function, re-execute JSP pre-compile and compile the JSP file or all JSP files that reference the tag file.
- When a class file included in the JSP working directory of an application in exploded archive format is to be copied from the development environment and updated, copy all the class files executing JSP pre-compile from the development environment.
- When JSP pre-compile is executed using the `cjjspc` command and an error occurs during the translation of JSP file or tag file, an error message is output to the console. Note that when JSP pre-compile is executed using the `cjstartapp` command and an error occurs during translation of JSP file or tag file, an error message is output to the servlet log.

(5) Renaming a J2EE application

When you replace a J2EE application, you can *manage the J2EE application generations*, if you rename and store the existing J2EE application in advance. Note that if required, you can easily return to the J2EE application before replacement.

Rename the J2EE applications using the server management commands. When a J2EE application with the same name as the name you want to rename to already exists, you cannot rename the application with that name. Names are not case-sensitive.

Note that when renaming a J2EE application, lookup name also needs to be changed. When the Enterprise Bean that uses home interface or component interface performing remote call is included in the configuration element of the J2EE application, reacquire RMI-IIOP stubs and interfaces.

Acquire RMI-IIOP stubs and interfaces using server management commands (`cjgetstubsjar`). You cannot acquire the same when the J2EE application has not been executed even once. Further, an error occurs while acquiring RMI-IIOP stubs and interfaces in the following cases:

- When only WAR is included in the J2EE application

- When all the home interfaces and component interfaces used in Enterprise Bean are locally invoked
- When Message-driven Bean is the only Enterprise Bean

To rename a J2EE application:

1. Terminate the J2EE application to be renamed.

Execute the server management command (`cjstopapp`).

2. Rename the J2EE application.

Execute the server management command (`cjrenameapp`).

The execution format and example are described below.

Execution format

```
cjrenameapp J2EE-server-name -name Old-J2EE-application-name -newname New-J2EE-application-name
```

Execution example

```
cjrenameapp MyServer -name Appl -newname Applbak
```

3. Start the J2EE application.

Execute the server management command (`cjstartapp`).

4. Acquire RMI-IIOP stubs and interfaces when home interface or component interface in Enterprise Beans contained in the J2EE application is defined such that remote invocation is executed.

Execute the server management command (`cjgetstubsjar`).

The execution format and example are described below.

Execution format

```
cjgetstubsjar J2EE-server-name -name New-J2EE-application-name -d Path-of-the-directory-that-stores-RMI-IIOP-stub-and-interface
```

Execution example

```
cjgetstubsjar MyServer -name Applbak -d temp
```

5.7 Accessing network resources from J2EE applications

In Windows, you must specify the settings to access other hosts from J2EE applications with path of the network resources specified in a UNC or a network drive. This section describes about accessing network resources from J2EE applications. Note that in UNIX, network resources can be accessed without using the functionality described in this section.

The following table describes the organization of this section:

Table 5–27: Organization of this section (Accessing network resources from J2EE applications)

Category	Title	Reference
Description	Overview of accessing network resources	5.7.1
Settings	Settings for accessing network resources	5.7.2
Notes	Problems during operations	5.7.3

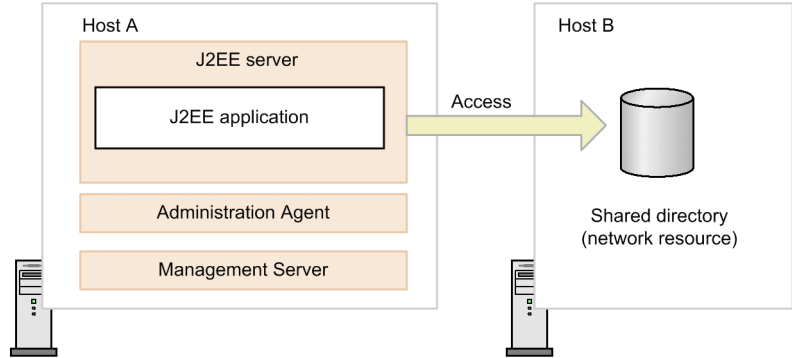
Note:

Function-specific explanation is not available for "Implementation" and "Operations".

5.7.1 Overview of accessing network resources

The functionality to access network resources is used from J2EE applications, on the J2EE server running in Windows, for accessing the resources on other hosts. Using this functionality you can specify a UNC or a network drive to access the path of the resources of other hosts from the J2EE application. The following figure gives an overview of accessing network resources:

Figure 5–17: Overview of accessing network resources



In this figure, the shared directory of HostB is assumed to be allocated to X drive of HostA. You can use the functionality for accessing network resources and access the HostB shared directory allocated to X drive from the J2EE application of HostA. Note that to use this functionality, you must start the Administration Agent on HostA where the J2EE application is running, using an account of the Administrators group.

5.7.2 Settings for accessing network resources

To access the network resources, specify the following settings. Note that the following description uses Windows Server 2019 as an example:

1. From the **Control Panel**, open **Management Tools**, and then **Services**.

2. Change the logon account of Administration Agent to an account belonging to the Administrators group.
Change the logon account `Cosminexus Management Server - Administration Agent` to an account belonging to the Administrators group of the local host.
3. In `adminagentuser.cfg`, specify the network drive to be used in the J2EE application.
Add the following description to *Cosminexus-installation-directory*\manager\config\adminagentuser.cfg. In this example, `\\host\dir` is allocated to X drive.
`add.network.drive=X=\\host\dir`
Note that you implement these settings only when using network drives for accessing network resources. Settings are not required when you use the UNC to access resources.

Tip

Settings for network drive allocation in step 3

Perform the following settings before invoking the Administration Agent. The specified network drive is allocated when the Administration Agent is invoked. However, the network drive is not allocated if the logon account of the Administration Agent is a local system account. The detailed description of the contents specified in step 3 is as follows:

- Files used
Cosminexus-installation-directory\manager\config\adminagentuser.cfg
- Settings
In the `add.network.drive` key, specify the name and the directory path of the drive you want to allocate as the network drive.
Specification example: `X=\\host\dir`
Note that if you specify these settings in the UNIX environment, KEOS21401-E message is output to *Manager-log-output-directory*/adminagent.err and the Administration Agent terminates with end code 1.
Note that you can also allocate multiple network drives.

The following table describes the example of settings and operations of the `add.network.drive` key:

Table 5–28: Example of settings and operations of the `add.network.drive` key

Specified value	Setting example	Operation
No key	(<code>add.network.drive</code> is not defined)	The network drive is not allocated.
No value	<code>add.network.drive=</code>	
Specify one network drive	<code>add.network.drive=X=\\host\dir</code>	Outputs KEOS21304-I message and allocates <code>\\host\dir</code> to X drive.
Specify multiple network drives	<code>add.network.drive=X=\\host\dir</code> <code>add.network.drive=Y=\\host\dir2</code>	Outputs KEOS21304-I message and allocates <code>\\host\dir</code> to X drive and <code>\\host\dir2</code> to Y drive ^{#1} .
Specify multiple network drives (when the drive name is the same)	<code>add.network.drive=X=\\host\dir</code> <code>add.network.drive=X=\\host\dir2</code>	Outputs KEOS21304-I message and allocates <code>\\host\dir</code> to X drive ^{#1} . Outputs KEOS21305-W message and fails in an attempt to allocate the <code>\\host\dir2</code> ^{#2} .

Note:

If the logon account of the Administration Agent is a local system account, the KEOS21307-W message is output to *Manager-log-output-directory*\adminagent.err and the processing continues without allocating the network drive.

#1

If the specified network drive is allocated successfully, the KEOS21304-I message is output to *Manager-log-output-directory*\adminagent.err and the processing continues.

#2

If an attempt to allocate the specified network drive fails, the KEOS21305-W message is output to *Manager-log-output-directory*\adminagent.err and the processing continues.

4. From the **Control Panel**, open **User Account** and then **Manage network passwords**.

If the logon account is the domain user, open **Manage passwords**. In this window, specify the settings to omit the access authentication processing for accessing the network resources. Implement these settings from the desktop of the logon account that invoked the Administration Agent.

5. In the **Save user name and password** dialog box, specify the host name, user name, and password for the resource to be accessed.

However, the specified user must be allowed to access the shared directory.

Tip

Specifying the host name for resource to be accessed in step 5

Specify the host name and not the IP address in the 'host name to be accessed' that you will specify in step 5. If *IP-address* is specified for the host name in the network drive settings in step 5, and if the host to be allocated in step 3 is *host-name*, an attempt to allocate the host fails as you cannot login to the host you want to access according to Windows specifications.

The following table uses the allocation of host name 'host' (IP=10.10.10.10) as an example to describe whether the network can be allocated for different combinations of the host names specified in step 3 and the host names specified in step 5:

Table 5–29: Availability of network allocation

Host name specified in step 5	Host name specified in step 3	Allocation
host	add.network.drive=X=\\host\dir	Y
	add.network.drive=X=\\10.10.10.10\dir	Y
10.10.10.10	add.network.drive=X=\\host\dir	--
	add.network.drive=X=\\10.10.10.10\dir	Y

Legend:

Y: Network can be allocated

--: Network cannot be allocated

5.7.3 Problems during operations

If the Administration Agent is invoked using an account that does not belong to the Administrators group, the following events occur:

- Occurrence timing

When Cosminexus Management Server - Administration Agent is invoked from a Windows service or when the `adminagentctl start` command is executed

- Events

An attempt to invoke the Administration Agent fails. Also, "KEOS21108-E The administration agent could not be started. (detail = The logon user is not using a local system account, or does not belong to the Administrators group)" is output to the event log.

As an action, use the account for logging on to Cosminexus Management Server - Administration Agent from the Windows service as a local system account, or change the account to one belonging to the Administrators group.

5.8 Precautions when a J2EE application is operating

In AIX, if an error `exec error: parameter list or environment list is too long` (in the case of environments other than Japanese environment: `exec error: Arg list too long`) occurs during the deployment of a J2EE application, the length of the argument of the `java2iio` command exceeds the specified value of the ARG/ENV list which is the kernel parameter of the OS. By default, this error occurs when the number of home and component interfaces of an EnterpriseBean containing an application to be deployed exceeds approximately 570 (assuming that the average length of the interface names that include package names is 40 characters). If an error occurs, implement the following measures:

1. Execute the following command and check the value of the ARG/ENV list.

```
lsattr -E -l sys0 -a ncargs
```

Note that the default value is 6 (Units: 4 KB block).

2. Execute the following command and change to a value greater than the calculated value of the ARG/ENV list.

```
chdev -l sys0 -a ncargs=value-of-ARG/ENV-list-after-change
```

Specify *value-of-ARG/ENV-list-after-change* in the range from 6 through 128. Specify a value more than the value calculated in the following formula:

Value of the ARG/ENV list after change $\geq ((A + (B \times C))^{\#} + 4,095) / 4,096$

- A: Approximately 1,600 bytes (number of command bytes except for the parameter passed to the `vbj` command)
- B: Average length of the class name that includes the package name (Units: Bytes)
- C: Number of home and component interfaces

#: With $A + (B \times C)$, you can calculate the length of the command that is executed in bytes.

3. Redeploy the J2EE application.

6

Audit Log Output Functionality (INTENTIONALLY DELETED)

INTENTIONALLY DELETED

6.1 INTENTIONALLY DELETED

INTENTIONALLY DELETED

7

Database Audit Trail Linkage Functionality (INTENTIONALLY DELETED)

INTENTIONALLY DELETED

7.1 INTENTIONALLY DELETED

INTENTIONALLY DELETED

8

Statistical Output using Management Commands

This chapter describes the output of statistics using the management commands.

8.1 Organization of this chapter

This chapter is organized as the following table.

Table 8–1: Organization of this chapter (Output of statistics using management commands)

Category	Title	Reference
Description	Overview of statistics output using management commands	8.2
	Methods to output server statistics	8.3
	Items that can be checked by the statistics monitoring	8.4

Note:

The function-specific explanation is not available for "Implementation", "Operations", "Settings", and "Precautions".

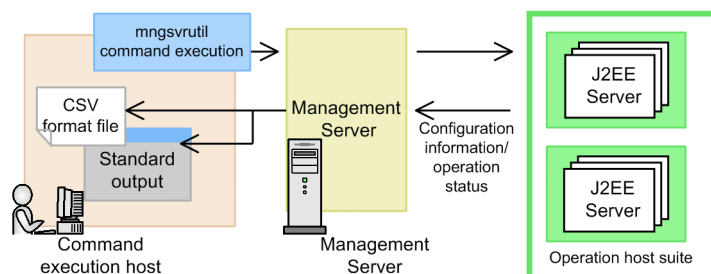
8.2 Overview of statistics output using management commands

You can output the statistics using the management command (`mngsvrutil`) of the Management Server. You can output the statistics to the files and check for each management domain. The files will be in the CSV format.

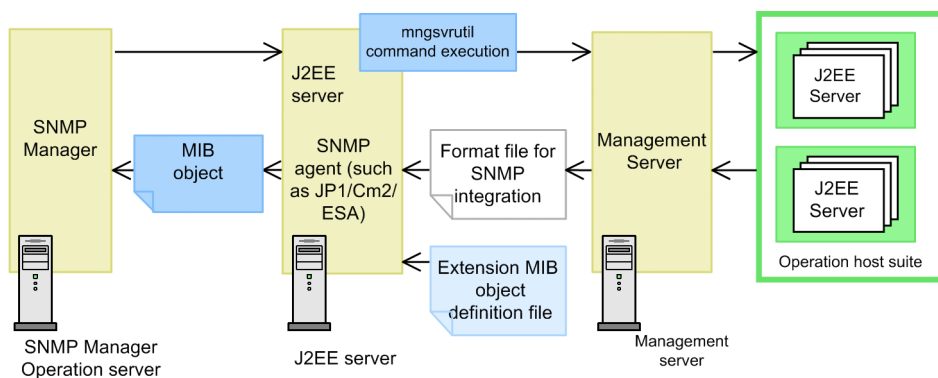
The following figure shows the flow of the output in a statistics file.

Figure 8–1: Flow of the output in a statistics file

- When you output a file of CSV format



- When you output a file of SNMP format is (after output, convert to MIB object)



Reference note

You can use the management command or the `adminagentcheck` command to check the status of the Administration Agent and the Management Server.

Tip

In normal operations, you use the statistics file to monitor statistics. Only when you want to obtain detailed information than the information obtained in a statistics using file, you use the management commands for monitoring the statistics. For details on monitoring of statistics using the statistics file, see [3.2 Overview of statistics collection functionality](#).

8.3 Methods to output server statistics

The statistics on the operational statuses of the J2EE server or batch server can be monitored by using the management command (`mngsvrutil`).

To monitor the statistics, you specify the subcommand `get` in the management command (`mngsvrutil`), and then execute the command. This command can be used to send the operating information of the J2EE server to the standard output. The command can also be used to output the information to a CSV file or an SNMP integration format file.

You can specify the target to which the statistics are to be output in the arguments of `get`.

The execution format and example of execution while acquiring the statistics of a J2EE server are described below:

Execution format

```
mngsvrutil -m Management-Server-host-name[:port-number] -u management-user  
-ID -p management-password -t J2EE-server-name get domain-name-or-category  
-to-be-obtained
```

Execution example

```
mngsvrutil -m mnghost -u user01 -p pw1 -t J2EEServer1 get j2eeContainer
```

For details on the `mngsvrutil` command, subcommands, and information that can be acquired, see *mngsvrutil (Management Server management command)* and *7.3 Details of subcommands of the mngsvrutil command* in the *uCosminexus Application Server Command Reference Guide*.

8.4 Items that can be checked by statistics monitoring

Statistics monitoring allows you to monitor the operational statuses of the J2EE server and batch server.

The following table describes the items that can be monitored as the statistics:

Table 8–2: Items that can be displayed by monitoring the statistics (for J2EE servers)

Monitoring target		Items that can be monitored
J2EE server	J2EE container	<ul style="list-style-type: none">• J2EE server name• Naming Service host• Naming Service port number• Container start time
	EJB container	<ul style="list-style-type: none">• Container name
	Web container	Basic information <ul style="list-style-type: none">• Container name• Web container start time• Port number of a management server• Port number used for communication with a Web server• Upper limit set for the socket backlog for communication with a Web server• Upper limit set for number of running threads (processing requests)• Maximum number of concurrently executing threads• Default pending queue size Statistics <ul style="list-style-type: none">• Number of connections between a Web server and Web container• Number of running threads (HTTP server connector)• Number of requests pending for Web containers• Upper limit of the number of concurrently executing threads• Number of running threads• Number of default pending requests• Number of requests exceeding the default pending queue• Number of running threads (HTTP server)
	JavaVM	Basic information <ul style="list-style-type: none">• JavaVM name• JavaVM version• Maximum memory used by JavaVM Statistics <ul style="list-style-type: none">• Free memory of JavaVM• Total memory space of JavaVM
Applications	J2EE application	<ul style="list-style-type: none">• Application name (display name)• Description• Time of deployment
	EJB application	<ul style="list-style-type: none">• EJB-JAR name• Description
	Enterprise Bean (Stateful Session Bean)	Basic information <ul style="list-style-type: none">• Enterprise Bean name• Description• Bean name (for internal identification)• Home interface name

Monitoring target		Items that can be monitored
		<ul style="list-style-type: none"> • Local Home interface name • Component interface name • Local Component interface name • EJB class name • Transaction type • Value of maximum concurrent connections • Value of maximum concurrent executions • Passive session timeout • Timeout for running sessions <p>Statistics</p> <ul style="list-style-type: none"> • Number of sessions connected currently • Number of running sessions • Number of passive sessions • Number of sessions waiting for connection <p>Home interface</p> <ul style="list-style-type: none"> • Home interface name • Response • EJB method execution time <p>Local Home interface</p> <ul style="list-style-type: none"> • Home interface name • Response • EJB method execution time <p>Component interface</p> <ul style="list-style-type: none"> • Component interface name • Response • EJB method execution time <p>Local Component interface</p> <ul style="list-style-type: none"> • Component interface name • Response • EJB method execution time
	Enterprise Bean (Stateless Session Bean)	<p>Basic information</p> <ul style="list-style-type: none"> • Enterprise Bean name • Description • Bean name (for internal identification) • Home interface name • Local Home interface name • Component interface name • Local Component interface name • EJB class name • Transaction type • Bean instance pool <p>Statistics</p> <ul style="list-style-type: none"> • Number of sessions waiting for connection • Current value of the Bean instance pool • Number of Session Beans in use • Number of unused Session Beans <p>Home interface Same as Stateful Session Beans</p> <p>Local Home interface Same as Stateful Session Beans</p>

Monitoring target		Items that can be monitored
		Component interface Same as Stateful Session Beans Local Component interface Same as Stateful Session Beans
	Enterprise Bean (Entity Bean)	Basic information <ul style="list-style-type: none"> • Enterprise Bean name • Description • Bean name (for internal identification) • Home interface name • Local Home interface name • Component interface name • Local Component interface name • EJB class name • EntityBean persistent type • EntityBean cache model • Value of maximum concurrent connections • Bean instance pool • Connection timeout Statistics <ul style="list-style-type: none"> • Number of sessions connected currently • Number of sessions waiting for connection • Current value of the Bean instance pool • Number of Entity Beans in use • Number of unused Entity Beans Home interface Same as Stateful Session Beans Local Home interface Same as Stateful Session Beans Component interface Same as Stateful Session Beans Local Component interface Same as Stateful Session Beans
	Enterprise Bean (Message-driven Bean)	Basic information <ul style="list-style-type: none"> • Enterprise Bean name • Description • Bean name (for internal identification) • EJB class name • Transaction type • Destination type^{#1} • Bean instance pool Statistics <ul style="list-style-type: none"> • Number of sessions connected currently • Current value of the Bean instance pool • EJB method execution time
	Web Application	Basic information <ul style="list-style-type: none"> • Context root • Number of dedicated threads • Maximum number of concurrently executing threads • Size of the pending queue for Web applications Statistics

Monitoring target		Items that can be monitored
		<ul style="list-style-type: none"> Number of valid sessions Upper limit of the number of concurrently executing threads Number of running threads Number of requests pending for Web applications Number of requests exceeding the pending queue for Web applications
	Servlet	<ul style="list-style-type: none"> Servlet name Implementation class name of the servlet Servlet execution frequency Servlet failure frequency Servlet execution time Output data size
	URL	<ul style="list-style-type: none"> URL URL invocation frequency URL invocation failure frequency URL execution time Output data size
Resources	Data source (SimpleJTA)	Basic information <ul style="list-style-type: none"> Resource name Resource type Description Authentication type Login timeout User ID Connection pool Statistics <ul style="list-style-type: none"> Resource name Current Pool value (total) Used connections Unused connections
	Data source (FullJTA)	Basic information <ul style="list-style-type: none"> Resource name Resource type Description Login timeout User ID Connection pool Statistics <ul style="list-style-type: none"> Resource name Current value of a pool Used connections Unused connections Execution time of the getConnection() method Execution time of the getXACConnection() method Frequency of failure frequency of the getConnection() method Frequency of FATAL errors occurring in the Connection
	Resource adapter	Basic information <ul style="list-style-type: none"> Resource name Resource type

Monitoring target		Items that can be monitored
		<ul style="list-style-type: none"> • Description • Name of the vendor providing the resource adapter • Version of JCA specifications to which the resource adapter conforms • Resource adapter version • Type of EIS to be connected • Interface name of the ConnectionFactory • Implementation class name of the ConnectionFactory • Implementation class name of the ManagedConnectionFactory • Interface name of the Connection • Implementation class of the Connection • Transaction support model • Information of set properties • User ID • Connection pool^{#2} <p>Statistics</p> <ul style="list-style-type: none"> • Resource name • Current value of a pool (total)^{#2} • Number of connections in use^{#2} • Number of unused connections^{#2} • Execution frequency of the createManagedConnection() method of ManagedConnectionFactory^{#2} • Execution frequency of the getConnection() method of ManagedConnection^{#2} • Execution frequency of the cleanup() method of ManagedConnection^{#2} • Execution frequency of the destroy() method of ManagedConnection^{#2} • Execution time of the allocateConnection() method of ConnectionManager^{#2} • Execution time of the createManagedConnection() method of ManagedConnectionFactory^{#2} • Frequency of failure of the allocateConnection() method of ConnectionManager^{#2} • Frequency of FATAL errors occurring in ManagedConnection^{#2}
Service	Transaction	<p>Basic information</p> <ul style="list-style-type: none"> • Service name • Service type • Default value of transaction timeout <p>Statistics</p> <ul style="list-style-type: none"> • Number of active transactions • Average transaction time

Note:

The displayed items differ depending on the used functions (whether the function that controls the maximum number of concurrently executing threads is used), option settings, and used resources.

#1

If a resource adapter compliant with the Connector 1.5 specifications is used, Other is output.

#2

If Cosminexus RM is used, these items are output as the statistics of Cosminexus RM. You cannot confirm these items by monitoring the statistics of a Management Server.

Table 8–3: Items that can be displayed using statistics monitoring (in batch servers)

Monitoring target		Items that can be monitored
Batch server	J2EE container	<ul style="list-style-type: none"> Batch server name Naming Service host Naming Service port number Container start time
	JavaVM	Basic information <ul style="list-style-type: none"> JavaVM name JavaVM version Maximum memory used by JavaVM Statistics <ul style="list-style-type: none"> Free memory of JavaVM Total memory space of JavaVM
Resources	Data source (SimpleJTA)	Basic information <ul style="list-style-type: none"> Resource name Resource type Description Authentication type Login timeout User ID Connection pool Statistics <ul style="list-style-type: none"> Resource name Current Pool value (total) Used connections Unused connections
	Data source (FullJTA)	Basic information <ul style="list-style-type: none"> Resource name Resource type Description Login timeout User ID Connection pool Statistics <ul style="list-style-type: none"> Resource name Current value of a pool Used connections Unused connections Execution time of the getConnection() method Execution time of the getXACConnection() method Frequency of failure frequency of the getConnection() method Frequency of FATAL errors occurring in the Connection
	Resource adapter	Basic information <ul style="list-style-type: none"> Resource name Resource type Description Name of the vendor providing the resource adapter Version of JCA specifications to which the resource adapter conforms Resource adapter version

Monitoring target		Items that can be monitored
		<ul style="list-style-type: none"> • Type of EIS to be connected • Interface name of the ConnectionFactory • Implementation class name of the ConnectionFactory • Implementation class name of the ManagedConnectionFactory • Interface name of the Connection • Implementation class of the Connection • Transaction support model • Information of set properties • User ID • Connection pool[#] <p>Statistics</p> <ul style="list-style-type: none"> • Resource name • Current pool value (total)[#] • Number of connections in use[#] • Number of unused connections[#] • Execution frequency of the createManagedConnection() method of ManagedConnectionFactory[#] • Execution frequency of the getConnection() method of ManagedConnection[#] • Execution frequency of the cleanup() method of ManagedConnection[#] • Execution frequency of the destroy() method of ManagedConnection[#] • Execution time of the allocateConnection() method of ConnectionManager[#] • Execution time of the createManagedConnection() method of ManagedConnectionFactory[#] • Frequency of failure of the allocateConnection() method of ConnectionManager[#] • Frequency of FATAL errors occurring in ManagedConnection[#]
Service	Transaction	<p>Basic information</p> <ul style="list-style-type: none"> • Service name • Service type • Default value of transaction timeout <p>Statistics</p> <ul style="list-style-type: none"> • Number of active transactions • Average transaction time

Note:

The displayed items differ depending on the used functions (whether the function that controls the maximum number of concurrently executing threads is used), option settings, and used resources.

[#]

If Cosminexus RM is used, these items are output as the statistics of Cosminexus RM. You cannot confirm these items by monitoring the statistics of a Management Server.



Tip

Setting the sampling time of statistical information

Specify sampling time of the statistical information, if the statistics are to be displayed.

When you specify a sampling time, data within the time specified in the sampling time (this is called *N seconds*) can be displayed as statistical information, by tracing back the time when the window is displayed or updated. This statistical information is N seconds peak and Average value in N seconds. The maximum and minimum values are extracted from the data after the sampling starts.

You can specify a sampling time using the subcommand set of the operation management command (`mngsvrutil`). For details on the commands, see *7.3 Details of subcommands of the mngsvrutil command* in the *uCosminexus Application Server Command Reference Guide*.

9

Automatic Execution of Processing Using Management Event Notification and Management Action

This chapter describes the automatic execution of the processing by using a management event.

9.1 Organization of this Chapter

This chapter is organized as the following table.

Table 9–1: Organization of this chapter (Automatic execution of processing using management event notification and management action)

Category	Title	Reference
Description	Overview of management event notification and management action	9.2
	Controlling the execution of management action	9.3
Settings	Settings for automatic execution of processes by management events	9.4

Note:

The function-specific explanation is not available for "Implementation", "Operations", and "Notes".

9.2 Overview of Management Event Notification and Management Action

If failure, resource depletion, and other phenomena occur on the J2EE server or batch server in the management domain, Management Server is notified of them via events called *Management events*. Issuance of Management events is triggered by messages that are output during operation of the J2EE server or batch server. By defining the operations, to be performed when a Management event is notified, in the Management Server, the action can be executed automatically when the management event occurs. This action is called a *Management action*.

Important note

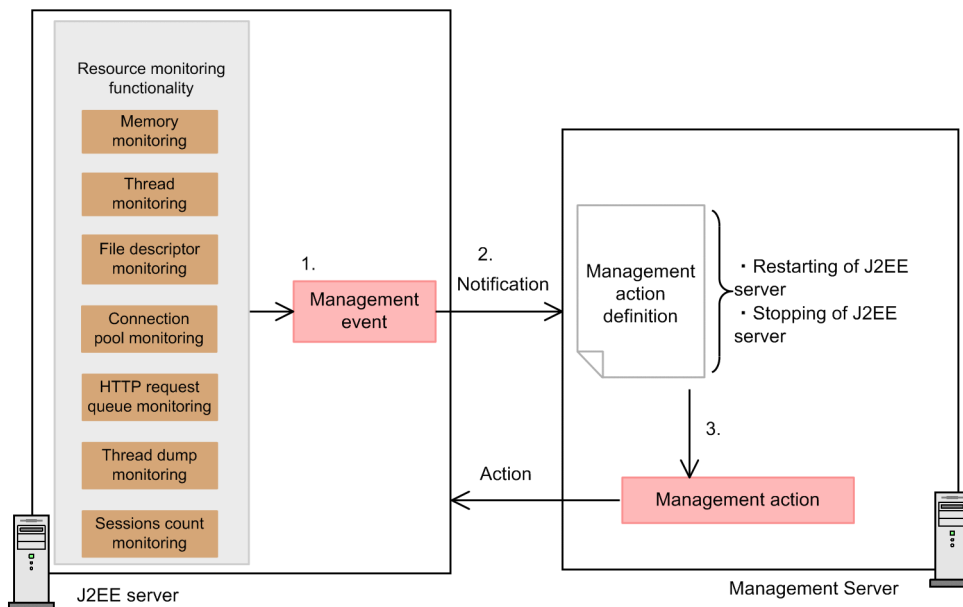
The following messages that are output by the J2EE server or batch server do not trigger issuance of Management events:

- KDJE90001-E
- KDJE90002-E
- KDJE90003-E
- KDJE90005-W
- KDJE90006-W
- KDJE90009-W

For details on the messages, see the manual *uCosminexus Application Server Messages*.

The following figure shows the flow from the time a Management event is issued until a management action is executed in the Management Server:

Figure 9–1: Management event and action



In the example, a management event is issued when the message is output to indicate that the threshold value specified in the resource depletion monitoring functionality has been exceeded. The flow of processes in the figure are explained below:

1. A management event is issued when the threshold value specified in the resource to be monitored is exceeded.

2. The management event is notified to the Management Server.
3. The process is executed automatically according to the management action definitions defined in the Management Server.

In the definition of the management action, define the action corresponding to the Management event sent from the J2EE or batch server. The management action must be defined in advance.

If the resource depletion monitoring functionality is used to perform memory monitoring, symptoms that can trigger Full GC can be detected. If a symptom that can trigger Full GC occurs during memory monitoring, the symptom is reported to Management Server as a Management event. At this time, if an action that shuts down and restarts the services on the J2EE server or batch server is defined as a reaction to reception of that Management event, the action is automatically performed for the server. This helps to prevent the termination of the request processing.

9.3 Controlling the Execution of Management Action

This section describes how to control the execution of the management actions.

You can control the execution of management actions by restraining the execution of identical management actions at regular intervals and setting the maximum number of concurrent executions. In this way, you can prevent the repeated execution of identical management actions and the concentration of the execution of the Management actions.

The execution of management action refers to the process from the execution of the command, specified as the Management action, until the command terminates or is timed out.

The following table describes the method of controlling the execution of the Management actions:

Table 9–2: Controlling the execution of management action

Method of controlling	Description
Restraint time control	After executing the Management action, the execution of identical management actions is restrained within a fixed time. As a result, the management actions to be executed for the management event occurring before a fixed time are consolidated.
Controlling the number of concurrent executions	Restricts the number of concurrent executions of the identical management actions. As a result, it becomes possible to prevent repeated execution of the identical management actions during the execution of the management action. The number of synchronous executions is applied to each management action ID.

When the two methods mentioned above are specified at the same time, the execution of management actions is restrained within the restraint time even when the maximum number of concurrent executions is not reached.

The methods of controlling the management action are explained below with the help of examples:

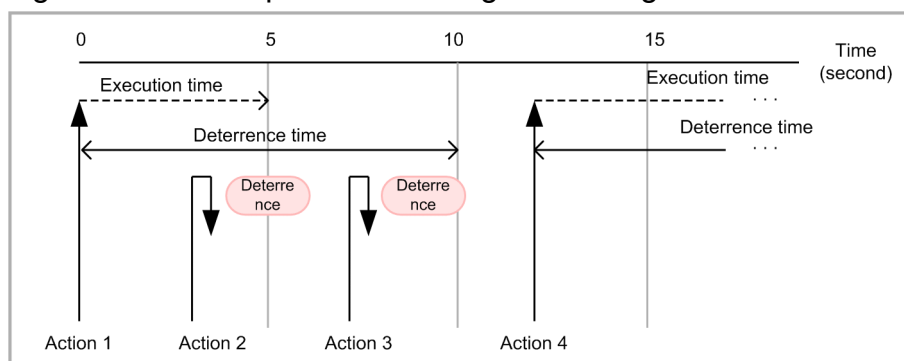
- **Example of setup-1**

Assume the following settings:

- Execution time of the management action: 5 seconds
- Restraint time: 10 seconds
- Maximum number of concurrent executions: 2

The following figure illustrates an example of controlling the management actions:

Figure 9–2: Example of controlling the management actions-1



Legend:

- > :Execution time of the Management action
- <-----> :Deterrence time of the Management action
- > :Management action

In the example illustrated in this figure, after action 1, the execution of action 2 and action 3 is restrained because the execution of identical actions is restrained for 10 seconds. Action 4 can be executed after the restraint time has elapsed.

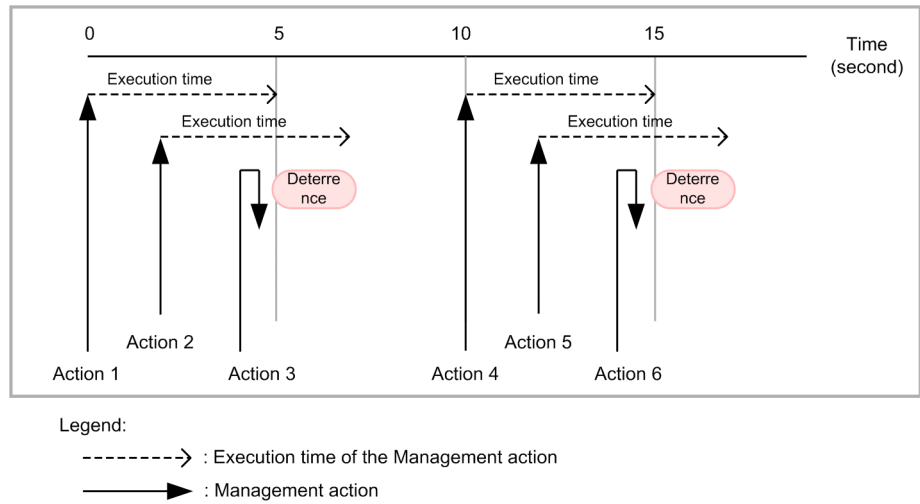
• **Setup example-2**

Assume the following settings:

- Execution time of the management action: 5 seconds
- Restraint time: 0 seconds (not restrained)
- Maximum number of concurrent executions: 2

The following figure illustrates an example of controlling the management actions:

Figure 9–3: Example of controlling the management actions-2



In the example illustrated in this figure, there is no restriction of restraint time, and therefore, action 2 is executed after the execution of action 1. At this time, the execution of action 3 is restrained, because the number of concurrent executions of the identical management actions has reached the maximum value. Action 4 can be executed after the execution of action 1 is complete. Action 5 and action 6 are controlled in the same way as action 2 and action 3.

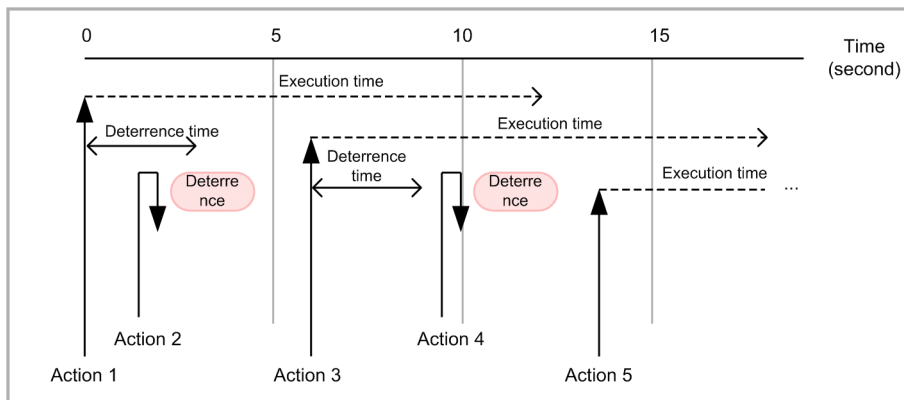
• **Setup example-3**

Assume the following settings:

- Execution time of the management action: 12 seconds
- Restraint time: 3 seconds
- Maximum number of concurrent executions: 2

The following figure illustrates an example of controlling the management actions:

Figure 9-4: Example of controlling the management actions-3



Legend:

- > : Execution time of the Management action
- <-----> : Deterrence Time of the Management action
- > : Management action

In the example illustrated in this figure, execution of action 2 is restrained because execution of identical management actions is restrained for 3 seconds after the execution of action 1. Execution of action 4 is restrained after the execution of action 3, because the number of concurrent executions of identical management actions has reached the maximum value. Action 5 can be executed after the execution of action 1 is complete.

9.4 Settings for automatic execution of processes by management events

This section describes settings for automatic execution of processes by Management events.

Management events are used to notify Management Server of failure, resource depletion, and other phenomena that occurred on the J2EE server or batch server in the management domain. All messages that are output by the J2EE server or batch server during its operation can be used as triggers to issue Management events. By defining the operations, to be performed when a Management event is notified, in the Management Server, the action can be executed automatically when the management event occurs. This action is called a management action.

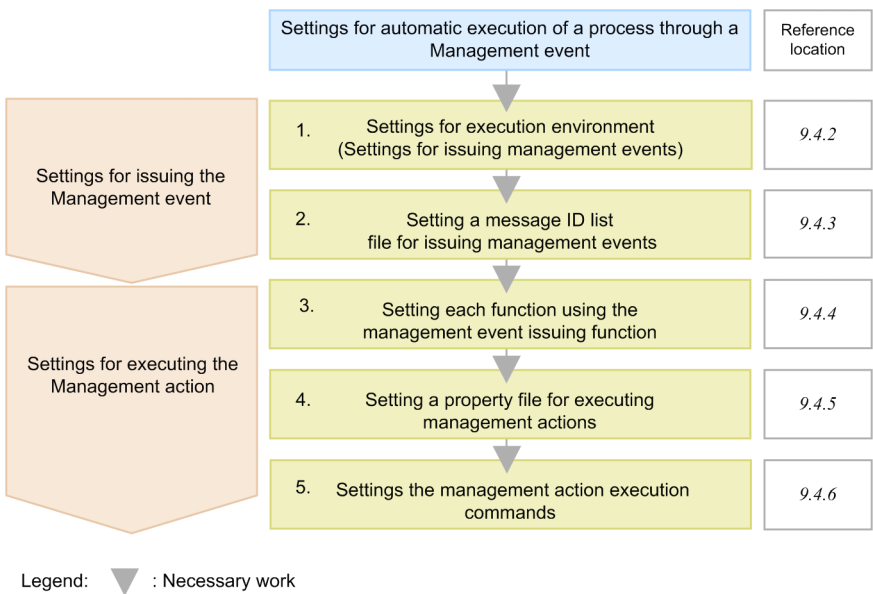
For example, if the resource depletion monitoring functionality is used to monitor memory usage, a Management event can be issued before Full GC occurs by using a message that is output as an alert when a threshold is exceeded. On the J2EE server or batch server, a Management action that shuts down and restarts the services can be defined. In this case, the defined action can be automatically performed, upon detection of a symptom that can trigger Full GC, to prevent the processing of a request from being stopped.

By detecting events such as a failure using a Management event and defining a corresponding action for the Management event as a Management action, you can automatically execute the processes for the occurred events such as actions for preventing failures.

9.4.1 Procedure to Set Automatic Execution of Processes by Management Events

The following figure describes the procedure for setting automatic execution of processes by Management events.

Figure 9–5: Procedure for setting the automatic execution of processes by management events



Stages 1. to 5. of the figure are explained below:

1. Make the settings to issue Management events in the execution environment.
Enable issuing of Management events and set the operation to be performed when Management events are issued.
For details, see [9.4.2 Settings for execution environment](#).
2. Set a message ID list file for issuing Management events.

In the list file, specify a message for which a Management event is to be issued. For details, see [9.4.3 Setting a Message ID List File for Issuing Management Events](#).

3. Set each function using the Management event issuing function.

Use the Management event issuing function to set the functions that notify Management events to the Management Server.

Furthermore, make the settings to execute Management actions corresponding to the set Management events. For details, see [9.4.4 Setting Each Function Using the Management Event Issuing Function](#).

4. Set a property file for executing Management actions.

Define the Management actions and mapping of message IDs with Management actions in a property file. For details, see [9.4.5 Setting a Property File for Executing Management Actions](#).

5. Set the management action execution commands.

Create a command file (batch file or shell script file) to code commands to be executed as Management actions. For details, see [9.4.6 Settings of Management Action Execution Commands](#).

9.4.2 Settings for execution environment

To issue a management event, you must specify settings for a J2EE server.

(1) Setting J2EE servers

You implement the J2EE server settings in the Easy Setup definition file. You specify the issuing definition for the management event in the `<configuration>` tag of the logical J2EE server (`j2ee-server`), in the Easy Setup definition file.

The following table describes the issuing definition for management events defined in the Easy Setup definition file:

Table 9–3: Issuing definition for management events in the Easy Setup definition file (J2EE server)

Items	Parameters to be specified	Setting contents
Application of issuing Management events	<code>ejbserver.manager.agent.MEventAgent.enabled</code>	Specifies whether to enable the issuing of management event. To issue a management event, specify <code>true</code> .
Setting the operation to be performed when a Management event is issued	<code>manager.mevent.send.timeout</code>	Specifies the send timeout value (seconds) for the management event.
	<code>manager.mevent.retry.limit</code>	Specifies the retry limit for the management event.
	<code>manager.mevent.retry.interval</code>	Specifies the retry interval (seconds) for the management event.
	<code>manager.mevent.send.max</code>	Specifies the maximum number of concurrently issued management events.
	<code>manager.mevent.message_id.list</code>	Specifies the path of the message ID list file.
	<code>manager.mevent.sender.bind.host</code>	Specifies the presence or absence of a fixed local address.

For details on the Easy Setup definition file and the parameters to be specified, see [4.3 Easy Setup definition file](#) in the *uCosminexus Application Server Definition Reference Guide*.

9.4.3 Setting a Message ID List File for Issuing Management Events

Code the message ID of the message for which you want to issue a Management event in the message ID list file used for issuing Management events. For details on the message ID list file used for issuing Management events, see 8.2.12 *Message ID list file for issuing Management events* in the *uCosminexus Application Server Definition Reference Guide*.

(1) Contents specified in the file

Code the message ID of the message for which you want to issue a Management event in the message ID list file used for issuing Management events.

A message that is output while the J2EE server and batch server are running can issue a Management event. The messages and user logs output during starting and stopping do not issue Management events.

If you omit the settings of the message ID list file used for issuing Management events, Management events are issued with the default message IDs. If you want to use default settings, you need not set the message ID list file used for issuing Management events. For details on default message IDs, see the description of the message ID list file used for issuing Management events. For message ID list file used for issuing Management events, see 8.2.12 *Message ID list file for issuing Management events* in the *uCosminexus Application Server Definition Reference Guide*.

(2) Storage location of the file

A sample of the message ID list file used for issuing Management events is stored in the following location. Copy the sample file to create a list file.

- **In Windows**

`Cosminexus-installation-directory\manager\config\templates\mevent.midlist.conf`

- **In UNIX**

`/opt/Cosminexus/manager/config/templates/mevent.midlist.conf`

You specify the path of the created list file in the `manager.mevent.message_id.list` parameter of the `<configuration>` tag for the logical J2EE server (`j2ee-server`) in the Easy Setup definition file.

(3) Example of creating a file

The following is an example of creating a message ID list file for issuing Management events:

```
# Monitoring of resources
# : Status of memory (Java Heap)
KDJE34500-W

# : Number of file descriptors
KDJE34520-W

# : Number of threads
KDJE34540-W
# : Number of thread dump files
-KDJE34580-W
KDJE34581-E

# : Number of HTTP requests in queue
-KDJE34621-W

# : Number of HTTP sessions
```



```
KDJE34640-W

# : Status of connection pool
-KDJE34660-W
KDJE34661-W

# Monitoring of execution time of user program
KDJE52702-W
KDJE52703-W
KDJE52705-W
KDJE52713-E
```

If you code a plus (+) sign before a message ID, that message issues a Management event. If you code a minus (-) sign before a message ID, that message will not issue a Management event. If you omit plus (+) and minus (-) signs, that message issues a Management event.

9.4.4 Setting Each Function Using the Management Event Issuing Function

Set the functions that issue Management events.

As an example, this subsection describes the required settings to issue management events in the following functions:

- Function to monitor resource depletion
- Function to monitor the execution time of J2EE applications

(1) Settings for monitoring resource depletion

Specify the resource-monitoring interval and threshold value to monitor resources and make the settings so that a Management event is issued when the threshold is exceeded.

For details on the functionality and settings to monitor resource depletion, see [4.3 Resource depletion monitoring functionality and output of resource depletion monitoring information](#).

(2) Settings for monitoring the J2EE application execution time

When monitoring the execution time of a request make the settings so that if an error such as indefinite loop occurs, a Management event is issued. For details on the functionality and settings for monitoring the J2EE application execution time, see [5.3 Monitoring and Canceling a J2EE Application During Runtime](#).

Specify timeout in the target of the method timeout function such as request processing of Web applications and method invocation processing of EJBs. For details on the settings, see [5.3.8 Precautions When Implementing](#).

9.4.5 Setting a Property File for Executing Management Actions

The definition of the management action, and the mapping between the message IDs and management actions are defined in the *property file for executing management action* (`maction.properties`). For `maction.properties`, see [8.2.10 maction.properties \(Property file for execution of Management actions\)](#) in the *uCosminexus Application Server Definition Reference Guide*.

(1) Storage location of the file

The storage location of `maction.properties` is as follows:

- **In Windows**

`Cosminexus-installation-directory\manager\config\maction.properties`

- **In UNIX**

`/opt/Cosminexus/manager/config/maction.properties`

(2) Example of setting the file

The following is an example of setting the property file used for executing Management actions in Windows:

```
# Defining the Management action
maction.restart.command=c:\\tmp\\command1.bat
maction.restart.timeout=12
maction.restart.timeout.forced_stop=true
maction.restart.exclusive_time=60
maction.restart.max_executable_actions=1

# Mapping of the message ID and Management action
maction.message.KDJE11111-E.mactions=restart
maction.message.KDJE22222-E.mactions=restart

# Mapping of the logical server and Management action
maction.server.j2ee1.mactions=restart
maction.server.j2ee2.mactions=restart
maction.server.j2eeClstr1.mactions=restart
```

This setting example defines `restart` as an ID to identify Management action (Management action ID). This section describes the operation and settings of the Management action of `restart`:

- Run the command file `command1.bat` as a command to execute the Management action.
- Set 12 seconds as a timeout for the Management action execution command.
- If the Management action execution command is timed out, the command is forcefully terminated.
- Set the suppression time of the Management action as 60 seconds, and the number of concurrent executions of Management actions as 1.[#]
- Execute this management action if messages `KDJE11111-E` and `KDJE22222-E` are output from the J2EE server `j2ee1`, `j2ee2`, and `j2eeClstr1`.

#

Management actions are distinguished by Management action IDs, and you can execute the same Management action for multiple servers and for different message IDs. By setting the suppression time and the number of concurrent executions, and controlling the execution of Management actions, you can prevent the aggregation and duplication of Management actions. For details on the execution control, see [9.3 Controlling the Execution of Management Action](#).

(3) Notes

The following are the precautions related to the property file for executing Management actions:

- **Priority of properties**

The logical server and cluster specified by the following keys contain the inclusive relation of the J2EE server < J2EE server cluster < service unit < physical tier:

- `maction.server.Logical-server-name.mactions`
- `maction.unit.Web-system-name.service-unit-name.mactions`
- `maction.tier.Web-system-name.Physical-tier-type-name.mactions`

Therefore, to define different Management actions for each logical server containing inclusive relationships such as a J2EE server cluster and J2EE servers that are the components of that J2EE server cluster, execute any one of the Management actions with the priority order described below:

1. A J2EE server from where a Management event is issued
2. A J2EE server cluster including the J2EE server from where a Management event is issued
3. A service unit including the J2EE server from where a Management event is issued
4. A physical tier including the J2EE server from where a Management event is issued

A Web system, service unit, and physical tier are the concepts to build a system using the Smart Composer function. For the Smart Composer functionality, see *1.1.3 Smart Composer functionality* in the *uCosminexus Application Server System Setup and Operation Guide*.

- **Specification order of Management actions**

If the specification order of Management actions differs in the following two keys, the specification order of the `maction.message.message-ID.mactions` key is given priority:

- `maction.message.message-ID.mactions`
This key maps message IDs with Management actions.
- `maction.server.Logical-server-name.mactions`
This key maps logical servers with Management actions.

(Examples)

In this example, `act1` is given priority.

```
maction.message.KDJE99999-E.mactions=act1,act2
```

```
maction.server.J2EE01.mactions=act3,act2,act1
```

9.4.6 Settings of Management Action Execution Commands

You can code the commands to be executed as Management actions in the command files (batch files or shell script files). Create command files as required. Note that for batch servers, you cannot use request scheduling by CTM, and therefore, CTM- related description is not applicable.

The following are the environment variables that can be used in the command files and samples of command files:

(1) Environment variables that can be used in command files

The following table lists the environment variables that you can use in the command files:

Table 9–4: Environment variables that can be used in the command files of the management action execution commands

Environment variable	Description
<code>COSMI_MNG_MACT_LSNAME</code>	Logical server name from where the Management event is issued.

Environment variable	Description
COSMI_MNG_MACT_HOST	Host name of a logical server from where the Management event is issued.
COSMI_MNG_MACT_MSG_ID	Message ID issued by the Management event.
COSMI_MNG_MACT_MSG_TEXT	Message text issued by the Management event.
COSMI_MNG_MACT_CTM	Logical CTM name specified for the logical server from where the management event is issued.
COSMI_MNG_MACT_WEBSYSTEM	Web system name to which the logical server, from where the Management event is issued, belongs.
COSMI_MNG_MACT_UNIT	Service unit name to which the logical server, from where the Management event is issued, belongs.
COSMI_MNG_MACT_TIER	Physical tier type to which the logical server, from where the Management event is issued, belongs.
COSMI_MNG_MACT_OPTN	An option string notified by the Management event. The embedded characters of the message are used as option string. N is an integer from 0 and above, and the environment variable expressing the first option string is COSMI_MNG_MACT_OPT0. For details on the set strings, see the information displayed in the variable values of the message text of messages that issue Management events.
COSMI_MNG_MACT_MNGSVR_PORT	HTTP port number of a Management Server.
COSMI_MNG_MACT_RNAME	Actual server name of the logical server from where the Management event is issued.
COSMI_MNG_MACT_NS_HOSTPORT	Host name and port number of the naming service used by the logical server from where the Management event is issued. For example: HostA:900, when the host name is HostA and the port number is 900.

(2) Samples of command files

Reference the provided samples of command files and create a command file. The samples are stored in the following location:

- **In Windows**

Cosminexus-installation-directory\manager\examples\maction

- **In UNIX**

/opt/Cosminexus/manager/examples/maction

For using a sample, change the arguments of the `mngsvrutil` command (such as `-m`, `-u`, and `-p` options), and the storage destination of a log file in conformity with the environment.

(a) Sample to restart the server that issues Management events

File name of a sample

- In Windows: `mActionSample_restartServer.bat`
- In UNIX: `mActionSample_restartServer.sh`

Operating the sample

Restart the logical server from where the Management event is issued.

You can use this sample to issue Management events with the resource depletion monitoring function.

1. The resource depletion monitoring functionality monitors the memory usage of a Java VM, and outputs a message if Full GC is likely to occur when the threshold value that is set is exceeded.
2. The Management event issuing function executes a Management action after issuing a Management event in correspondence with the output message, and restarts the J2EE server from where the Management event is issued.

By terminating the J2EE server from where the Management event is issued, you can block the J2EE application. Furthermore, if the J2EE server uses CTM for the load balancing of requests, you can terminate the J2EE server for distributing new requests to other J2EE servers.

The sample to restart the server from where the Management event is issued is described below:

- In Windows

```
(Omitted)
...
rem Management action sample for restart server.

setlocal

set LSNAME=%COSMI_MNG_MACT_LSNAME%
set MSGID=%COSMI_MNG_MACT_MSG_ID%

set CJCLDELLOG=%COSMINEXUS_HOME%\CC\client\bin\cjcldellog.bat
set LOG_ROOT_DIR=%SystemDrive%\<MyLogDir>
if not exist "%LOG_ROOT_DIR%" mkdir "%LOG_ROOT_DIR%"
call "%CJCLDELLOG%" -t 30d -f "%LOG_ROOT_DIR%"
set LOG_DIR=%LOG_ROOT_DIR%\%date:/%=
if not exist "%LOG_DIR%" mkdir "%LOG_DIR%"
set LOG=%LOG_DIR%\%time::=%_%LSNAME%_%MSGID%.txt

echo %0 > "%LOG%"

set MNGSVR=localhost:%COSMI_MNG_MACT_MNGSVR_PORT%
set UID=<User-id>
set PWD=<Password>

echo mngsvrutil.exe -m %MNGSVR% -u %UID% -p %PWD% -t %LSNAME% -s stop serv
er >> "%LOG%" 2>&1
mngsvrutil.exe -m %MNGSVR% -u %UID% -p %PWD% -t %LSNAME% -s stop server >
> "%LOG%" 2>&1
set RET=%ERRORLEVEL%
echo %RET% >> "%LOG%"

if not %RET% == 0 goto END
echo mngsvrutil.exe -m %MNGSVR% -u %UID% -p %PWD% -t %LSNAME% -s start ser
ver >> "%LOG%" 2>&1
mngsvrutil.exe -m %MNGSVR% -u %UID% -p %PWD% -t %LSNAME% -s start server >
> "%LOG%" 2>&1
set RET=%ERRORLEVEL%
echo %RET% >> "%LOG%"

:END
exit %RET%
```

- In UNIX

```

(Omitted)
...
# Management action sample for restart server.

LSNAME=${COSMI_MNG_MACT_LSNAME}
MSGID=${COSMI_MNG_MACT_MSG_ID}

LOG_ROOT_DIR=/tmp/<MyLogDir>
if [ ! -d ${LOG_ROOT_DIR} ]; then
    /bin/mkdir ${LOG_ROOT_DIR}
fi
/usr/bin/find ${LOG_ROOT_DIR}/* -depth -mtime +30 | /usr/bin/xargs /bin/rm -fr
LOG_DIR=${LOG_ROOT_DIR}/`/bin/date +%y%m%d`
if [ ! -d ${LOG_DIR} ]; then
    /bin/mkdir ${LOG_DIR}
fi
LOG=${LOG_DIR}/`/bin/date +%H%M%S`_${LSNAME}_${MSGID}.txt

echo $0 > "${LOG}"

MNGSVR=localhost:${COSMI_MNG_MACT_MNGSVR_PORT}
USERID=<User-id>
PASSWD=<Password>

echo ./mngsvrutil -m ${MNGSVR} -u ${USERID} -p ${PASSWD} -t ${LSNAME} -s start server >> "${LOG}" 2>&1
./mngsvrutil -m ${MNGSVR} -u ${USERID} -p ${PASSWD} -t ${LSNAME} -s stop server >> "${LOG}" 2>&1
RET=$?
echo ${RET} >> "${LOG}"

if [ $RET -eq 0 ]; then
    echo ./mngsvrutil -m ${MNGSVR} -u ${USERID} -p ${PASSWD} -t ${LSNAME} -s start server >> "${LOG}" 2>&1
    ./mngsvrutil -m ${MNGSVR} -u ${USERID} -p ${PASSWD} -t ${LSNAME} -s start server >> "${LOG}" 2>&1
    RET=$?
    echo ${RET} >> "${LOG}"
fi

exit ${RET}

```

(b) A sample for blocking a service unit belonging to the server issuing Management events

Reference the sample provided for blocking the service units belonging to the server from where the management event is issued, and create a sample. The sample file name and operations are as follows:

File name of a sample

- In Windows: mActionSample_closeUnit.bat
- In UNIX: mActionSample_closeUnit.sh

Operating the sample

When errors such as resource depletion occur in a J2EE server belonging to a service unit, block the server from where the Management event is issued.

The sample for blocking service units belonging to the server from where the Management event is issued is described below:

- In Windows

```
(Omitted)
...
rem Management action sample for close unit.

setlocal

set LSNAME=%COSMI_MNG_MACT_LSNAME%
set MSGID=%COSMI_MNG_MACT_MSG_ID%
set WEBSYSTEM=%COSMI_MNG_MACT_WEBSYSTEM%
set UNIT=%COSMI_MNG_MACT_UNIT%

set CJCLDELLOG=%COSMINEXUS_HOME%\CC\client\bin\cjcldellog.bat
set LOG_ROOT_DIR=%SystemDrive%\<MyLogDir>
if not exist "%LOG_ROOT_DIR%" mkdir "%LOG_ROOT_DIR%"
call "%CJCLDELLOG%" -t 30d -f "%LOG_ROOT_DIR%"
set LOG_DIR=%LOG_ROOT_DIR%\%date:/%=%
if not exist "%LOG_DIR%" mkdir "%LOG_DIR%"
set LOG=%LOG_DIR%\%time:=%_%LSNAME%_%MSGID%.txt

echo %0 > "%LOG%"

if "%WEBSYSTEM%" == "" goto :ERR
if "%UNIT%" == "" goto :ERR

set MNGSVR=localhost:%COSMI_MNG_MACT_MNGSVR_PORT%
set UID=<User-id>
set PWD=<Password>

echo cmx_stop_target.exe -m %MNGSVR% -u %UID% -p %PWD% -mode ALL -s %WEBSY
STEM% -unit %UNIT% >> "%LOG%" 2>&1
cmx_stop_target.exe -m %MNGSVR% -u %UID% -p %PWD% -mode ALL -s %WEBSYSTEM
% -unit %UNIT% >> "%LOG%" 2>&1
set RET=%ERRORLEVEL%
echo %RET% >> "%LOG%"
goto :END

:ERR
set RET=2

:END
exit %RET%
```

- In UNIX

```
(Omitted)
...
# Management action sample for close unit.

LSNAME=${COSMI_MNG_MACT_LSNAME}
```

```

MSGID=${COSMI_MNG_MACT_MSG_ID}
WEBSYSTEM=${COSMI_MNG_MACT_WEBSYSTEM}
UNIT=${COSMI_MNG_MACT_UNIT}

LOG_ROOT_DIR=/tmp/<MyLogDir>
if [ ! -d ${LOG_ROOT_DIR} ]; then
    /bin/mkdir ${LOG_ROOT_DIR}
fi
/usr/bin/find ${LOG_ROOT_DIR} -depth -mtime +30 -mindepth 1 -exec /bin/rm
-fr {} \;
LOG_DIR=${LOG_ROOT_DIR}/`/bin/date +%y%m%d`
if [ ! -d ${LOG_DIR} ]; then
    /bin/mkdir ${LOG_DIR}
fi
LOG=${LOG_DIR}/`/bin/date +%H%M%S`_${LSNAME}_${MSGID}.txt

echo $0 > "${LOG}"

if [ "${WEBSYSTEM}" = "" -o "${UNIT}" = "" ]; then
    exit 2
fi

MNGSVR=localhost:${COSMI_MNG_MACT_MNGSVR_PORT}
USERID=<User-id>
PASSWD=<Password>

echo ./cmx_stop_target -m ${MNGSVR} -u ${USERID} -p ${PASSWD} -mode ALL
-s ${WEBSYSTEM} -unit ${UNIT} >> "${LOG}" 2>&1
./cmx_stop_target -m ${MNGSVR} -u ${USERID} -p ${PASSWD} -mode ALL -s ${WE
BSYSTEM} -unit ${UNIT} >> "${LOG}" 2>&1
RET=$?
echo ${RET} >> "${LOG}"

exit ${RET}

```

(3) Operations of Management action execution commands

The following section describes the operation of management action execution commands:

- The Management Server does not acquire the standard output and standard error output from the commands to execute a Management action. To acquire the standard output and standard error output of a command, information must be output to a file during command execution.
- The Management Server runs commands to execute Management actions. The executing user is the user who executes the Management Server. The environment variables are inherited from the environment variables set in the Management Server.
- The *working directory for the management action execution commands* is *Cosminexus-installation-directory\manager\bin* (in Windows) or */opt/Cosminexus/manager/bin* (in UNIX).

10

Collecting CTM Statistics (INTENTIONALLY DELETED)

INTENTIONALLY DELETED

10.1 INTENTIONALLY DELETED

INTENTIONALLY DELETED

11

Output of the Console Log

This chapter describes the output functionality of the console log such as the standard output and the standard error output of processes invoked by the Administration Agent.

11.1 Organization of this chapter

This chapter is organized as the following table.

Table 11–1: Organization of the chapter (Output of the console log)

Category	Title	Reference
Description	Output target of the console log	11.2
Settings	Settings for acquiring the console log	11.3

Note:

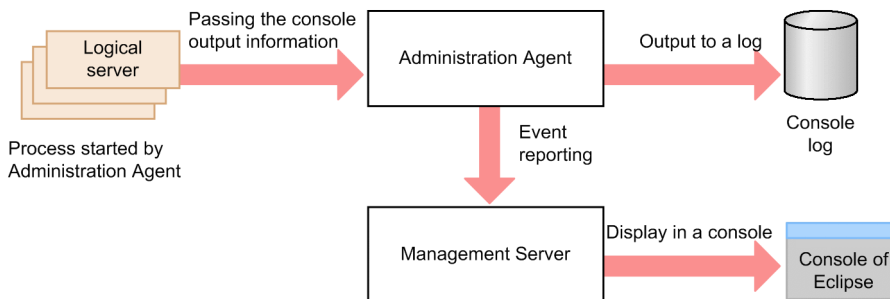
The function-specific explanation is not available for "Implementation", "Operations", and "Precautions".

11.2 Output target of the console log

This section describes the output target of the console log.

When you use the Management Server for operations, you can acquire the standard output or the standard error output of the processes executed by the Administration Agent as the *console output information*. The following figure shows how to get the console output information:

Figure 11–1: Acquiring the console output information



The console output information that the Administration Agent obtains from each process is output in the log file (*console log*).

When using Eclipse plugins that use Management Server, the console output information is also displayed on the Eclipse console via the Management Server. The console output information is notified to the Management Server as an event. The event is retained in the event queue, and is then extracted from the sequential event queue and notified to the Management Server. If the event cannot be notified due to reasons such as the Management Server is not running, the event is resent until it can be notified. If the events in the event queue exceed the queue size, the earlier events are removed from the queue.

11.2.1 Target operations of the console log output

When operating the logical server, J2EE application or the resource by using the following methods, you can obtain the console log because the Administration Agent starts or stops the processes:

- When using the Smart Composer functionality commands
- When using the management command (`mngsvrutil`) of the Management Server
- When using the Eclipse plugins that use Management Server

11.2.2 Target processes of the console log output

The following processes are targets for console log output:

- The processes that configure the logical server
- The processes that operate the J2EE application or the resource

In Windows, however, you cannot obtain the console log from the processes that have been indirectly started. For more information on the logical server that is indirectly invoked, see [2.3 Mechanism of Starting and Stopping the Logical Server](#).



Important note

If a process started by Administration Agent outputs a large amount of text data in a short time, the target processes might stop temporarily, depending on the time when Administration Agent performs GC.

11.3 Settings for Acquiring the Console Log

When using the Management server, you can acquire the standard output, standard error output of the process started by the Administration Agent as the console output information. This subsection describes the changes in the settings for acquiring the console log. When changing the settings for the output of the log files (console log) of the console output information, specify the following keys in `adminagent.properties`:

- `adminagent.process.consolelog.enabled`
Specify whether to output the console output information. By default, `true` is specified and output occurs.
- `adminagent.userserver.process.console_log.enabled`
Specify whether to output the console output information of the logical J2EE server to the console log. By default, `false` is specified and output does not occur.
If you specify `true` in the `adminagent.j2ee.process.console_log.enabled` key, you also need to specify `true` in the `adminagent.process.consolelog.enabled` key.
- `adminagent.userserver.process.console_event.enabled`
Specify whether to output the console output information of the logical user server to the console log. By default, `false` is specified and output does not occur.
If you specify `true` in the `adminagent.userserver.process.console_log.enabled` key, you also need to specify `true` in the `adminagent.process.consolelog.enabled` key.
- `adminagent.j2ee.process.console_log.enabled`
Specify whether to display the console output information of the logical J2EE server in the Eclipse plugin that uses Management Server. By default, `false` is specified and the console output information is not displayed.
If you specify `true` in the `adminagent.j2ee.process.console_event.enabled` key, you also need to specify `true` in the `adminagent.process.consolelog.enabled` key.
- `adminagent.userserver.process.console_log.enabled`
Specify whether to display the console output information of the logical user server in the Eclipse plugin that uses Management Server. By default, `false` is specified and the console output information is not displayed.
If you specify `true` in the `adminagent.userserver.process.console_event.enabled` key, you also need to specify `true` in the `adminagent.process.consolelog.enabled` key.
- `adminagent.process.consolelog.filenum`
Specify the number of console logs.
- `adminagent.process.consolelog.filesize`
Specify the maximum size for each console log.

12

Operating a JP1 Integrated System (INTENTIONALLY DELETED)

INTENTIONALLY DELETED

12.1 INTENTIONALLY DELETED

INTENTIONALLY DELETED

13

Centralized Monitoring of the System (Integrating with JP1/IM) (INTENTIONALLY DELETED)

INTENTIONALLY DELETED

13.1 INTENTIONALLY DELETED

INTENTIONALLY DELETED

14

Job-based Automatic Operations of a System (Integrating with JP1/AJS) (INTENTIONALLY DELETED)

INTENTIONALLY DELETED

14.1 INTENTIONALLY DELETED

INTENTIONALLY DELETED

15

Linking with Cluster Software

This chapter describes the operations of the systems linked with the cluster software.

15.1 Organization of this chapter

This chapter is organized as the following table.

Table 15–1: Organization of this chapter (Linking with cluster software)

Category	Title	Reference
Description	Operations that can be implemented by linking with cluster software	15.2
	Prerequisites for linking with the cluster software	15.3

Note:

The function-specific explanation is not available for "Implementation", "Settings", "Operations", and "Notes".

15.2 Operations that can be implemented by linking with cluster software

This section describes an overview about the settings for integrating with the cluster software.

The cluster software is a program that aims to improve the reliability and operating rate of the system, and implements switching of the systems containing server programs. In Application Server, you can integrate the system with the following cluster software:

- **Windows Server Failover Cluster**

This cluster software is compatible with Windows Server 2019 Standard or Datacenter, and with Windows Server 2016 Standard or Datacenter.

- **HA monitor**

This cluster software is compatible with AIX or Linux.

When you operate a system that is linked with the cluster software, if a failure occurs in the Application Server, the application server switches automatically. Therefore, you can execute the recovery process of the Application Server wherein the failure has occurred using the standby recovery server. Also, when failure occurs in the Management Server, you can switch to the Management Server that is set up in the standby mode. As a result, you can reduce the system downtime, and can improve the reliability and operating rate of the system.

Important note

When monitoring the failure of Administration Agent and Management Server, and then switching the systems linked with the cluster software, do not set up automatic restart in Administration Agent and Management Server.

In the systems of the Application Server, you can operate the systems linked with the cluster software for reducing the system downtime as per the failure. You can continue the operations even when failure occurs. In the Application Server, the following node switching functionality is used based on the linkage with cluster software:

- 1-to-1 node switching system

- 1-to-1 node switching of the Management Server

Using a configuration in which the Management Server of the executing node and the Management Server of the standby node have the 1-to-1 ratio, you can continue the operations by switching the nodes when a failure occurs in the Management Server of the executing node.

- Mutual node switching

In this system, the Application Server of the executing node and the Application Server of the standby node are specified in the 1-to-1 ratio, and while each Application Server operates as an executing node, each Application Server functions as the standby node for the other Application Server. This enables the execution of economical operations using a few Application Server machines.

- Node switching system for the host unit management model

In this system, deploy multiple (1-to-N) executing node Application Server machines (hosts) and one standby node Application Server machine (host), and then deploy Management Server and Administration Agent in each Application Server machine. When a failure occurs in the Application Server machine of the executing node, you can continue the operations by performing node switching with the Application Server machine of the standby node.

Reference note

You use the following cluster software to build and operate the 1-to-1 node switching system and node switching system for the host unit management model in Application Server:

- Windows Server Failover Cluster (In Windows)
- HA monitor (In AIX or Linux)

You must understand the following terminology when you link a system with the cluster software.

In a cluster software, the generic name for the entire system that contains programs and communication equipment apart from the hardware required for the business processing is called a *node*. The names for distinguishing the nodes running in systems are the executing node and the standby node.

- **Executing node**

A node that has a server running currently. If node switching occurs, this node becomes the standby node.

- **Standby node**

A node that has a server currently in the standby mode. If node switching occurs, this node becomes the executing node.

Furthermore, the names for distinguishing the nodes in the system settings when the system is not running are the active node and the spare node.

- **Active node**

A node that is initially started as an executing node.

- **Spare node**

A node that is initially started as a standby node.

In Application Server, by integrating the systems with the cluster software, you can operate the system using the node-switching configuration of the J2EE application execution environment.

The following table lists and describes the references of the node switching functionality according to the linkage with the cluster software.

Table 15–2: Node switching functionality based on linkage with cluster software

Functionality		Reference
1-to-1 node switching system	1-to-1 node switching of Management Server	17.2
	Mutual node switching	18.2
Node switching system for the host unit management model		16.2

15.2.1 Systems that operate the executing node and the standby node in the 1-to-1 ratio (1-to-1 node switching system)

The *1-to-1 node switching system* is a system in which the executing node and the standby node are in 1-to-1 ratio. Application Server supports the operations in the 1-to-1 node switching system of Management Server. However, in the execution environment of the batch application, the Management Server is not deployed, and therefore, the operations cannot be executed in the 1-to-1 node switching system of the Management Server.

In the 1-to-1 node switching system, if some failure occurs in the executing node, the cluster software detects the failure, automatically switches to the standby node, and continues the operations. Also, even when a failure does not occur, if the running system has preventive maintenance or other requirements, you can switch to the standby node in a planned manner using the operations of the operator.

For details on the 1-to-1 node switching system, see [17. 1-to-1 Node Switching System \(Linked with Cluster Software\)](#).

15.2.2 System operating on mutual standby (mutual node switching systems)

The *mutual node switching system* is a system in which the configuration of the 1-to-1 node switching system is used, and while each server operates as an active node, each server become the spare node for the other server. The Application Server supports operations of the mutual switching systems of the Application Server.

In the mutual node switching system, a J2EE application or batch application is operated respectively on each server and when a failure occurs, the node is switched automatically to the mutual standby node, and after that the operations are continued. Also, same as in the case of the 1-to-1 node switching system, even when failure does not occur, if the running system has preventive maintenance or other requirements, you can switch to the standby node in a planned manner using the operations of the operator.

For details on the mutual node switching system, see [18. Mutual Node Switching System \(Linked with Cluster Software\)](#).

15.2.3 System that operates the standby node as the recovery server (N-to-1 recovery system)

The *N-to-1 recovery system* is a system in which a recovery server is deployed as the standby node with respect to multiple (N number of) Application Servers of the executing node in a cluster configuration.

Execute a business in Application Server of the executing node. If a failure occurs in Application Server in which you are executing a business, the recovery server concludes the transaction that was in progress in Application Server in which the failure occurred.

Note that you cannot use the N-to-1 recovery system in a batch server.

For details on the N-to-1 recovery system, see the chapter [19. N-to-1 Recovery System \(Linked with Cluster Software\)](#).

15.2.4 System in which Application Server machines (host) of the executing node and the standby node operate in a N-to-1 ratio (Node switching system for the host unit management model)

The node switching system for the host unit management model supports the operations in a node switching system for the host unit management model with a configuration in which Management Server and Administration Agent are deployed in each of the N number of Application Servers of the executing node and also in Application Server of the standby node.

In the node switching system for the host unit management model, if a failure occurs in Management Server or Administration Agent of the executing node, the cluster software detects the failure, and continues the operations by switching to the standby node.

For details on the node switching systems for the host unit management model, see the chapter *16. Node Switching System for the Host Management Model (Linked with Cluster Software)*.

15.3 Prerequisites for linking with the cluster software

This section describes the prerequisites for operating a system linked with the cluster software.

15.3.1 Servers for which nodes are to be switched

The following table describes and lists the servers for which nodes are to be switched, when operating the systems linked with the cluster software:

Table 15–3: Servers for which nodes are to be switched

Node switching types	Servers for which nodes are to be switched	Example of system configurations when operating the systems by linking with cluster software
1-to-1 node switching system	Management Server	See <i>17.3.1 Example system configuration for a 1-to-1 node switching system</i> .
Mutual node switching system	Application server	See <i>18.3.1 Example of system configuration of the mutual node switching system</i> .
Node switching system for the host unit management model	Application Server of the host unit management model	See <i>16.2.1 Example of system configuration of the node switching system for the host management model</i> .

15.3.2 Usage of light transaction functionality

For linking with the cluster software, the types of node switching that you can use depend on whether the light transaction functionality is enabled. The following table lists the node switching types and the usage of the light transaction functionality. Note that the executing node and the standby node are required to be specified in the same operation mode.

Table 15–4: Node switching types and usage of the light transaction functionality

Node switching types	Light transaction	
	Enabled	Disabled
1-to-1 node switching system	Y	Y
Mutual node switching system	Y	Y
Node switching system for the host unit management model	Y	Y

Legend:

Y: Available

15.3.3 Operating methods of the systems

When the system is linked with the cluster software, regardless of the node switching types, you must use the Management Server to operate the system.

For details on the settings to link the systems with the cluster software, see [15.2 Operations that can be implemented by linking with cluster software](#).

Reference note

For the 1-to-1 node switching system of the Management Server, the cluster software monitors the Management Server.

If Management Server or Administration Agent is included in a node switching system, make sure that Management Server or Administration Agent starts before the system starts.

For details about how to set the preceding operation, see [2.6 Settings for starting and stopping the system](#).

15.3.4 Prerequisites for the executing node

In an N-to-1 node switching system, the executing node Application Server must satisfy the following prerequisites. However, the executing node does not have any prerequisites to satisfy in 1-to-1 node switching systems and mutual node switching systems.

- Use a global transaction.
- Use the CORBA Naming Service and the transaction service as an in-process.
- Save the status file of the transaction service in the directory of the shared disk device.

15.3.5 Prerequisites for the standby node

The prerequisites of the standby node are as follows:

When operating the systems linked with the cluster software, the standby node waits for node switching with the *cold standby*. The cold standby is a method in which a standby node server is started after the node switching occurs.

(1) In 1-to-1 node switching systems, mutual node switching systems, and node switching systems for the host unit management model

In 1-to-1 node switching systems, mutual node switching systems, and node switching systems for the host unit management model, Application Server related processes cannot be started in Application Server of the standby node. Therefore, note that you cannot execute the following operations:

- Deploying and undeploying applications in the J2EE server
Includes replacing J2EE applications
- Management operations by the Management Server and the Administration Agent

Also, you cannot access a shared disk device from Application Server of the standby node. When J2EE servers are used and when there is a global transaction, the definition information of the transaction service is stored in the shared disk device. However, you cannot change this definition information from Application Server of the standby node.

16

Node Switching System for the Host Management Model (Linked with Cluster Software)

This chapter describes the systems to be operated on the node switching system for the host management model.

16.1 Organization of this chapter

The following table describes the organization of this chapter:

Table 16–1: Organization of this chapter (Node switching system for the host management model (Integrating with cluster software))

Category	Title	Reference
Description	System configuration and operations of the node switching system for the host management model	16.2
Settings	Setting up the node switching system for the host management model	16.3
Operations	Starting and stopping the node switching system for the host management model	16.4

Note:

The function-specific explanation is not available for "Implementation" and "Precautions".

The *node switching system for the host management model* is a system in which you deploy multiple (1-to-N) executing node Application Server machines (hosts) and one standby node Application Server machine (host), and then deploy Management Server and Administration Agent on each node.

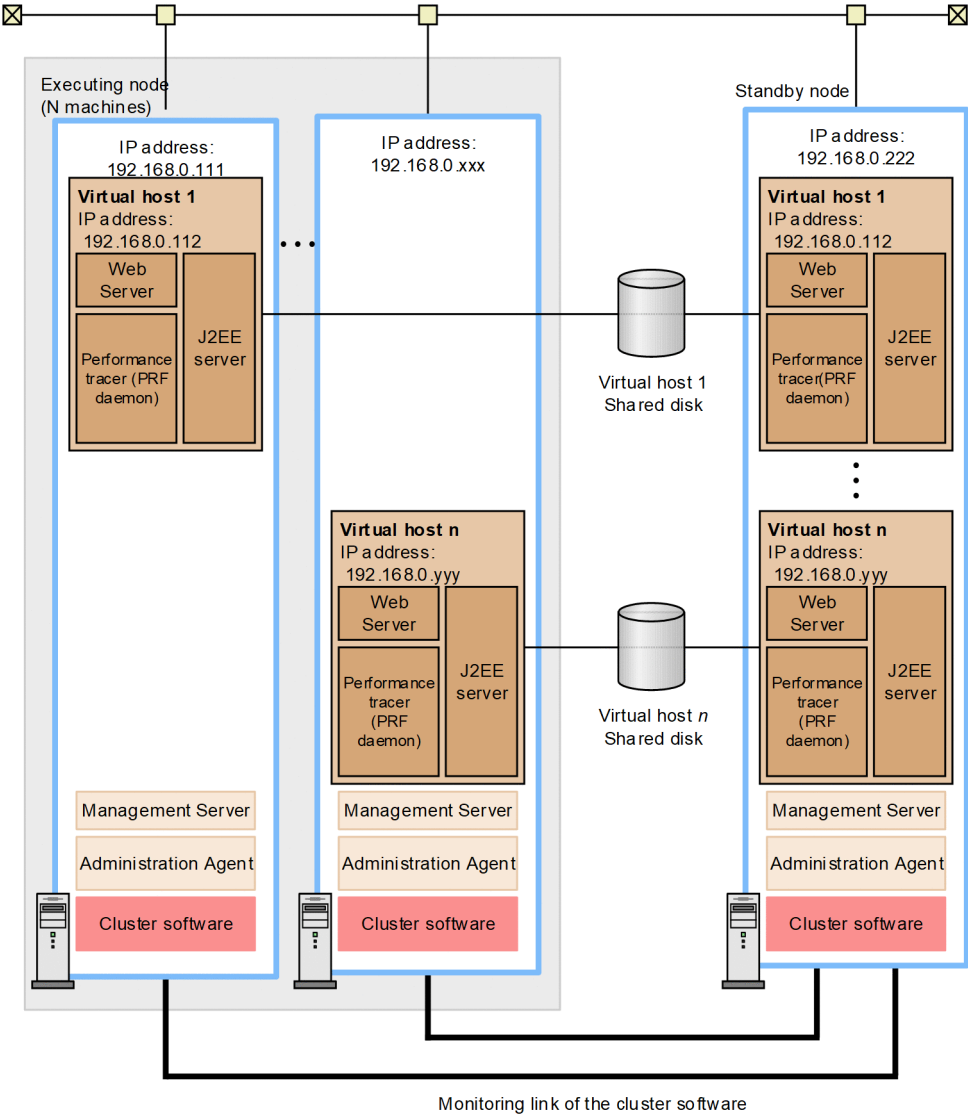
16.2 System configuration and operations of the node switching system for the host management model

This section shows an example configuration of the node switching system for the host management system and provides an overview of node switching procedure.

16.2.1 Example of system configuration of the node switching system for the host management model

The following figure shows a configuration example of the node switching system for the host management model of N executing nodes:

Figure 16–1: Example of system configuration of the node switching system for the host management model



The description of the node switching system for the host management model is as follows:

- This is a cluster configuration system in which N executing nodes and one standby node are deployed on the Application Server machine (host) and Management Server, and Administration Agent is deployed on each node. For every executing node Application Server, there are one or multiple servers with virtual host defined. On the other hand, multiple virtual hosts are defined for one standby node Application Server. For details on the virtual hosts, see [18.3.1 Example of system configuration of the mutual node switching system](#).

If a failure occurs in the executing node Application Server, the node switches to the standby node. For example, if failure occurs in Management Server of the virtual host 1 of the executing node in figure, the node is switched to the virtual host 1 of the standby node.

- A virtual IP address assigned by the cluster software is used for Application Server operation on the executing node and for Application Server operation on the standby node.

If you provide only one standby node for two or more executing nodes, you must specify real IP addresses for Management Server and Administration Agent so that the node on which they operate does not change when node switchover occurs. If you provide one standby node for one executing node, you can specify virtual or real IP addresses for Management Server and Administration Agent.

Note that you can execute the following operations in the node switching system for the host management model:

Using shared disk devices

A shared disk device is used to inherit the transaction information, such as the OTS status during node switching. Note that if the cluster software is an HA monitor, shared disk devices are not required when using the local transactions.

16.2.2 Node switching timing

Node switching is implemented at the following timing:

- When Administration Agent or Management Server are down^{#1}.
- When the automatic restart frequency of the logical server exceeds the automatic restart frequency specified during system setup^{#2}.
- When a failure occurs in the node hardware or cluster software.

#1

If automatic restart is specified for Administration Agent and Management Server, node switching is not executed during this time. For executing the node switching during this time, do not set up automatic restart for Administration Agent and Management Server.

#2

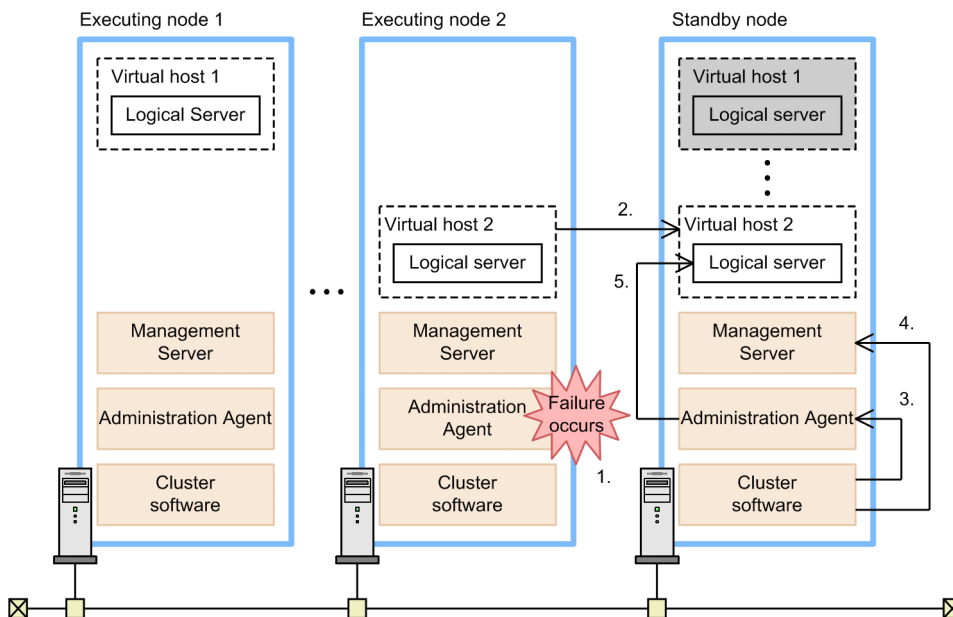
You must specify the settings for switching the node (`true`) in the `com.cosminexus.mngsvr.logical_server_abnormal_stop.exit` key of the `mserver.properties` (Management Server environment setup file), when the logical server status is "abnormal termination" (the automatic restart frequency is exceeded or when 0 is specified as the automatic restart frequency and a failure is detected). For details on the settings for `mserver.properties` (Management Server environment setup file), see [16.3.3 Editing the setup file](#). If automatic restart is specified for Management Server, node switching is not executed during this time. For executing the node switching during this time, do not set up automatic restart for Management Server. If automatic restart is specified, use the `mngautorun` command to cancel the settings. For details about the `mngautorun` command, see *mngautorun (Set up/canceling the set up of autostart and autorestart)* in the *uCosminexus Application Server Command Reference Guide*.

In a configuration in which two or more executing nodes exist, do not specify `true` for the `com.cosminexus.mngsvr.logical_server_abnormal_stop.exit` key. If you do so, multiple systems of these executing nodes might start on the standby node.

16.2.3 Node switching procedure

The following figure shows the node switching procedure for the host management model:

Figure 16–2: Node switching procedure



1. Detect the failure.

If the Administration Agent of the executing node 2 is down due to a failure, the cluster software of the standby node detects the failure of the Administration Agent of the executing node 2.

2. Switch the node.

The cluster software of the standby node switches the virtual host 2 of the executing node 2 to the virtual host 2 of the standby node. In this case, the shared disk device is switched and the IP address of the virtual host is inherited.

3. Invoke the Administration Agent of the standby node.

The cluster software on the standby node starts Administration Agent in only the case where a virtual IP address is used for Administration Agent.

4. Invoke Management Server of the standby node.

The cluster software on the standby node starts Management Server in only the case where a virtual IP address is used for Management Server.

5. Invoke the logical server.

The Administration Agent invokes the virtual host 2 logical server of the standby node.

The Administration Agent, Management Server, and logical server are invoked by the start server script registered in the cluster system. For details on the start server script, see the OS manual in use.

16.3 Setting up the node switching system for the host management model

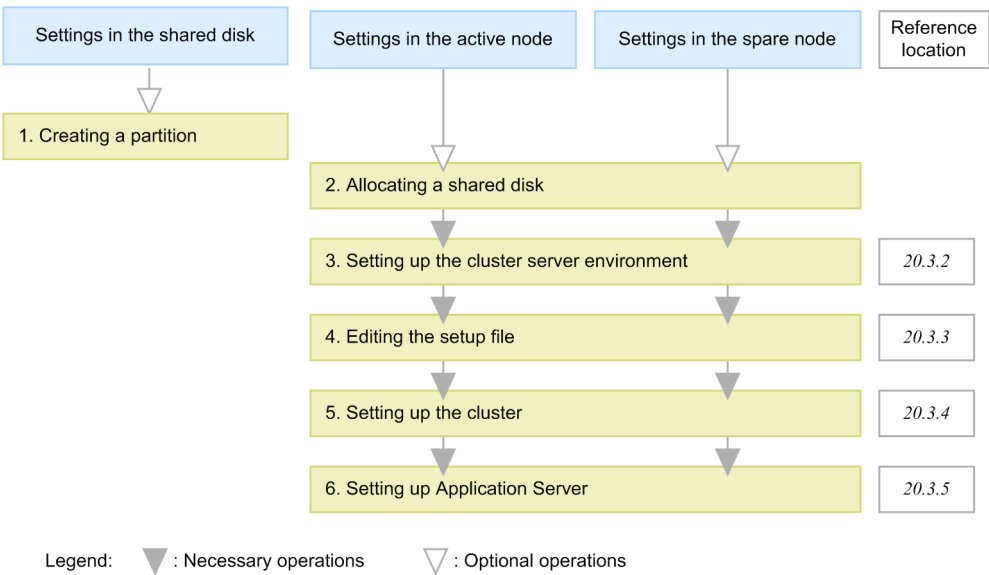
The node switching system for the host management model is a system in which you deploy multiple executing nodes (1-to-N) and one standby node on the Application Server machine (host), and then deploy Management Server and Administration Agent on each node.

This section describes how to set up the node switching system for the host management model.

16.3.1 Procedure for setting up the node switching system for the host management model

The following figure shows the procedure for setting up the node switching system for the host management model:

Figure 16–3: Procedure for setting up the node switching system for the host management model



The points 1 to 6 in the figure are described as follows:

1. Create a partition in the shared disk, and then build the file system.
When using a global transaction, create the storage location for the transaction information. Note that this step is not required when using the local transaction, if the HA monitor is used.
2. Allocate a shared disk to the system.
When allocating a shared disk to the system, check that you allocate the same drive character in the virtual host 1 of the active node 1 and the virtual host 1 of the standby node, and the virtual host 2 of the active node 2 and the virtual host 2 of the standby node.
3. Set up the cluster server environment in Management Server.
Set up the cluster server environment (each logical server and the hosts on which the logical servers are set up) in the Easy Setup definition file of the Smart Composer functionality of Management Server. For details, see [16.3.2 Cluster server environment settings](#).
4. Edit the setup file.

Set up various definition files used on the Administration Agent and Management Server. For details, see [16.3.3 Editing the setup file](#).

5. Set up the cluster.

Create the cluster software environment settings and the script files for monitoring the Management Server and Administration Agent. For details, see [16.3.4 Cluster settings](#).

6. Set up Application Server.

The J2EE application deploys Application Server in a cluster configuration and sets up the resource adapter. For details, see [16.3.5 Application Server settings](#).

Implement the above procedure for both, active node and standby node.

Reference note

The logical servers are deployed on different hosts, but externally operate as the same logical server. On the other hand, Management Servers operate separately on different hosts and have respective management domains. Define one virtual host in one management domain and set up as active node Application Server host and standby node Application Server host respectively. A virtual host controls the starting and stopping of Application Server using the Administration Agent, but allocates the same IP address as the one used for operations and is defined so that it appears to be the same host.

16.3.2 Cluster server environment settings

This subsection describes the points to be noted about Management Server when you specify the cluster server environment settings when integrating with the cluster software.

Reference note

You must set up the Management Server on the host where the Management Server will be used for the first time. For details about how to set up Management Server, see [4.1.14 Setting the management functionality](#) in the *uCosminexus Application Server System Setup and Operation Guide*.

(1) Settings in the Easy Setup definition file

Define each logical server and the hosts on which the logical servers are set up, in the Easy Setup definition file of the Smart Composer functionality of Management Server.

Note that you can also set up the cluster server environment by using the management portal. For details about operations via the management portal, see the manual *uCosminexus Application Server Management Portal User Guide*.

(a) Defining the hosts

Define the hosts in the management domain. Set up the virtual IP address to be used for the individual server operations in the `<host-name>` tag for the virtual host of the active node and standby node.

(b) Setting up the logical server

The points to be noted when setting up the logical J2EE server (`j2ee-server`) and logical Web server (`web-server`) are as follows:

- **Setting up the logical J2EE server (j2ee-server)**

Set up a virtual IP address and a real IP address in the parameters in the `<configuration>` tag of the logical J2EE server (j2ee-server) as and when required.

The following table describes the parameters to be specified in the `<configuration>` tag of the logical J2EE server (j2ee-server):

Table 16–2: Parameters to be specified in the `<configuration>` tag of the logical J2EE server (j2ee-server)

Parameter name	Content	Set IP address
<code>vbroker.se.iiop_tp.host</code>	Host name or IP address of the EJB container for J2EE servers	Virtual IP address
<code>webserver.connector.http.bind_host</code>	Local IP address used on the server for managing the Web container, or a resolvable local host name	Virtual IP address
<code>webserver.connector.nio_http.bind_host</code>	IP address or host name used for the NIO HTTP server	Virtual IP address
<code>webserver.connector.http.permitted.hosts</code>	IP address or host name of the host that is allowed to access the server for managing the Web container	Real IP address, Virtual IP address, localhost ^{#1}
<code>add.jvm.arg</code>	Parameters for invoking JavaVM ^{#2}	<code>-Djava.rmi.server.hostname = virtual-IP-address</code>

#1

The settings described here allow access from the same host as the logical J2EE server. To allow access from another host for operations, add the IP address or the name of that host.

#2

Set up the parameters for all the J2EE servers of the spare node. The settings need not be done for the active node. Also, in the 1-to-1 node switching system, the settings need not be done for both the active node and spare node.

- **Setting up the logical Web server**

Set up the following parameter in the `<configuration>` tag of the logical Web server (web-server) and fix the host. Set up the virtual IP address as the IP address.

- `Listen` parameter (IP address or host name that accepts the request)

16.3.3 Editing the setup file

This subsection describes the points to be noted when you edit the setup file used in Management Server, while integrating the system with the cluster software. For details about the `adminagent.properties` file, see 8.2.1 *adminagent.properties (Administration Agent property file)* in the *uCosminexus Application Server Definition Reference Guide*.

(1) Administration Agent settings

Among the items to be specified in `adminagent.properties` (Administration Agent property file), this section describes the settings to be noted when integrating with cluster software.

- `adminagent.adapter.bind_host`

If you provide only one standby for two or more executing nodes, you must specify a real IP address for the `adminagent.adapter.bind_host` key.

If you provide one standby node for one executing node, you can specify a virtual IP address or real IP address for the `adminagent.adapter.bind_host` key.

(2) Management Server settings

Among the items to be specified in `mserver.properties` (Management Server environment setup file), this section describes the settings to be noted when integrating with cluster software.

- `com.cosminexus.mngsvr.logical_server_abnormal_stop.exit`
Specify the settings (`true`) for switching the node when the logical server status is "abnormal termination" (when exceeds the automatic restart frequency or when 0 is specified as the automatic restart frequency and a failure is detected).

A settings example is as follows:

```
com.cosminexus.mngsvr.logical_server_abnormal_stop.exit = true
```

Important note

If automatic restart is specified for Management Server, node switching is not executed during this time. For executing the node switching during this time, do not specify automatic restart for Management Server.

In a configuration in which two or more executing nodes exist, do not specify `com.cosminexus.mngsvr.logical_server_abnormal_stop.exit=true`. If you do so, multiple systems of these executing nodes might start on the standby node.

- `webserver.connector.http.bind_host`
If you provide only one standby node for two or more executing nodes, you must specify a real IP address for the `webserver.connector.http.bind_host` key.
If you provide one standby node for one executing node, you can specify a virtual or real IP address for the `webserver.connector.http.bind_host` key.
- `mngsvr.myhost.name`
If you provide only one standby node for two or more executing nodes, you must specify a real IP address for the `mngsvr.myhost.name` key.
If you provide one standby node for one executing node, you can specify a virtual or real IP address for the `mngsvr.myhost.name` key.

(3) Settings for the property file for issuing Management events

The `mevent.logical-server-name.properties` file (Property file for issuing Management events) is a setup file related to the J2EE servers; therefore, the file is created after you set up the system using the Smart Composer functionality. Specify the following key after the system setup:

- `manager.mevent.send.host`
If you provide only one standby node for two or more executing nodes, you must specify a real IP address for the `manager.mevent.send.host` key.
If you provide one standby node for one executing node, you can specify a virtual or real IP address for the `manager.mevent.send.host` key.

(4) Settings for the Management Server management command (mngsvrutil)

Specify the following keys in the client-side definition file (`.mngsvrutilrc`) or client-side common definition file (`mngsvrutilcl.properties`) of the Management Server management command (`mngsvrutil`):

- `mngsvrutil.connect.host`

If you provide only one standby node for two or more executing nodes, you must specify a real IP address for the `mngsvrutil.connect.host` key.

If you provide one standby node for one executing node, you can specify a virtual or real IP address for the `mngsvrutil.connect.host` key.

- `mngsvrutil.target_name`

Specify the virtual IP address in the `mngsvrutil.target_name` key. Check that you specify this key only when specifying the host name.

Reference note

Use the `.mngsvrutilrc` file to set up individual default values for each client. If you want to set up a common default value for all the clients, use the `mngsvrutilcl.properties` file. Note that when you are using both the files, the `.mngsvrutilrc` file is applied. The `mngsvrutilcl.properties` file is not read.

The `mngsvrutilcl.properties` file is used when the `mngsvrutil` command is executed with the script for monitoring, starting, and stopping Management Server.

(5) Settings for the commands provided in the Smart Composer functionality

Specify the following key in the client settings property file (`.cmxrc`) or client common settings property file (`cmxclient.properties`):

- `cmx.connect.host`

If you provide only one standby node for two or more executing nodes, you must specify a real IP address for the `cmx.connect.host` key.

If you provide one standby node for one executing node, you can specify a virtual or real IP address for the `cmx.connect.host` key.

Reference note

Use the `.cmxrc` file to set up individual default values for each client. If you want to set up a common default value for all the clients, use the `cmxclient.properties` file. Note that when you are using both the files, the `.cmxrc` file is applied. The `cmxclient.properties` file is not read.

(6) Settings for the invoke monitor command (In Windows)

You can only use the invoke monitor command in Windows.

Specify the following key in the `.mngsvrmonitorrrc` file (invoke monitor setup file for JP1/IM integration). For details about the `.mngsvrmonitorrrc` file, see 8.2.17 *.mngsvrmonitorrrc (Settings file of the monitor startup command for JP1/IM integration)* in the *uCosminexus Application Server Definition Reference Guide*.

- `mngsvrmonitor.connect.host`

If you provide only one standby node for two or more executing nodes, you must specify a real IP address for the `mngsvrmonitor.connect.host` key.

If you provide one standby node for one executing node, you can specify a virtual or real IP address for the `mngsvrmonitor.connect.host` key.

16.3.4 Cluster settings

Specify the following settings for a cluster:

- Creating a script file
- Cluster software environment settings

(1) Creating a script file

Create script files for starting and stopping Management Server, Administration Agent, and Application Server. You must also determine the cluster monitoring method in the script file. For details on creating the script files, see the OS manual in use.

This subsection describes the monitoring targets and monitoring methods of node switching for the host management model.

(a) Monitoring targets

- Management Server
- Administration Agent
- Logical server #

#

The node is switched when the logical server status becomes "abnormal termination" (the automatic restart frequency is exceeded or when 0 is specified as the automatic restart frequency and a failure is detected). Specify the following key in `mserver.properties` (Management Server environment setup file):

```
com.cosminexus.mngsvr.logical_server_abnormal_stop.exit = true
```

If automatic restart is specified for Management Server, node switching is not executed during this time. For executing the node switching during this time, do not specify automatic restart for Management Server.

In a configuration in which two or more executing nodes exist, do not specify `true` for the `com.cosminexus.mngsvr.logical_server_abnormal_stop.exit` key. If you do so, multiple systems of these executing nodes might start on the standby node.

(b) Monitoring methods

This subsection describes the monitoring methods to be implemented in node switching for the host management model.

Use one of the following methods according to the operation or environment used:

- Process monitoring

This method does not place too much load on the monitoring process.

- **Process monitoring and health check**

The monitoring accuracy is higher than the method of monitoring only the processes, but this method places some load on the monitoring process.

To monitor a logical server, see the notes described in (a) *Monitoring targets*.

- **Process monitoring**

This method monitors the existence of Management Server processes. The following table describes the names of the monitored processes:

Table 16–3: Monitored processes

Monitoring target	Process name	
	In Windows	In UNIX
Management Server	mngsvr.exe	cjstartsv#1,#2
Administration Agent	adminagent.exe	adminagent
Logical server	mngsvr.exe	cjstartsv#1

#1

The `cjstartsv` process has the same name as the process name of the J2EE server. You identify the process of Management Server by the server name.

#2

The command line name of Management Server begins with the following character string:

`/opt/Cosminexus/CC/server/bin/cjstartsv cosmi_m`

`cosmi_m` is the default server name. If Management Server is set up with a server name other than `cosmi_m`, change the server name.

- **Health check**

This method monitors the existence of Management Server processes and whether the existing processes are running. Confirm by executing the following commands in Management Server:

- When monitoring Management Server and logical server
`mngsvrutil` command
- When monitoring the Administration Agent
`adminagentcheck` command

(2) Cluster software environment settings

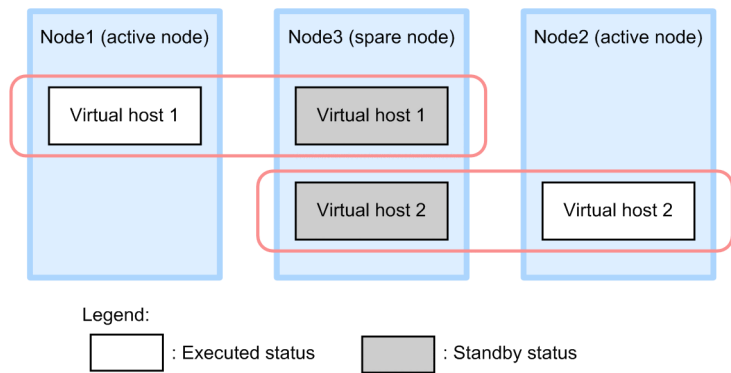
For details on the cluster software environment settings, see the cluster software documentation in use.

16.3.5 Application Server settings

To set up Application Server, invoke the Administration Agent and Management Server of the active node and standby node respectively. After invocation, set up the logical server, deploy the setup file, and then import the applications and resource adapters into the J2EE server. The procedure to set up Application Server for 2 active nodes (Node1, Node2) and 1 standby node (Node3) is as follows:

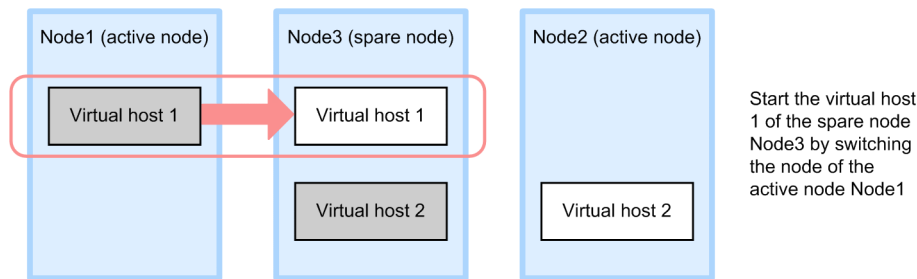
1. For Node1 and Node2, the cluster software starts the node.

The operating state is normal with the active node Application Server (virtual host 1 of Node1 and virtual host 2 of Node2) in running state and the standby node Application Server (virtual host 1 and virtual host 2 of Node3) in standby state.

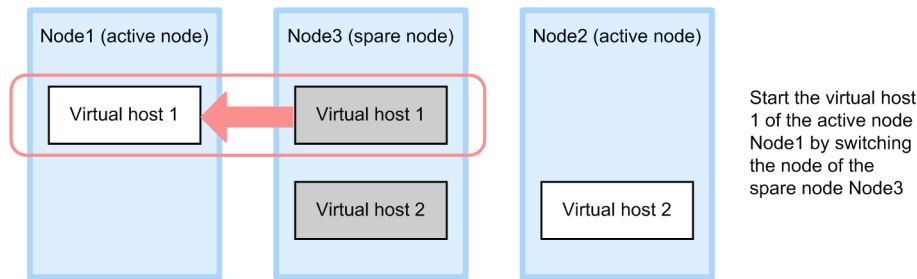


- Based on the Easy Setup definition file that is setup, the Web system is set up and started using the Smart Composer functionality commands for the active node Application Server (virtual host 1 of Node1 and virtual host 2 of Node2). Also, the server management commands are used to import the J2EE applications and resource adapters, and then the imported J2EE applications and resource adapters are started.
- For details about operations via server management commands, see 3. *Basic Operations of Server Management Commands* in the *uCosminexus Application Server Application Setup Guide*.

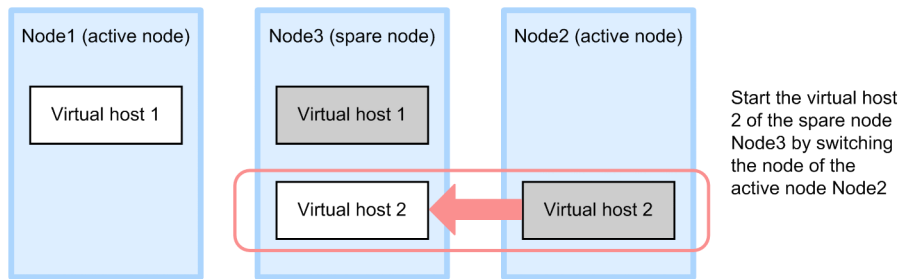
- The cluster software switches the node of virtual host 1.
- The nodes of active node Application Server (virtual host 1 of Node1) are switched and the standby node Application Server (virtual host 1 of Node3) is started.



- Implement step 2. for the standby node Application Server (virtual host 1 of Node3).
 - The cluster software returns the node of virtual host 1.
- The nodes of the standby node Application Server (virtual host 1 of Node3) are switched and the active node Application Server (virtual host 1 of Node1) is started. The system returns to a normal operating state.



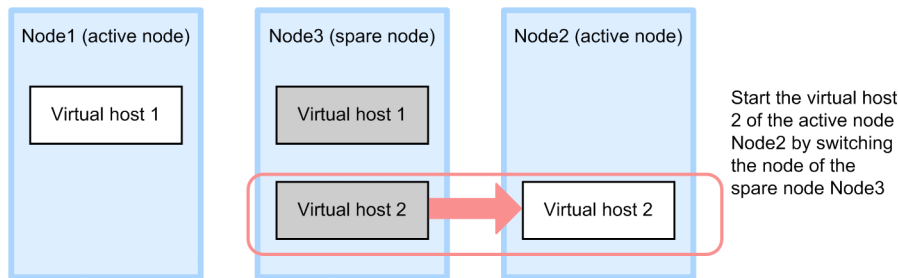
- The cluster software switches the node of virtual host 2.
- The nodes of the active node Application Server (virtual host 2 of Node2) are switched and the standby node Application Server (virtual host 2 of Node3) is started.



7. Implement step 2 for the standby node Application Server (virtual host 2 of Node3).

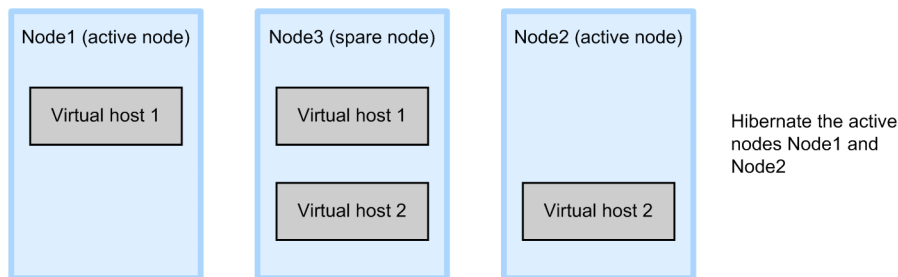
8. The cluster software returns the node of virtual host 2.

Nodes of the standby node Application Server (virtual host 2 of Node3) are switched and the active node Application Server (virtual host 2 of Node2) is started. The system returns to a normal operating state.



9. In Node1 and Node2, the cluster software stops the node.

The active node Application Server (virtual host 1 of Node1 and virtual host 2 of Node2) stops and is in a waiting state.



16.4 Starting and stopping the node switching system for the host management model

To use and operate the node switching system for the host management model, you must first specify the required environment settings, such as preparing the hosts for the N active nodes and one standby node and registering the script for monitoring, starting, or stopping Management Server and Administration Agent that are monitored by the cluster software. For details on how to specify the settings, see [16.3.1 Procedure for setting up the node switching system for the host management model](#).

This subsection describes the points to be noted when starting and stopping a cluster configuration of the active node Application Server (active node 1 of Node1 and active node 2 of Node2) and the standby node Application Server (standby node 1 and standby node 2 of Node3).

- When you start the operations, start the active node Application Server (active node 1 of Node1 and active node 2 of Node2) and set the standby node Application Server (standby node 1 and standby node 2 of Node3) in the standby state.
- You cannot start the active node and standby node Application Servers concurrently. When you need to replace J2EE applications, stop the service, and then switch the active node and standby node. Stop the J2EE applications one by one, replace them, and then set up and deploy the J2EE applications as and when required.

17

1-to-1 Node Switching System (Linked with Cluster Software)

This chapter describes a system in which the executing node and the standby node are operated as the 1-to-1 ratio (1-to-1 node switching system).

17.1 Organization of this chapter

This chapter is organized as the following table.

Table 17–1: Organization of this chapter (1-to-1 node switching system (integrated with cluster software))

Category	Title	Reference
Description	Overview of 1-to-1 node switching system	17.2
	System configuration and operations of 1-to-1 node switching system	17.3
Settings	Settings of the 1-to-1 node switching system of Application Server (In Windows)	17.4
	Settings of the 1-to-1 node switching system in the Management Server (In Windows)	17.5
	Settings of the 1-to-1 node switching system of Application Server (In UNIX)	17.6
	Settings of the 1-to-1 node switching system for the Management Server (In UNIX)	17.7
Operations	Starting and stopping the 1-to-1 node switching system in Application Server (In Windows)	17.8
	Starting and stopping a 1-to-1 node switching system for the Management Server (In Windows)	17.9
	Starting and stopping a 1-to-1 node switching system for the Management Server (In UNIX)	17.10
Confirmation of the settings	Confirming the settings of a node switching system	17.11

Note:

Function-specific explanation is not available for "Implementation" and "Precautions".

17.2 Overview of 1-to-1 node switching system

The *1-to-1 node switching system* is the system in which the executing node and the standby node have 1-to-1 ratio. Application Server supports the 1-to-1 node switching system of Management Server. However, you cannot use the 1-to-1 node switching system of Management Server in the execution environment of batch applications.

- **1-to-1 node switching system of Management Server**

In this configuration, one Management Server machine of the standby node is prepared for one Management Server machine of the executing node. If a problem occurs in the Management Server machine of the executing node and the machine stops, or if the Management Server process terminates, the cluster software starts the Management Server machine of the standby node and switches the processing.

17.3 System configuration and operations of 1-to-1 node switching system

This section describes the configuration examples of the 1-to-1 node switching system, and the flow of the node switching processing.

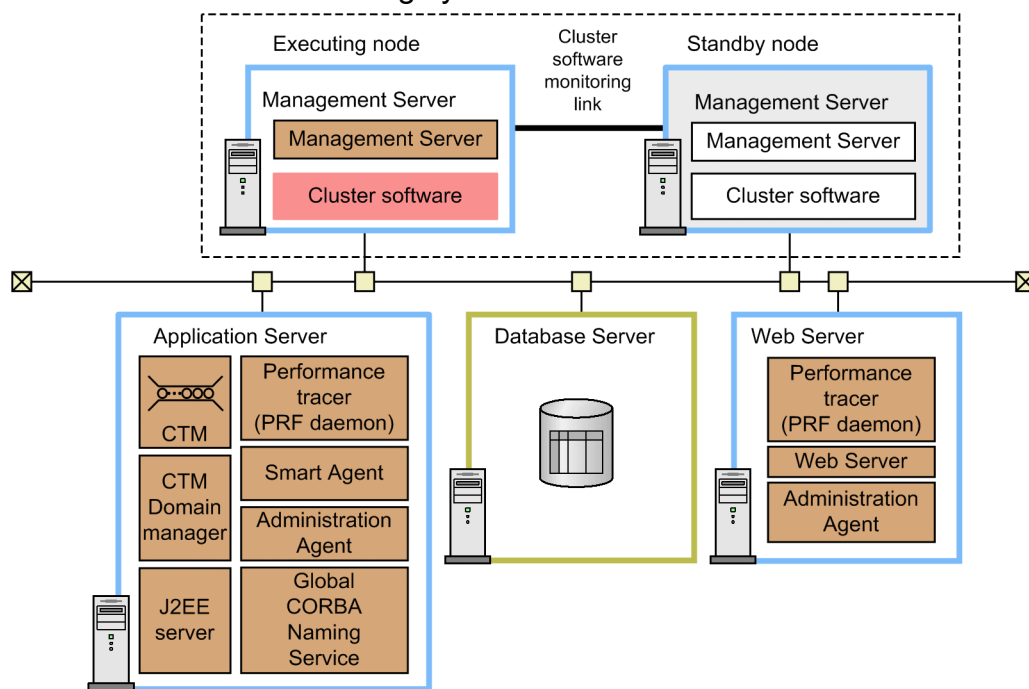
17.3.1 Example system configuration for a 1-to-1 node switching system

The following figure shows an example of system configuration for operating Management Server in the 1-to-1 node switching system by linking with cluster software:


- To operate the Management Server in the 1-to-1 node switching system

The following figure shows the example of system configuration for operating the Management Server in the 1-to-1 node switching system. The Management Server appears to restart with the same logical address from the Web server and from the Application Server during node switching.

Figure 17–1: Example system configuration for operating the Management Server in the 1-to-1 node switching system



Legend:

 : It is the unit of node switching. It appears to be a single server, when viewed from outside the server.

Reference note

In the Management Server, the Management Server-related settings such as the configuration information about the management domain are stored in a file. In the 1-to-1 node switching system of the Management Server, if there are changes in the management domain configuration, the information defined in the executing node (such as various definition files of the Management Server, and the J2EE applications and the resource adapters defined in the Management Server) must be copied into the standby node before starting the system so that the executing node information is inherited in the standby node. For details on how to copy

the Management Server-related settings, see [17.5.4 Copying the Management Server settings from the active node to the spare node](#).

You can execute the following types of operations in the 1-to-1 node switching system:

For JP1 integration:

In a cluster software configuration, you can integrate the system with JP1.

For integrating the system with JP1, the Application Server also requires JP1/Base. You must manage JP1 in the cluster software separately from the Application Server.

Integration with cluster software in a database server:

You can also use a cluster software configuration in the database server. In this case, if only the virtual address (logical address) is recognized in the Application Server, you cannot recognize that the database server is using the cluster software.

Application of the load balancer

Though not described in this system configuration example, you can also apply the load balancer by preparing several Web servers in the same configuration. As a result, you can improve the reliability and operating rate of the Web server.

For details on the system configuration to be used for operating Application Server in the 1-to-1 node switching system, see [3.11.1 Configuration in which executing node and standby node of Application Server are in 1-to-1 ratio \(when transaction service is not used\)](#) and [3.11.2 Configuration in which executing node and standby node of Application Server are in 1-to-1 ratio \(when using transaction service\)](#) in the *uCosminexus Application Server System Design Guide*. For details on the system configuration to be used for operating Management Server in the 1-to-1 node switching system, see [3.11.3 Configuration in which executing node and standby node of Management Server are in 1-to-1 ratio](#) in the *uCosminexus Application Server System Design Guide*.

17.3.2 Timing for node switching

Implement the node switching at the following timing:

- When the cluster software detects node failure[#] such as hardware failure
- When the process being monitored (Administration Agent or Management Server) is down
- When the node switching command of the cluster software is used
- When other events in which the cluster software switches the node occur

#

Indicates failure occurring in nodes except the servers. This includes hardware failure in nodes and cluster software failure.

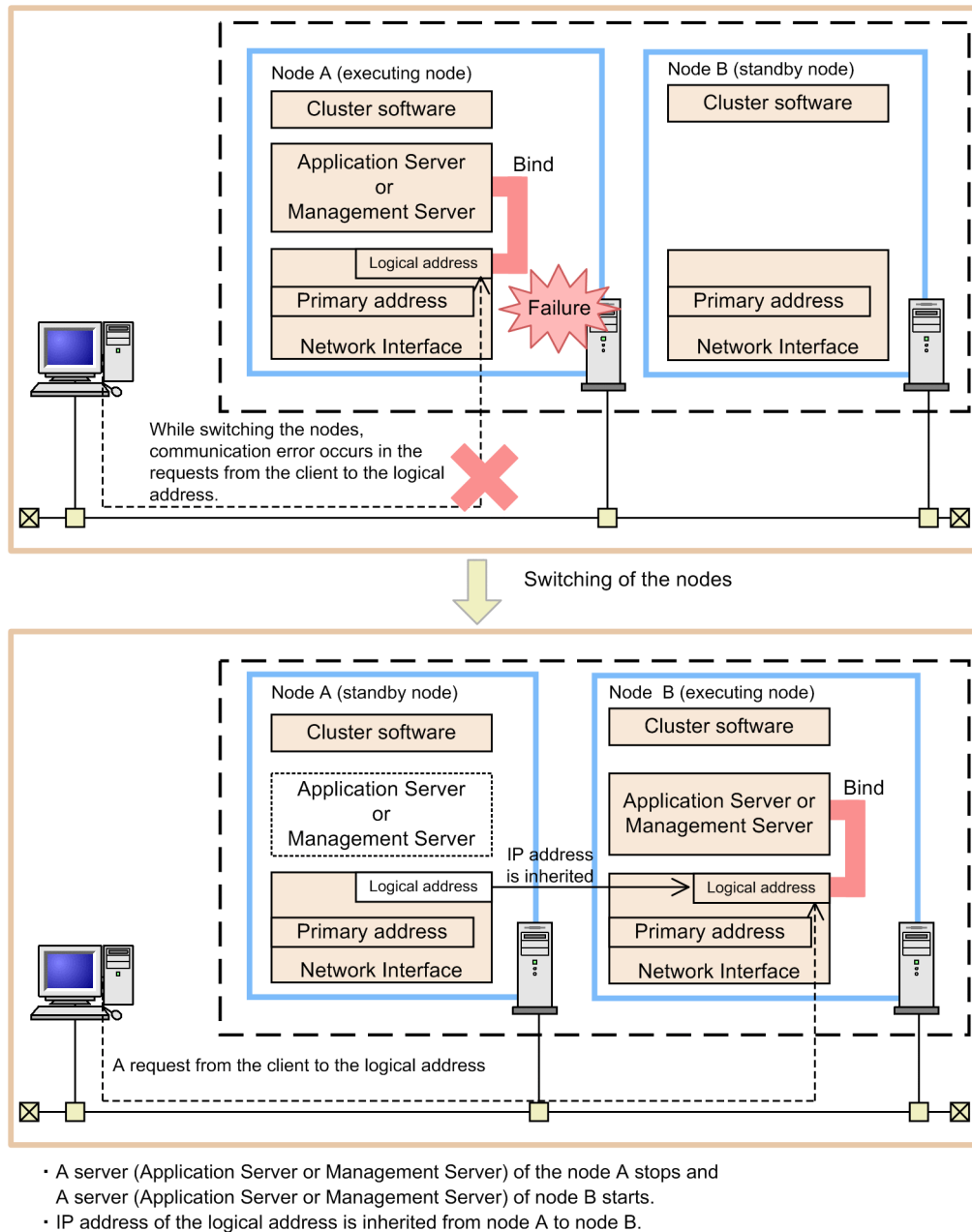
17.3.3 Flow of the node switching process in the 1-to-1 node switching system

This subsection describes the operations and the process flow during the node switching in 1-to-1 node switching system.

(1) Node switching operations

The following figure shows the node switching in a 1-to-1 node switching system. This figure describes an example in which failure occurs in node A and the system switches over to node B:

Figure 17–2: Node switching operations in the 1-to-1 node switching system



(2) Flow of node switching process

There are two node switching methods that you can apply in the 1-to-1 node switching system; one is automatic node switching method and the other is planned node switching method. The *automatic node switching* is a method in which the cluster software automatically switches the nodes. If a failure occurs in the executing node, automatic node switching occurs. On the other hand, the *planned node switching* is a method in which the operator switches the nodes in a planned manner during the node maintenance. The operator switches the nodes from the executing node, using the cluster software commands.

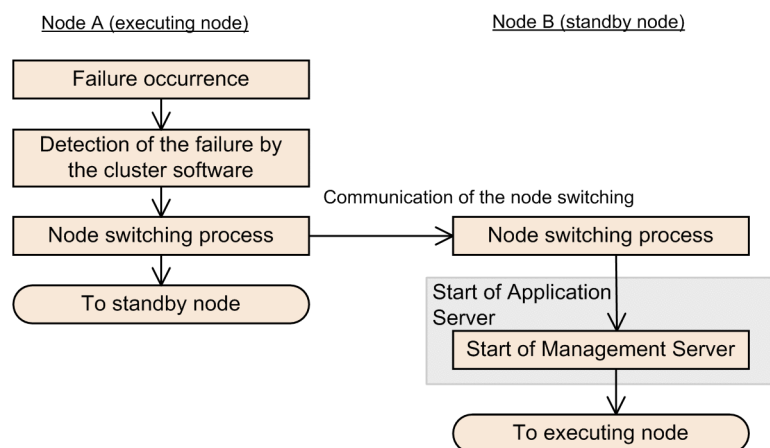
The following points separately describe the flow of the node switching process for automatic node switching and planned node switching:

(a) Flow of automatic node switching process

This subsection describes the flow of automatic node switching process of a 1-to-1 node switching system. This subsection describes the flow for switching to node B that is the standby node when failure occurs in node A that is the executing node.

The following figure shows the flow of automatic node switching process in a 1-to-1 node switching system in the Management Server:

Figure 17–3: Flow of automatic node switching process in the 1-to-1 node switching system of the Management Server

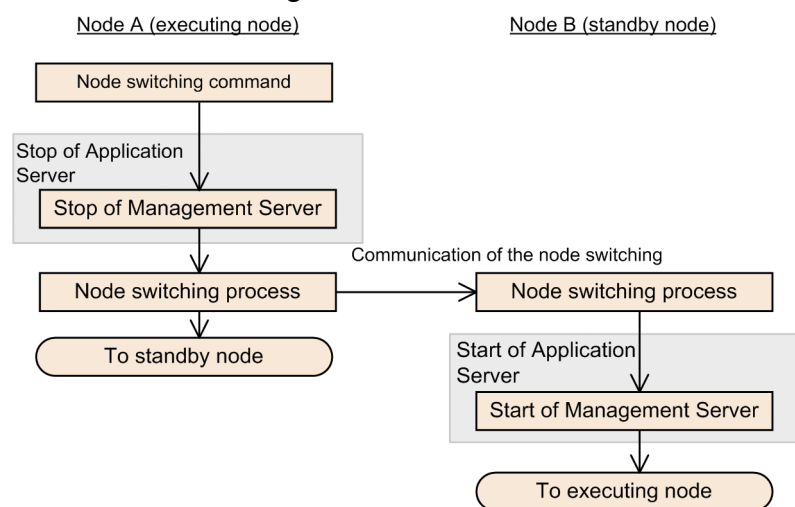


(b) Flow of planned node switching process

This subsection describes the flow of planned node switching process in a 1-to-1 node switching system. This subsection describes the process flow when you are using the cluster software commands to switch the executing node A into the standby node, and the standby node B into the executing node.

The following figure shows the flow of planned node switching process into a 1-to-1 node switching system of the Management Server:

Figure 17–4: Flow of planned node switching process in the 1-to-1 node switching system of the Management Server



17.3.4 System operations during node switching

This subsection describes the system operations during the node switching.

(1) Inheriting the HTTP session information

When you do not use the session failover functionality, the HTTP session information^{#1} managed by the Web container and the attribute information^{#2} specified in `ServletContext` remains only in the memory of the J2EE server, and this information is not shared between the executing node and the standby node. After the nodes are switched, the HTTP session information is lost. Therefore, invalid operations might be executed in the Web applications and Web client where the session is already running. In order to prevent this, you specify settings in applications so that if inconsistency occurs in the session information, appropriate action can be taken.

When using the session failover functionality, you can maintain the HTTP session information even after the node switching. However, you can only maintain the session information that is inherited with the session failover functionality.

In order to maintain the session after node switching, you must uniquely set up the session in the applications so that the session is managed by DBMS, and can be referenced from both the executing node and the standby node.

#1

Indicates information stored in the `javax.servlet.http.HttpSession` object acquired using the `javax.servlet.http.HttpServletRequest#getSession` method.

#2

Indicates information stored in the `javax.servlet.ServletContext` object acquired using the `javax.servlet.ServletContext#getServletContext` method.

(2) Inheriting the EJB session information

The session information[#] managed by the EJB container is only in the memory of the J2EE server, and therefore, once the node is switched, the session information is lost (the session information is not shared between the executing node and standby node). Therefore, normal operations might not continue in the EJB client that is already running the session. If the session information is lost, you are required to design the EJB client so that the process from the Home lookup will re-execute.

Note that the same problem occurs in the CTM environment.

#

Indicates the object information of the home interface and component interface.

(3) Inheriting the transaction information

The inheriting of the transaction information depends on the types of transaction. The inheriting of the transaction information for local transaction and global transaction is as follows:

(a) In the local transaction

In a local transaction, the transaction information is not inherited when the nodes are switched.

When the system is down due to failure and if node switching occurs due to process failure in the J2EE server, both commit and rollback are not executed for a running transaction.

Alike planned node switching, when the J2EE server terminates normally, the transaction does not remain in the running state.

(b) In the global transaction

In a global transaction, the transaction inherited after node switching is used to execute the transaction recovery process when the J2EE server starts. As a result, the appropriate process from commit or rollback is implemented.

(4) Operations of the Administration Agent

In the Administration Agent, information is not inherited directly between the executing node and the standby node. After node switching, the logical server coded in the starting script of the Administration Agent will start. For details on the starting script of Administration Agent, see [17.6.5 Creating a shell script file](#).

(5) Key information output to the Trace based Performance Analysis

The key information for identifying the series of processes is provided in the information output to the Trace based Performance Analysis. This key information includes the IP address used for creation. When there is a host with several IP addresses, the key information includes one of the IP addresses of that host. Therefore, in the Trace based Performance Analysis that is output to the executing node host, the *stationary IP address* (IP address that does not move to other nodes by node switching) and the *alias IP address* (IP address dynamically allocated by cluster software) might be included as the key information. Note that subsequently, the IP address included in the key information might change before or after the node switching.

For the key information that is output to the Trace based Performance Analysis, see [7.2 Overview of the trace based performance analysis](#) in the *uCosminexus Application Server Maintenance and Migration Guide*.

17.3.5 Inheriting information during node switching

This subsection describes the information that is inherited and information that is not inherited using the standby node during the node switching.

- **Information that is inherited**

- OTS transaction information (in global transaction)
After the node is switched from the executing node to the standby node, the OTS transaction information is inherited, and the transaction recovery process is executed.

- **Information that is not inherited**

Note that when node switching occurs, the following information is not inherited in the node after the switching:

- HTTP session information[#] of applications managed by the Web container
- Attribute information set in ServletContext of the Web container
- Session information for the Bean of the home interface and the component interface
- Information related to the operating conditions of applications, status of JNDI cache, connection pool, and CTM queue.

#

When the session failover functionality is used, only the session information set as the target for inheritance can be inherited.

17.4 Settings of the 1-to-1 node switching system of Application Server (In Windows)

The 1-to-1 node switching system is a system in which you set the executing node and the standby node in 1-to-1 ratio. In Application Server, you can build and operate a system in which you deploy one Application Server of the executing node for one Application Server of the standby node, and a system in which you deploy one Management Server of the executing node for one Management Server of the standby node. Use the management portal to build a system in which you deploy Application Servers in 1-to-1 ratio. For details on the management portal, see the *uCosminexus Application Server Management Portal User Guide*. This section describes the settings for a node switching system in which you deploy the Application Server of the executing node and Application Server of the standby node in a 1-to-1 ratio.

In node switching systems, where you deploy the executing node and the standby node of Application Server in 1-to-1 ratio, if a machine failure or failures such as termination of the Administration Agent occur in the Application Server of the executing node, the cluster software detects the failure, and continues the operations by automatically switching to the standby node. Also, even if a failure does not occur and the operating system requires preventive maintenance, the operator can switch to the standby node in a planned manner.

For details on the 1-to-1 node switching system of Application Server, see [17.2 Overview of 1-to-1 node switching system](#). For system configuration, see [3.11.1 Configuration in which executing node and standby node of Application Server are in 1-to-1 ratio \(when transaction service is not used\)](#) and [3.11.2 Configuration in which executing node and standby node of Application Server are in 1-to-1 ratio \(when using transaction service\)](#) in the *uCosminexus Application Server System Design Guide*.

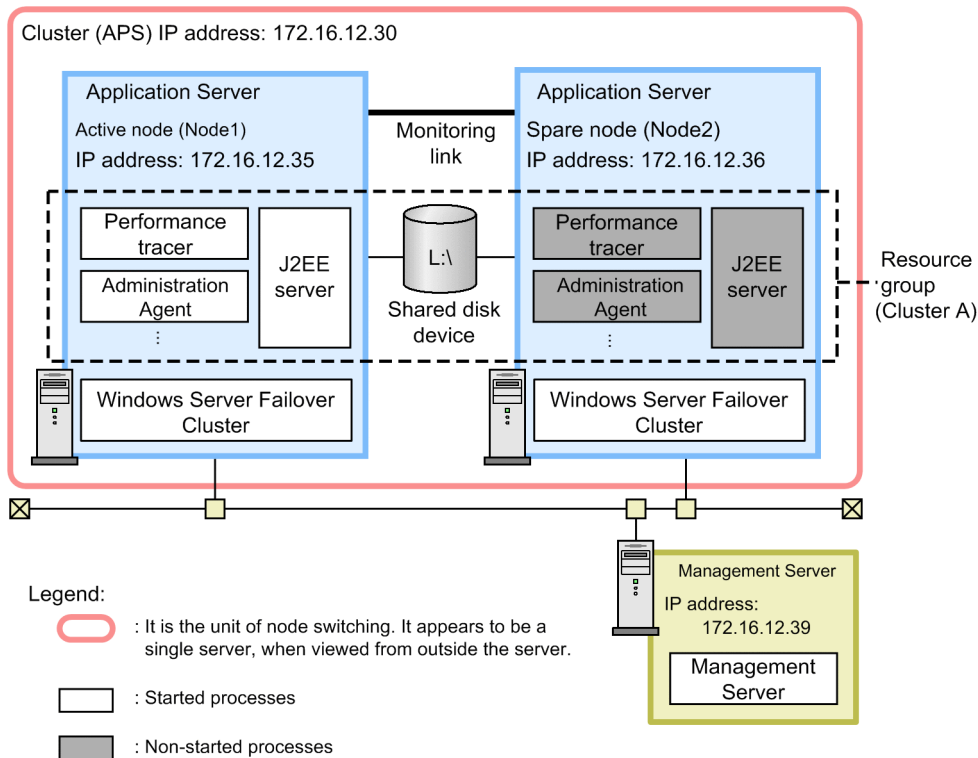
17.4.1 Procedure for setting the 1-to-1 node switching system of Application Server

This subsection describes the example of system configuration and the procedure for setting the system for the 1-to-1 node switching system of Application Server.

(1) Example system configuration

The following figure shows an example of configuration for the 1-to-1 node switching system of Application Server. Note that the subsequent points describe examples of building the system using this example of system configuration.

Figure 17–5: Configuration example of the 1-to-1 node switching system of Application Server (In Windows)



In this example, the executing node (active node) of Application Server and the standby node (spare node) of Application Server are deployed in 1-to-1 ratio. Also, Management Server is deployed in a machine different from Application Server.

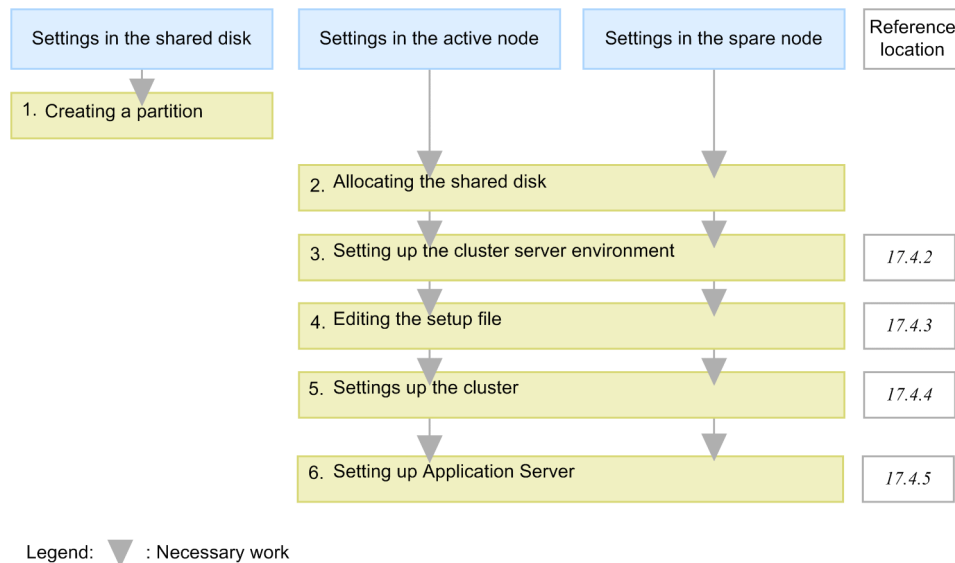
In the 1-to-1 node switching system of Application Server, first of all you build a cluster, and then you define Application Server as the active node (Node1 in the example) and the spare node (Node2 in the example) within the cluster (APS in the example). Also, the active node and the spare node are defined as one resource group (ClusterA in the example) containing the 'physical disk resource', 'IP address resource', 'network name resource', and 'general-purpose script resource'.

To deploy Application Server in a cluster configuration, set the *alias IP address*, and specify settings so that the running node inherits the alias IP address and the client does not consider the nodes in the cluster. In the 1-to-1 node switching system, the alias IP address becomes the address that Windows Server Failover Cluster allocates dynamically (*cluster IP address*).

(2) Procedure for system setup

When integrating with Windows Server Failover Cluster, you must set the management portal and cluster administrator. The following figure shows the procedure for setting the 1-to-1 node switching system of Application Server:

Figure 17–6: Procedure for setting the 1-to-1 node switching system of Application Server (In Windows)



The following points describe the step 1 through 6 in the above figure:

1. Create a partition in the shared disk to build a file system.
Create the storage location for the cluster management file of Windows Server Failover Cluster. Also, when you are using a global transaction, create the storage location for the transaction information. Note that you might store the cluster management file and the transaction information in the same partition.
2. Allocate the shared disk to the system.
When allocating the shared disk to the system, allocate the same drive character to the active node and spare node.
3. Set the environment of the cluster server with the management portal.
Specify the settings for using Windows Server Failover Cluster in 'Configuration definition of management domain' and 'Environment setup of a logical server' of the management portal. For details, see [17.4.2 Setting the environment of the cluster server](#).
4. Edit the configuration files.
Set various definition files of Administration Agent, Management Server, and Cosminexus HTTP Server. For details, see [17.4.3 Editing the configuration files](#).
5. Setting a cluster
Create a script file for monitoring the environment settings of the cluster software and Administration Agent. For details, see [17.4.4 Setting a cluster](#).
6. Set Application Server with the management portal and cluster administrator.
Use the management portal and the cluster administrator to deploy Application Server in the cluster configuration, and then set the J2EE applications and resource adapters. For details, see [17.4.5 Setting Application Server](#).

17.4.2 Setting the environment of the cluster server

This subsection describes the points for you to consider when specifying the settings with the management portal during integration with Windows Server Failover Cluster.

Reference note

To set the operating environment of the system with the management portal, you must start Management Server beforehand. Also, you must set up Management Server on the host in which you will be using Management Server for the first time. For details on setting the operating environment with the management portal, and the operating procedures and screens, see the *uCosminexus Application Server Management Portal User Guide*.

(1) Settings in the 'Configuration definition of management domain'

Define the configuration of the host for setting the logical servers and the configuration of each logical server. At this point, set the cluster IP address in 'host name' in the **Host Definition** screen.

(2) Settings in the 'Environment setup of logical server'

Make a note of the following points when setting the Logical J2EE Server, Logical Web Server, and Logical Naming Service:

(a) Setting the logical J2EE server

- **Host fixing**

Set 'Fix' in 'Fix Host' at the following locations:

- 'Fix Host' under 'Setting the Management Agent' in the J2EE Container Settings screen
- 'Fix Host' in the EJB Container Settings screen
- 'Fix Host' under 'Connecting to the Web Server' and 'Setting the server for management' in the Web Container Settings screen

When you fix the host, the host name defined in the 'Configuration definition of management domain' will be used.

- **Setting the global transaction**

Specify the following settings when using a global transaction:

- Set 'Light transaction functionality' to 'Disable' in the Transaction Settings screen.
- Specify the directory of the shared disk device in the 'Status file storage destination of in-process OTS' in the Transaction Settings screen.

Setup example

L:\Group1\otsstatus

(b) Setting the Logical Web Server

Set 'Fix' in 'Fix Host' in the Web Server Settings screen.

(c) Setting the Logical Naming Service

Set 'Fix' in 'Fix Host' in the Naming Service Settings screen. At this point, do not check the 'Use the host of Administration Agent as fixed host' checkbox when setting the Naming Service (started as an out-process) used by the J2EE server. By specifying the settings in this way, the host name specified in the Adding Naming Service screen of 'Configuration definition of management domain' is used.

17.4.3 Editing the configuration files

Set various definition files of Administration Agent, Management Server, and Cosminexus HTTP Server. This subsection describes the file settings that you must be careful about when integrating with Windows Server Failover Cluster.

(1) Setting the Administration Agent

Among the items to be set in `adminagent.properties` (Administration Agent property file), the settings that you must be careful about when integrating with Windows Server Failover Cluster are as follows.

For `adminagent.properties`, see 8.2.1 *adminagent.properties (Administration Agent property file)* in the *uCosminexus Application Server Definition Reference Guide*.

- **adminagent.adapter.bind_host key**

Specify the cluster IP address with the `adminagent.adapter.bind_host` key. Only the cluster IP address receives the requests for management. Specify as described in the following example:

```
adminagent.adapter.bind_host=172.16.12.30
```

- **adminagent.cluster.localaddress.check key**

Specify the settings for stopping the logical server or Administration Agent of the standby node that did not stop due to a failure during node switching of Application Server.

```
adminagent.cluster.localaddress.check=true
```

(2) Setting Cosminexus HTTP Server (When integrating with Web server)

Among the items to be set in `httpsd.conf` (Cosminexus HTTP Server definition file), the settings that you must be careful about when integrating with Windows Server Failover Cluster are as follows. For details on each directive of the `httpsd.conf` file, see the *uCosminexus Application Server HTTP Server User Guide*.

- **Listen directive and BindAddress directive**

Specify the cluster IP address in the `Listen` directive and the `BindAddress` directive. The cluster IP address provides services to the client.

- **ProxyPass directive and ProxyPassReverse directive**

For the forwarding-destination address, specify a cluster IP address, a host name that can resolve to a cluster IP address, or the constant `localhost`.

- **ServerName directive**

Specify the cluster IP address or a host name that you can resolve to the cluster IP address in the `ServerName` directive.

(3) Setting the Smart Agent (when using CTM)

When you use CTM to integrate with Management Server, you must edit the following file:

- **localaddr**

Create `localaddr`, and in `localaddr`, you set the stationary IP address and cluster IP address to enable the Smart Agent to use the stationary IP address and alias IP address for communication. For details on the settings of `localaddr`, see [Appendix A. Settings of Cosminexus TPBroker for Integrating Cluster Software \(in Windows\)](#).

17.4.4 Setting a cluster

You execute the following settings for a cluster:

- Create a script file
- Environment settings of the cluster software

(1) Create a script file

Create a script file for starting and stopping a monitoring target. Also, you must determine how to monitor a cluster in the script file. For details on how to create the script files, see the manual of the OS that is being used.

This subsection describes the monitoring targets and the monitoring methods for 1-to-1 node switching systems of Application Server.

(a) Monitoring targets

The monitoring target for 1-to-1 node switching systems of Application Server is as follows:

- Administration Agent

The node switching is executed when Administration Agent ends.

(b) Monitoring method

You monitor the availability of processes of Administration Agent. You monitor the following process:

- `adminagent.exe`

(2) Environment settings for Windows Server Failover Cluster

For details on the environment settings of Windows Server Failover Cluster, see the documentation of the cluster software that is being used.

17.4.5 Setting Application Server

When setting Application Server, deploy Application Server in a cluster configuration using the management portal and cluster administrator. Also, set up the logical servers and distribute the configuration information, and then import the J2EE applications and resource adapters. Following is the procedure to set Application Server:

1. Start Management Server.

For starting the Management Server, see *4.1.1 Procedure for starting a system* and *4.1.2 Methods of starting a system* in the *uCosminexus Application Server Management Portal User Guide*.

2. In Node1, set the resource group ClusterA to online.

3. Set up the J2EE server with the management portal.

Set up the J2EE server with the Setup screen of 'Configuration definition of management domain'. For details on performing operations with the management portal and details on the screens, see the *uCosminexus Application Server Management Portal User Guide*.

4. Using the management portal, distribute the information set in the J2EE server to Node1.

Distribute the information set in the J2EE server to Node1 in the Distribution of Setup Information screen of 'Environment setup of the logical server'.

5. Use the server management commands to import the J2EE application and resource adapter in the J2EE server of Node1. Set and start the imported J2EE applications and resource adapters.

For the settings of the J2EE applications, see *4.1.29 Setting and starting the business application (when using CUI)* in the *uCosminexus Application Server System Setup and Operation Guide* and for the settings of the resource adapters, see the following subsections of the *uCosminexus Application Server System Setup and Operation Guide*:

- *4.1.26 Setting DB Connector (when using CUI)*
- *4.1.27 Setting resource adapters other than DB Connector (when using CUI)*
- *4.1.28 Starting the resource adapter (when using CUI)*

For operations of the server management commands, see *3. Basic Operations of Server Management Commands* in the *uCosminexus Application Server Application Setup Guide*.

Execute the server management commands after starting the J2EE server and its prerequisite processes. Once you have completed the operations with server management commands, stop the server management commands.

6. With the cluster administrator, execute 'Move Group' for the resource group ClusterA to switch the nodes, and then change the resource owner to Node2.
7. Execute step 3. to step 5. in Node2.
8. With the cluster administrator, execute 'Move Group' for the resource group ClusterA to switch the nodes, and then return the resource owner to Node1.

17.5 Settings of the 1-to-1 node switching system in the Management Server (In Windows)

The 1-to-1 node switching system is a system in which the executing node and the standby node have the 1-to-1 ratio. In Application Server, you can build and operate a system in which one Management Server of the executing node is deployed for one Management Server of the standby node. You use the Smart Composer functionality to build a node switching system by deploying Management Servers having the 1-to-1 ratio. This section describes the settings for a node switching system in which Management Server of the executing node and Management Server of the standby node are deployed with the 1-to-1 ratio.

In a node switching system where Management Servers are deployed with the 1-to-1 ratio, if a machine failure occurs or if Management Server processes terminate in Management Server of the executing node, the cluster software detects the failure, automatically switches to a standby node, and continues operations. Also, even if failure does not occur and the running system requires preventive maintenance, the operator can switch to a standby node in a planned manner.

For details on the 1-to-1 node switching system of the Management Server, see [17.2 Overview of 1-to-1 node switching system](#). For system configuration, see [3.11.3 Configuration in which executing node and standby node of Management Server are in 1-to-1 ratio](#) in the *uCosminexus Application Server System Design Guide*.

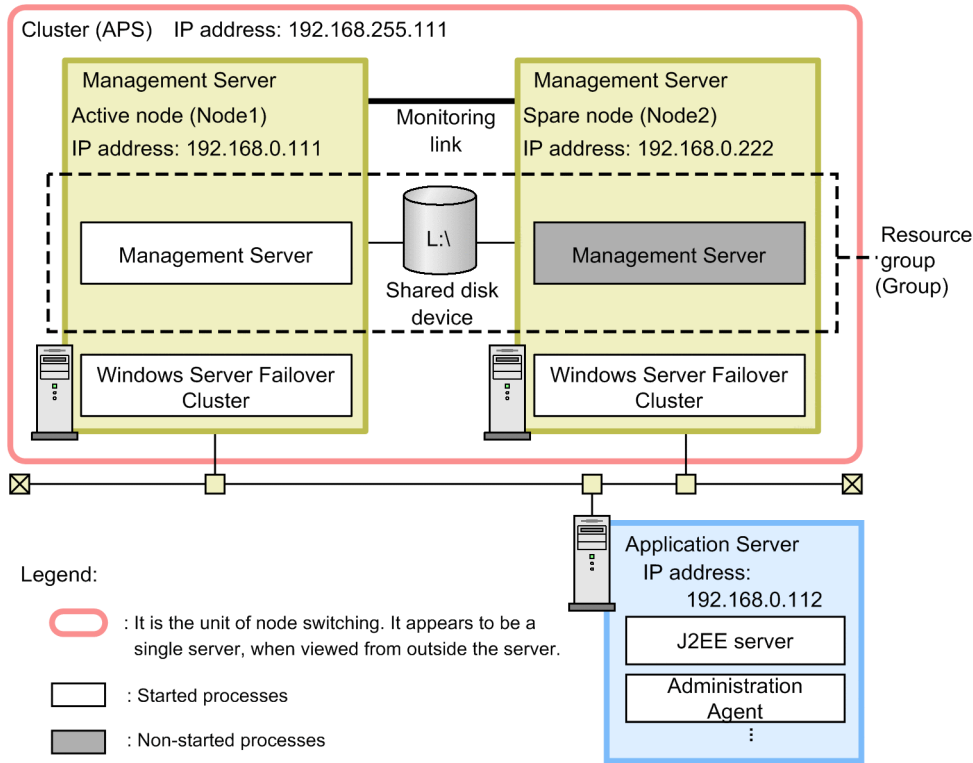
17.5.1 Procedure for setting the 1-to-1 node switching system in the Management Server

This subsection describes the example of system configuration and the procedure for setting the 1-to-1 node switching system of Application Server.

(1) System configuration example

The following figure shows the example configuration for the 1-to-1 node switching system of the Management Server. Note that the subsequent points describe the system configuration example using this configuration example.

Figure 17–7: Configuration example of 1-to-1 node switching system for the Management Server (in Windows)



The executing node (active node) of the Management Server and the standby node (spare node) of the Management Server are deployed in 1-to-1 ratio.

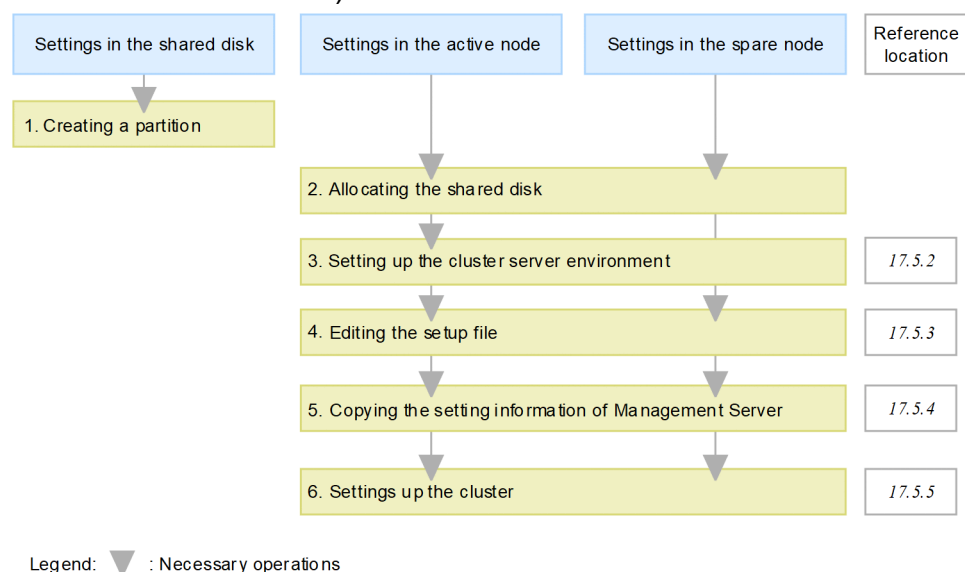
In the 1-to-1 node switching system of the Management Server, first the cluster is built, and then the Management Server is defined as the active node (Node1 in the example) and spare node (Node2 in the example) in the cluster (APS in the example). Also, the active node and spare node are defined as one resource group (Group in the example) containing 'physical disk resource', 'IP address resource', 'network name resource', and 'general-purpose script resource'.

Note that in order to deploy the application server in a cluster configuration, set the alias IP address, and specify settings such that the running node inherits the alias IP address so that the client does not consider the nodes in the cluster. When there is 1-to-1 node switching system, the alias IP address is dynamically allocated by Windows Server Failover Cluster (cluster IP address).

(2) Procedure of system setup

When integrating the system with Windows Server Failover Cluster, settings must be specified for the Management Server and the Cluster Administrator. The following figure shows the procedure for setting up the 1-to-1 node switching system of the Management Server.

Figure 17–8: Procedure for setting the 1-to-1 node switching system in the management server (in Windows)



The following points describe the steps 1 through 6 in the above figure:

1. Create a partition in the shared disk to build a file system.
You create a location for storing the cluster management file of Windows Server Failover Cluster. Also, when using the global transaction, create the location for storing the transaction information. The cluster management file and the transaction information might be stored in the same partition.
2. Allocate a shared disk to the system.
When allocating a shared disk to the system, allocate the same drive character to the active node and the spare node.
3. Set up the cluster server environment in the Management Server.
In the Easy Setup definition file of the Smart Composer functionality for the Management Server, specify the settings for using Windows Server Failover Cluster. For details, see [17.5.2 Environment settings in the cluster server](#).
4. Edit the setup files.
Specify the settings for various definition files of the Management Server that cannot be set up in the Easy Setup definition file. For details, see [17.5.3 Editing the setup files](#).
5. Copy the Management Server settings from the active node to the spare node.
To set the same environment for the active node and the spare node, copy the Management Server-related settings of the active node into the spare node. For details, see [17.5.4 Copying the Management Server settings from the active node to the spare node](#).
6. Set up the cluster.
Create a script file for monitoring the environment settings of the cluster software and Management Server. For details, see [17.5.5 Setting a cluster](#).

17.5.2 Environment settings in the cluster server

This subsection describes the points that you must note when you specify the settings in the Management Server while integrating the system with Windows Server Failover Cluster.



Reference note

You must set up the Management Server on the host in which the Management Server is used for the first time. For the Management Server setup, see *4.1.14 Setting the management functionality* in the *uCosminexus Application Server System Setup and Operation Guide*.

(1) Settings in the Easy Setup definition file

In the Easy Setup definition file for the Smart Composer functionality of the Management Server, you define the host on which the logical server is set up and define the configuration of each logical server. Also, you set up the logical server environment and start and stop operations in the `<configuration>` tag of each logical server, as and when required. You use the Smart Composer functionality commands to set up the Web system based on an already set up Easy Setup definition file.

As and when required, use the server management commands to import and start the J2EE applications and resource adapters.

17.5.3 Editing the setup files

You can edit the setup files. This subsection describes the setup files that you must take care of while integrating the system with Windows Server Failover Cluster. Note that different settings are required in the active node and the spare node.

1. Management Server settings
2. Settings for the management command (`mngsvrutil`) of the Management Server

You specify step 1 and 2 in the active node.

Specify step 2 only in the spare node. You can specify step 1 by copying the Management Server settings from the active node, so the settings are not required here.

(1) Management Server settings

From the items set up in the Management Server environment settings file (`mserver.properties`), this point describes the settings that you must note while integrating the system with Windows Server Failover Cluster. For `mserver.properties`, see *8.2.6 mserver.properties (Management Server environment settings file)* in the *uCosminexus Application Server Definition Reference Guide*.

- `mngsvr.myhost.name`
Specify the cluster IP address with the `mngsvr.myhost.name` key. Specify as described in the following example:
`mngsvr.myhost.name=192.168.255.111`

(2) Settings for the management command (mngsvrutil) in the Management Server

In the home directory of the local system account, you prepare the client definition file (`.mngsvrutilrc`) for the management command (`mngsvrutil`) of the Management Server. You specify the user ID and password of the management user of the Management Server. Also, set appropriate access privileges for the client definition file.

You use this file to operate the `mngsvrutil` command in the script for monitoring, starting, and stopping the Management Server.

For the client-side definition file (`.mngsvrutilrc`) of the `mngsvrutil` command, see 8.2.14 *.mngsvrutilrc (Client-side definition file of the mngsvrutil command)* in the *uCosminexus Application Server Definition Reference Guide*.

17.5.4 Copying the Management Server settings from the active node to the spare node

To set up the same environment for the active node and the spare node, copy the Management Server-related settings of the active node to the spare node. Use the save and recovery command in the Management Server settings to copy settings such as various definition files of the Management Server set up as the active node, and copy J2EE applications and resource adapters registered in the Management Server from the active node to the spare node.

(1) Procedure for copying the Management Server settings

To copy the Management Server settings:

1. Execute the `mstrexport` command in the active node.

The `mstrexport` command collects the Management Server settings of the active node to be executed by the `mstrexport` command and saves the collected information in a ZIP file. You specify the file name of the saved ZIP file in the `mstrexport` command argument.

- **Command storage location**

Cosminexus-installation-directory\manager\bin

- **Execution example**

```
mstrexport C:\work\mstruct.zip
```

Note that you can execute the `mstrexport` command regardless of the starting and stopping of the Management Server that is executed by the `mstrexport` command.

2. Copy the ZIP file saved in step 1 from the active node into the spare node.

3. Execute the `mstrimport` command in the spare node.

You use the `mstrimport` command for deploying the copied ZIP file in the Management Server of the spare node that is executed by the command. As a result, you can specify the same settings for the Management Server of the active node and Management Server of the spare node.

Specify the file name of the copied ZIP file in the `mstrimport` command argument.

- **Command storage location**

Cosminexus-installation-directory\manager\bin

- **Execution example**

```
mstrimport D:\recovery\mstruct.zip
```

Note that you can execute the `mstrimport` command only when the Management Server to be executed by the `mstrimport` command is not running.

For commands, see *mstrexport (save Management Server management file)* and *mstrimport (restore Management Server management file)* in the *uCosminexus Application Server Command Reference Guide*.

(2) Defining the files to be collected

You can specify the file (file that defines the save target for the Management Server administrative file) that describes the collection target of the `mstrexport` command as the argument of the `mstrexport` command.

Execution example

```
mstrexport c:\work\mstruct.zip "C:\Documents  
and Settings\MyUser\filelist.txt"
```

By default, you can collect the information required for building and operating a system in the Management Server such as various definition files of the Management Server and the J2EE applications and resource adapters registered in the Management Server using the `mstrexport` command. In addition to this information, if you want to add user-created commands in the collection target of the `mstrexport` command, specify the absolute path of the file to be collected in the file that defines the save target for the Management Server administrative file.

Coding example of file

```
${cosminexus.home}/manager/apps/MyApp.ear  
D:/home/confdir/message1.conf
```

For files, see *8.2.13 Definition files to be saved for the Management Server management files* in the *uCosminexus Application Server Definition Reference Guide*.

17.5.5 Setting a cluster

You execute the following settings for a cluster:

- Create a script file
- Environment settings of the cluster software

(1) Create a script file

This subsection describes the monitoring targets and monitoring methods for 1-to-1 node switching systems of Management Server.

You create a script file for starting and stopping a monitoring target. Also, you must determine how to monitor a cluster in the script file. For details on how to create the script files, see the manual of the OS that is being used.

(a) Monitoring target

The monitoring target for 1-to-1 node switching systems of Management Server is as follows:

- Management Server

The node switching is executed when Management Server ends.

(b) Monitoring method

You monitor the availability of processes of Management Server. You monitor the following process:

- `mngsvr.exe`

(2) Environment settings of Windows Server Failover Cluster

For details on the environment settings of Windows Server Failover Cluster, see the manual of the cluster software that is being used.

17.6 Settings of the 1-to-1 node switching system of Application Server (In UNIX)

The 1-to-1 node switching system is a system in which you set the executing node and the standby node in 1-to-1 ratio. In Application Server, you can build and operate a system in which one Application Server of the executing node is deployed for one Application Server of the standby node and one Management Server of the executing node is deployed for one Management Server of the standby node. Use the management portal to build a system in which you can deploy Application Server in 1-to-1 ratio. For details on the management portal, see the *uCosminexus Application Server Management Portal User Guide*. This section describes the settings for a node switching system in which you deploy the Application Server of the executing node and Application Server of the standby node in 1-to-1 ratio.

In a node switching system, where the executing node and the standby node of Application Server are deployed in 1-to-1 ratio, if a machine failure or failures such as termination of the Administration Agent occur in Application Server of the executing node, the HA monitor detects the failure, and continues the operations by automatically switching to the standby node. Also, when a failure does not occur and the operating system requires preventive maintenance, the operator can switch to the standby node in a planned manner.

Note that you can use the operations of the 1-to-1 node switching system of Application Server only in AIX or Linux.

For details on the functions, see [17.2 Overview of 1-to-1 node switching system](#). For system configuration, see [3.11.1 Configuration in which executing node and standby node of Application Server are in 1-to-1 ratio \(when transaction service is not used\)](#) and [3.11.2 Configuration in which executing node and standby node of Application Server are in 1-to-1 ratio \(when using transaction service\)](#) in the *uCosminexus Application Server System Design Guide*. For HA monitor, see the manual *High-reliability System Monitoring Functionality HA Monitor*.

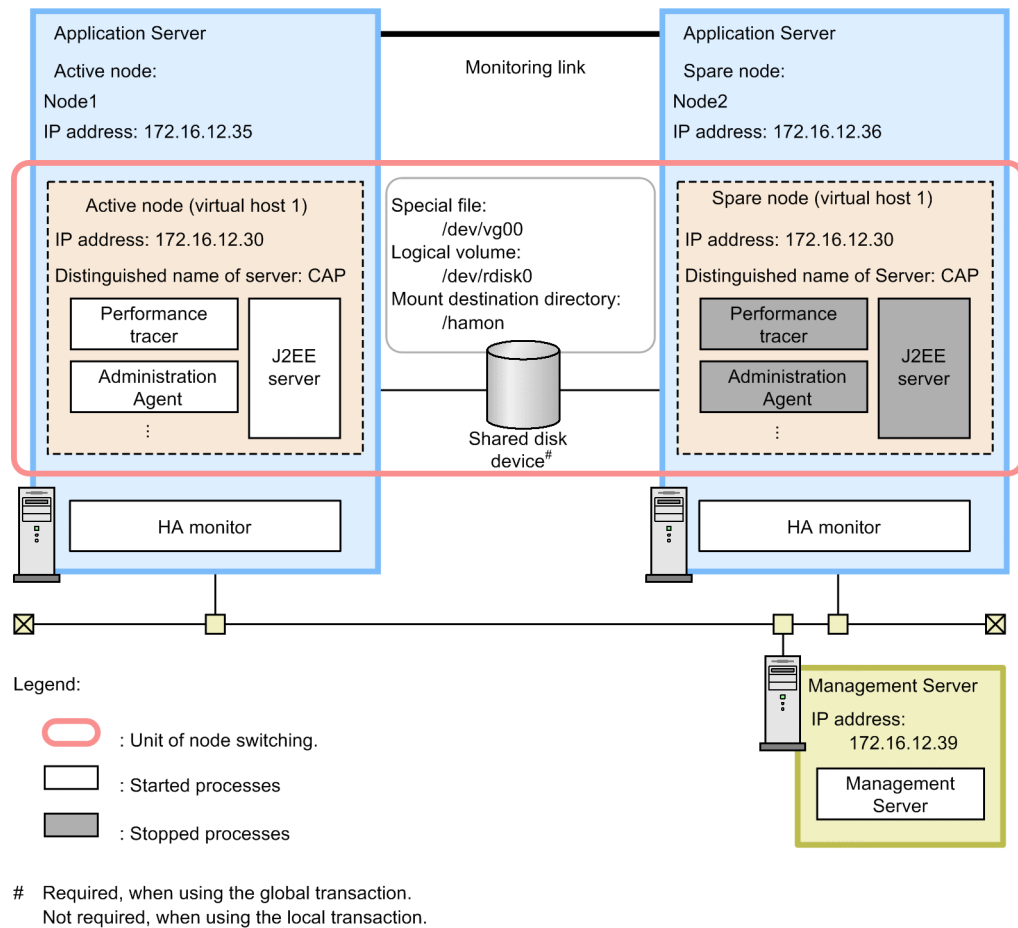
17.6.1 Procedure for setting the 1-to-1 node switching system of Application Server

This subsection describes an example of system configuration and the procedure for setting the system.

(1) Example of system configuration

The following figure shows an example of configuration for the 1-to-1 node switching system of Application Server. Note that the subsequent points describe the examples of building the system using this example of system configuration.

Figure 17–9: Configuration example of the 1-to-1 node switching system of Application Server (In UNIX)



In this example, the executing node (active node) of Application Server and the standby node (spare node) of Application Server are deployed in 1-to-1 ratio. Also, Management Server is deployed in a machine different from Application Server.

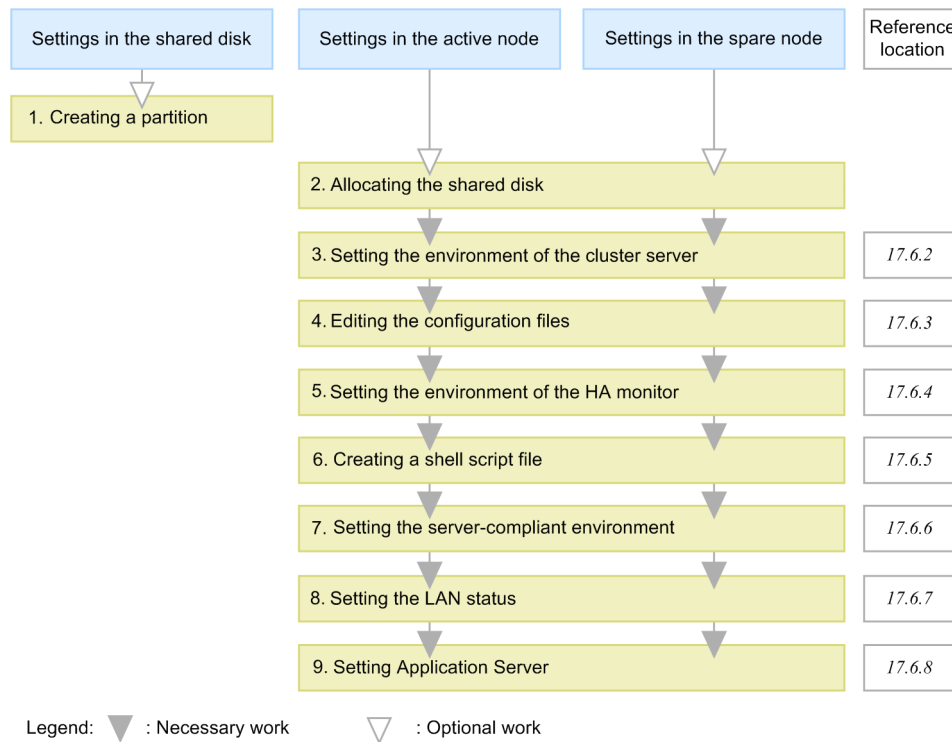
In the 1-to-1 node switching system of Application Server, first of all build a cluster, and then define Application Server as the active node and spare node within the cluster. Also, specify the distinguished name of the server (server optional name) for the commands used in the HA monitor and the messages to be output. Specify the same name (CAP in the example) for the active node and spare node.

To deploy Application Server in a cluster configuration, set the alias IP address, and specify settings so that the running node inherits the alias IP address and the client does not consider the nodes in the cluster. In case of the 1-to-1 node switching system, the alias IP address becomes the address that the HA monitor allocates dynamically.

(2) Procedure for system setup

When integrating with the HA monitor, you must set the management portal and the files of the HA monitor. The following figure shows the procedure for setting the 1-to-1 node switching system of Application Server:

Figure 17–10: Procedure for setting the 1-to-1 node switching system of Application Server (In UNIX)



Steps 1. to 9. of the figure are as follows:

- When using a global transaction, create a partition in the shared disk to build a file system.
To use a global transaction, create the storage location of the transaction information.
- When using a global transaction, allocate the shared disk to the system.
When allocating the shared disk to the system, specify the same mount destination directory in the active node and spare node.
- Set the environment of the cluster server with the management portal.
Specify the settings for using the HA monitor in 'Configuration definition of management domain' and 'Environment setup of a logical server' of the management portal. For details, see [17.6.2 Setting the environment of the cluster server](#).
- Edit the configuration files.
Set various definition files of the Administration Agent, Management Server, and Cosminexus HTTP Server. For details, see [17.6.3 Editing the configuration files](#).
- Set the environment of the HA monitor.
Define the environment of the HA monitor with the `sysdef` file of the HA monitor. For details, see [17.6.4 Setting the environment of the HA monitor](#).
- Create the shell script file.
Create the shell script file for monitoring the Administration Agent and for starting and stopping the Administration Agent and logical servers. For details, see [17.6.5 Creating a shell script file](#).
- Set the server-compliant environment.

Using the `servers` file of the HA monitor, define the environment of the active node server and the spare node server to be operated on the nodes. For details, see [17.6.6 Setting the server-compliant environment](#).

8. Set the status of LAN.

Using the LAN status setup file of the HA monitor, specify the IP address of the LAN adapter, and define switching of LAN in the HA monitor. For details, see [17.6.7 Setting the LAN status](#).

9. Using the management portal and commands of the HA monitor, specify the settings for Application Server.

Using the management portal and commands of the HA monitor, deploy Application Server in a cluster configuration, and set the J2EE applications and resource adapters. For details, see [17.6.8 Setting Application Server](#).

Important note

The precautions to be taken during system setup when integrating with the HA monitor are as follows:

- **Setting the client for connecting to Application Server integrated with the HA monitor**

In the client used to connect to Application Server integrated with the HA monitor, specify Application Server with the alias IP address when invoking Application Server integrated with the HA monitor during EJB lookup.

17.6.2 Setting the environment of the cluster server

This subsection describes the points for you to consider when specifying the settings with the management portal during integration with the HA monitor.

Reference note

To set the operating environment of the system with the management portal, you must start Management Server beforehand. Also, you must set up Management Server on the host on which Management Server is used for the first time. For details on setting the operating environment with the management portal, and the operating procedures and screens, see the *uCosminexus Application Server Management Portal User Guide*.

(1) Settings in the 'Configuration definition of management domain'

Define the configuration of the host on which you will set the logical servers, and also the configuration of each logical server. At this point, set the alias IP address in 'host name' in the Host Definition screen.

(2) Settings in the 'Environment setup of a logical server'

Note the following points when setting the Logical J2EE Server, Logical Web Server, and Logical Naming Service:

(a) Setting the logical J2EE server

- **Host fixing**

Set 'Fix' in 'Fix Host' at the following locations:

- 'Fix Host' under 'Setting the Management Agent' in the J2EE Container Settings screen
- 'Fix Host' in the EJB Container Settings screen

- 'Fix Host' under 'Connecting to the Web Server' and 'Setting the server for management' in the Web Container Settings screen

When you fix the host, the host name defined in the 'Configuration definition of management domain' is used.

- **Setting the global transaction**

Specify the following settings when using a global transaction:

- Set 'Light transaction functionality' to 'Disable' in the Transaction Settings screen.
- Specify the directory of the shared disk device in the 'Status file storage destination of in-process OTS' in the Transaction Settings screen.

Setup example

/hamon

(b) Setting the Logical Web Server

Set 'Fix' in 'Fix Host' in the Web Server Settings screen.

(c) Setting the Logical Naming Service

Set 'Fix' in 'Fix Host' in the Naming Service Settings screen. At this point, do not check the 'Use the host of Administration Agent as fixed host' checkbox when setting the Naming Service (started as an out-process) used by the J2EE server. By specifying the settings in this way, the host name specified in the Adding Naming Service screen of 'Configuration definition of management domain' is used.

17.6.3 Editing the configuration files

Set various definition files of the Administration Agent, Management Server, and Cosminexus HTTP Server. This subsection describes the file settings that you must be careful about when integrating with the HA monitor.

(1) Setting the Administration Agent

Among the items to be set in `adminagent.properties` (Administration Agent property file), the settings that you must be careful about when integrating with the HA monitor are as follows. For `adminagent.properties`, see 8.2.1 *adminagent.properties (Administration Agent property file)* in the *uCosminexus Application Server Definition Reference Guide*.

- **adminagent.adapter.bind_host key**

Specify the alias IP address with the `adminagent.adapter.bind_host` key. Only the alias IP address receives the requests for management. Specify as described in the following example:

```
adminagent.adapter.bind_host=172.16.12.30
```

- **adminagent.cluster.localaddress.check key**

Specify the settings for stopping the logical server or Administration Agent of the standby node that did not stop due to a failure during node switching of Application Server.

```
adminagent.cluster.localaddress.check=true
```

(2) Setting Cosminexus HTTP Server (When integrating with Web server)

Among the items to be set in `httpsd.conf` (Cosminexus HTTP Server definition file), the settings that you must be careful about when integrating with the HA monitor are as follows. For details on each directive of the `httpsd.conf` file, see the *uCosminexus Application Server HTTP Server User Guide*.

- **Listen directive and BindAddress directive**

Specify the alias IP address in the `Listen` directive and `BindAddress` directive. Only the alias IP address provides services to the client.

- **ProxyPass directive and ProxyPassReverse directive**

For the forwarding-destination address, specify a cluster IP address, a host name that can resolve to a cluster IP address, or the constant `localhost`.

- **ServerName directive**

Specify the alias IP address or a host name that you can resolve to the alias IP address in the `ServerName` directive.

(3) Setting the Smart Agent (When using CTM)

When you use CTM to integrate with Management Server, you must edit the following file:

- **localaddr**

Create `localaddr` and set the stationary IP address and alias IP address to enable the Smart Agent to use the IP addresses for communication. For details on the settings of `localaddr`, see [Appendix B. Settings of Cosminexus TPBroker for Integrating Cluster Software \(In UNIX\)](#).

(4) Setting the management command (mngsvrutil) of Management Server

In the home directory of the root of active and spare nodes, prepare the definition file at client side (`.mngsvrutilrc`) of the management command (`mngsvrutil`) of Management Server, and then set the user ID and password of the management user of Management Server. Also, set appropriate access privileges for the definition file at client side.

This file is the script for monitoring the processes of the Administration Agent and for starting and stopping the Administration Agent and logical servers. You can also use this file for executing the `mngsvrutil` command.

For the client-side definition file (`.mngsvrutilrc`) of the `mngsvrutil` command, see 8.2.14 *.mngsvrutilrc (Client-side definition file of the mngsvrutil command)* in the *uCosminexus Application Server Definition Reference Guide*.

17.6.4 Setting the environment of the HA monitor

Depending on the system you are using, define the environment of the HA monitor in the definition file `sysdef`.

For details on setting the environment of the HA monitor, see the manual *High reliability System Monitoring Functionality HA Monitor*.

17.6.5 Creating a shell script file

Create the following shell script files to monitor the processes of the Administration Agent and to start and stop the Administration Agent and logical servers:

- Shell script file for monitoring the processes of the Administration Agent
- Shell script file for starting the Administration Agent and logical servers
- Shell script file for stopping the Administration Agent and logical servers

Use the same shell script file in Application Server of the active node and Application Server of the spare node, and deploy in the same path.

(1) Shell script file for monitoring the processes of the Administration Agent

An example of the shell script file for monitoring the processes of the Administration Agent (manager_adminagent_monitor.sh) is as follows:

```
#!/bin/sh

LOGDIR=/home/manager/hamon/log
AA=/opt/Cosminexus/manager/bin/adminagent

logg()
{
    echo `date '+[%Y/%m/%d %H:%M:%S]`'"[$$]: $1" \
    >> ${LOGDIR}/adminagent.log 2>&1
}

logg "### $0: started. ###"
while true
do
    CHECK=`ps -ef | grep $AA | grep -v grep`

    if [ "$CHECK" = "" ]
    then
        logg "### $0: stop. ###"
        exit 0
    fi
    sleep 10
done
```

(2) Shell script file for starting the Administration Agent and logical servers

An example of the shell script file for starting the Administration Agent and logical servers (manager_adminagent_start.sh) is as follows:

```
#!/bin/sh

LOGDIR=/home/manager/hamon/log
SCRIPTDIR=/home/manager/hamon/bin
MNGDIR=/opt/Cosminexus/manager

logg()
{
    echo `date '+[%Y/%m/%d %H:%M:%S]`'"[$$]: $1" \
    >> ${LOGDIR}/adminagent.log 2>&1
}

# make adminagent.access.info
logg "### $0: make adminagent.access.info ###"
echo 172.16.12.30:28080,hostA:20295 > $MNGDIR/tmp/adminagent.access.info
```

```

# start Administration Agent
logg "### $0: starting Administration Agent. ###"
$MNGDIR/bin/adminagentctl start
if [ $? -eq 0 ] ; then
    logg "### $0: Administration Agent start normally. ###"
else
    logg "### $0: Administration Agent cannot start. ###"
    exit 1
fi

sleep 10

# start logical server
logg "### $0: starting logical servers. ###"
$MNGDIR/bin/mngsvrutil -m mnghost:28080 -t lserver1 -s start server
$MNGDIR/bin/mngsvrutil -m mnghost:28080 -t lserver2 -s start server
$MNGDIR/bin/mngsvrutil -m mnghost:28080 -t lserver3 -s start server

exit 0

```

Use the shell script file to create the access information file (/opt/Cosminexus/manager/tmp/adminagent.access.info)#, and to start the Administration Agent and logical servers.

#

The access information file is deleted after starting the Administration Agent and is restarted when accessing the file from Management Server.

Here, specify the settings in such a way so that the contents are the same as those re-created when you access the access information file from Management Server. Based on this file, each logical server is restarted when node switching occurs due to failure of a node.

The settings to be specified in the shell script file are as follows:

Creating the access information file

Coding corresponding to the shell script file in the example

```

echo 172.16.12.30:28080,hostA:20295
    > $MNGDIR/tmp/adminagent.access.info

```

Format

```
<Mng_ip>:<Mng_port>,<AA_host>:<AA_port>
```

The contents to be specified are as follows:

- Mng_ip
Specifies the IP address of the host in which Management Server exists.
- Mng_port
Specifies the HTTP port number for connecting to Management Server.
- AA_host
Specifies the host name of the Administration Agent. Specify the value specified in the host in 'Configuration definition of management domain' of the management portal of Management Server.
- AA_port

Specifies the port number of the Administration Agent. Specify the value specified in the port number of the Administration Agent in 'Configuration definition of management domain' of the management portal of Management Server.

Starting the logical servers

Specify the settings for starting the logical servers.

Use the `mngsvrutil` command to start the logical servers. For the `mngsvrutil` command, see *mngsvrutil* (Management Server management command) in the *uCosminexus Application Server Command Reference Guide*.

(3) Shell script file for stopping the Administration Agent and logical servers

An example of the shell script file for stopping the Administration Agent and logical servers (`manager_adminagent_stop.sh`) is as follows:

```
#!/bin/sh

LOGDIR=/home/manager/hamon/log
MNGDIR=/opt/Cosminexus/manager

logg ()
{
    echo `date '+[%Y/%m/%d %H:%M:%S] '`"[$$]: $1" \
    >> ${LOGDIR}/adminagent.log 2>&1
}

# stop logical server
logg "### $0: stop logical servers. ###"
$MNGDIR/bin/mngsvrutil -m mnghost:28080 -t lserver3 -s stop server
$MNGDIR/bin/mngsvrutil -m mnghost:28080 -t lserver2 -s stop server
$MNGDIR/bin/mngsvrutil -m mnghost:28080 -t lserver1 -s stop server

# stop Administration Agent
logg "### $0: stopping Administration Agent. ###"
$MNGDIR/bin/adminagentctl stop
```

17.6.6 Setting the server-compliant environment

When setting the server-compliant environment with the HA monitor, define the environment of the executing server and the standby server to be operated on the nodes.

In the definition file called *servers*, define the server-compliant environment for 1-to-1 node switching of Application Server. The following table describes the settings for the server-compliant environment:

Table 17–2: Settings for the server-compliant environment (in 1-to-1 node switching of Application Server)

Operand	Setting contents
name	This operand specifies the shell script file for starting the Administration Agent and logical servers. Specification example: <code>/home/manager/hamon/bin/manager_adminagent_start.sh</code>

Operand	Setting contents
alias	This operand specifies the identification name of the server. You can specify the same name in the active node and spare node. Specification example: CAP
acttype	This operand specifies the startup method of the server. Here, specify <code>monitor</code> because you start the server with the HA monitor command.
termcommand	This operand specifies the shell script file for stopping the Administration Agent and logical servers. Specification example: <code>/home/manager/hamon/bin/manager_adminagent_stop.sh</code>
initial	This operand specifies the status when you start the server. <ul style="list-style-type: none"> In the active node Specify <code>online</code>. In the spare node Specify <code>standby</code>.
disk	This operand specifies the character-type special file name of the shared disk device. Specification example: <code>/dev/vg00</code>
lan_updown	This operand specifies whether or not to use the status setup file of LAN. Here, specify <code>use</code> because you are using the status setup file of LAN.
fs_name	This operand specifies the absolute path name of the logical volume corresponding to the file system to be switched. Note that you require this setting only when using <code>\$TPFS</code> in a UNIX file. Specification example: <code>/dev/rdisk0</code>
fs_mount_dir	This operand specifies the absolute path name of the mount-destination directory of the file system to be switched. Note that you require this setting only when using <code>\$TPFS</code> in a UNIX file. Specification example: <code>/hamon</code>
patrolcommand	This operand specifies the shell script file for monitoring the processes of the Administration Agent. Specification example: <code>/home/manager/hamon/bin/manager_adminagent_monitor.sh</code>
servexec_retry	This operand specifies the restart frequency when the system detects a failure. Here, specify <code>0</code> because you switch the node without restarting when the system detects a failure.
waitserv_exec	This operand specifies whether to wait for completion of the execution command when you execute the startup completion process of the Administration Agent and logical servers. Here, specify <code>yes</code> because the server waits for completion of execution.

For details on setting the server-compliant environment, see the manual *High-reliability System Monitoring Functionality HA Monitor*.

17.6.7 Setting the LAN status

This subsection defines switching of LAN in the HA monitor by specifying the IP address of the LAN adapter in the LAN status setup file of the HA monitor.

Specify the alias IP address in the following files:

- `server-identification-name.up` **file**
Use this file during LAN connection. Specify the alias IP address to be added to the LAN adapter.
- `server-identification-name.down` **file**
Use this file when switching LAN. Specify the alias IP address to be deleted from the LAN adapter.

In *server-identification-name*, specify the value of *alias* of server-compliant environment settings (*servers* file).

For details on setting the LAN status, see the manual *High-Reliability System Monitoring Functionality HA Monitor*.

17.6.8 Setting Application Server

When setting Application Server, deploy Application Server in a cluster configuration using the management portal and the commands of the HA monitor. Also, set up the logical server and distribute the configuration information, and then import J2EE applications and resource adapters. Following is the procedure to set Application Server:

1. Start Management Server.

For starting the Management Server, see *4.1.1 Procedure for starting a system* and *4.1.2 Methods of starting a system* in the *uCosminexus Application Server Management Portal User Guide*.

2. In Node1 and Node2, execute the `monbegin` command of the HA monitor, start Application Server of the active node as the executing node and set Application Server of the spare node to the standby state.

- **Command execution example in Node1**

Execute # `monbegin CAP`, and then start the active node Node1.

- **Command execution example in Node2**

Execute # `monbegin CAP`, and then set the spare node Node2 to the standby status.

3. With the management portal, set up the J2EE server of the active node Node1.

Set up the J2EE server with the Setup screen of 'Configuration definition of management domain'. For details on the operating procedures and screens when using the management portal, see the *uCosminexus Application Server Management Portal User Guide*.

4. Using the management portal, distribute the information set in the J2EE server to the active node Node1.

Distribute the information set in the J2EE server to the active node Node1 with the Distribution of Setup Information screen of 'Environment setup of a logical server'.

5. Use the server management commands to import the J2EE application and resource adapter in the J2EE server of the active node Node1. Set and start the imported J2EE application and resource adapter.

For operations of the server management commands, see *3. Basic Operations of Server Management Commands* in the *uCosminexus Application Server Application Setup Guide*.

Execute the server management commands after starting the J2EE server and its prerequisite processes. Once you have completed the operations with server management commands, stop the server management commands.

6. Execute the `monswap` command of the HA monitor in Node1 to switch nodes.

- **Command execution example in Node1**

Execute # `monswap CAP` and switch the nodes with the server identification name CAP. The spare node Node2 is started and the active node Node1 is set to the standby state.

7. Execute step 3. to step 5. in the spare node Node2.

As the settings specified in the spare node are exactly the same as the active node, the setup procedure and contents are the same as the active node.

8. Execute the `monswap` command of the HA monitor in Node2, switch nodes and then return to the normal operation state.

- **Command execution example in Node2**

Execute `# monswap CAP`, and switch the nodes with the server identification name `CAP`. The active node Node1 is started and the spare node Node2 is set to the standby state.

17.7 Settings of the 1-to-1 node switching system for the Management Server (In UNIX)

The 1-to-1 node switching system is a system in which the executing node and the standby node are in the 1-to-1 ratio. With the Application Server, you can build and operate a system in which one Management Server of the executing node is deployed for one Management Server of the standby node. You use the Smart Composer functionality to build a node switching system by deploying the Management Servers in the 1-to-1 ratio. This section describes the settings for a node switching system in which the Management Server of the executing node and the Management Server of the standby node are deployed in the 1-to-1 ratio.

In a node switching system where the Management Servers are deployed in the 1-to-1 ratio, if machine failure occurs or if the Management Server process terminates in the Management Server of the executing node, the HA monitor detects the failure, automatically switches to a standby node, and then continues the operations. Also, if a running system requires preventive maintenance, the operator can switch to a standby node in a planned manner, even if a failure does not occur.

Note that you can only use the operations of the 1-to-1 node switching system of the Management Server in AIX or in Linux.

For details on the functionality, see [17.2 Overview of 1-to-1 node switching system](#). For system configuration, see [3.11.3 Configuration in which executing node and standby node of Management Server are in 1-to-1 ratio](#) in the *uCosminexus Application Server System Design Guide*. Also, for details on the HA monitor, see the manual *Reliable System Monitoring Functionality HA Monitor*.

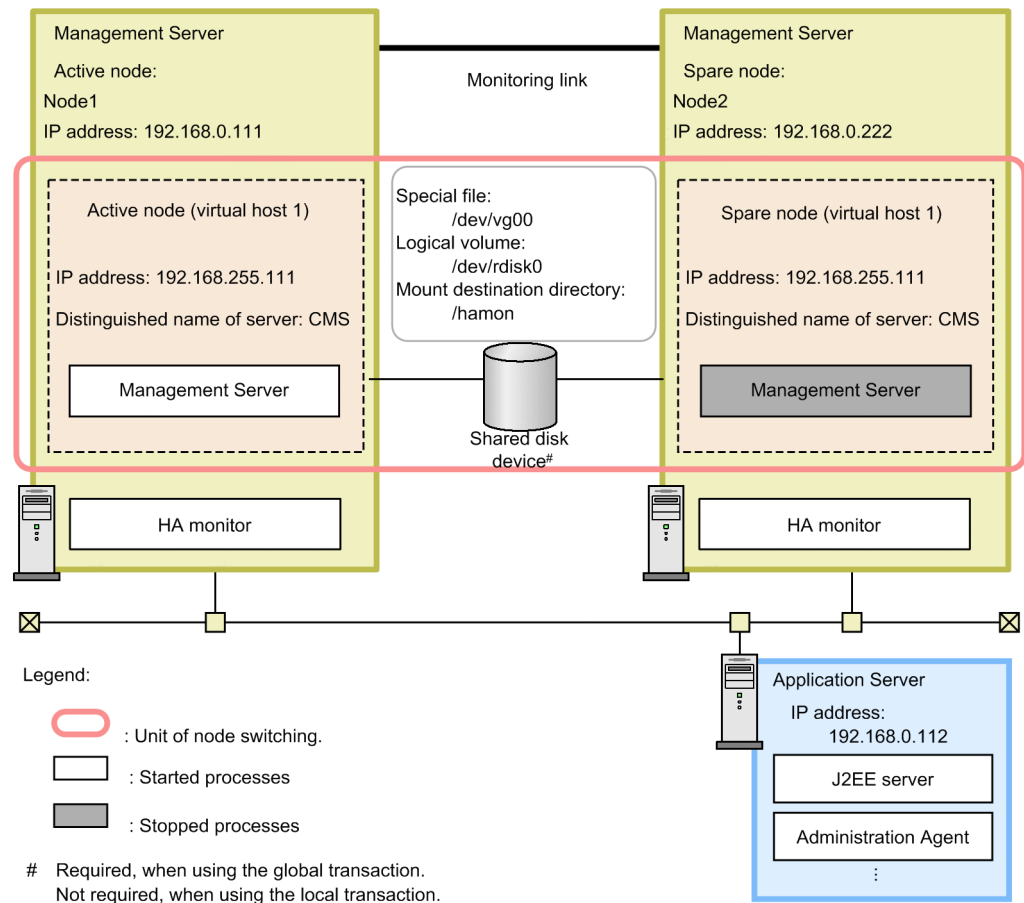
17.7.1 Procedure for setting the 1-to-1 node switching system in the Management Server

This subsection describes the example system configuration and the procedure for specifying the system settings.

(1) System configuration example

The following figure shows the example configuration for the 1-to-1 node switching system of the Management Server. Note that the subsequent points describe the example system configuration using this example configuration.

Figure 17–11: Configuration example of 1-to-1 node switching system for the Management Server
(In UNIX)



The executing node of the management server (active node) and the standby node of the management server (spare node) are deployed as 1-to-1.

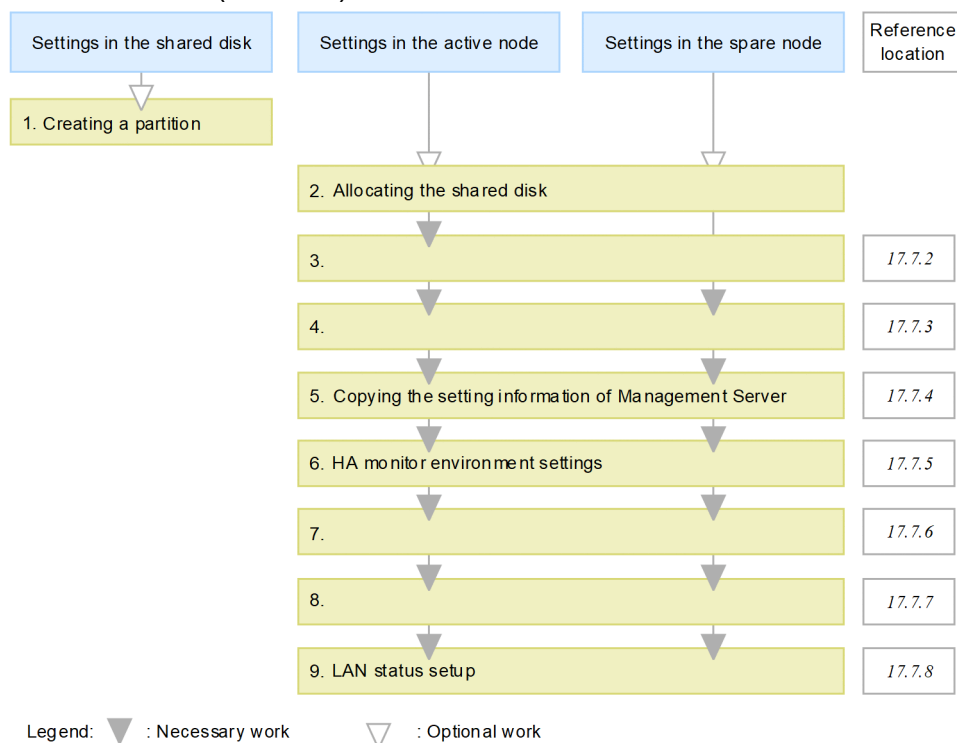
In the 1-to-1 node switching system of the Management Server, first the cluster is built, and then the management server is defined as the active node and the spare node in the cluster. Also, specify the server identification name (alias name of the server) for the used commands, and the messages output to the HA monitor. Specify the same name in the active node and the spare node (such as CMS in the example).

Note that in order to deploy the Application Server in a cluster configuration, set up the alias IP address and specify the settings in such a way so that the running node inherits the alias IP address, and the client does not realize the nodes in the cluster. When there is a 1-to-1 node switching system, the alias IP address is dynamically allocated by the HA monitor.

(2) Procedure of the system setup

For integrating the system with the HA monitor, settings must be specified for the Management Server and the HA monitor file. The following figure shows the procedure for setting up the 1-to-1 node switching system of the Management Server:

Figure 17–12: Procedure for setting the 1-to-1 node switching system in the Management Server
(In UNIX)



Stages 1. to 9. of the figure are explained below:

- When using the global transaction, create a partition in the shared disk and build the file system.
To use the global transaction, create the location to store the transaction information.
- When using the global transaction, allocate a shared disk to the system.
When allocating a shared disk to the system, specify the same directory to mount the shared disk in the active node and the spare node.
- Set up the cluster server environment in the Management Server.
In the Easy Setup definition file of the Smart Composer functionality for the Management Server, you specify the settings for using the HA monitor. For details, see [17.7.2 Environment settings in the cluster server](#).
- Edit the setup files.
Specify the settings for various definition files of the Management Server that cannot be specified in the Easy Setup definition file. For details, see [17.7.3 Editing the setup files](#).
- Copy the Management Server settings from the active node to the spare node.
To set the same environment for the active node and the spare node, copy the Management Server-related settings of the active node into the spare node. For details, see [17.7.4 Copying the Management Server settings from the active node to the spare node](#).
- Set up the HA monitor environment.
Define the HA monitor environment in the `sysdef` file of the HA monitor. For details, see [17.7.5 HA monitor environment settings](#).
- Create the shell script file.

Create the shell script file to monitor the Management Server and to start and stop the Management Server. For details, see [17.7.6 Creating the shell script file](#).

8. Set up a server-compliant environment.

Define the environment for the active node server and spare node server to be operated on the node, in the servers file of the HA monitor. For details, see [17.7.7 Server-compliant environment settings](#).

9. Set up the LAN status.

In the LAN status setup files of the HA monitor, specify settings such as the IP address of the LAN adapter to define the switching of LAN in the HA monitor. For details, see [17.7.8 LAN status setup](#).

17.7.2 Environment settings in the cluster server

This subsection describes the points that you must note while specifying the settings in the Management Server, for integrating the system with the HA monitor.



Reference note

You must set up the Management Server on the host on which the Management Server is used for the first time. For the Management Server setup, see [4.1.14 Setting the management functionality](#) in the *uCosminexus Application Server System Setup and Operation Guide*.

(1) Settings in the Easy Setup definition file

In the Easy Setup definition file of the Smart Composer functionality of the Management Server, you define the host on which the logical server is set up, and define the configuration of each logical server. Also, set up the logical server environment and the start and stop operations in the `<configuration>` tag of each logical server, as and when required. You use the Smart Composer functionality commands to set up the Web system based on an already set up Easy Setup definition file.

After starting the set up Web system with the Smart Composer functionality commands, use the server management commands to import and start the J2EE applications and resource adapters, as and when required.

17.7.3 Editing the setup files

Edit the setup files. This subsection describes the setup files for which you must be careful when you integrate the system with the HA monitor. Note that different settings are required in the active node and the spare node.

1. Management Server settings
2. Settings for the management command (`mngsvrutil`) of the Management Server

Specify step 1 and 2 in the active node.

Specify only step 2 in the spare node. For step 1, you can copy the setting information of the Management Server from the active node, so the settings need not be specified here.

(1) Management Server settings

From the items set up in the Management Server environment settings file (`mserver.properties`), this point describes the settings that you must note for integrating the system with the HA monitor. For `mserver.properties`, see 8.2.6 *mserver.properties (Management Server environment settings file)* in the *uCosminexus Application Server Definition Reference Guide*.

- `mngsvr.myhost.name`

You specify the Alias IP address with the `mngsvr.myhost.name` key. Specify as described in the following example:

```
mngsvr.myhost.name=192.168.255.111
```

(2) Settings for the management command (mngsvrutil) in the Management Server

In the home directory of the local system account, prepare the client definition file (`.mngsvrutilrc`) for the management command (`mngsvrutil`) of the Management Server and specify the user ID and password of the management user of the Management Server. Also, set appropriate access privileges for the client definition file.

Use this file to execute the `mngsvrutil` command in the script for monitoring the Management Server processes and in the script for starting and stopping the Management Server.

For the client definition file (`.mngsvrutilrc`) of the `mngsvrutil` command, see 8.2.14 *.mngsvrutilrc (Client-side definition file of the mngsvrutil command)* in the *uCosminexus Application Server Definition Reference Guide*.

17.7.4 Copying the Management Server settings from the active node to the spare node

To set up the same environment for the active node and the spare node, you copy the Management Server-related settings from the active node to the spare node. You use the save and recovery command of the Management Server settings for copying various definition files of the Management Server that is specified in the active node, and the J2EE applications and resource adapters that are registered in the Management Server from the active node into the spare node.

(1) Procedure for copying the Management Server settings

To copy the Management Server settings:

1. Execute the `mstrexport` command in the active node.

The `mstrexport` command collects the Management Server settings of the active node to be executed using the `mstrexport` command, and saves the collected information in a ZIP file. You specify the file name of the saved ZIP file in the `mstrexport` command argument.

- **Command storage location**

```
/opt/Cosminexus/manager/bin
```

- **Execution example**

```
mstrexport /tmp/work/mstruct.zip
```

Note that you can execute the `mstrexport` command regardless of the starting and stopping of the Management Server to be executed using the `mstrexport` command.

2. Copy the ZIP file saved in step 1 from the active node into the spare node.

3. Execute the `mstrimport` command in the spare node.

The `mstrimport` command deploys the copied ZIP file in the Management Server of the spare node to be executed using the command. As a result, you can specify the same settings for the Management Server of the active node and the Management Server of the spare node.

You specify the file name of the copied ZIP file in the `mstrimport` command argument.

- **Command storage location**

`/opt/Cosminexus/manager/bin`

- **Execution example**

`mstrimport /tmp/recovery/mstruct.zip`

Note that you can execute the `mstrimport` command only when the Management Server to be executed using the `mstrimport` command is not running.

For commands, see *mstrexport (save Management Server management file)* and *mstrimport (restore Management Server management file)* in the *uCosminexus Application Server Command Reference Guide*.

(2) Defining the files to be collected

You can specify the file (file that defines the save target for the Management Server administrative file) that describes the collection target of the `mstrexport` command as the argument of the `mstrexport` command.

Execution example

```
mstrexport /tmp/work/mstruct.zip "/work/filelist.txt"
```

By default, you can collect the information required for building and operating a system in the Management Server such as various definition files of the Management Server, and the J2EE applications and resource adapters that are registered in the Management Server using the `mstrexport` command. In addition to this information, if you want to add user-created commands in the collection target of the `mstrexport` command, specify the absolute path of the file to be collected in the file that defines the save target for the Management Server administrative file.

Coding example of file

```
${cosminexus.home}/manager/apps/MyApp.ear  
/home/confdir/message1.conf
```

For files, see *8.2.13 Definition files to be saved for the Management Server management files* in the *uCosminexus Application Server Definition Reference Guide*.

17.7.5 HA monitor environment settings

According to the system being used, define the HA monitor environment in the definition file called as *sysdef*.

For details on the settings for the HA monitor environment, see the manual *Reliable System Monitoring Functionality HA Monitor*.

17.7.6 Creating the shell script file

Create the following shell script files for monitoring the Management Server process, and for starting and stopping the Management Server:

- Shell script file for monitoring the Management Server process
- Shell script file for starting the Management Server
- Shell script file for stopping the Management Server

You use the same shell script file in the Management Server of the active node and the Management Server of the spare node, and deploy the file in the same path.

(1) Shell script file for monitoring the Management Server process

An example of the shell script file for monitoring the Management Server process (`manager_mngsvr_monitor.sh`) is as follows:

```
#!/bin/sh

LOGDIR=/home/manager/hamon/log
MNGDIR=/opt/Cosminexus/manager

logg ()
{
    echo `date '+[%Y/%m/%d %H:%M:%S]'`"[$$]: $1" \
    >> ${LOGDIR}/mngsvr.log 2>&1
}

logg "### $0: started. ###"
while true
do
    $MNGDIR/bin/mngsvrutil -m 192.168.255.111:28080 check mngsvr
    if [ $? -ne 0 ]
    then
        logg "### $0: stop. ###"
        exit 0
    fi
    sleep 10
done
```

In this shell script file, you check the operational status of the Management Server with the `check` command of `mngsvrutil`. You specify the Alias IP address in the argument host name of the `-m` option of the `mngsvrutil` command.

(2) Shell script file for starting the Management Server

An example of the shell script file for starting the Management Server (`manager_mngsvr_start.sh`) is as follows:

```
#!/bin/sh

LOGDIR=/home/manager/hamon/log
MNGDIR=/opt/Cosminexus/manager
RETRY_COUNT=20
RETRY_INTERVAL=10
```

```

logg ()
{
    echo `date '+[%Y/%m/%d %H:%M:%S]'`"[$$]: $1" \
    >> ${LOGDIR}/mngsvr.log 2>&1
}
# start Management Server
logg "### $0: starting Management Server. ###"
$MNGDIR/bin/mngsvrctl start &

I=0
while [ $I -lt $RETRY_COUNT ] ; do
    $MNGDIR/bin/mngsvrutil -m 192.168.255.111:28080 check mngsvr
    if [ $? -eq 0 ] ; then
        break
    fi
    sleep $RETRY_INTERVAL
    I=`expr $I + 1`
done
exit 0

```

(3) Shell script file for stopping the Management Server

An example of the shell script file for stopping the Management Server (`manager_mngsvr_stop.sh`) is as follows:

```

#!/bin/sh

LOGDIR=/home/manager/hamon/log
MNGDIR=/opt/Cosminexus/manager
logg ()
{
    echo `date '+[%Y/%m/%d %H:%M:%S]'`"[$$]: $1" \
    >> ${LOGDIR}/mngsvr.log 2>&1
}
# stop Management Server
logg "### $0: stop Management Server. ###"
$MNGDIR/bin/mngsvrctl stop
exit 0

```

17.7.7 Server-compliant environment settings

In the server-compliant environment settings of the HA monitor, define the environment for the executing server and the standby server to be operated on the node.

You define the server-compliant environment for the 1-to-1 node switching of the Management Server in the definition file named `servers`. The following table lists the contents to be set up in the server-compliant environment settings:

Table 17–3: Contents to be set up in the server-compliant environment settings (in 1-to-1 node switching of the management server)

Operand	Setting contents
name	Specify the shell script file for starting the Management Server. Example specification: <code>/home/manager/hamon/bin/manager_mngsvr_start.sh</code>

Operand	Setting contents
alias	Specify the server identification name. Specify the same name in the active node and the spare node. Example specification: CMS
acttype	Specify the server starting method. Specify 'monitor' since the server is started using the HA monitor command.
termcommand	Specify the shell script file for stopping the management server. Example specification: /home/manager/hamon/bin/manager_mngsvr_stop.sh
initial	Specify the server startup status. <ul style="list-style-type: none"> For the active node Specify 'online'. For the spare node Specify 'standby'.
disk	Specify the name of the character type special file for the shared disk device. Example specification: /dev/vg00
lan_updown	Specify whether to use the LAN status setup files. Specify use since the LAN status setup files are used here.
fs_name	Specify the absolute path name of the logical volume corresponding to the switched file system. Note that this setting is required only when \$TPFS is used in the UNIX file. Example specification: /dev/rdisk0
fs_mount_dir	Specify the absolute path name of the directory to mount the switched file system. Note that this setting is required only when \$TPFS is used in the UNIX file. Example of specification: /hamon
patrolcommand	Specify the shell script file for monitoring the Management Server process. Example specification: /home/manager/hamon/bin/manager_mngsvr_monitor.sh
servexec_retry	Specify the frequency of retrying when failure is detected. Specify 0 since the node is switched without retrying when failure occurs.
waitserv_exec	Specify whether to wait for the execution of the start command to finish when executing the start completion process of the management server. Specify yes since the server waits for the execution to finish.

The examples of the servers file are described below:

Example of servers file (in the active node)

An example of the servers file for the active node is as follows:

```
server name    /home/manager/hamon/bin/manager_mngsvr_start.sh,
alias         CMS,
acttype       monitor,
termcommand   /home/manager/hamon/bin/manager_mngsvr_stop.sh,
initial       online,
disk          /dev/vg00,
lan_updown    use,
fs_name       /dev/rdisk0,
fs_mount_dir  /hamon,
patrolcommand /home/manager/hamon/bin/manager_mngsvr_monitor.sh,
servexec_retry 0,
waitserv_exec yes;
```

Example of servers file (in the spare node)

An example of the servers file for the spare node is as follows:


```
server name    /home/manager/hamon/bin/manager_mngsvr_start.sh,  
alias          CMS,  
acttype        monitor,  
termcommand    /home/manager/hamon/bin/manager_mngsvr_stop.sh,  
initial        standby,  
disk           /dev/vg00,  
lan_updown     use,  
fs_name        /dev/rdisk0,  
fs_mount_dir   /hamon,  
patrolcommand  /home/manager/hamon/bin/manager_mngsvr_monitor.sh,  
servexec_retry 0,  
waitserv_exec  yes;
```

For details on the server-compliant environment settings, see the manual *Reliable System Monitoring Functionality HA Monitor*.

17.7.8 LAN status setup

You define the settings for switching LAN in the HA monitor by specifying the IP address of the LAN adapter in the LAN status setup files of the HA monitor.

You set up the alias IP address in the following files:

- *Server-identification-name*.up **file**

Use this file to connect LAN. You specify the alias IP address to be added to the LAN adapter.

- *Server-identification-name*.down **file**

Use this file to disconnect LAN. You specify the alias IP address to be deleted from the LAN adapter.

Specify the alias value of the server-compliant environment settings (servers file) in the *server-identification-name*.

For details on the LAN status setup, see the manual *Reliable System Monitoring Functionality HA Monitor*.

17.8 Starting and stopping the 1-to-1 node switching system in Application Server (In Windows)

This section describes the procedure for starting and stopping the system when you are using 1-to-1 node switching system in Application Server. This section also describes the procedure for starting and stopping the system when you are performing maintenance of the J2EE server or the batch server after starting the operation.

In the *1-to-1 node switching system of Application Server*, use the cluster service. For details on the cluster service, see the OS manual.

When performing the operation using the 1-to-1 node switching system, you must specify beforehand the required environment settings, such as preparing the two hosts as the active node and spare node, and registering the script for monitoring, starting, and stopping the Administration Agent in which you run the cluster service. For details on how to specify the settings, see *17.4 Settings of the 1-to-1 node switching system of Application Server (In Windows)*.

Note that the description in this section is based on the prerequisite that before you execute planned node switching according to the procedure; the active node host must be running as the executing node host.

Important note

If node switching occurs as a result of failure of the Administration Agent, the processes of each logical server remains in the host in which the Administration Agent has failed. If processes are still running, you cannot start the logical servers when node switching occurs, and therefore, you must stop all processes before occurrence of node switching. Apart from this, execute the tasks beforehand for proper start of applications.

Tip

When you use the 1-to-1 node switching system, start Management Server in Management Server. In each Application Server host, do not specify the settings for starting the Administration Agent together with the OS.

17.8.1 Starting the 1-to-1 node switching system of Application Server

This subsection describes how to start the 1-to-1 node switching system of Application Server.

To start the 1-to-1 node switching system of the Application Server, you must start Management Server beforehand. If you have not yet started Management Server, start it first. For starting the Management Server, see *4.1.1 Procedure for starting a system* and *4.1.2 Methods of starting a system* in the *uCosminexus Application Server Management Portal User Guide*.

Following is the procedure to start the 1-to-1 node switching system of Application Server:

1. From the **Start** menu, select **Control Panel - Performance and Maintenance - Management Tools**, and then select **Cluster Administrator**.
The Cluster Administrator starts.
2. In the console tree (left pane), select the active node, and then from the **File** menu, choose **Start Cluster Service**.
The cluster service of the active node starts.
3. In the console tree (left pane), select the spare node, and then from the **File** menu, choose **Start Cluster Service**.

The cluster service of the spare node starts.

4. In the console tree (left pane), select the resource group that contains the active node and spare node, and then from the **File** menu, choose **Set to Online**.

The 1-to-1 node switching system starts.

17.8.2 Stopping the 1-to-1 node switching system of Application Server

To stop the 1-to-1 node switching system of Application Server:

1. From the **Start** menu, select **Control Panel - Performance and Maintenance - Management Tools**, and then select **Cluster Administrator**.

The Cluster Administrator starts.

2. In the console tree (left pane), select the resource group that contains the executing node and standby node, and then from the **File** menu, choose **Set to Offline**.

The resource group containing the executing node and standby node goes offline.

3. In the console tree (left pane), select the spare node, and then from the **File** menu, choose **Stop Cluster Service**.

The cluster service of the spare node stops.

4. In the console tree (left pane), select the active node, and then from the **File** menu, choose **Stop Cluster Service**.

The cluster service of the active node stops.

Manually stop Management Server at the end. For stopping the Management Server, see *4.1.3 Procedure for stopping a system* and *4.1.4 Methods of stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.

17.8.3 Starting and stopping during planned node switching in the 1-to-1 node switching system of Application Server

This subsection describes the procedure for switching the executing node and standby node of 1-to-1 node switching system in Application Server in cases other than those where an error has occurred. When you are switching nodes, Application Server in the standby node host must be in the standby state.

1. From the **Start** menu, select **Control Panel - Performance and Maintenance - Management Tools**, and then select **Cluster Administrator**.

The Cluster Administrator starts.

2. In the console tree (left pane), select the resource group that contains the executing node and standby node, and then from the **File** menu, choose **Migrate Group**.

The nodes switch.

17.8.4 Starting and stopping the system during maintenance of the 1-to-1 node switching system of Application Server

This subsection describes the procedures for starting and stopping the system when you perform maintenance of a 1-to-1 node switching system in Application Server.

Note that these procedures are applicable when Application Server is already running in the executing node host.

(1) Procedure when you do not need to restart the system during maintenance

1. In the executing node host, stop the running J2EE application and resource adapter.
In case of a batch server, check that the batch application is not running and then stop the resource adapter.
For stopping the system, see *Appendix F.4 How to terminate a system* in the *uCosminexus Application Server Command Reference Guide*.
2. Use the server management commands to execute the maintenance process.
For operations of the server management commands, see 3. *Basic Operations of Server Management Commands* in the *uCosminexus Application Server Application Setup Guide*.
3. Start the J2EE application and resource adapter that you stopped in step 1.
In case of the J2EE server, start the J2EE application and resource adapter. In case of the batch server, start the resource adapter.
For the startup methods, see *Appendix C.2 How to start a system* in the *uCosminexus Application Server Command Reference Guide*.
4. As and when required, check the operation of the J2EE application and resource adapter.
5. From the **Start** menu, select **Control Panel - Performance and Maintenance - Management Tools**, and then select **Cluster Administrator**.
The Cluster Administrator starts.
6. In the console tree (left pane), select the resource group that contains the executing node and standby node, and then from the **File** menu, choose **Migrate Group**.
The nodes switch.
7. In the executing node host following switching, execute step 1. to step 6.
8. To return the post-switching executing node host to the executing node, in the console tree (left pane), select the resource group that contains the executing node and standby node, and then from the **File** menu, choose **Migrate Group**.
Maintenance is complete.

(2) When you need to restart the system during maintenance (stopping both nodes concurrently)

1. When you have not specified the settings for batch stop of the logical servers, stop each logical server in the executing node host.
This step is not required when you have specified the settings for batch stop with the Management Server. For stopping the logical servers, see the following manuals:
 - In a system executing J2EE applications
See 4.1.3 *Procedure for stopping a system* and 4.1.4 *Methods of stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.
 - In a system executing batch applications
See 6.1.3 *Procedure for stopping a system* and 6.1.4 *Methods of stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.

2. From the **Start** menu, select **Control Panel - Performance and Maintenance - Management Tools**, and then select **Cluster Administrator**.
The Cluster Administrator starts.
3. In the console tree (left pane), select the resource group that contains the executing node and standby node, and then from the **File** menu, choose **Set to Offline**.
The resource group containing the executing node and standby node goes offline.
4. In the console tree (left pane), select the spare node, and then from the **File** menu, choose **Stop Cluster Service**.
The cluster service of the spare node stops.
5. In the console tree (left pane), select the active node, and then from the **File** menu, choose **Stop Cluster Service**.
The cluster service of the active node stops.
6. In the active node and spare-node host, execute the maintenance operation including changing the definition file.
7. In the console tree (left pane), select the active node, and then from the **File** menu, choose **Start Cluster Service**.
The cluster service of the active node starts.
8. In the console tree (left pane), select the spare node, and then from the **File** menu, choose **Start Cluster Service**.
The cluster service of the spare node starts.
9. In the console tree (left pane), select the resource group that contains the active node and spare node, and then from the **File** menu, choose **Set to Online**.
The 1-to-1 node switching system restarts.
10. When you have not specified the settings for batch start of the logical servers, start each logical server in the executing node host.
This step is not required when you have specified the settings for batch start with Management Server. For the startup method, see the following manuals:
 - In a system executing J2EE applications
See *4.1.1 Procedure for starting a system* and *4.1.2 Methods of starting a system* in the *uCosminexus Application Server Management Portal User Guide*.
 - In a system executing batch applications
See *6.1.1 Procedure for starting a system* and *6.1.2 Methods of starting a system* in the *uCosminexus Application Server Management Portal User Guide*.
11. In the executing node host, confirm the operation related to definition changes as and when required.
Maintenance is complete.

(3) When you need to restart the system during maintenance (avoid stopping both the nodes concurrently)

1. In the spare-node host, execute the maintenance operation including changing the definition file.
2. From the **Start** menu, select **Control Panel - Performance and Maintenance - Management Tools**, and then select **Cluster Administrator**.
The Cluster Administrator starts.
3. In the console tree (left pane), select the resource group that contains the executing node and standby node, and then from the **File** menu, choose **Migrate Group**.

Application Server running on the executing node host stops, and Application Server starts on the node that was in standby status until now. As a result, the host for which you have already performed maintenance becomes the executing node host.

4. In the post-switching executing node host, confirm the operation related to definition changes as and when required.
5. In the post-switching standby node host, execute the maintenance operation including editing of definition files.
6. To return the post-switching executing node host to the executing node, in the console tree (left pane), select the resource group that contains the executing node and standby node, and then from the **File** menu, choose **Migrate Group**.
7. In the post-switching executing node host, confirm the operation related to definition changes.
Maintenance is complete.

17.9 Starting and stopping a 1-to-1 node switching system for the Management Server (In Windows)

This section describes how to start and stop the system when you use the 1-to-1 node switching system for the Management Server.

The 1-to-1 node switching system for the Management Server uses the cluster service. For details on the cluster service, see the OS manual.

To use the 1-to-1 node switching system for the Management Server, you must first specify the necessary environment settings such as preparing the two hosts of the active node and spare node, and registering the script for monitoring, starting, and stopping the Management Server that is the target of the cluster service. For details on the setup methods, see [17.5 Settings of the 1-to-1 node switching system in the Management Server \(In Windows\)](#). For details on the active node and spare node, see [15.2 Operations that can be implemented by linking with cluster software](#).

17.9.1 Starting the 1-to-1 node switching system for the Management Server

This subsection describes how to start the 1-to-1 node switching system for the Management Server.

To start the 1-to-1 node switching system for the Management Server, you must first start the Administration Agent of the Application Server that you want to start. If the Administration Agent is not running, start the Administration Agent first. For starting the Administration Agent, see [4.1.2 Methods of starting a system](#) in the *uCosminexus Application Server Management Portal User Guide*.

To start the 1-to-1 node switching system for the Management Server:

1. From the **Start** menu, choose **Control Panel - Performance and Maintenance - Management Tools**, and then select **Cluster Administrator**.
The Cluster Administrator starts.
2. In the console tree (left pane), select the active node of the Management Server, and then from the **File** menu, choose **Start Cluster Service**.
The cluster service of the active node starts.
3. In the console tree (left pane), select the spare node of the Management Server, and then from the **File** menu, choose **Start Cluster Service**.
The cluster service of the spare node starts.
4. In the console tree (left pane), select the resource group that contains the active node and spare node of the Management Server, and then from the **File** menu, choose **Set to Online**.
The 1-to-1 node switching system for the Management Server will start.

17.9.2 Stopping the 1-to-1 node switching system for the Management Server

This subsection describes how to stop the 1-to-1 node switching system for the Management Server.

1. From the **Start** menu, choose **Control Panel - Performance and Maintenance - Management Tools**, and then select **Cluster Administrator**.
The Cluster Administrator starts.
2. In the console tree (left pane), select the resource group that contains the executing node and standby node of the Management Server, and then from the **File** menu, choose **Set to Offline**.
The resource group containing the executing node and standby node of the Management Server goes offline.
3. In the console tree (left pane), select the spare node of the Management Server, and then from the **File** menu, choose **Stop Cluster Service**.
The cluster service of the spare node of the Management Server will stop.
4. In the console tree (left pane), select the active node of the Management Server, and then from the **File** menu, choose **Stop Cluster Service**.
The cluster service of the active node of the Management Server will stop.
5. The Administration Agent of the application server will stop.
For stopping the Administration Agent, see *4.1.4 Methods of stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.

17.9.3 Starting and stopping a system when there is planned node switching in the 1-to-1 node switching system for the Management Server

This subsection describes how to switch the executing node and standby node of the 1-to-1 node switching system for the Management Server for the issues other than errors. When the node is switched, the Application Server must be in the standby status in the standby node host.

1. If you change the Management Server settings while the system is running, collect the Management Server settings of the executing node and copy from the executing node to the standby node. Also, execute the `mstrimport` command to apply the copied Management Server settings in the standby node.
There would be same settings in the Management Server of the executing node and the Management Server of the standby node.
For details on how to copy the Management Server settings from the executing node to the standby node and to apply the settings in the standby node, see the description related to the settings for the 1-to-1 node switching system in *17.7.4 Copying the Management Server settings from the active node to the spare node*.
2. From the **Start** menu, choose **Control Panel - Performance and Maintenance - Management Tools**, and then select **Cluster Administrator**.
The Cluster Administrator starts.
3. In the console tree (left pane), select the resource group that contains the executing node and standby node, and then from the **File** menu, choose **Migrate Group**.
The node will switch.

17.10 Starting and stopping a 1-to-1 node switching system for the Management Server (In UNIX)

This section describes how to start and stop the system when you use *HA monitor-based 1-to-1 node switching system*. This section also describes the starting and stopping procedure when maintenance is performed for the Management Server after starting the operations. Note that the HA monitor can only be used in AIX or Linux.

To use the 1-to-1 node switching system for operations, you must first specify the necessary environment settings such as preparing the two hosts of the active node and spare node and registering the script for monitoring, starting, and stopping the Management Server (in the 1-to-1 node switching system of the Management Server) to be monitored by the HA monitor. For details on method of setup, see *17.6 Settings of the 1-to-1 node switching system of Application Server (In UNIX)* and *17.7 Settings of the 1-to-1 node switching system for the Management Server (In UNIX)*.

The following table lists and describes the operations that can be executed in the 1-to-1 node switching system of Management Server and the references for the operations:

Table 17–4: Operations that can be executed in the 1-to-1 node switching system

Operation	1-to-1 node switching system of the Management Server
Starting the system	17.10.1
Stopping the system	17.10.2
Starting and stopping the system when there is planned node switching of the executing node and the standby node	17.10.3
Starting and stopping the system for maintenance	17.10.5

The following commands are provided by the HA monitor. For details, see the manual *Reliable System Monitoring Functionality HA Monitor*.

- `monbegin` (Starting a server that does not have an interface with the HA monitor)
- `monend` (Communication to stop a server that does not have an interface with the HA monitor)
- `monsbystp` (Stopping a standby server)
- `monswap` (Planned node switching)

17.10.1 Starting the 1-to-1 node switching system

This subsection describes the points you must keep in mind when you start the system using the 1-to-1 node switching system and the procedure for starting the system.

(1) Points to remember when starting the system

Keep the following points in mind when you start a system using the 1-to-1 node switching system:

- The HA monitor deployed on the host of active node and spare node starts simultaneously as the OS.
- If you are using the Management Server to start the 1-to-1 node switching system of Application Server, make sure that you start the Management Server beforehand. For starting the Management Server, see *4.1.1 Procedure for starting a system* and *4.1.2 Methods of starting a system* in the *uCosminexus Application Server Management Portal User Guide*.

- When starting the 1-to-1 node switching system of the management server, start the Administration Agent of the Application Server existing in the management domain in advance, if the application server is deployed on another host. For starting the Administration Agent, see *4.1.2 Methods of starting a system* in the *uCosminexus Application Server Management Portal User Guide*.

(2) Procedure to start the system

To start a system using the 1-to-1 node switching system:

1. Execute the `monbegin` command in the active node host to start the active node host as the executing node.

```
# monbegin server-identification-name
```

In the underlined part, specify the server identification name specified in the operand 'alias' of the servers file.

As a result, the processes defined in the script file to start the Administration Agent are executed and the active node host will start as the executing node.

2. Execute the `monbegin` command in the spare node host to start the spare node host as the standby node.

```
# monbegin server-identification-name
```

In the underlined part, specify the server identification name specified in the operand 'alias' of the servers file.

As a result, the spare node host becomes the standby node and prepares for the failure in the executing node.

Important note

If you have not specified settings to collectively start the logical servers exist on the management domain when the Management Server starts, you start each logical server for the active node host. Note that if you have specified settings to collectively start the logical servers present in the management domain when the Management Server starts, you need not start the logical servers manually. For the startup methods, see the following manuals:

- In a system executing J2EE applications
See *4.1.1 Procedure for starting a system* and *4.1.2 Methods of starting a system* in the *uCosminexus Application Server Management Portal User Guide*.
- In a system executing batch applications
See *6.1.1 Procedure for starting a system* and *6.1.2 Methods of starting a system* in the *uCosminexus Application Server Management Portal User Guide*.

Reference note

For details on the servers file definition (server environment settings in the HA monitor), see [17.7.7 Server-compliant environment settings](#).

17.10.2 Stopping the 1-to-1 node switching system

This subsection describes the procedure for stopping the 1-to-1 node switching system.

The stopping procedure will be described for the following two cases:

- When stopping both the executing node and the standby node hosts
- When stopping only the standby node host

(1) When stopping both the executing node and the standby node hosts

To stop both the executing node and the standby node hosts:

1. If you have not specified the settings to collectively stop the logical servers exist on the management domain in the script for stopping the Administration Agent, stop each logical server of the executing node host.

You need not execute this step, if you specify settings to stop the logical server in the script for stopping the Administration Agent. For stopping the logical servers, see the following manuals:

- In a system executing J2EE applications
See *4.1.4 Methods of stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.
- In a system executing batch applications
See *6.1.4 Methods of stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.

2. Execute the `monend` command in the executing node host to stop the Administration Agent of the executing node host.

```
# monend server-identification-name
```

In the underlined part, specify the server identification name specified in the operand 'alias' of the servers file.

As a result, the Administration Agent of the executing node host will stop. Also, stop instruction is automatically sent to the standby node by the HA monitor and the Administration Agent of the standby node host also stops.

Note that if you are using the Management Server in the 1-to-1 node switching system of Application Server, manually stop the Management Server at the end. For stopping the Management Server, see *4.1.3 Procedure for stopping a system* and *4.1.4 Methods of stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.



Reference note

For details on the servers file definition (server environment settings in the HA monitor), see [17.7.7 Server-compliant environment settings](#).

(2) When stopping only the standby node host

To end the standby status in the standby node without stopping the executing node, execute the following command in the standby node host:

```
# monsbystp server-identification-name
```

In the underlined part, specify the server identification name specified in the operand 'alias' of the `servers` file.

17.10.3 Starting and stopping a system when there is a planned node switching in the 1-to-1 node switching system

This subsection describes the planned node switching of the executing node and standby node for the issues other than the problems in the 1-to-1 node switching system. When the node is switched, the Application Server must be in standby status in the standby node host.

The following points describe how to switch the executing node and standby node in the 1-to-1 node switching system in a planned manner. Note that the active node host is presumed to be running as the executing node.

1. If you have not specified the settings to collectively stop the logical servers exist in the management domain in the script for stopping the Administration Agent, stop each logical server of the executing node host.

You need not execute this step if you specify settings to stop the logical server in the script for stopping the Administration Agent. For stopping the logical servers, see the following manuals:

- In a system executing J2EE applications
See *4.1.4 Methods of stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.
- In a system executing batch applications
See *6.1.4 Methods of stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.

2. Execute the monswap command in the executing node host to switch the nodes.

```
# monswap server-identification-name
```

In the underlined part, specify the server identification name specified in the operand 'alias' of the servers file.

The processes defined in the script for stopping the Administration Agent are executed and the Application Server of the executing node host stops. Thereafter, the Application Server will start in the node that was in standby status until now. Thus, the planned node switching procedure is complete.

Reference note

- For details on how to start and stop the 1-to-1 node switching system for maintenance, see [17.10.5 Starting and stopping the 1-to-1 node switching system of the Management Server for maintenance](#).
- For details on the servers file definition (server environment settings in the HA monitor), see [17.7.7 Server-compliant environment settings](#).

17.10.4 Starting and stopping the system during maintenance of the 1-to-1 node switching system of Application Server

This subsection describes the procedure for starting and stopping the system when you perform maintenance of the 1-to-1 node switching system of Application Server.

The stopping procedure for the following three cases is as follows:

- When you do not need to restart the system during maintenance
- When you need to restart the system during maintenance (stopping both nodes concurrently)
- When you need to restart the system during maintenance (avoid stopping both the nodes concurrently)

Note that these procedures are applicable when Application Server is already running in the executing node host.

Important note

If node switching occurs as a result of failure of the Administration Agent, the processes of each logical server remains in the host in which the Administration Agent has failed. If processes are still running, you cannot start the logical servers when node switching occurs, and therefore, you must stop all processes before occurrence of node switching. Apart from this, execute the tasks for proper start of applications beforehand.

(1) When you do not need to restart the system during maintenance

To start and stop the system when you do not need to perform restart during maintenance:

1. In the executing node host, stop the running J2EE application and resource adapter.
In case of a batch server, check that the batch application is not running, and then stop the resource adapter.
For stopping the system, see the following manuals:
 - In a system executing J2EE applications
See *4.1.3 Procedure for stopping a system* and *4.1.4 Methods of stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.
 - In a system executing batch applications
See *6.1.3 Procedure for stopping a system* and *6.1.4 Methods of stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.
2. Use the server management commands to execute the maintenance process.
For operations of the server management commands, see *3. Basic Operations of Server Management Commands* in the *uCosminexus Application Server Application Setup Guide*.
3. Start the J2EE application and resource adapter that you stopped in step 1.
In case of the J2EE server, start the J2EE application and resource adapter. In case of the batch server, start the resource adapter.
For the startup methods, see the following manuals:
 - In a system executing J2EE applications
See *4.1.1 Procedure for starting a system* and *4.1.2 Methods of starting a system* in the *uCosminexus Application Server Management Portal User Guide*.
 - In a system executing batch applications
See *6.1.1 Procedure for starting a system* and *6.1.2 Methods of starting a system* in the *uCosminexus Application Server Management Portal User Guide*.
4. As and when required, check the operation of the J2EE application and resource adapter.
5. In the executing node host (active node host), execute the `monswap` command and switch the nodes.
The executing node host (active node host) changes to the standby node host, and the host that earlier was the standby node (spare-node host) becomes the executing node host.
6. In the executing node host following switching (spare-node host), execute step 1. to step 4.
7. In the post-switching executing node host (spare-node host), execute the `monswap` command again as and when required.
Execute this step to return the active node host to the executing node.
This completes the start and stop process during maintenance.

(2) When you need to restart the system during maintenance (How to stop both the nodes concurrently)

Following is the procedure to start and stop the system by stopping both the nodes concurrently when you need to perform restart during maintenance:

1. When you have not specified the settings for batch stop of the logical servers of the management domain in the script for stopping the Administration Agent, stop each logical server in the executing node host.

This step is not required when you have specified the settings for stopping the logical servers in the script for stopping the Administration Agent. For stopping the logical servers, see the following manuals:

- In a system executing J2EE applications
See *4.1.4 Methods of stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.
- In a system executing batch applications
See *6.1.4 Methods of stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.

2. In the executing node host, execute the `monend` command and stop the system.

```
# monend server-identification-name
```

In the underlined part, specify the identification name of the server specified in the operand `alias` in the `servers` file.

The executing node host stops. Also, an instruction for stopping is output automatically in the standby node by the HA monitor, and the standby node host also stops.

3. In the active node and spare node host, execute the maintenance operation including changing the definition file.
4. In the active node host, execute the `monbegin` command and start the Administration Agent of the active node host.

```
# monbegin server-identification-name
```

In the underlined part, specify the identification name of the server specified in the operand `alias` in the `servers` file.

By doing this, the active node host starts operating as the executing node.

5. Execute the `monbegin` command in the spare-node host.

```
# monbegin server-identification-name
```

In the underlined part, specify the identification name of the server specified in the operand `alias` in the `servers` file.

By doing this, the spare-node host starts operating as the standby node.

6. When you have not specified the settings for batch start of the logical servers of management domain in the script for starting the Administration Agent, start each logical server of the executing node host.

This step is not required when you have specified the settings for starting the logical servers in the script for starting the Administration Agent. For starting the logical servers, see the following manuals:

- In a system executing J2EE applications
See *4.1.2 Methods of starting a system* in the *uCosminexus Application Server Management Portal User Guide*.
- In a system executing batch applications
See *6.1.2 Methods of starting a system* in the *uCosminexus Application Server Management Portal User Guide*.

7. In the executing node host, confirm the operation related to definition changes as and when required.

Reference note

For details on defining the `servers` file (setting the server environment with the HA monitor), see [17.6.6 Setting the server-compliant environment](#).

(3) When you need to restart the system during maintenance (How to avoid stopping both the nodes concurrently)

To start and stop the system by stopping both nodes concurrently when you need to perform restart during maintenance:

1. In the standby node host, execute the maintenance operation including changing the definition file.
2. In the executing node host, execute the `monswap` command and switch the nodes.

```
# monswap server-identification-name
```

In the underlined part, specify the identification name of the server specified in the operand `alias` in the `servers` file.

Application Server running on the executing node host stops, and Application Server starts on the node that was in the standby status until now. As a result, the host for which you have already performed maintenance becomes the executing node host.

3. In the post-switching executing node host (spare node host), confirm the operation related to definition changes as and when required.
4. In the post-switching standby node host (active node host), execute the maintenance operation including editing of definition files.
5. In the post-switching executing node host (spare node host), execute the `monswap` command again as and when required.
6. In the post-switching executing node host (active node host), confirm the operation related to definition changes as and when required.

Execute this step when returning the active node host to the executing node.

This completes the start and stop process during maintenance.

Reference note

For details on defining the `servers` file (setting the server environment with the HA monitor), see [17.6.6 Setting the server-compliant environment](#).

17.10.5 Starting and stopping the 1-to-1 node switching system of the Management Server for maintenance

This subsection describes how to start and stop the 1-to-1 node switching system of the Management Server for maintenance in order to change the Management Server settings.

Note that this is a procedure for the case in which the Management Server is already running in the executing node host.

1. Execute the `mstrexport` command in the executing node host (active node host).

```
# mstrexport mstruct.zip
```

The Management Server settings of the active node host are automatically collected and the collected information is saved in a ZIP file.

In the underlined part, specify the path of the ZIP file that saves the automatically collected settings.

2. Copy the ZIP file saved in step 1 from the executing node to the standby node.

3. Execute the `mstrimport` command in the standby node host (spare node host).

```
# mstrimport mstruct.zip
```

In the underlined part, specify the path of the ZIP file copied in step 2.

As a result, the Management Server settings copied in step 2. are deployed in the standby node host and the Management Server of the executing node and the standby node have the same settings.

4. Execute the `monswap` command in the executing node host to switch the nodes.

```
# monswap server-identification-name
```

In the underlined part, specify the server identification name specified in the operand 'alias' of the servers file.

The Management Server of the executing node host stops, and then the Management Server starts through the node that was in standby status until now. The executing node settings are applied in the system when the Management Server starts.

5. As and when required, check the operations related to the changes in definitions in the executing node host (spare node host) after the nodes are switched.

6. As and when required, re-execute the `monswap` command in the executing node host (spare node host) after the nodes are switched.

Execute the command to return the active node host to the executing node.

7. As and when required, check the operations related to the changes in definitions in the executing node host (active node host) after the nodes are switched.

Therefore, the procedure for starting and stopping the system for maintenance is complete.



Reference note

For details on the servers file definition (server environment settings in the HA monitor), see [17.7.7 Server-compliant environment settings](#).

17.11 Confirming the settings of a node switching system

The J2EE server and HTTP Server settings to be specified on the active and spare nodes are identical in principle. Compare and confirm the content of the settings files shown in the following sections.



Important note

For the J2EE server, the definition might differ depending on the component to be used. Make sure that component-specific property settings comply with the specifications of the relevant component.

17.11.1 J2EE server settings

Table 17–5: Settings files for the J2EE server

No.	Settings file	File location
1	usrconf.cfg (option definition file for J2EE servers)	<ul style="list-style-type: none">• In Windows <i>Cosminexus-installation-directory</i>\CC\server\usrconf\ejb\server-name\• In UNIX /opt/Cosminexus/CC/server/usrconf/ejb/server-name/
2	usrconf.properties ^{#1} (user property file for J2EE servers)	
3	server.policy (security policy file for J2EE servers)	
4	Connector property file	This file is obtained by executing the <code>cjgetrarprop</code> (get the RAR file property) command. ^{#2}

#1: The setting of the `webserver.connector.http.permitted.hosts` key can be different from that for HTTP server.

#2: You cannot check the user name and password settings because the command cannot obtain them.

17.11.2 HTTP Server settings

Table 17–6: Settings files for HTTP Server

No.	Settings file	File location
1	httpsd.conf	<ul style="list-style-type: none">• In Windows <i>HTTP-Server-installation-directory</i>\servers\HWS_logical-server-name\conf\• In UNIX /opt/hitachi/httpsd/servers/HWS_logical-server-name/conf/
2	mime.types	

17.11.3 Management server settings

For details about the settings file for the management server, see *8.1 List of files used with Cosminexus Manager* in the manual *uCosminexus Application Server Definition Reference Guide*.

18

Mutual Node Switching System (Linked with Cluster Software)

This chapter describes the systems that link with the cluster software and operate with the mutual node switching systems.

18.1 Organization of this chapter

This chapter is organized as the following table:

Table 18–1: Organization of this chapter (Mutual node switching system (Linked with cluster software))

Category	Title	Reference
Description	Overview of the mutual node switching system	18.2
	System configuration and operations of mutual node switching systems	18.3
Settings	Settings for the mutual node switching system (In Windows)	18.4
	Settings for the mutual node switching system (In UNIX)	18.5
Operations	Starting and stopping the mutual node switching system (In Windows)	18.6
	Starting and stopping the mutual node switching system (In UNIX)	18.7

Note:

The function-specific explanation is not available for "Implementation" and "Precautions".

18.2 Overview of the mutual node switching system

The mutual node switching system is a system in which two servers are configured as a 1-to-1 node switching system. The two servers run as the active nodes, and each server becomes the spare node for the other server.

This section describes an example system configuration for a mutual standby system and the flow of the node switching process. The timing for node switching and the system operations during the switching of nodes are the same as for a 1-to-1 node switching configuration. The following table lists the reference sections:

Table 18–2: Node switching operations and their corresponding references

Node switching operations	Reference
Timing for node switching	17.3.2
System operations during node switching	17.3.4
Inheriting information during node switching	17.3.5

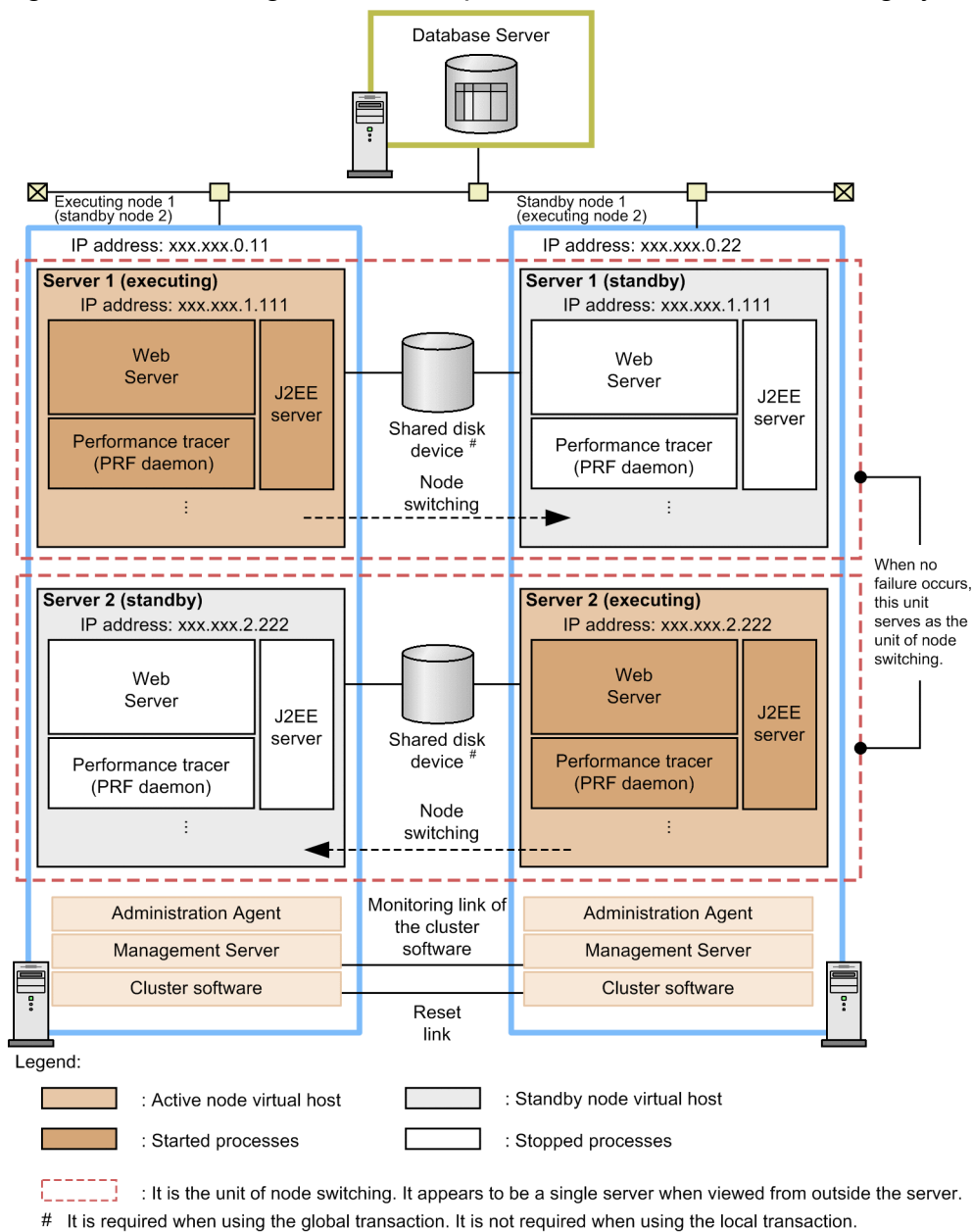
18.3 System configuration and operations of mutual node switching systems

This section describes an example configuration for a mutual node switching systems and the flow of the node switching process.

18.3.1 Example of system configuration of the mutual node switching system

The following figure shows an example configuration for a mutual node switching system that is integrated with cluster software. From outside the server such as from the client or database, Application Server appears to restart with the same logical address during the node switching.

Figure 18–1: Configuration example for a mutual node switching system



The mutual node switching system shown in the figure is as follows:

- Each Application Server of the executing node 1 (standby node 2) and standby node 1 (executing node 2) is managed in a different management domain.
- In each management domain, two virtual hosts of the active node and spare node are defined. The virtual hosts of the active node and spare node are used as the executing node Application Server and the standby node Application Server respectively.
- When failure occurs, the virtual hosts in the mutual management domains are switched. For example, as shown in the figure, if a failure occurs in the active node virtual host of server 1, the node is switched to the spare node virtual host of server 1.
- The IP address to be used in the operations of each server is the IP address dynamically allocated by the cluster software (*alias IP address*). For the mutual node switching system of the figure, 'xxx.xxx.1.111' and 'xxx.xxx.2.222' become the alias IP addresses. Note that the cluster software switches LAN for each IP address, and therefore, a unique value is allocated to the cluster IP address of server 1 and server 2. During node switching, the alias IP address is deleted in the active node virtual host, and the alias IP address is added in the spare node virtual host so the processing is continued.
- To send requests from the Management Server to the Administration Agent, use the IP address that does not move to another node due to node switching (*stationary IP address*) is used. In the mutual node switching system of the figure, 'xxx.xxx.0.11' of the active node 1 and 'xxx.xxx.0.22' of the spare node 1 become the stationary IP addresses.

Reference note

A virtual host is a configuration in which multiple different IP addresses can be allocated to one machine, and the machine can be used as multiple physical hosts. The virtual hosts in the same management domain can control operations such as starting and stopping Application Server using one Administration Agent, but the IP addresses used for operations are different, so the virtual hosts are handled as apparently different physical hosts.

You can execute the following types of operations in the mutual node switching systems:

Using the shared disk device

The use of the shared disk device is different in the local transaction and the global transaction.

- In a local transaction
The shared disk device is not required. In the local transaction, no session information is to be inherited between the executing node and standby node, so the shared disk device is not used.
- In the global transaction
The shared disk device is required. The shared disk device is used for inheriting the transaction information such as the OTS status during the node switching.

For JP1 integration

In a cluster software configuration, you can integrate the system with JP1.

For integrating with JP1, Application Server also requires JP1/Base. You must manage JP1 in the cluster software separately from Application Server.

Integration with cluster software in a database server

You can also use a cluster software configuration in the database server. In this case, if only the virtual address (logical address) is recognized in Application Server, you need not be aware that the database server is using the cluster software.

Application of the load balancer

Though not described in this system configuration example, you can also apply the load balancer by providing several Web servers in the same configuration. As a result, you can improve the reliability and operating rate of the Web server.

For details on the system configuration of the mutual node switching system, see *3.11.4 Configuration in which executing node and standby node of Application Server are mutually standby* in the *uCosminexus Application Server System Design Guide*.

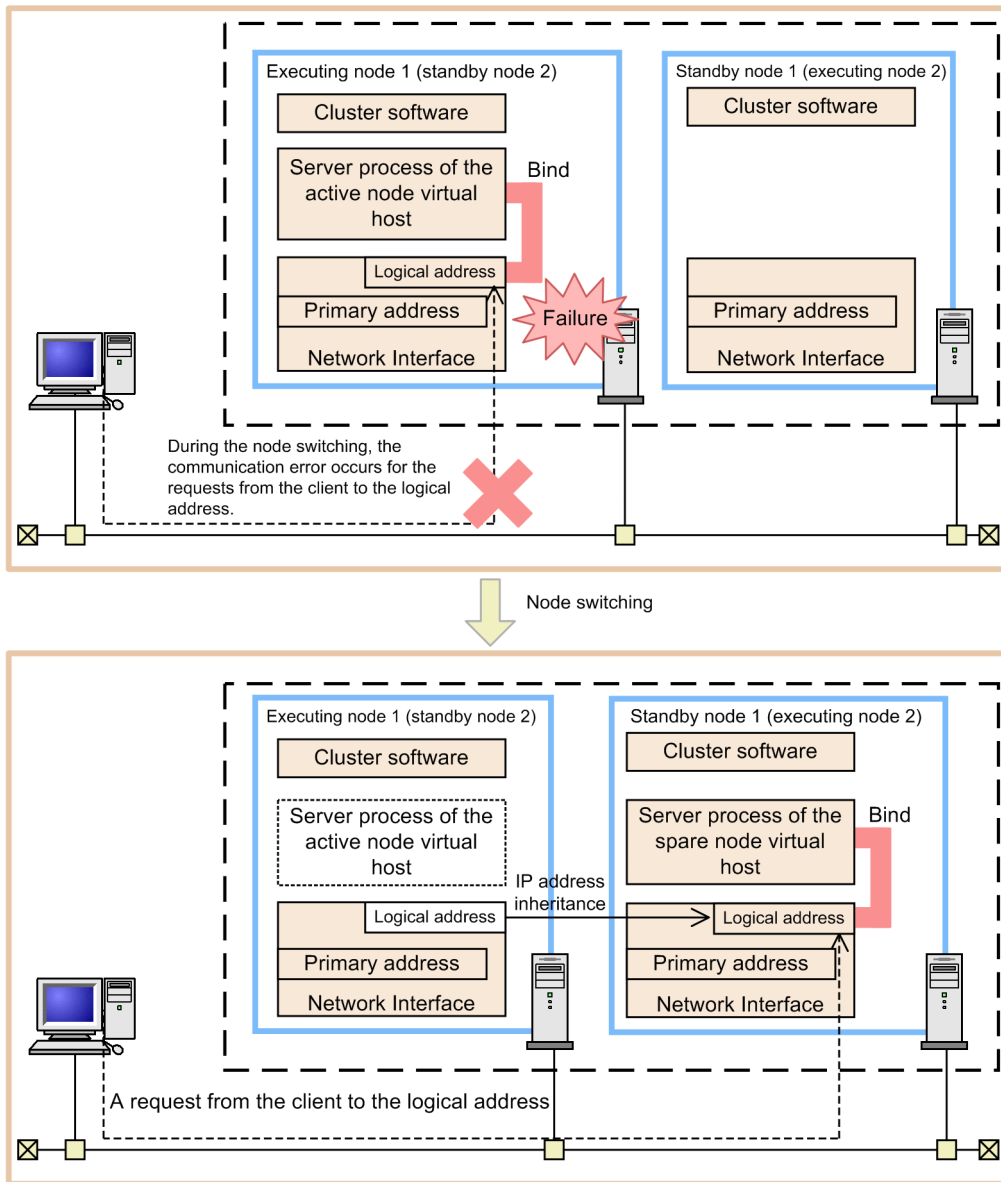
18.3.2 Flow of the node switching process in the mutual node switching system

This subsection describes the operations and the flow of processes during the node switching in mutual node switching systems.

(1) Node switching operations

The node switching is performed in the mutual node switching system, as shown in the following figure. The following figure shows an example when a failure occurs in the active node virtual host of the executing node 1 (standby node 2) and the node is switched to the spare node virtual host of the standby node 1 (executing node 2):

Figure 18–2: Node switching operations in the mutual node switching system



- The server process of the active node virtual host stops and the server process of spare node virtual host starts.
- IP address of the logical address is inherited from the active node virtual host to the spare node virtual host.

(2) Flow of the node switching process

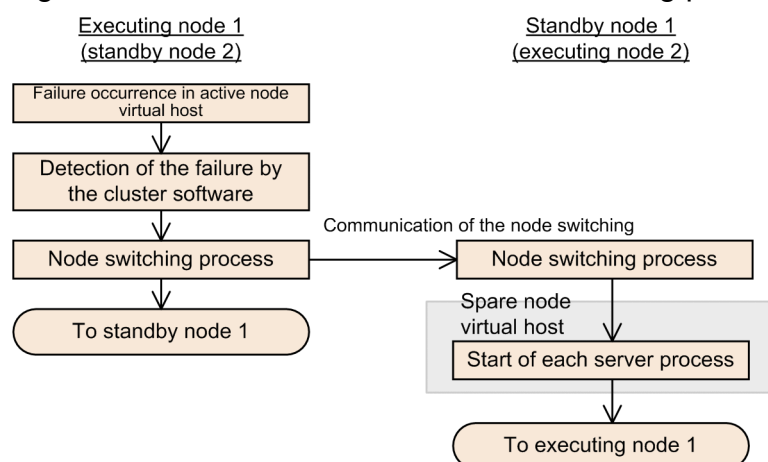
The node switching methods that can be applied in the mutual node switching system are the automatic node switching and the planned node switching. The *automatic node switching* is a method in which the cluster software automatically switches the node. An automatic node switching occurs, when a failure occurs in the executing node. On the other hand, the *planned node switching* is a method in which the operator switches the nodes in a planned manner during the node maintenance. The node is switched using the cluster software commands from the executing node.

The following points separately describe the flow of the node switching process for automatic node switching and for planned node switching:

(a) Flow of automatic node switching process

The following figure shows the flow of automatic node switching process in a mutual node switching system. The figure shows the process flow when a failure occurs in the active node virtual host of the executing node 1 (standby node 2) and the node is switched to the spare node virtual host of the standby node 1 (executing node 2):

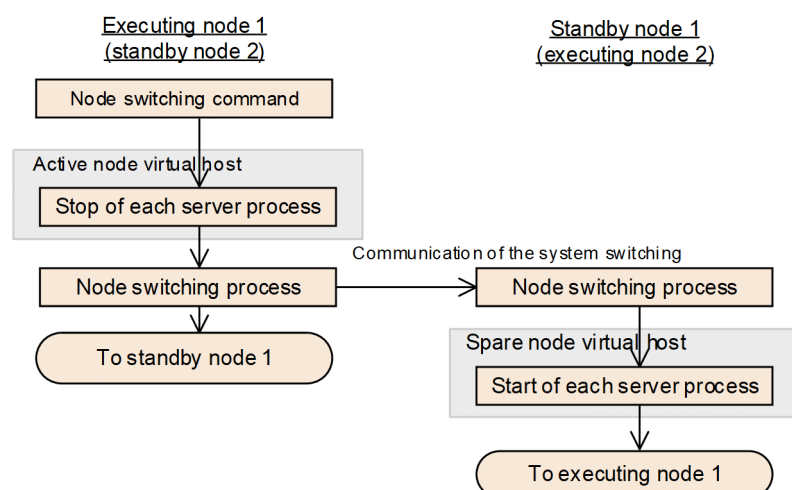
Figure 18–3: Flow of automatic node switching process in the mutual node switching system



(b) Flow of planned node switching process

The following figure shows the flow of planned node switching process in the mutual node switching system. The figure shows the process flow when the cluster software commands are used for switching the active node virtual host of the executing node 1 (standby node 2) to the spare node virtual host and the spare node virtual host of the standby node 1 (executing node 2) to the active node virtual host:

Figure 18–4: Flow of planned node switching process in the mutual node switching system



18.4 Settings for the mutual node switching system (In Windows)

The mutual node switching system is one of the configurations in which the executing node and the standby node of Application Server are arranged in a 1-to-1 ratio, and while each Application Server is operating as an executing node, each Application Server is set as the standby node for the other Application Server (*mutual standby configuration*). Deploy the same types of J2EE servers in each Application Server, and start different J2EE servers in advance, each Application Server operates as an executing node, and each Application Server functions as a standby node for the other Application Server. If a failure occurs in any one of the nodes, the system switches to the other node. This enables the economical operations using a few Application Server machines. You use the Smart Composer functionality to build this system. This subsection describes the settings for a mutual node switching system.

For details on the mutual node switching systems, see [18.2 Overview of the mutual node switching system](#). For the system configuration, see [3.11.4 Configuration in which executing node and standby node of Application Server are mutually standby](#) in the *uCosminexus Application Server System Design Guide*.

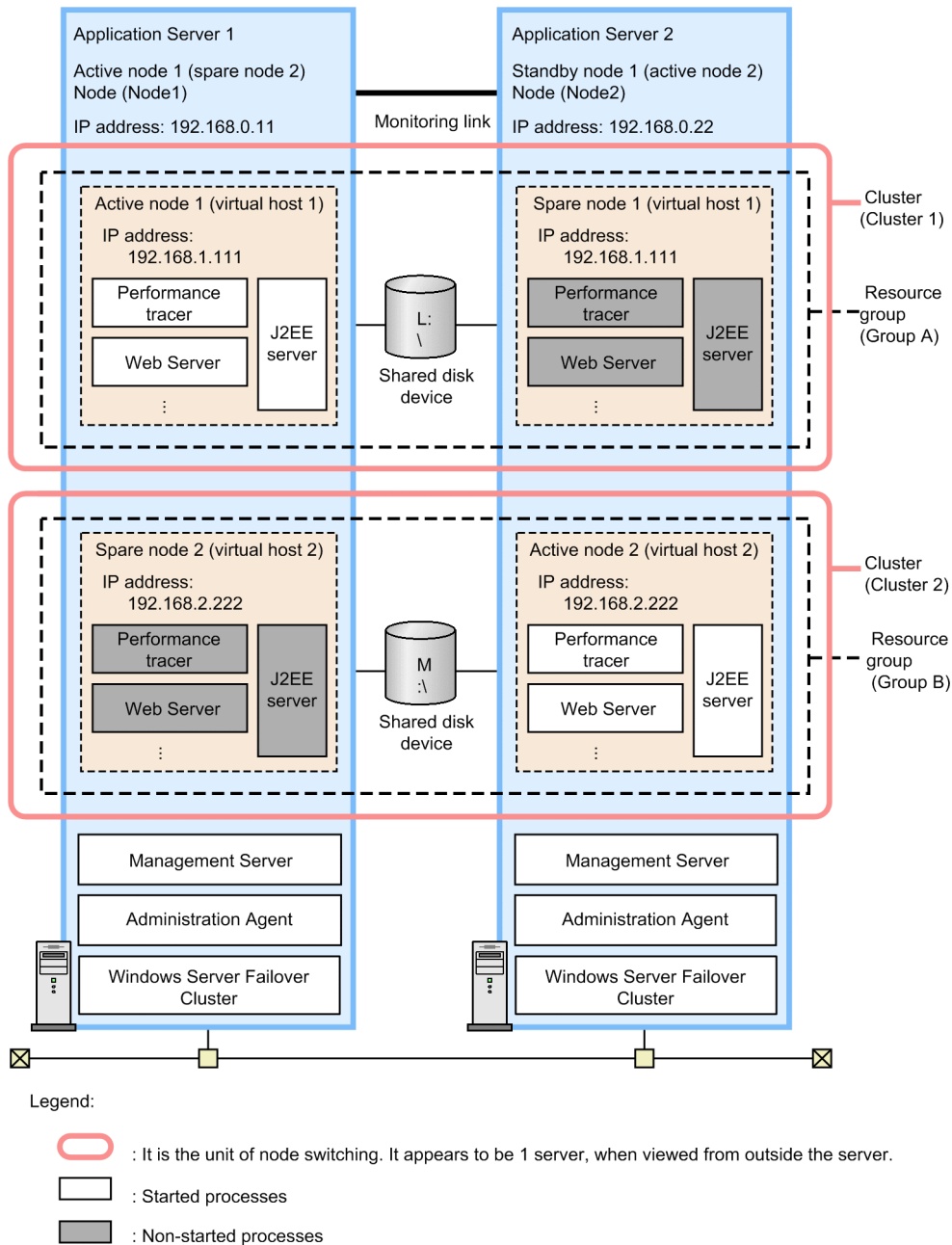
18.4.1 Procedure for setting up the mutual node switching systems

This subsection describes the example of the system configuration and the procedures for specifying the system settings when integrating with Windows Server Failover Cluster.

(1) System configuration examples

The following figure shows an example configuration of a mutual node switching system. Note that the subsequent points describe the system configuration based on this configuration example.

Figure 18–5: Configuration example of mutual node switching system (in Windows)



In this example, two types of application servers (assumed to be application server 1 and application server 2) are used. The executing node (active node 1) and standby node (spare node 2) of Application Server 1 and the executing node (active node 2) and standby node (spare node 1) of Application Server 2 are deployed in 1-to-1 ratio. Each Application Server deploys a Management Server having a separate management domain, and the Management Server is started on both the machines.

In the mutual node switching system, two virtual hosts are defined in one management domain and are configured as the host of the active node Application Server and the host of the spare node Application Server respectively. In this example, the combination is as follows:

- Active node 1 (virtual host 1) of Node 1 and spare node 1 (virtual host 1) of Node 2
- Active node 2 (virtual host 2) of Node 2 and spare node 2 (virtual host 2) of Node 1

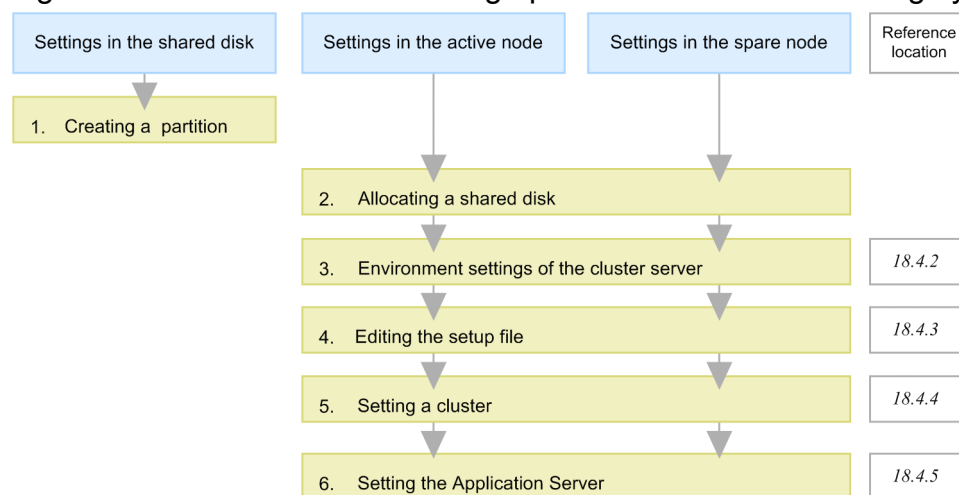
The virtual host controls the starting and stopping of Application Server using one Administration Agent, but the IP addresses allocated for operations are different, so the virtual hosts are defined as apparently different hosts.

Note that in order to deploy Application Server in a cluster configuration, set the alias IP address and specify settings in such a way so that the running node inherits the alias IP address, and the client does not realize the nodes in the cluster. For the mutual node switching system, the alias IP address is the address (cluster IP address) that is dynamically allocated by the cluster software. For the IP address to be used in the operations of Application Server, you use the cluster IP address, and for sending the requests from the Management Server to the Administration Agent, you use the IP address (stationary IP address) that does not move to other nodes due to node switching.

(2) Procedure for system setup

When integrating the system with Windows Server Failover Cluster, you must set up Management Server and Cluster Administrator. The following figure shows the procedure for setting up the mutual node switching system:

Figure 18–6: Procedure for setting up the mutual node switching system (in Windows)



Legend: ▼ : Necessary work

The following points describe the step 1 through 6 of the above figure:

1. Create a partition in the shared disk to build the file system.

You create a location for storing the cluster management file of Windows Server Failover Cluster. Also, when using the global transaction, create the location for storing the transaction information. The cluster management file and the transaction information can be stored in the same partition.

2. Allocate a shared disk to the system.

When allocating a shared disk to the system, allocate the same drive characters to the active node 1 and spare node 1, and to the active node 2 and spare node 2.

3. Set up the cluster server environment in the Management Server.

In the Easy Setup definition file of the Smart Composer functionality for Management Server, specify the settings for using Windows Server Failover Cluster. For details, see [18.4.2 Environment settings of the cluster server](#).

4. Edit the setup file.

Specify the settings for various definition files of Administration Agent, Management Server, and Cosminexus HTTP Server that cannot be set up in the Easy Setup definition file. For details, see [18.4.3 Editing the setup file](#).

5. Setting a cluster.

Create a script file for monitoring the environment settings of the cluster software, Management Sever, and Administration Agent. For details, see [18.4.4 Setting a cluster](#).

6. Set up Application Server in Management Server and Cluster Administrator.

Use Management Server and Cluster Administrator to deploy Application Server in the cluster configuration, and to set up J2EE applications and resource adapters. For details, see [18.4.5 Setting Application Server](#).

18.4.2 Environment settings of the cluster server

This subsection describes the points to be remembered when you specify the settings in Management Server for a system that is linked with Windows Server Failover Cluster.

Reference note

The Management Server must be set up on the host in which Management Server is used for the first time. For the Management Server setup, see [4.1.14 Setting the management functionality](#) in the *uCosminexus Application Server System Setup and Operation Guide*.

(1) Settings of the Easy Setup definition file

In the Easy Setup definition file of the Smart Composer functionality for the Management Server, you define the Web system and hosts. Note the following points when you define the hosts to set up logical servers, and define the configuration of each logical server:

- **Defining a Web system**

Define the Web system that will be used in the Smart Composer functionality. Specify a different Web system for each virtual host.

- **Defining the host**

You define a host that exists on the management domain. You specify the cluster IP address to be used in the server operations in the `<host-name>` tag for the virtual host of the active node and spare node respectively. In the example, the following is specified:

- In active node 1 of Node 1 and spare node 1 of Node 2: 192.168.1.111
- In active node 2 of Node 2 and spare node 2 of Node 1: 192.168.2.222

Also, in the `<agent-host>` tag, specify the stationary IP address of the host. In the example, the following is specified:

- In Node 1: 192.168.0.11
- In Node 2: 192.168.0.22

Note the following points when you specify the `<configuration>` tag of the logical J2EE server (j2ee-server) and logical Web server (web-server).

(a) Setting logical J2EE servers

- **Settings for fixing the host**

You specify any value in the following parameters of the `<configuration>` tag for the logical J2EE server (j2ee-server), and fix the management host and operation host:

- `mngagent.connector.host` parameter[#] (Host name or IP address of the Administration Agent)

- `vbroker.se.iiop_tp.host` parameter (Host name or IP address of the EJB container for each J2EE server)
- `webserver.connector.nio_http.bind_host` parameter (Host name or IP address used for the NIO HTTP server)

When you specify the `mngagent.connector.port` parameter along with the `mngagent.connector.host` parameter, specify a different port number for each virtual host.

- **Global transaction settings**

For using the global transaction, specify the following settings in the following parameters of the `<configuration>` tag for the logical J2EE server (`j2ee-server`):

- Specify 'false' in the `ejbserver.distributedtx.XATransaction.enabled` parameter (specify settings to enable or disable the light transaction functionality). By default, 'false' is set up.
- Specify the directory for the shared disk device in the `ejbserver.distributedtx.ots.status.directory1` parameter (location for saving the backup of the status file of the in-process OTS). By default, `otsstatus` is set up.

Example settings

`L:\Group1\otsstatus`

(b) Setting the logical Web server

The following settings must be specified in the `<configuration>` tag of the logical Web server (`web-server`):

- **Settings for fixing the host**

Specify the IP address or host name that receives the request in the `Listen` parameter and fix the host for the Web server.

18.4.3 Editing the setup file

You edit the setup file. This subsection describes the setup files for which you must take precautions when linking the system with Windows Server Failover Cluster.

(1) Setting the administration agent

Of the items set up in the `adminagent.properties` (Administration Agent property file), this point describes the settings that you must note when integrating the system with Windows Server Failover Cluster. For `adminagent.properties`, see 8.2.1 *adminagent.properties (Administration Agent property file)* in the *uCosminexus Application Server Definition Reference Guide*.

- `adminagent.adapter.bind_host` key
Specify the stationary IP address (in the example, 192.168.0.11 or 192.168.0.22) in the `adminagent.adapter.bind_host` key. The management request is received only in the stationary IP address.

(2) Settings for the management command (mngsvrutil) in the Management Server

Under the *Cosminexus-installation-directory*\manager\config directory, provide the common definition file at the client side (`mngsvrutilcl.properties`) for the management command (`mngsvrutil`) in the Management

Server, and specify the user ID and password of the management user in the Management Server. Also, set up appropriate access privileges for the common definition file that is at the client side.

You use this file to execute the `mngsvrutil` command in the script for monitoring the Administration Agent, and in the script for starting and stopping logical servers.

For the client-side common definition file (`mngsvrutilcl.properties`) of the `mngsvrutil` command, see *8.2.16 mngsvrutilcl.properties (Client-side shared definition file of the mngsvrutil command)* in the *uCosminexus Application Server Definition Reference Guide*.

(3) Setting Management Server

Among the items to be set in `mserver.properties` (Management Server environment settings file), the settings that you must be careful about when integrating with Windows Server Failover Cluster are as follows. For the `mserver.properties` file, see *8.2.6 mserver.properties (Management Server environment settings file)* in the *uCosminexus Application Server Definition Reference Guide*.

- **com.cosminexus.mngsvr.logical_server_abnormal_stop.exit**

Specify the settings for switching the nodes when the operating status of the logical servers is 'abnormal termination' (when the system has exceeded the automatic restart frequency, or when the system detects a failure at automatic restart frequency 0).

A setup example is as follows:

```
com.cosminexus.mngsvr.logical_server_abnormal_stop.exit=true
```

Important note

If automatic restart is specified for Management Server, node switching is not executed during this time. For executing the node switching during this time, do not set up automatic restart for Management Server.

Set the following items as and when required:

- **webserver.connector.http.bind_host**

Specify the real IP address in the `webserver.connector.http.bind_host` key.

- **mngsvr.myhost.name**

Specify the real IP address in the `mngsvr.myhost.name` key.

- **com.cosminexus.mngsvr.management.host**

Specify the virtual IP address in the `com.cosminexus.mngsvr.management.host` key.

18.4.4 Setting a cluster

You execute the following settings for a cluster:

- Create a script file
- Environment settings of the cluster software

(1) Create a script file

Create a script file for starting and stopping a monitoring target. Also, you must determine how to monitor a cluster in the script file. For details on how to create the script files, see the documentation of the OS that is being used.

This subsection describes the monitoring targets and the monitoring methods for mutual node switching systems.

(a) Monitoring targets

The monitoring targets for mutual node switching systems are as follows:

- Management Server
- Administration Agent

The node switching is executed when Management Server or Administration Agent ends.

(b) Monitoring method

You monitor the availability of processes of the monitoring targets. The following table lists the processes to be monitored.

Table 18–3: Process to be monitored

Monitoring target	Process name
Management Server	mngsvr.exe
Administration Agent	adminagent.exe

(2) Environment settings of Windows Server Failover Cluster

For details on the environment settings of Windows Server Failover Cluster, see the documentation of the cluster software that is being used.

18.4.5 Setting Application Server

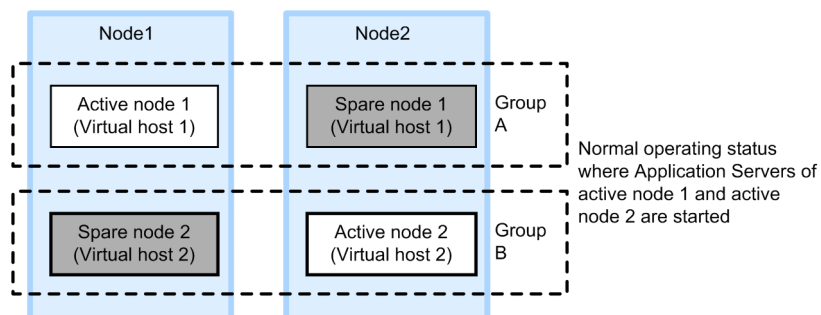
In the Application Server settings, you use Management Server and Cluster Administrator to deploy Application Server in the cluster configuration. Also, you import and start the J2EE applications and resource adapters in the J2EE server. To set Application Server:

1. Start the Administration Agent and Management Server in Node 1 and Node 2 respectively.

For details on how to start the Administration Agent and Management Server, see [2.6 Settings for starting and stopping the system](#).

2. Start the cluster service and set the resource groups GroupA and GroupB to online.

The operating status becomes normal wherein Application Server of the active node (active node 1 of Node 1 and active node 2 of Node 2) is started, and Application Server of the spare node (spare node 1 of Node 2 and spare node 2 of Node 1) is in the standby state.



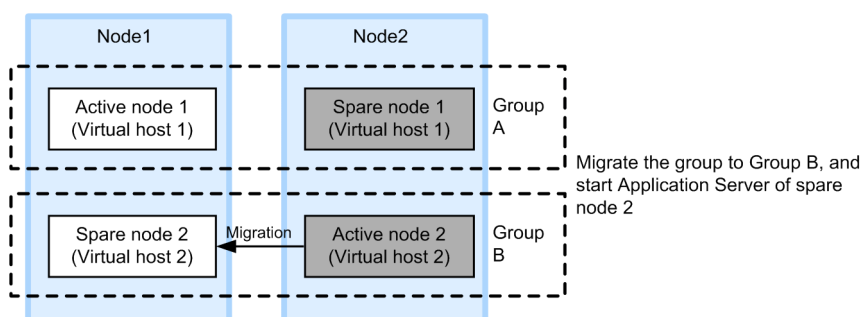
- For Application Server of the active node (active node 1 of Node 1 and active node 2 of Node 2), you use the Smart Composer functionality commands of the Management Server for setting up and starting the Web system based on the already set up Easy Setup definition file. Also, use the server management commands to import the J2EE applications and resource adapters and to start the imported J2EE applications and resource adapters.

For the settings of the J2EE applications, see 4.1.29 *Setting and starting the business application (when using CUI)* in the *uCosminexus Application Server System Setup and Operation Guide* and for the settings of the resource adapters, see the following subsections of the *uCosminexus Application Server System Setup and Operation Guide*:

- 4.1.26 *Setting up DB Connector (when using CUI)*
- 4.1.27 *Setting up resource adapters other than DB Connector (when using CUI)*
- 4.1.28 *Starting the resource adapters (when using CUI)*

For operations of the server management commands, see 3. *Basic Operations of Server Management Commands* in the *uCosminexus Application Server Application Setup Guide*.

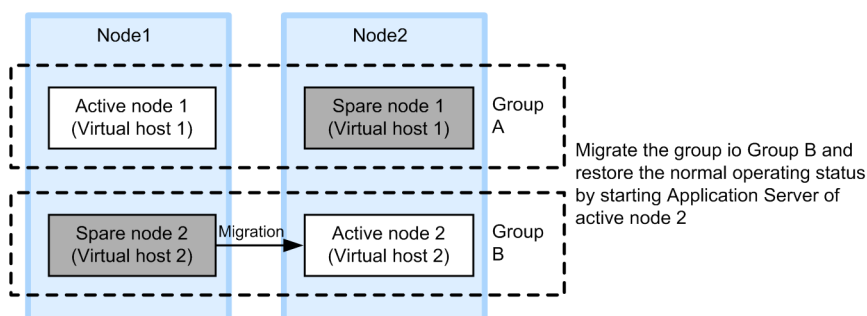
- Execute 'Migrate Group' for the resource group GroupB in the Cluster Administrator, switch the nodes, and start Application Server of the spare node 2 in Node 1.



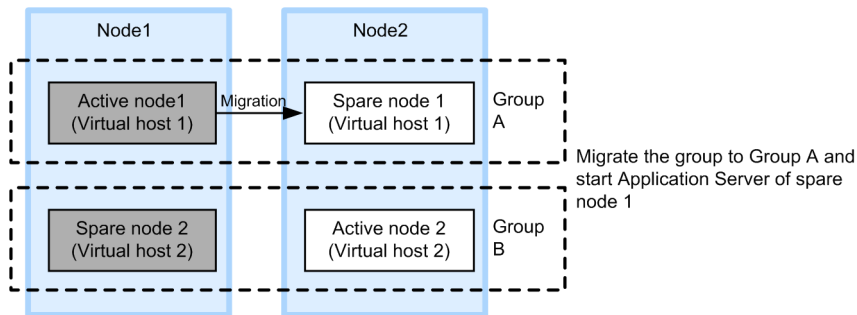
- For Application Server of the spare node 2 in Node 1, you use the Smart Composer functionality commands of the Management Server for setting up and starting the Web system based on an already set up Easy Setup definition file. Also, use the server management commands to import the J2EE applications and resource adapters, and to start the imported J2EE applications and resource adapters.

Import the same J2EE applications and resource adapters as the active node 2 of Node 2 in the spare node 2 of Node 1.

- Execute 'Migrate Group' for the resource group GroupB in the Cluster Administrator, switch the nodes, start Application Server of the active node 2 in Node 2, and return to the normal operating state.



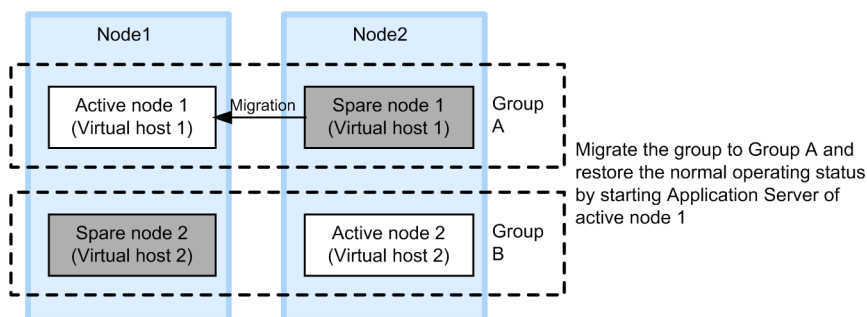
- Execute 'Migrate Group' for the resource group GroupA, switch the nodes, and start Application Server of the spare node 1 in Node 2.



8. For Application Server of the spare node 1 in Node 2, you use the Smart Composer functionality commands of the Management Server for setting up and starting the Web system based on the already set up Easy Setup definition file. Also, use the server management commands to import the J2EE applications and resource adapters, and to start the imported J2EE applications and resource adapters.

Import the same J2EE applications and resource adapters as the active node 1 of Node 1 in the spare node 1 of Node 2.

9. Execute 'Migrate Group' for the resource group GroupA, switch the nodes, start Application Server of the active node 1 in Node 1, and return to the normal operating state.



18.5 Settings for the mutual node switching system (In UNIX)

The mutual node switching system is one of the configurations in which the executing node and standby node of Application Server are arranged in the 1-to-1 ratio. Each Application Server operates as an executing node, and one Application Server is set as the standby node of other Application Server and the vice versa (*mutual standby configuration*). By deploying the same types of J2EE servers in each Application Server and by starting different J2EE servers in advance, each Application Server operates as an executing node and simultaneously functions as the standby node of the other Application Server. If a failure occurs in any one of the nodes, the system switches to the other node. This enables the economical operations using a few application server machines. You use the Smart Composer functionality to build this system. This subsection describes the settings for the mutual node switching systems.

Note that the operations of the mutual node switching system can only be used in AIX or in Linux.

For details on the functionality, see [18.2 Overview of the mutual node switching system](#). For the system configuration, see [3.11.4 Configuration in which executing node and standby node of Application Server are mutually standby](#) in the *uCosminexus Application Server System Design Guide*. Also, for HA monitor, see the manual *Reliable System Monitoring Functionality HA Monitor*.

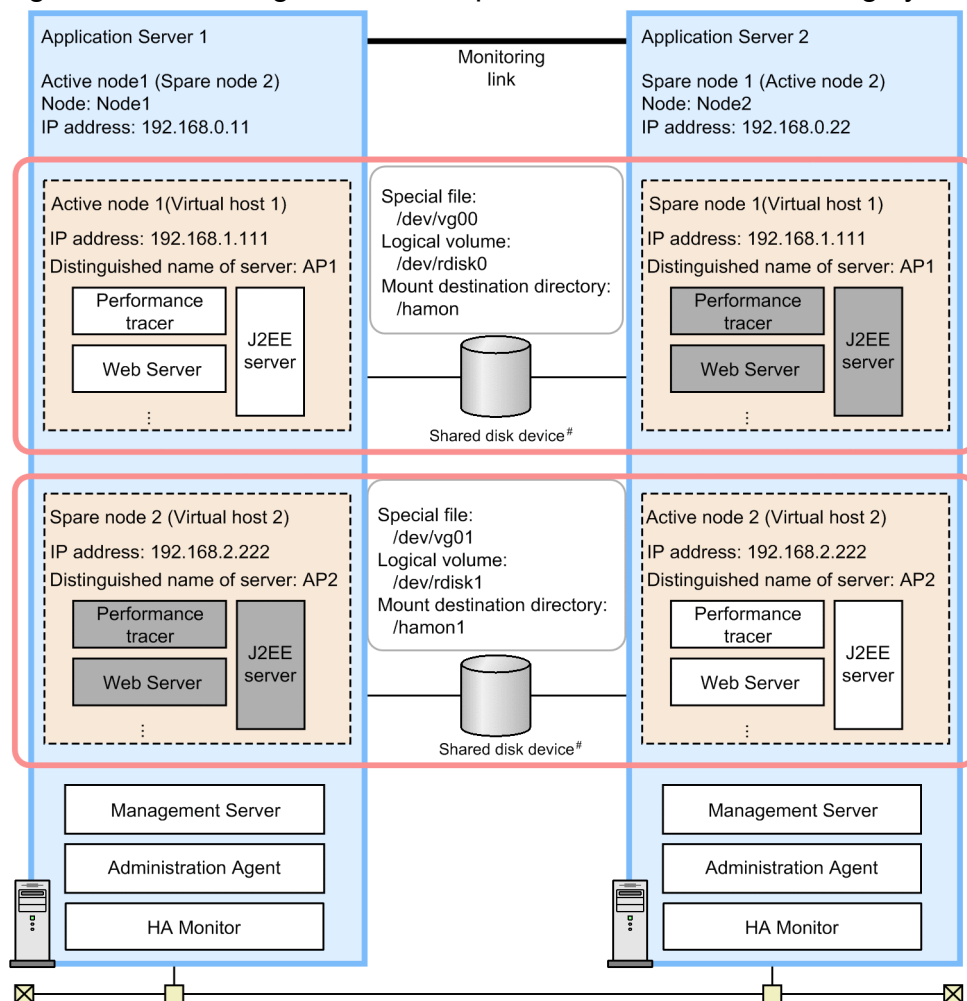
18.5.1 Procedure for setting up the mutual node switching system

This subsection describes the example system configuration and the procedure for specifying the system settings.

(1) System configuration example

The following figure shows the configuration example for the mutual node switching system. Note that the subsequent points describe the system configuration example using this configuration example.

Figure 18–7: Configuration example of mutual node switching system (In UNIX)



Legend:

 : It is the unit of node switching.

 : Started processes

 : Non-started processes

It is required when using a global transaction.
It is not required when using a local transaction.

In this example, two types of Application Servers (assumed to be Application Server 1 and Application Server 2) are used. The executing node (active node 1) and standby node (spare node 2) of Application Server 1 and the executing node (active node 2) and standby node (spare node 1) of Application Server 2 are deployed in the 1-to-1 ratio. Each Application Server deploys a Management Server having a separate management domain, and the Management Server is started on both the machines.

In the mutual node switching system, two virtual hosts are defined in one management domain, and are configured as the host of the active node Application Server and the host of the spare node Application Server respectively. In this example, the combination is as follows:

- Active node 1 (virtual host 1) of Node 1 and spare node 1 (virtual host 1) of Node 2
- Active node 2 (virtual host 2) of Node 2 and spare node 2 (virtual host 2) of Node 1

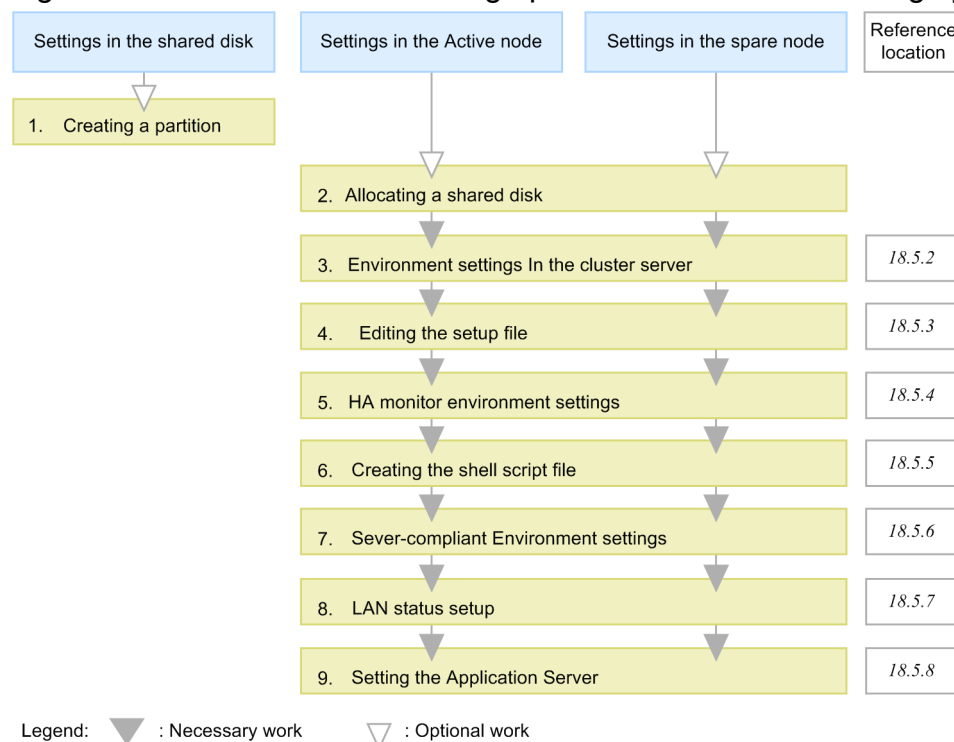
The virtual host controls the starting and stopping of Application Server using one Administration Agent, but the IP addresses allocated for operations are different, and therefore, the virtual hosts are defined as apparently different hosts.

Note that in order to deploy Application Server in a cluster configuration, specify the alias IP address and specify settings in such a way so that the running node inherits the alias IP address, and the client does not realize the nodes in the cluster. For the mutual node switching system, the alias IP address is the address that is dynamically allocated by the HA monitor. For the IP address to be used for Application Server operations, you use the alias IP address, and for sending requests from the Management Server to the Administration Agent, you use the IP address that does not move to other nodes due to node switching (stationary IP address).

(2) Procedure of system setup

For linking the system with the HA monitor, you must specify the settings of the Management Server and HA monitor files. The following figure shows the procedure for setting up a mutual node switching system:

Figure 18–8: Procedure for setting up the mutual node switching system (in UNIX)



Stages 1. to 9. of the figure are explained below:

1. When using the global transaction, create a partition in the shared disk, and build the file system.
To use the global transaction, create the location to store the transaction information.
2. When using the global transaction, allocate a shared disk to the system.
For allocating a shared disk to the system, specify the same directory to mount the shared disk in the active node 1 and spare node 1, active node 2 and spare node 2.
3. Set up the cluster server environment in the Management Server.
In the Easy Setup definition file of the Smart Composer functionality for the Management Server, specify the settings for using the HA monitor. For details, see [18.5.2 Environment settings in the cluster server](#).
4. Edit the setup file.
Specify the settings for various definition files of Administration Agent, Management Server, and Cosminexus HTTP Server that you cannot set up in the Easy Setup definition file. For details, see [18.5.3 Editing the setup file](#).
5. Set up the HA monitor environment.

Define the HA monitor environment in the `sysdef` file of the HA monitor. For details, see [18.5.4 HA monitor environment settings](#).

6. Create the shell script file.

Create the shell script file to monitor the Administration Agent and to start and stop the logical servers. For details, see [18.5.5 Creating the shell script file](#).

7. Set up a server-compliant environment.

Define the environment for the active node server and spare node server to be run with the node using the `servers` file of the HA monitor. For details, see [18.5.6 Server-compliant environment settings](#).

8. Set up the LAN status.

In the LAN status setup file of the HA monitor, specify settings such as the IP address of the LAN adapter to define the switching of LAN in the HA monitor. For details, see [18.5.7 LAN status setup](#).

9. Set up Application Server using Management Server and HA monitor commands.

Use Management Server and HA monitor commands to deploy Application Server in the cluster configuration, and to set up the J2EE applications and resource adapters. For details, see [18.5.8 Setting Application Server](#).

18.5.2 Environment settings in the cluster server

This subsection describes the points to be remembered when you specify the settings in the Management Server when the system is integrated with the HA monitor.



Reference note

The Management Server must be set up on the host in which the Management Server is used for the first time. For the Management Server setup, see [4.1.14 Setting the management functionality](#) in the *uCosminexus Application Server System Setup and Operation Guide*.

(1) Settings in the Easy Setup definition file

In the Easy Setup definition file of the Smart Composer functionality of the Management Server, define the Web system and host. Note the following when you define the host on which the logical server is to be set up and define the configuration of each logical server:

- **Defining the Web system**

Define the Web system that will be used in the Smart Composer functionality. Specify a different Web system for each virtual host.

- **Defining the host**

Define the host exists on the management domain. Specify the Alias IP address to be used in the server operations in the `<host-name>` tag for the virtual host of the active node and spare node respectively. In the example, the following is specified:

- In active node 1 of Node 1 and spare node 1 of Node 2: 192.168.1.111
- In active node 2 of Node 2 and spare node 2 of Node 1: 192.168.2.222

Also, in the `<agent-host>` tag, specify the stationary IP address of the host. In the example, the following is specified:

- In Node 1: 192.168.0.11

- In Node 2: 192.168.0.22

Note the following points when you specify the `<configuration>` tag of the logical J2EE server (j2ee-server) and logical Web server (web-server).

(a) Setting logical J2EE servers

• Settings for fixing the host

Specify any value in the following parameters in the `<configuration>` tag of the logical J2EE server (j2ee-server) and fix the management host and operation host:

- `mngagent.connector.host` parameter[#] (Host name or IP address of the Administration Agent)
- `vbroker.se.iioptp.host` parameter (Host name or IP address of the EJB container for each J2EE server)
- `webserver.connector.nio_http.bind_host` parameter (Host name or IP address used for the NIO HTTP server)

[#] When you specify the `mngagent.connector.port` parameter along with the `mngagent.connector.host` parameter, specify a different port number for each virtual host.

• Global transaction settings

When the global transaction is used, specify the following settings in the following parameters in the `<configuration>` tag of the logical J2EE server (j2ee-server):

- Specify 'true' in the `ejbserver.distributedtx.XATransaction.enabled` parameter (specify settings to enable or disable the light transaction functionality). By default, 'false' is set up.
- Specify the directory for the shared disk device in the `ejbserver.distributedtx.ots.status.directory1` parameter (location for saving the backup of the status file of the in-process OTS). By default, `otsstatus` is set up.

Example settings

/hamon

(b) Setting the logical Web server

The following settings must be specified in the `<configuration>` tag of the logical Web server (web-server):

• Settings for fixing the host

Specify the IP address or host name that receives the request in the `Listen` parameter and fix the host for the Web server.

18.5.3 Editing the setup file

You edit the setup file. This subsection describes the setup files for which you need to be careful when the system is linked with the HA monitor.

(1) Setting the administration agent

Of the items set up in the `adminagent.properties` (Administration Agent property file), this point describes the settings that you must note for integrating the system with the HA monitor. For `adminagent.properties`, see 8.2.1 *adminagent.properties (Administration Agent property file)* in the *uCosminexus Application Server Definition Reference Guide*.

- `adminagent.adapter.bind_host` key

Specify the stationary IP address (in the example, 192.168.0.11 or 192.168.0.22) in the `adminagent.adapter.bind_host` key. The management request is only received in the stationary IP address.

(2) Settings for the management command (mngsvrutil) in the Management Server

Under the `/opt/Cosminexus/manager/config` directory, provide the common definition file at client side (`mngsvrutilcl.properties`) for the management command (`mngsvrutil`) for the Management Server and specify the user ID and password of the management user of the Management Server. Also, set appropriate access privileges for the common definition file at the client.

You use this file to execute the `mngsvrutil` command in the script for monitoring the Administration Agent process, and in the script for starting and stopping the logical server.

For the client-side common definition file (`mngsvrutilcl.properties`) of the `mngsvrutil` command, see *8.2.16 mngsvrutilcl.properties (Client-side shared definition file of the mngsvrutil command)* in the *uCosminexus Application Server Definition Reference Guide*.

(3) Setting Management Server

Among the items to be set in `mserver.properties` (Management Server environment settings file), the settings that you must be careful about when integrating with the HA monitor are described as follows. For the `mserver.properties` file, see *8.2.6 mserver.properties (Management Server environment settings file)* in the *uCosminexus Application Server Definition Reference Guide*.

- **`com.cosminexus.mngsvr.logical_server_abnormal_stop.exit`**

Specify the settings for switching the nodes when the operating status of the logical servers is 'abnormal termination' (when the system has exceeded the automatic restart frequency, or when the system detects a failure at automatic restart frequency 0).

A setup example is as follows:

```
com.cosminexus.mngsvr.logical_server_abnormal_stop.exit=true
```



Important note

If automatic restart is specified for Management Server, node switching is not executed during this time. For executing the node switching during this time, do not set up automatic restart for Management Server.

Set the following items as and when required:

- **`webserver.connector.http.bind_host`**

Specify the real IP address in the `webserver.connector.http.bind_host` key.

- **`mngsvr.myhost.name`**

Specify the real IP address in the `mngsvr.myhost.name` key.

- **`com.cosminexus.mngsvr.management.host`**

Specify the virtual IP address in the `com.cosminexus.mngsvr.management.host` key.

18.5.4 HA monitor environment settings

According to the system being used, define the HA monitor environment in the definition file called as *sysdef*.

For details on the settings for the HA monitor environment, see the manual *Reliable System Monitoring Functionality HA Monitor*.

18.5.5 Creating the shell script file

Create the following shell script files to monitor the Administration Agent process and to start and stop the logical servers:

- Shell script file for monitoring the Administration Agent processes
- Shell script file for starting the logical server
- Shell script file for stopping the logical server

Note that when you have specified the settings for switching the nodes when the operating status of the logical servers is 'abnormal termination' (when the system has exceeded the automatic restart frequency or when the system detects a failure at automatic restart frequency 0), add the monitoring process of Management Server in the shell script file.

(1) Shell script file for monitoring the Administration Agent process

An example of the shell script file for monitoring the Administration Agent process (*manager_adminagent_monitor.sh*) is as follows:

```
#!/bin/sh

LOGDIR=/home/manager/hamon/log
AA=/opt/Cosminexus/manager/bin/adminagent
MNGDIR=/opt/Cosminexus/manager

logg ()
{
    echo `date '+[%Y/%m/%d %H:%M:%S] '`"[$$]: $1" \
    >> ${LOGDIR}/adminagent.log 2>&1
}

logg "### $0: started. ###"
while true
do
    $MNGDIR/bin/mngsvrutil -m 192.168.0.1-to-1 :28080 -t 192.168.1-to-1 1-to
-1 check adminAgent
    if [ $? -ne 0 ]
    then
        logg "### $0: stop. ###"
        exit 0
    fi
    sleep 10
done
```

In this shell script file, you check the operational status of the Administration Agent using the `check` command of `mngsvrutil`. Specify the stationary IP address of the host in the argument `host` name of the `-m` option and the alias IP address of the virtual host of the active node in the argument of the `-t` option of the `mngsvrutil` command.

Note that this shell script file describes the monitoring of the Administration Agent that runs on the active node 1 (spare node 2) of the Node 1. For using the shell script file in the spare node 1 (active node 2) of Node 2, replace the argument `host` name of the `-m` option with the stationary IP address (192.168.0.22) of Node 2 and the argument in the `-t` option with the alias IP address (192.168.2.222) of the virtual host of the active node.

(2) Shell script file for starting the logical server

An example of the shell script file for starting the logical server (`manager_apserver1_start.sh`) is as follows:

```
#!/bin/sh

MNGDIR=/opt/Cosminexus/manager
logg ()
{
    echo `date '+[%Y/%m/%d %H:%M:%S]`"[$$]: $1" \
    >> ${LOGDIR}/adminagent.log 2>&1
}
# start logical server
logg "### $0: starting logical servers. ###"
$MNGDIR/bin/mngsvrutil -m 192.168.0.1-to-1 :28080 -t performance-tracer -s start server
$MNGDIR/bin/mngsvrutil -m 192.168.0.1-to-1 :28080 -t J2EE-server-1 -s start server
$MNGDIR/bin/mngsvrutil -m 192.168.0.1-to-1 :28080 -t Web-server-1 -s start server
exit 0
```

In this shell script file, you start the logical server. Specify the stationary IP address of the host in the argument `host` name of the `-m` option and the name of the logical server to be started in the argument of the `-t` option of the `mngsvrutil` command.

Note that this shell script file describes the starting of the logical server in the active node 1 of Node 1.

(3) Shell script file for stopping the logical server

An example of the shell script file for stopping the logical server (`manager_apserver1_stop.sh`) is as follows:

```
#!/bin/sh

MNGDIR=/opt/Cosminexus/manager
logg ()
{
    echo `date '+[%Y/%m/%d %H:%M:%S]`"[$$]: $1" \
    >> ${LOGDIR}/adminagent.log 2>&1
}
# stop logical server
logg "### $0: stop logical servers. ###"
$MNGDIR/bin/mngsvrutil -m 192.168.0.1-to-1 :28080 -t Web-server-1 -s stop server graceful:300
$MNGDIR/bin/mngsvrutil -m 192.168.0.1-to-1 :28080 -t J2EE-server-1 -s stop server
```

```
$MNGDIR/bin/mngsvrutil -m 192.168.0.1-to-1 :28080 -t performance-tracer -s s  
top server
```

In this shell script file, you stop the logical server. Specify the stationary IP address of the host in the argument `host` name of the `-m` option and the name of the logical server to be stopped in the argument of the `-t` option of the `mngsvrutil` command.

Note that this shell script file illustrates the stopping of the logical server in the active node 1 of Node 1.

18.5.6 Server-compliant environment settings

In the server-compliant environment settings of the HA monitor, you define the environment for the executing server and the standby server to be operated on the node.

Define the server-compliant environment for mutual node switching system in the definition file called `servers`. The following table lists the contents to be set up in the server-compliant environment settings:

Table 18–4: Contents to be set up in the server-compliant environment settings (In the mutual node switching system)

Operand	Setting contents
<code>name</code>	Specify the shell script file for starting the logical server. Example specification: <code>/home/manager/hamon/bin/manager_apserver1_start.sh</code>
<code>alias</code>	Specify the server identification name. Specify the same name in the corresponding active node and spare node. Example specification: AP1 (for active node 1 and spare node 1), AP2 (for active node 2 and spare node 2)
<code>acttype</code>	Specify the server starting method. Specify 'monitor' to start the server using the HA monitor command.
<code>termcommand</code>	Specify the shell script file for stopping the logical server. Example specification: <code>/home/manager/hamon/bin/manager_apserver1_stop.sh</code>
<code>initial</code>	Specify the server startup status. <ul style="list-style-type: none">For the active node Specify 'online'.For the spare node Specify 'standby'.
<code>disk</code>	Specify the special file name for the character type of the shared disk device. Example specification: <code>/dev/vg00</code> (for active node 1 and spare node 1), <code>/dev/vg01</code> (for active node 2 and spare node 2)
<code>lan_updown</code>	Specify whether to use the LAN status setup file. Specify 'use' since the LAN status setup file is used here.
<code>fs_name</code>	Specify the absolute path name of the logical volume corresponding to the switched file system. Note that this setting is required only when <code>\$TPFS</code> is used in the UNIX file. Example specification: <code>/dev/rdisk0</code>
<code>fs_mount_dir</code>	Specify the absolute path name of the directory to mount the switched file system. Note that this setting is required only when <code>\$TPFS</code> is used in the UNIX file. Example of specification: <code>/hamon</code>

Operand	Setting contents
patrolcommand	Specify the shell script file for monitoring the Administration Agent process. Example specification: /home/manager/hamon/bin/manager_adminagent_monitor.sh
servexec_retry	Specify the retry frequency, when failure is detected. Specify '0' since the node is switched without retrying when failure occurs.
waitserv_exec	Specify whether to wait for the completion of execution of the start command when executing the start completion process of the logical server. Specify 'yes' since the server waits for the completion of execution.

The examples of the servers file are described below:

Example of servers file (in the active node 1 (spare node 2) of Node 1)

An example of the servers file for the active node 1 (spare node 2) of Node 1 is as follows:

```
server name      /home/manager/hamon/bin/manager_apserver1_start.sh,
alias           AP1,
acttype         monitor,
termcommand     /home/manager/hamon/bin/manager_apserver1_stop.sh,
initial         online,
disk            /dev/vg00,
lan_updown      use,
fs_name         /dev/rdisk0,
fs_mount_dir    /hamon,
patrolcommand   /home/manager/hamon/bin/manager_adminagent_monitor.sh,
servexec_retry  0,
waitserv_exec   yes;

server name      /home/manager/hamon/bin/manager_apserver2_start.sh,
alias           AP2,
acttype         monitor,
termcommand     /home/manager/hamon/bin/manager_apserver2_stop.sh,
initial         standby,
disk            /dev/vg01,
lan_updown      use,
fs_name         /dev/rdisk1,
fs_mount_dir    /hamon1,
patrolcommand   /home/manager/hamon/bin/manager_adminagent_monitor.sh,
servexec_retry  0,
waitserv_exec   yes;
```

Example of servers file (in spare node 1 (active node 2) of Node 2)

An example of the servers file for the spare node 1 (active node 2) of Node 2 is as follows:

```
server name      /home/manager/hamon/bin/manager_apserver1_start.sh,
alias           AP1,
acttype         monitor,
termcommand     /home/manager/hamon/bin/manager_apserver1_stop.sh,
initial         standby,
disk            /dev/vg00,
lan_updown      use,
fs_name         /dev/rdisk0,
fs_mount_dir    /hamon,
patrolcommand   /home/manager/hamon/bin/manager_adminagent_monitor.sh,
servexec_retry  0,
waitserv_exec   yes;
```

```
server name      /home/manager/hamon/bin/manager_apserver2_start.sh,
alias           AP2,
acttype         monitor,
termcommand     /home/manager/hamon/bin/manager_apserver2_stop.sh,
initial         online,
disk            /dev/vg01,
lan_updown      use,
fs_name         /dev/rdisk1,
fs_mount_dir    /hamon1,
patrolcommand   /home/manager/hamon/bin/manager_adminagent_monitor.sh,
servexec_retry  0,
waitserv_exec   yes;
```

In this example, a directory on the shared disk is specified as the location for storing the status file of the in-process transaction service. Specify the location for storing the status file of the in-process transaction service in the `ejbserver.distributedtx.ots.status.directory1` parameter of the `<configuration>` tag. The `<configuration>` tag exists in the logical J2EE server (`j2ee-server`) of the Easy Setup definition file that is used for the Smart Composer functionality of the Management Server.

18.5.7 LAN status setup

Define the settings for switching LAN in the HA monitor by specifying the IP address of the LAN adapter in the LAN status setup file of the HA monitor.

Set up the Alias IP address in the following files:

- `<Server-identification-name>.up` **file**
Use this file to connect LAN. Specify the Alias IP address to be added to the LAN adapter.
- `<Server-identification-name>.down` **file**
Use this file to disconnect LAN. Specify the Alias IP address to be deleted from the LAN adapter.

Specify the alias value of the server-compliant environment settings (servers file) in the *server-identification-name*.

For details on the LAN status setup, see the manual *Reliable System Monitoring Functionality HA Monitor*.

18.5.8 Setting Application Server

In the Application Server settings, you use the Management Server and HA monitor commands to deploy Application Server in the cluster configuration. Also, you set up the logical server and distribute the settings to import and start the J2EE applications and resource adapters in the J2EE server. To set Application Server:

1. Start the Administration Agent and Management Server in Node 1 and Node 2 respectively.
For details on how to start the Administration Agent and Management Server, see [2.6 Settings for starting and stopping the system](#).
2. Run the `monbegin` command of the HA monitor in Node 1 and Node 2 and start Application Server of the active node as the executing node and set Application Server of the spare node to standby.
 - **Example of command execution in Node 1**
Run `'# monbegin AP1'` to start the active node 1 of Node 1.

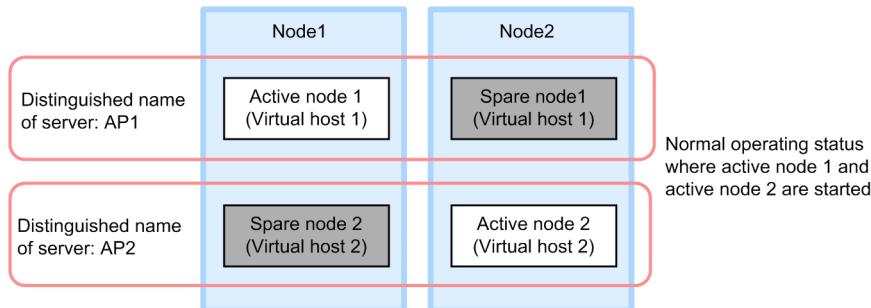
Run '# monbegin AP2' to set the spare node 2 of Node 1 to standby.

- **Example of command execution in Node 2**

Run '# monbegin AP1' to set the spare node 1 of Node 2 to standby.

Run '# monbegin AP2' to start the active node 2 of Node 2.

The operation status becomes normal wherein application server of active node (active node 1 of Node 1 and active node 2 of Node 2) is started and application server of spare node (spare node 1 of Node 2 and spare node 2 of Node 1) is in the standby state.



3. Use the Smart Composer functionality commands of the Management Server to set up, and start the Web system based on an already set up Easy Setup definition file for Application Server of the active node (active node 1 of Node 1 and active node 2 of Node 2). Also, use the server management commands to import the J2EE applications and resource adapters and to start the imported J2EE applications and resource adapters.

For the settings of the J2EE applications, see 4.1.29 *Setting and starting the business application (when using CUI)* in the *uCosminexus Application Server System Setup and Operation Guide* and for the settings of the resource adapters, see the following subsections of the *uCosminexus Application Server System Setup and Operation Guide*:

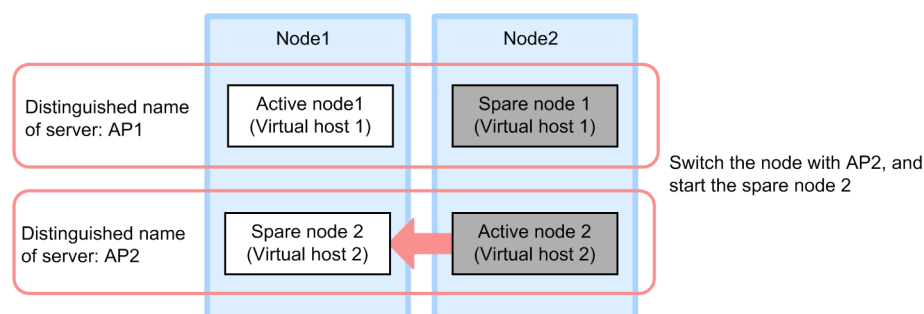
- 4.1.26 *Setting DB Connector (when using CUI)*
- 4.1.27 *Setting resource adapters other than DB Connector (when using CUI)*
- 4.1.28 *Starting the resource adapter (when using CUI)*

For operations of the server management commands, see 3. *Basic Operations of Server Management Commands* in the *uCosminexus Application Server Application Setup Guide*.

4. Run the monswap command of the HA monitor in Node 2 to switch the nodes.

- **Example of command execution in Node 2**

Run '# monswap AP2' to switch the node by server identification name 'AP2'. The spare node 2 of Node 1 is started and the active node 2 of Node 2 is in the standby state.



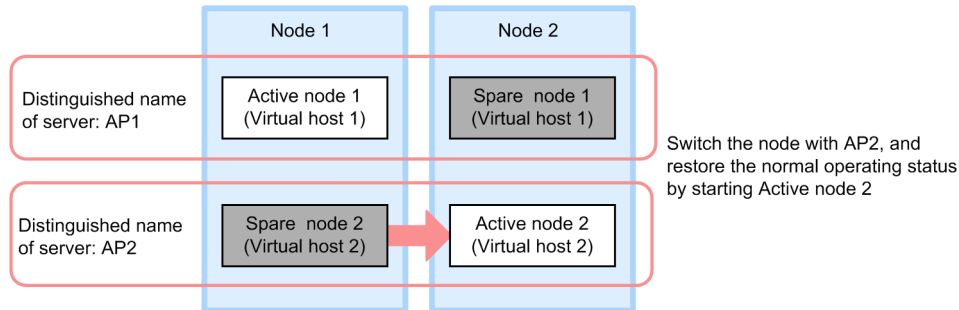
5. Use the Smart Composer functionality commands of the Management Server to set up and start the Web system based on an already set up Easy Setup definition file for Application Server of the spare node 2 in Node 1. Also, use the server management commands to import the J2EE applications and resource adapters and to start the imported J2EE applications and resource adapters.

Import the same J2EE applications and resource adapters as the active node 2 of Node 2 in the spare node 2 of Node 1.

6. Run the `monswap` command of the HA monitor in Node 1 to switch the nodes and return to the normal operating state.

- **Example of command execution in Node 1**

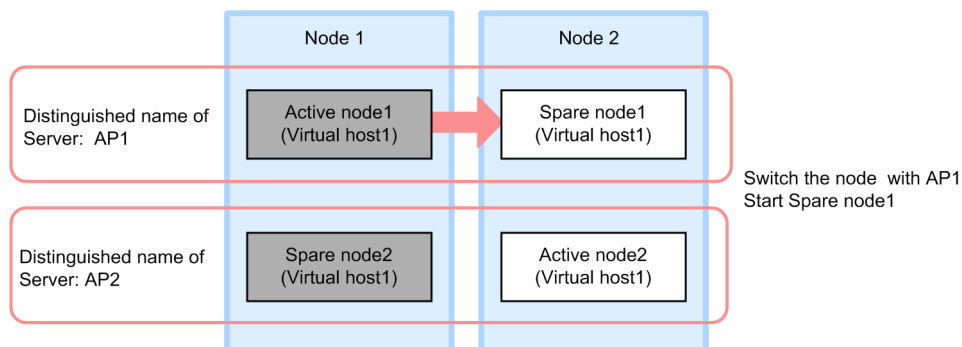
Run '# monswap AP2' to switch the node by the server identification name 'AP2'. The active node 2 of Node 2 is started and the spare node 2 of Node 1 is in the standby state.



7. Run the `monswap` command of the HA monitor in Node 1 to switch the nodes.

- **Example of command execution in Node 1**

Run '# monswap AP1' to switch the node with the server identification name 'AP1'. The spare node 1 of Node 2 is started and the active node 1 of Node 1 is in the standby state.



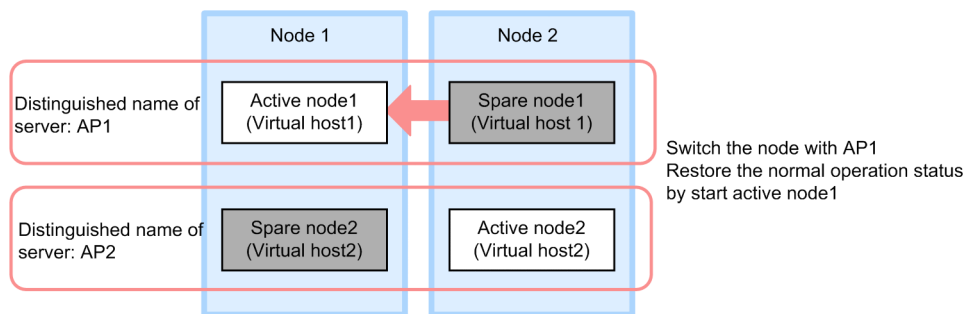
8. Use the Smart Composer functionality commands of the Management Server to set up and start the Web system based on an already set up Easy Setup definition file for Application Server of the spare node 1 in Node 2. Also, use the server management commands to import the J2EE applications and resource adapters and to start the imported J2EE applications and resource adapters.

Import the same J2EE applications and resource adapters as the active node 1 of Node 1 in the spare node 1 of Node 2.

9. Run the `monswap` command of the HA monitor in Node 2 to switch the nodes and return to the normal operating state.

- **Example of command execution in Node 2**

Run '# monswap AP1' to switch the node by the server identification name 'AP1'. The active node 1 of Node 1 is started and the spare node 1 of Node 2 is in the standby state.



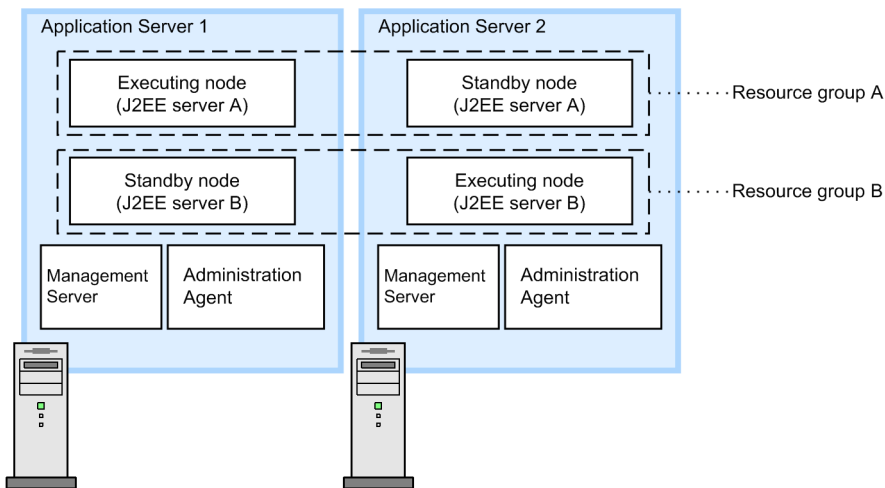
18.6 Starting and stopping the mutual node switching system (In Windows)

This section describes how to start and stop the system when you use the mutual node switching system.

The mutual node switching system is one of the configurations in which the executing node and standby node of Application Server are arranged in the 1-to-1 ratio. Each Application Server operates as an executing node, and each Application Server is set as the standby node of the other Application Server. For details on the executing node and the standby node, see *15.2 Operations that can be implemented by linking with cluster software*.

This section describes the configuration in which Application Servers are deployed as Application Server 1 and Application Server 2. The executing node of the J2EE server A and the standby node of J2EE server B are deployed in Application Server 1, and the standby node of J2EE server A and executing node of J2EE server B are deployed in Application Server 2. The following figure shows a configuration example. In this example, the executing node and the standby node of J2EE server A are set up in the resource group A, and the executing node and the standby node of J2EE server B are set up in the resource group B.

Figure 18–9: Configuration example of mutual standby system



Important note

If node switching occurs due to a failure in the Management Server or Administration Agent, the process of each logical server might remain in the failed host. If process remains in the host, the logical server cannot be started when node failback occurs, therefore, all the processes must be stopped before the node failback occurs. Moreover, you implement the operations for starting the business correctly in advance.

18.6.1 Starting the mutual node switching system

This subsection describes how to start the system when you use the mutual node switching system.

To start the mutual node switching system, you must first start the Management Server and Administration Agent. If the Management Server and Administration Agent are not running, start the Management Server and Administration Agent first.

The sequence for starting the mutual node switching system is as follows:

1. From the **Start** menu, choose **Control Panel - Performance and Maintenance - Management Tools**, and then select **Cluster Administrator**.
The Cluster Administrator starts.
2. In the console tree (left pane), select the node of Application Server 1, and then from the **File** menu choose **Start Cluster Service**.
The cluster service of the node for Application Server 1 starts.
3. In the console tree (left pane), select the node of Application Server 2, and then from the **File** menu choose **Start Cluster Service**.
The cluster service of the node for application server 2 starts.
4. In the console tree (left pane), select resource group A and then from the **File** menu, choose **Set to Online**.
The executing node (Application Server A) goes online.
5. In the console tree (left pane), select resource group B, and then from the **File** menu choose **Set to Online**.
The executing node (Application Server B) goes online.

18.6.2 Stopping the mutual node switching system

This subsection describes how to stop the mutual node switching system.

1. From the **Start** menu, choose **Control Panel - Performance and Maintenance - Management Tools**, and then select **Cluster Administrator**.
The Cluster Administrator starts.
2. In the console tree (left pane), select resource group B, and then from the **File** menu choose **Set to Offline**.
The resource group B goes offline.
3. In the console tree (left pane), select resource group A, and then from the **File** menu choose **Set to Offline**.
The resource group A goes offline.
4. In the console tree (left pane), select the node of Application Server 2, and then from the **File** menu choose **Stop Cluster Service**.
The cluster service of the node of Application Server 2 will stop.
5. In the console tree (left pane), select the node of Application Server 1, and then from the **File** menu choose **Stop Cluster Service**.
The cluster service of the node of Application Server 1 will stop.

Manually stop the Management Server and Administration Agent at the end. For stopping the Management Server and the Administration Agent, see the following manuals:

- In a system executing J2EE applications
See *4.1.3 Procedure for stopping a system* and *4.1.4 Methods of stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.
- In a system executing batch applications
See *6.1.3 Procedure for stopping a system* and *6.1.4 Methods of stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.

18.6.3 Starting and stopping a system for planned node switching in the mutual node switching system

This subsection describes how to switch the executing node and the standby node of the mutual node switching system in the cases other than when the trouble occurs. When the node is switched, Application Server must be in the standby state on the standby node host.

This subsection describes node switching for J2EE server A. You can switch the nodes using the same procedure for J2EE server B.

1. From the **Start** menu, choose **Control Panel - Performance and Maintenance - Management Tools**, and then select **Cluster Administrator**.

The Cluster Administrator starts.

2. In the console tree (left pane), select resource group A, and then from the **File** menu choose **Migrate Group**.

The node will switch.

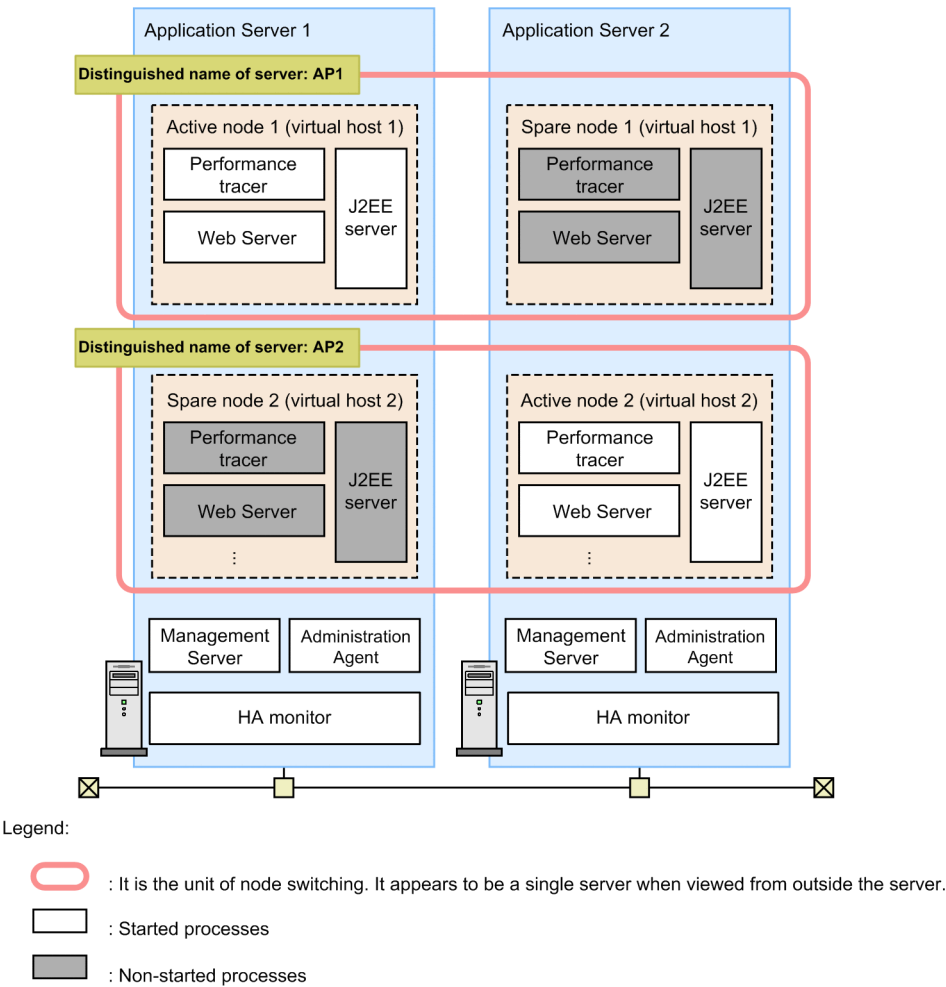
18.7 Starting and stopping the mutual node switching system (In UNIX)

This section describes how to start and stop the system when you use the HA monitor-based mutual node switching system. This section also describes the starting and stopping procedure when maintenance is performed for the J2EE server and batch server after the start of operations. Note that the HA monitor can only be used in AIX or in Linux.

To use the mutual node switching system, you must first specify the necessary environment settings such as preparing the two types of hosts as the active node and the spare node, and registering the script for monitoring, starting, and stopping the Administration Server to be monitored by the HA monitor. For details on the setup methods, see the description related to the HA monitor settings in *18.5.4 HA monitor environment settings*.

The following figure shows a configuration example of the mutual node switching system. This subsection describes how to start and stop the mutual node switching system based on this system configuration.

Figure 18–10: Configuration example of the mutual node switching system



From the servers file definitions (server environment settings in the HA monitor) in each Application Server in the figure, the following table lists the definition items related to the starting and stopping of the system.

Table 18–5: Definitions in the servers file for Application Server 1

Operand	Description	Specified value	
		Active node 1 (Virtual host 1)	Spare node 2 (Virtual host 2)
alias	Server identification name	AP1	AP2
initial	Server starting method in the HA monitor	online	standby

Table 18–6: Definitions in the servers file for Application Server 2

Operand	Description	Specified value	
		Spare node 1 (Virtual host 1)	Active node 2 (Virtual host 2)
alias	Server identification name	AP1	AP2
initial	Server starting method in the HA monitor	standby	online

For details on the servers file definitions, see [18.5.6 Server-compliant environment settings](#).

The following commands are provided by the HA monitor. For details, see the manual *Reliable System Monitoring Functionality HA Monitor*.

- monbegin (Starting a server that does not have an interface with the HA monitor)
- monend (Communication to stop the executing server that does not have an interface with the HA monitor)
- monsbystp (Stopping a standby server)
- monswap (Planned node switching)

Important note

If node switching occurs due to a failure in the Management Server or Administration Agent, the logical server processes might remain in the failed host. If processes remain in the host, the logical server cannot be started when node failback occurs, therefore, all the processes must be stopped before node failback occurs. In addition, implement operations for starting the business correctly in advance.

18.7.1 Starting the mutual node switching system

This subsection describes the points you must keep in mind when you start the system using the mutual node switching system and the procedure for starting the system.

(1) Points to remember when starting the system

Keep the following points in mind when you start a system using the mutual node switching system:

- The HA monitor deployed on the host of Application Server 1 and Application Server 2 starts simultaneously as the OS starts.

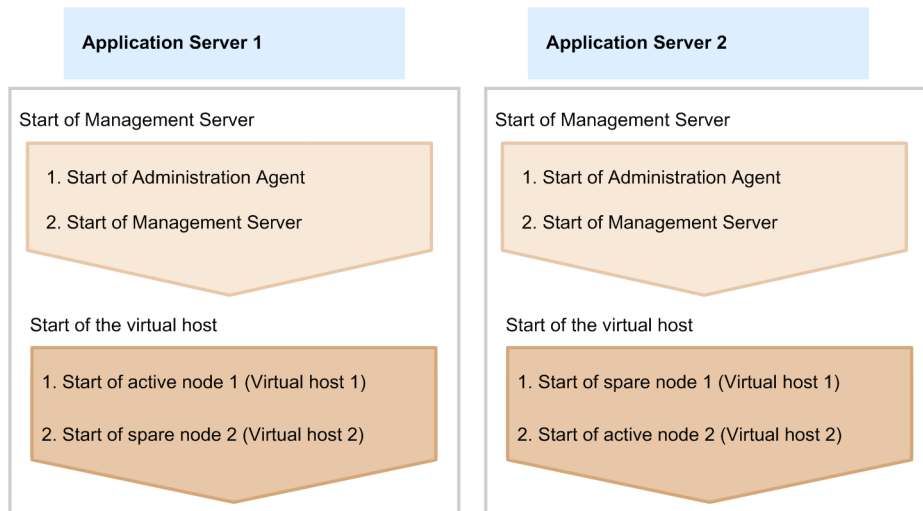
- Before starting the virtual host on the hosts of Application Server 1 and Application Server 2, first start the Management Server.

(2) Procedure to start the system

The following figure shows the flow for starting the mutual node switching system.

Start the system as per the flow shown in the following figure on the hosts of Application Server 1 and Application Server 2:

Figure 18–11: Flow for starting the mutual node switching system



Reference note

The virtual host can be started in any order on the hosts of Application Server 1 and Application Server 2. You may start either from active node virtual host or spare node virtual host. Note that this manual describes the procedure of starting in the order of virtual host 1 and virtual host 2.

For steps 1. and 2. in the figure for starting the Management Server, see the following manuals:

- In a system executing J2EE applications
See *4.1.1 Procedure for starting a system* in the *uCosminexus Application Server Management Portal User Guide*.
- In a system executing batch applications
See *6.1.1 Procedure for starting a system* in the *uCosminexus Application Server Management Portal User Guide*.

Note that if auto-start settings are specified, the Administration Agent and Management Server also start when the host starts, therefore, you can omit stages 1 and 2. For details on how to start the system, see [2.6 Settings for starting and stopping the system](#).

The procedure for starting the virtual host shown in the figure is described below. To start the virtual host in the hosts of Application Server 1 and Application Server 2:

- **To start the virtual host in application server 1**
 1. **Starting the active node 1 (virtual host 1)**
Run the `monbegin` command to start the active node 1 (virtual host 1) as the executing node.

```
# monbegin AP1
```

As a result, the process defined in the script file that starts the virtual host of the active node is executed and the virtual host of the active node is started as the executing node.

2. Starting the spare node 2 (virtual host 2)

Run the `monbegin` command to start the spare node 2 (virtual host 2) as the standby node.

```
# monbegin AP2
```

As a result, the virtual host of the spare node becomes the standby node and prepares for the failure in the executing node.

• To start the virtual host in Application Server 2

1. Starting the spare node 1 (virtual host 1)

Run the `monbegin` command to start the spare node 1 (virtual host 1) as the standby node.

```
# monbegin AP1
```

As a result, the virtual host of the spare node becomes the standby node and prepares for the failure in the executing node.

2. Starting the active node 2 (virtual host 2)

Run the `monbegin` command to start the active node 2 (virtual host 2) as the executing node.

```
# monbegin AP2
```

As a result, the process defined in the script file that starts the virtual host of the active node is executed and the virtual host of the active node is started as the executing node.

18.7.2 Stopping the mutual node switching system

This subsection describes the procedure for stopping the mutual node switching system.

The stopping procedure will be described for the following two cases:

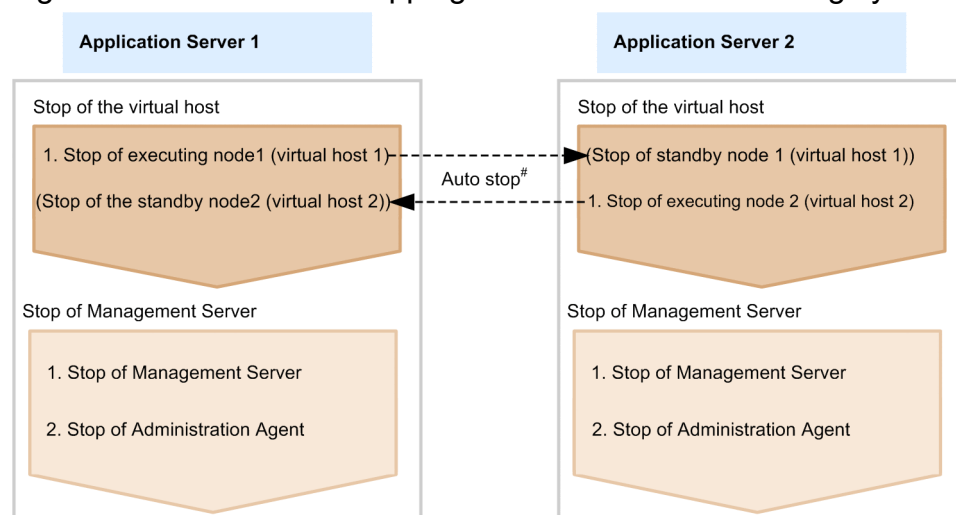
- When stopping the virtual hosts of both the executing node and the standby node
- When stopping only the virtual host of the standby node

(1) When stopping the virtual hosts of both the executing node and the standby node

The procedure for stopping the virtual hosts of both the executing node and standby node will be described below.

The following figure shows the flow for stopping the mutual node switching system when the virtual hosts of both the executing node and standby node are stopped.

Figure 18–12: Flow for stopping the mutual node switching system



Legend:

---▶ : Flow of the instructions for auto stop by HA monitor

After stopping of the executing node, stop instructions are sent to standby node by the HA monitor and standby node automatically stops.

Reference note

The virtual host 1 and virtual host 2 can be stopped in any order. This manual describes the procedure of stopping in the order of virtual host 1 and virtual host 2.

Stop all the virtual hosts, and then stop the Management Server for the hosts of Application Server 1 and Application Server 2.

For steps 1 and 2 in the figure for stopping Management Server, see the following manuals:

- In a system executing J2EE applications
See *4.1.3 Procedure for stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.
- In a system executing batch applications
See *6.1.3 Procedure for stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.

Note that if auto-stop settings are specified, the Management Server and Administration Agent also stop while the host stops, and therefore, you can omit the stages 1 and 2. For details on how to stop the systems, see [2.6 Settings for starting and stopping the system](#).

The procedure for stopping the virtual host shown in the figure is described below. To stop the virtual host in the hosts of Application Server 1 and Application Server 2:

- To stop the virtual host in Application Server 1
 1. Run the `monend` command in Application Server 1 to stop the virtual host 1 of the executing node.

```
# monend AP1
```

As a result, the executing node 1 (virtual host 1) of Application Server 1 will stop. Also, stop instruction is automatically sent to the standby node by the HA monitor and the standby node 1 (virtual host 1) of Application Server 2 also stops.

- To stop the virtual host in Application Server 2

1. Run the `monend` command in Application Server 2 to stop the virtual host 2 of the executing node.

```
# monend AP2
```

As a result, the executing node 2 (virtual host 2) of Application Server 2 will stop. Also, stop instruction is automatically sent to the standby node by the HA monitor and the standby node 2 (virtual host 2) of Application Server 1 also stops.

(2) When stopping only the virtual host of the standby node

To end the standby status in the virtual host of the standby node without stopping the virtual host of the executing node:

- To end the standby status in the standby node 2 (virtual host 2) of Application Server 1

Run the following command in the host of Application Server 1:

```
# monsbystp AP2
```

- To end the standby status in the standby node 1 (virtual host 1) of Application Server 2

Run the following command in the host of Application Server 2:

```
# monsbystp AP1
```

18.7.3 Starting and stopping a system for planned node switching in the mutual node switching system

This subsection describes the planned node switching of the executing node and standby node in the cases other than when the trouble occurs in the mutual node switching system. When the node is switched, the virtual host of the standby node must be at standby.

The following points describe how to switch the executing node and standby node of the virtual host 1 and virtual host 2 in a planned manner. Note that the active node host is presumed to be running as the executing node.

Reference note

For details on starting and stopping the mutual node switching system for maintenance, see [18.7.4 Starting and stopping the mutual node switching system for maintenance](#).

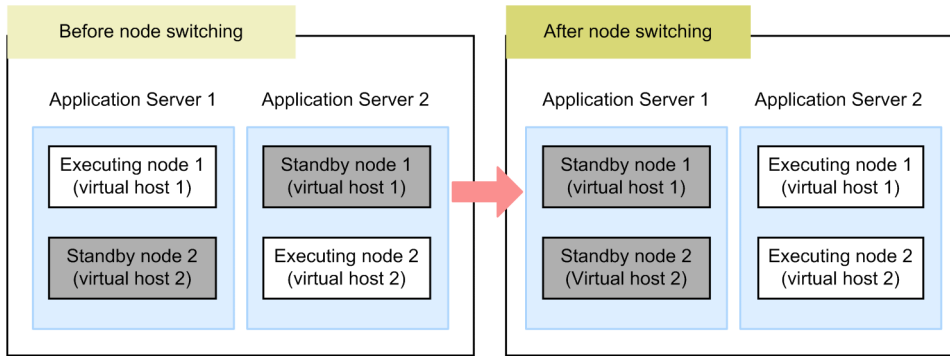
(1) Planned node switching for virtual host 1

To switch the nodes for the virtual host 1 in a planned manner:

1. Run the `monswap` command in Application Server 1 to switch the nodes.

```
# monswap AP1
```

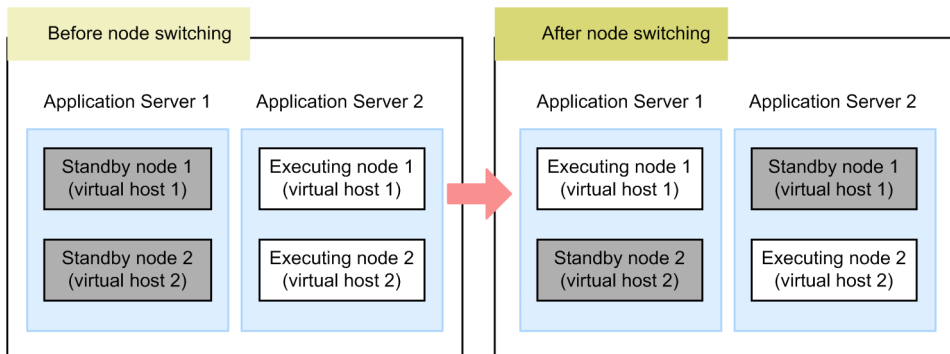
As a result, as shown in the figure, the executing node 1 (virtual host 1) of Application Server 1 switches to the virtual host of the standby node and the standby node 1 (virtual host 1) of Application Server 2 switches to the virtual host of the executing node.



2. To return to the normal executing node and standby node operations, run the `monswap` command in Application Server 2 and switch the nodes.

```
# monswap AP1
```

As a result, as shown in the figure, the executing node 1 (virtual host 1) of Application Server 2 switches to the virtual host of the standby node and the standby node 1 (virtual host 1) of Application Server 1 switches to the virtual host of the executing node. Thus, the executing node and standby node operations return to normal.



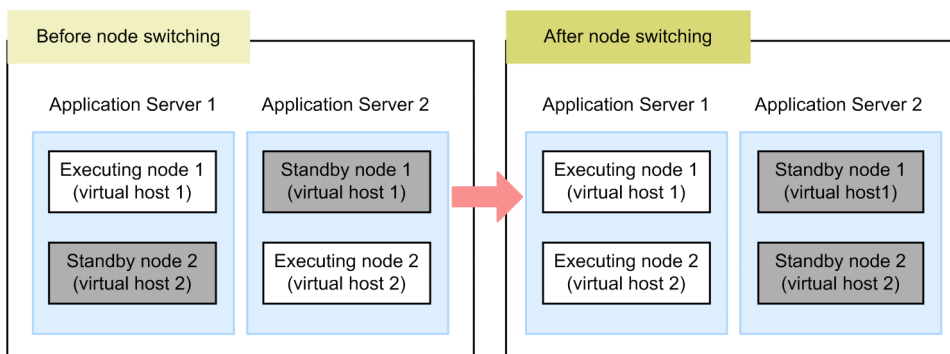
(2) Planned node switching for virtual host 2

Procedures to switch the nodes for the virtual host 2 in a planned manner:

1. Run the `monswap` command in Application Server 2 to switch the nodes.

```
# monswap AP2
```

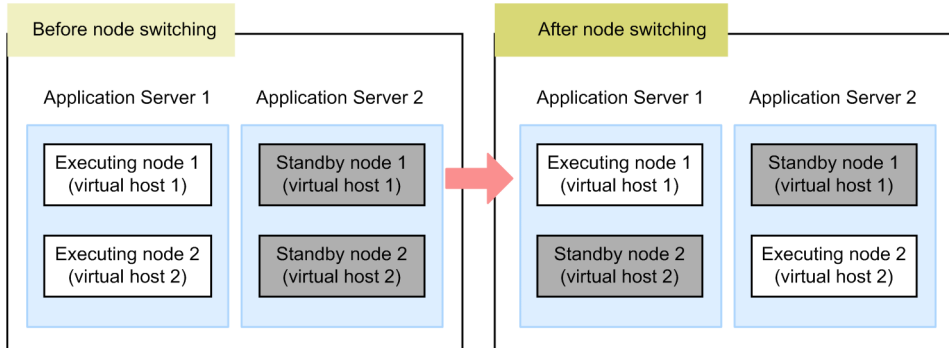
As a result, as shown in the figure, the executing node 2 (virtual host 2) of Application Server 2 switches to the virtual host of the standby node and the standby node 2 (virtual host 2) of Application Server 1 switches to the virtual host of the executing node.



2. To return to the normal executing node and standby node operations, run the `monswap` command in Application Server 1 and switch the nodes.

```
# monswap AP2
```

As a result, as shown in the figure, the executing node 2 (virtual host 2) of Application Server 1 switches to the virtual host of the standby node and the standby node 2 (virtual host 2) of Application Server 2 switches to the virtual host of the executing node. Therefore, the executing node and standby node operations return to normal.



18.7.4 Starting and stopping the mutual node switching system for maintenance

This subsection describes how to start and stop the mutual node switching system for maintenance.

The stopping procedure will be described for the following three cases:

- When maintenance that does not require restart
- When maintenance that requires restart (method of stopping both the nodes at the same time)
- When maintenance that requires restart (method of not stopping both the nodes at the same time)

Note that this is the procedure for the case where both the virtual hosts of the executing nodes are already running.

(1) When maintenance that does not require restart

To start and stop the system for maintenance that does not require system restart:

1. In the virtual host of the executing node (virtual host of the active node), stop the J2EE applications and resource adapters that are running.

When the batch server, confirm that the batch applications are not running and then stop the resource adapter.

For stopping the J2EE applications and resource adapters, see the following manuals:

- In a system executing J2EE applications
See *4.1.3 Procedure for stopping a system* and *4.1.4 Methods of stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.
- In a system executing batch applications
See *6.1.3 Procedure for stopping a system* and *6.1.4 Methods of stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.

2. Perform maintenance using server management commands.

For operations of the server management commands, see 3. *Basic Operations of Server Management Commands* in the *uCosminexus Application Server Application Setup Guide*.

3. Start the J2EE applications and resource adapters that were stopped in step 1.

For the J2EE server, the J2EE applications and resource adapters will start. The resource adapters will start even for the batch server.

For starting the J2EE applications and resource adapters, see the following manuals:

- In a system executing J2EE applications
See 4.1.1 *Procedure for starting a system* and 4.1.2 *Methods of starting a system* in the *uCosminexus Application Server Management Portal User Guide*.
- In a system executing batch applications
See 6.1.1 *Procedure for starting a system* and 6.1.2 *Methods of starting a system* in the *uCosminexus Application Server Management Portal User Guide*.

4. Check the operations of the J2EE applications and resource adapters as needed.

5. For the virtual host of the executing node (virtual host of the active node), run the `monswap` command to switch the nodes.

```
# monswap server-identification-name
```

In the underlined part, specify the server identification name of the virtual host of the executing node specified in the operand 'alias' of the servers file. The servers file exists in application server where the virtual host of the executing node is deployed.

The virtual host of the executing node (virtual host of the active node) switches to the virtual host of the standby node and the virtual host that was the standby node (virtual host of the spare node) becomes the virtual host of the executing node.

6. Execute steps 1 to 4 in the virtual host of the executing node (virtual host of the spare node) after the nodes are switched.

7. As needed, run the `monswap` command again in the executing node host (virtual host of the spare node) after the nodes are switched.

```
# monswap server-identification-name
```

In the underlined part, specify the same server identification name as that specified in the `monswap` command argument in step 5.

Execute this command to return the virtual host of the active node to the executing node.

Thus, the procedure for starting and stopping the system for maintenance is complete.

(2) When maintenance that requires restart (method of stopping both the nodes at the same time)

To start and stop the system for maintenance that requires system restart:

1. Run the `monend` command for the virtual host of the executing node to stop the system.

```
# monend server-identification-name
```

In the underlined part, specify the server identification name of the virtual host of the executing node specified in the operand 'alias' of the servers file in application server where the virtual host of the executing node is deployed.

The virtual host of the executing node will stop. Also, stop instruction is automatically sent to the standby node by the HA monitor and the virtual host of the standby node also stops.

2. Execute maintenance operations such as changing the definition files for the virtual hosts of both the active node and spare node.
3. Run the `monbegin` command for the virtual host of the active node.

```
# monbegin server-identification-name
```

In the underlined part, specify the same server identification name as that specified in the `monend` command argument in step 1.

As a result, the virtual host of the active node will start as the executing node.

4. Run the `monbegin` command for the virtual host of the spare node.

```
# monbegin server-identification-name
```

In the underlined part, specify the same server identification name as that specified in the `monend` command argument in step 1.

As a result, the virtual host of the spare node will start as the standby node.

5. As needed, check the operations related to changes in definitions in the virtual host of the executing node.



Reference note

For details on the servers file definition (server environment settings in the HA monitor), see [18.5.6 Server-compliant environment settings](#).

(3) When maintenance that requires restart (method of not stopping both the nodes at the same time)

To start and stop the system for maintenance that requires system restart:

1. Execute maintenance operations such as changing the definition files for the virtual host of the standby node.
2. Run the `monswap` command for the virtual host of the executing node to switch the nodes.

```
# monswap server-identification-name
```

In the underlined part, specify the server identification name of the virtual host of the executing node specified in the operand 'alias' of the servers file in Application Server where the virtual host of the executing node is deployed.

The virtual host of the executing node will be stopped, and then the standby node will start. As a result, the virtual host for which maintenance is complete becomes the virtual host of the executing node.

3. As needed, check the operations related to the changes in definitions for the executing node host (virtual host of the spare node) after the nodes are switched.
4. Execute maintenance operations such as editing the definition file in the standby node host (virtual host of the active node) after the nodes are switched.
5. As needed, run the `monswap` command again for the virtual host of the executing node (spare node host) after the nodes are switched.

```
# monswap server-identification-name
```

In the underlined part, specify the same server identification name as that specified in the `monswap` command argument in step 2.

Execute this command to return the virtual host of the active node to the executing node.

6. As needed, check the operations related to changes in definitions in the virtual host of the executing node (virtual host of the active node) after the nodes are switched.

Thus, the procedure for starting and stopping the system for maintenance is complete.

19

N-to-1 Recovery System (Linked with Cluster Software)

This chapter describes the systems that link with cluster software and operate with the N-to-1 recovery system.

19.1 Organization of this chapter

The following table describes the organization of this chapter:

Table 19–1: Organization of this chapter (N-to-1 recovery system (Linking with Cluster Software))

Category	Title	Reference
Description	Overview of the N-to-1 recovery system	19.2
	System configuration and operations of the N-to-1 recovery system	19.3
Settings	Settings for the N-to-1 recovery system (In Windows)	19.4
	Settings for the N-to-1 recovery system (In UNIX)	19.5
Operations	Starting and stopping the N-to-1 recovery system (In Windows)	19.6
	Starting and stopping the N-to-1 recovery system (In UNIX)	19.7

Note: The function-specific explanation is not available for "Implementation" and "Precautions".

19.2 Overview of the N-to-1 recovery system

The *N-to-1 recovery system* is a system in which you deploy one recovery server as the standby node for multiple (N number of) executing node Application Servers in a cluster configuration. However, for batch application execution environment, you cannot use the N-to-1 recovery system.

19.3 System configuration and operations of the N-to-1 recovery system

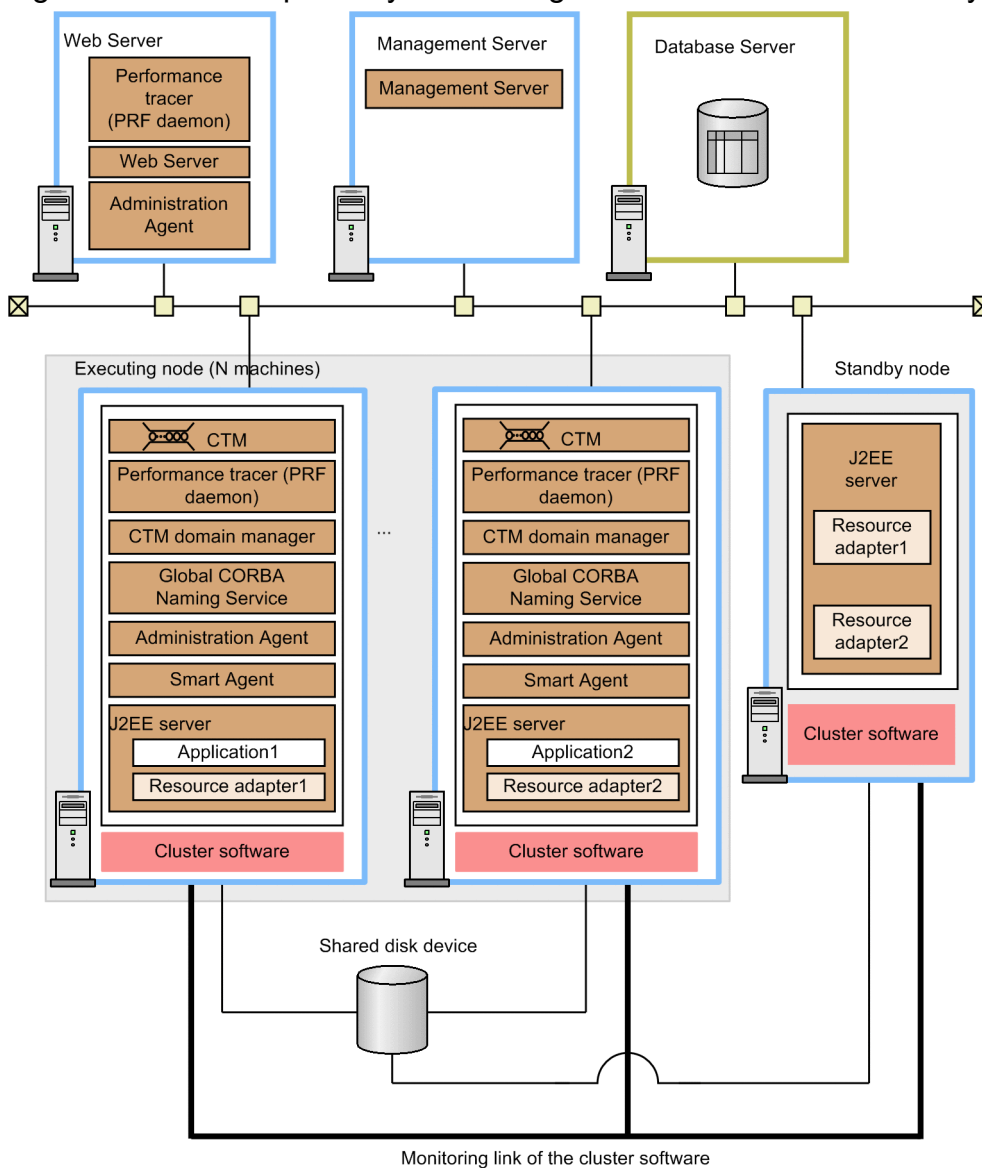
This section describes an example of system configuration and the flow of the recovery process.

19.3.1 Example of system configuration of the N-to-1 recovery system

The following figure shows an example of system configuration of the N-to-1 recovery system linked with cluster software. In a cluster configuration, if a failure occurs in the running Application Server, the standby node recovery server starts up and terminates the transaction of the executing node in which the failure occurred.

Import all resource adapters that are imported to the executing node, to the J2EE server of the standby node recovery server. Note that you need not deploy applications in the recovery server as the recovery server is only meant to perform recovery for the executing node Application Server in which the failure has occurred.

Figure 19–1: Example of system configuration of the N-to-1 recovery system



You can perform the following operations in the N-to-1 recovery system:

Using a shared disk device

You require a shared disk device to inherit the transaction information, such as the status of OTS. The transaction information is inherited between each executing node Application Server and standby node recovery server.

Applying the load balancer

Though not mentioned in this example of system configuration, you can apply a load balancer by preparing multiple Web servers with the same configuration. You can thereby improve the reliability and operating efficiency of Web servers.

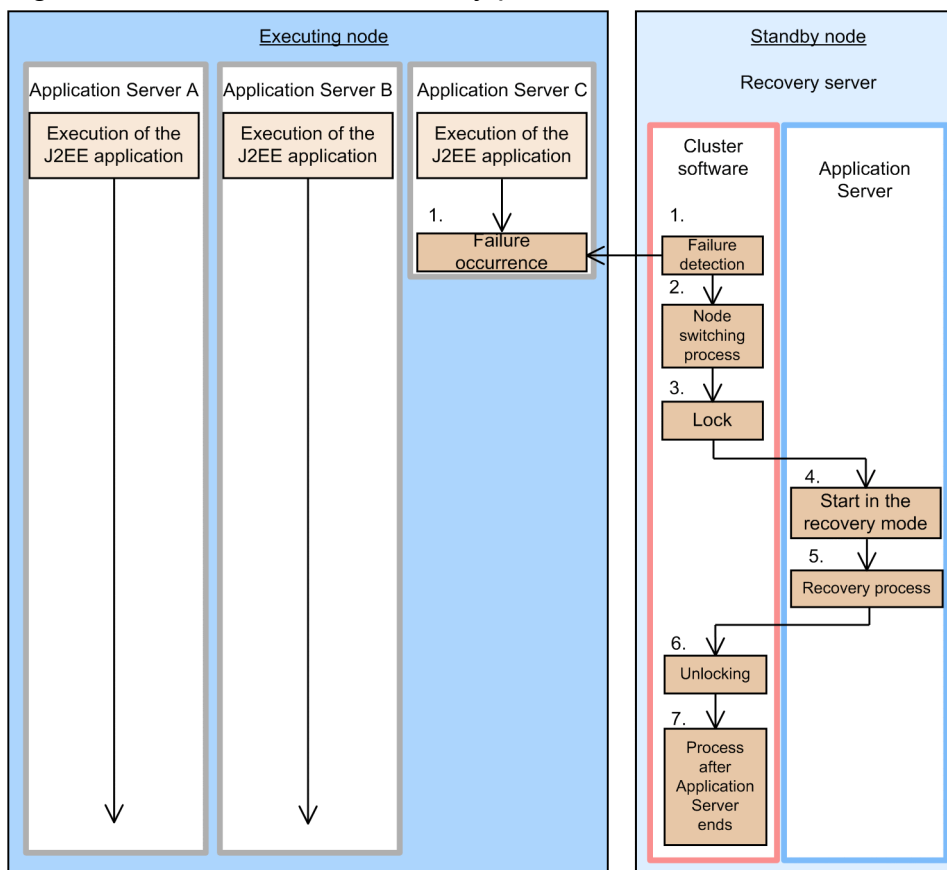
For details on the system configuration of an N-to-1 recovery system, see *3.11.5 Configuration using server exclusive for recovery (N-to-1 recovery system)* in the *uCosminexus Application Server System Design Guide*.

19.3.2 Flow of the recovery process

Perform the operations with the executing node arranged in a cluster configuration, and a single standby node that acts as the recovery server. If a failure occurs even in one of the multiple executing node Application Servers, the cluster software detects the failure and switches to the standby node recovery server. The standby node terminates the transaction of the executing node Application Server in which the failure occurred.

The following figure shows the flow of the recovery process:

Figure 19–2: Flow of the recovery process



1. The cluster software of the recovery server detects the failure.

If a failure occurs in the executing node Application Server C and if the server terminates, the cluster software of the standby node recovery server detects the termination of Application Server C.

2. Perform node switching with the cluster software of the recovery server.
Switch to the standby node recovery server. At this point, switch and mount the shared disk device, and set up the IP address of the virtual host.
3. Lock the recovery server.
Execute the recovery process sequentially in the recovery server. Before you execute the recovery process, lock the recovery server.
4. Execute the recovery command in the cluster software of the recovery server, and then start Application Server of the recovery server in the recovery mode.
5. In the recovery server, execute the recovery process of the transaction that was running in Application Server C in which the failure occurred.
Only the recovery process of the transaction is executed in the recovery server. Application Server of the recovery server terminates after the completion of the recovery process.
6. Release the lock of the recovery server.
7. Execute the process following the termination of Application Server in the cluster software of the recovery server.
Switch and unmount the shared disk device, and delete the IP address of the virtual host.

If a failure occurs in another executing node Application Server, execute the recovery process. Note that executing node Application Servers A and B are not affected by the failure in Application Server C and they continue with the application processing.

Notes

- If a failure occurs in multiple executing nodes and Application Servers terminate, the recovery server (standby node) executes the sequential recovery process exclusively.
- During the recovery process in the recovery server, the service port of the recovery server is blocked, and therefore, no processing is received.
- When the recovery processing does not terminate in the recovery server, timeout monitoring is executed.
- If a timeout occurs due to a failure in the database, execute the recovery process manually. For details, see *2.5.8 If a problem occurs in N-to-1 recovery systems* in the *uCosminexus Application Server Maintenance and Migration Guide*.
- Double failures such as termination of recovery servers are not supported.

19.3.3 Inheriting information during node switching

This subsection describes the information that the standby node inherits and the information that the standby node does not inherit during node switching.

- **Information that the standby node inherits**

- Inheriting the OTS transaction information (in a global transaction)
After you switch from the executing node to the standby node, the standby node inherits the OTS transaction information and performs the transaction recovery process.

- **Information that the standby node does not inherit**

Note that when node switching occurs, the post-switching node does not inherit the following information:

- HTTP session information of the application that the Web container manages

- Attribute information set in `ServletContext` of the Web container
- Session information of the Bean of the home interface and component interface
- Information about the operation status of applications, and the status of JNDI cache, connection pool and CTM queue

19.4 Settings for the N-to-1 recovery system (In Windows)

The N-to-1 recovery system has one configuration in which you prepare one standby node (**recovery server**) for N number of executing nodes. When using a global transaction in a configuration with redundant J2EE servers, use the N-to-1 recovery system to terminate the transaction when an error occurs in a particular J2EE server. Use the management portal to build this system. For details on the management portal, see the *uCosminexus Application Server Management Portal User Guide*. This section describes the settings of the N-to-1 recovery system.

Execute applications in N number of executing node Application Servers. If a failure occurs on Application Server in which applications are running, the cluster software detects this failure, and the recovery server terminates the transaction running on Application Server in which the failure occurred. Following this, application continues processing in the remaining executing node Application Servers that are operating.

For details on the N-to-1 recover system, see [19.2 Overview of the N-to-1 recovery system](#). For the system configuration, see [3.11.5 Configuration using server exclusive for recovery \(N-to-1 recovery system\)](#) in the *uCosminexus Application Server System Design Guide*.

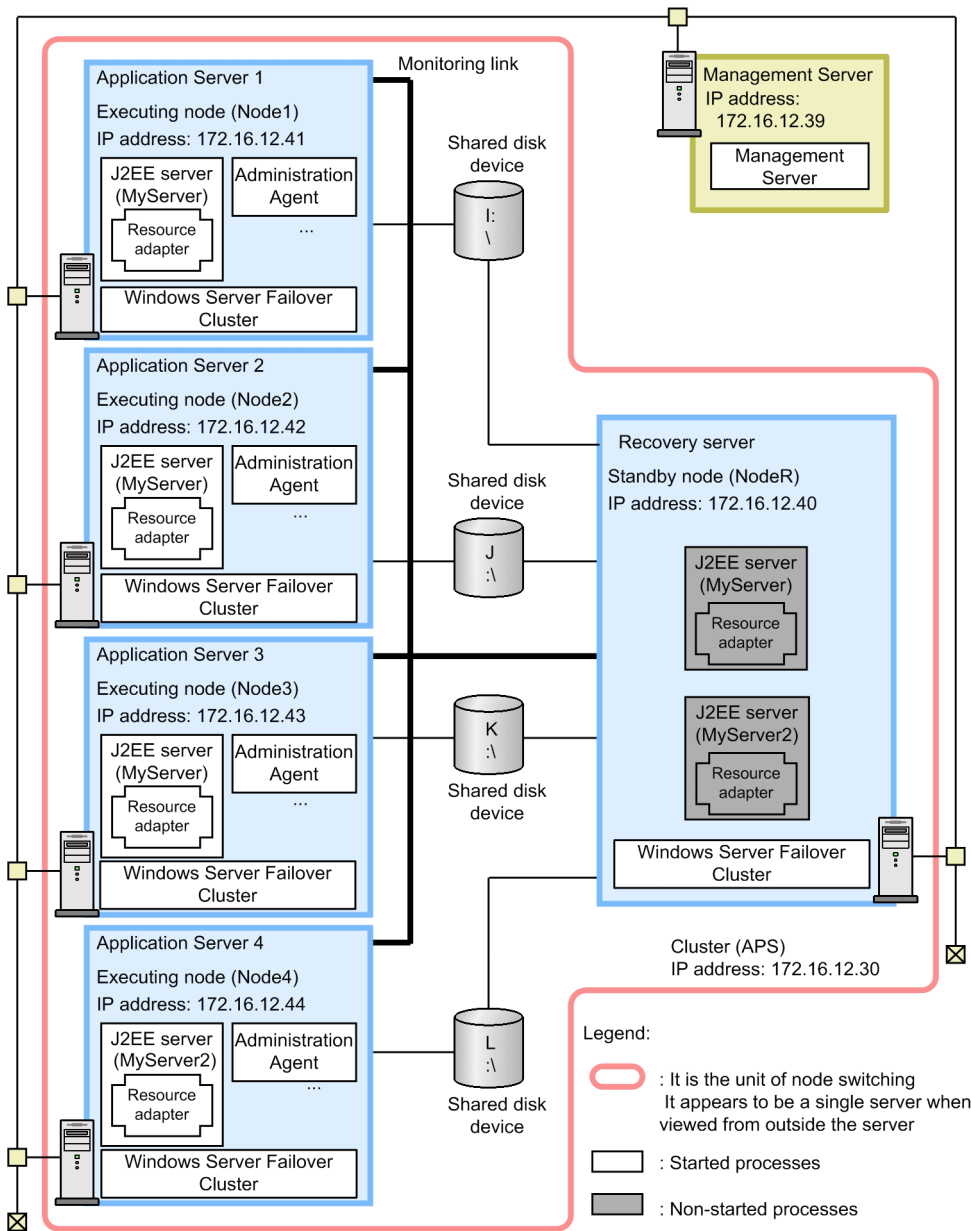
19.4.1 Procedure for setting the N-to-1 recovery system

This subsection describes an example of the system configuration and the procedure for setting up systems, when the systems are integrated with Windows Server Failover Cluster.

(1) Example of system configuration

The following figure shows an example of system configuration of the N-to-1 recovery system. Note that the subsequent sections describe examples of building the system using this example of system configuration.

Figure 19–3: Example of system configuration (In the N-to-1 recovery system)

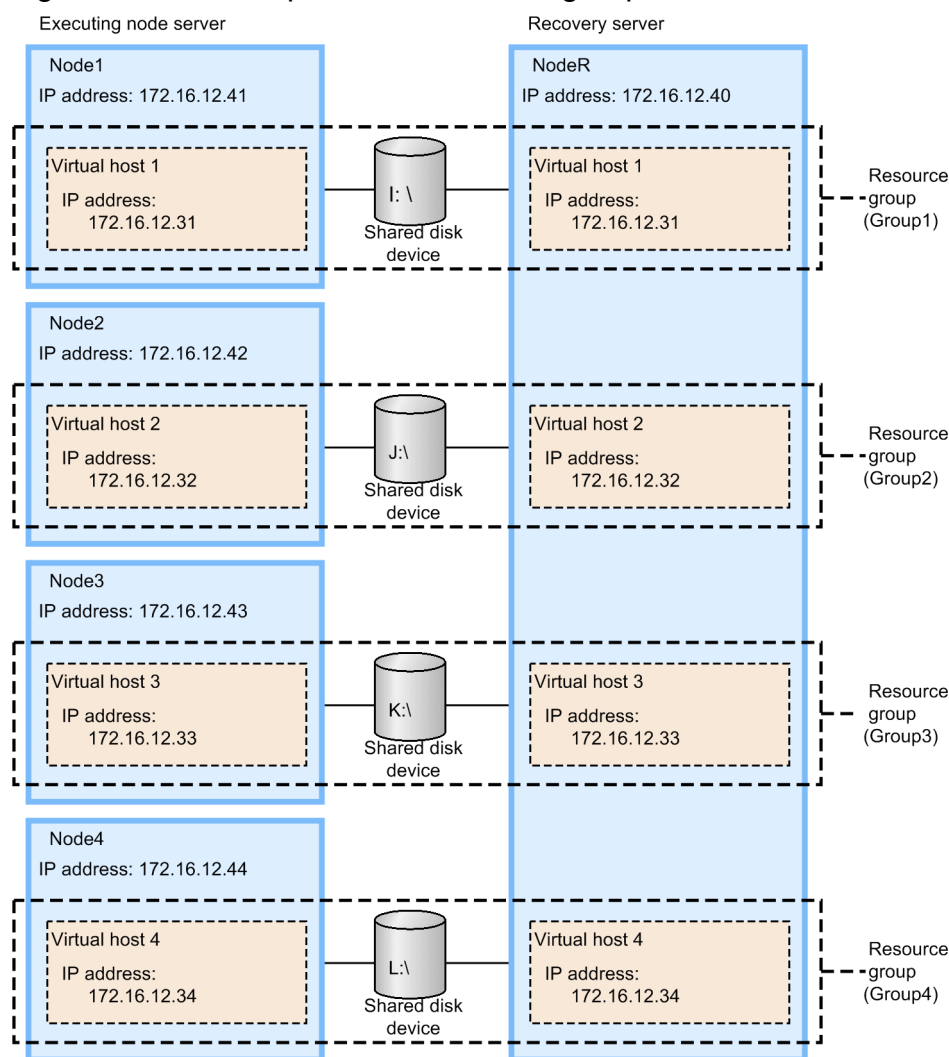


In this example, deploy four executing node servers and one recovery server. In three of the four executing nodes, deploy the same J2EE server called *MyServer* in which you deploy the same J2EE application and resource adapter. In the remaining one executing node, deploy a J2EE server called *MyServer2* that is different from the other J2EE server names and has a different J2EE application and resource adapter deployed in it.

In the N-to-1 recovery configuration, build multiple virtual hosts within one cluster. A combination of the executing node and recovery node configures the virtual host with the J2EE server deployed in a cluster configuration. In the N-to-1 recovery configuration, the J2EE server operates within the virtual host rather than the cluster. Therefore, set the IP address and host name of the virtual host.

A virtual host is defined as a resource group and includes the 'physical disk resource', 'IP address resource', 'network resource name', and 'general-purpose script resource'. The following figure shows an example of the resource groups described in this section:

Figure 19–4: Example of the resource group



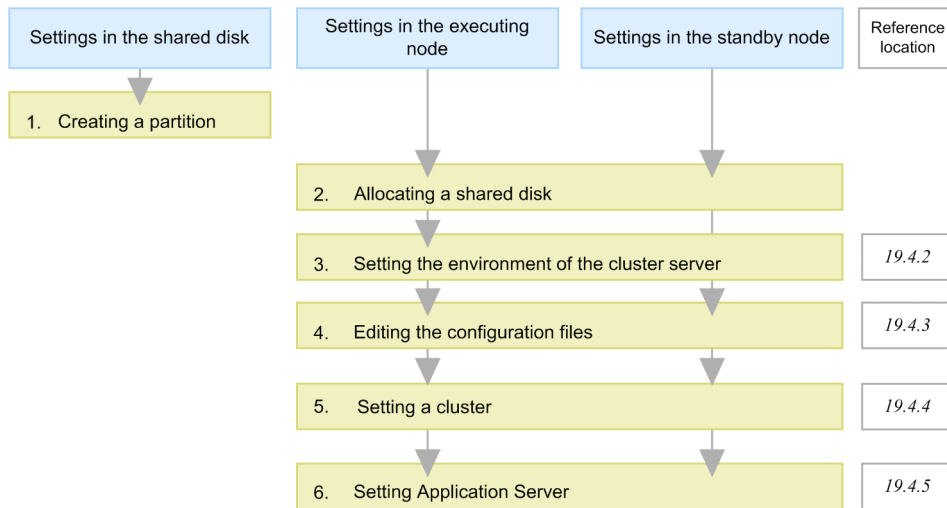
To deploy Application Server in a cluster configuration, set the alias IP address, and specify settings so that the running node inherits the alias IP address and the client does not consider the nodes in the cluster. In the N-to-1 recovery configuration, the alias IP address becomes the IP address of the virtual host.

In the J2EE server of the standby node, import and deploy a resource adapter with the same settings as all resource adapters that you are using in the J2EE servers of the executing node.

(2) Procedure for system setup

When integrating with Windows Server Failover Cluster, you must set up the management portal and cluster administrator. The following figure shows the procedure for setting the N-to-1 recovery system:

Figure 19–5: Procedure for setting the N-to-1 recovery system (In Windows)



Legend: ▼ : Necessary work

The following points describe the step 1 through 6 in the above figure:

1. Create N number of partitions in the shared disk to build a file system.
You create a storage location for the cluster management file of Windows Server Failover Cluster. Also, to use a global transaction in the N-to-1 recovery system, create the storage location for the transaction information. Note that you may store the cluster management file and transaction information in the same partition.
2. Allocate the shared disk to the system.
When allocating the shared disk to the system, allocate the same drive character to the executing node and standby node.
3. Set the environment of the cluster server with the management portal.
You specify the settings for using Windows Server Failover Cluster in 'Configuration definition of management domain' and 'Environment setup of a logical server' of the management portal. For details, see [19.4.2 Setting the environment of the cluster server](#).
4. Edit the configuration files.
You set up various definition files of Administration Agent, Management Server, and Cosminexus HTTP Server. For details, see [19.4.3 Editing the configuration files](#).
5. Setting a cluster.
You set up an environment for the cluster software. For details, see [19.4.4 Setting a cluster](#)
6. Set Application Server with the management portal and cluster administrator.
Use the management portal and cluster administrator to deploy Application Server in the cluster configuration, and then set the J2EE applications and resource adapters. For details, see [19.4.5 Setting Application Server](#).

19.4.2 Setting the environment of the cluster server

For details on the points to be considered when specifying the settings with the management portal during integration with Windows Server Failover Cluster, see [17.4.2 Setting the environment of the cluster server](#).

In the N-to-1 recovery configuration, set the IP address of the virtual host rather than the cluster IP address for the host name.

19.4.3 Editing the configuration files

You set up various definition files of Administration Agent, Management Server, and Cosminexus HTTP Server. This subsection describes the file settings that you must be careful about when linking with Windows Server Failover Cluster.

(1) Files to be set

The files that you must set are different for the executing node and standby node.

In the executing node

You must note the following settings for executing node:

- Setting the Administration Agent
- Setting Cosminexus HTTP Server
- Setting the worker
- Setting the Smart Agent

For details on setting these files, see [17.4.3 Editing the configuration files](#). In the N-to-1 recovery configuration, set the IP address of the virtual host rather than the cluster IP address in the host name.

In the standby node (dedicated for recovery)

You must note the following settings for standby node (dedicated for recovery):

- Setting the properties of the J2EE server

These settings are described as follows:

(2) Setting the properties of the J2EE server in the standby node

Note the following keys when setting properties in `usrconf.properties` in the J2EE server of the standby node:

- `ejbserver.distributedtx.XATransaction.enabled`
Set `true` to disable the light transaction.

19.4.4 Setting a cluster

You execute the following settings for a cluster:

- Create a script file
- Environment settings of the cluster software

(1) Create a script file

Code a script file for starting and stopping Application Server in the executing node using VBScript. Use the same script file in each node of the executing node, and allocate the script file to the same path.

The script file uses the general-purpose script as it is. For an example of the general-purpose script file, see *(2) Example of a general-purpose script file for the executing node*.

This subsection describes the monitoring targets and the monitoring methods for N-to-1 recovery systems.

(a) Monitoring targets

The monitoring target for 1-to-1 node switching systems of Application Server is as follows:

- Administration Agent

The node switching is executed when Administration Agent ends.

(b) Monitoring method

You monitor the availability of processes of Administration Agent. You monitor the following process:

- adminagent.exe

(2) Example of a general-purpose script file for the executing node

The following example describes a general-purpose script file used in the executing node of an N-to-1 recovery system:

```
Dim WshShell, fso, AccessInfoFile, oExec, oRun, oCosmiHome, oAdmin, oMngsvut
, oCjsleep
Dim oMngHost, oUser, oPass
Set WshShell = CreateObject("WScript.Shell")

oCosmiHome = WshShell.ExpandEnvironmentStrings("%COSMINEXUS_HOME%")
oAdminac = "" & oCosmiHome & "\manager\bin\adminagentctl""
oMngsvut = "" & oCosmiHome & "\manager\bin\mngsvrutil""
oCjsleep = "" & oCosmiHome & "\CC\server\bin\cjsleep""

oMngHost = "172.16.12.39" '--- Management Server
oUser = "admin" ' --- Management Server's userid
oPass = "admin" ' --- Management Server's password

'==== Open ====
Function Open( )
    Resource.LogInformation "===== Entering Open. ====="
    ' --- J2EE Server Name ---
    If Resource.PropertyExists("ServerName") = False Then
        Resource.LogInformation "### Private property ServerName is not set
. ###"
    End If
    ' --- Inprocess OTS Status Path ---
    If Resource.PropertyExists("StatusPath") = False Then
        Resource.LogInformation "### Private property StatusPath is not set
. ###"
    End If
    ' --- Cluster IP Address ---
    If Resource.PropertyExists("IPAddress") = False Then
        Resource.LogInformation "### Private property IPAddress is not set.
###"
    End If
    ' --- Cluster Host Name ---
    If Resource.PropertyExists("HostName") = False Then
        Resource.LogInformation "### Private property HostName is not set. #
##"
    End If
```

```

    Open = True
End Function

'==== Online ====
Function Online( )
    Resource.LogInformation "===== Entering Online. ====="

    '--- private properties ---
    If Resource.PropertyExists("ServerName") = False Or _
        Resource.PropertyExists("StatusPath") = False Or _
        Resource.PropertyExists("IPAddress") = False Or _
        Resource.PropertyExists("HostName") = False Then
        Resource.LogInformation "### Private property is not set. ###"
        Online = False
        Exit Function
    End If

    '--- adminagentctl ---
    Set oExec = WshShell.Exec(oAdminac & " start")
    Do While oExec.Status = 0
        WshShell.Run oCjsleep & " 1", 0, True
    Loop
    If oExec.ExitCode <> 0 Then
        Resource.LogInformation "### Administration Agent cannot start. ###"
rtn=" & oResult
        Resource.LogInformation oExec.StdErr.ReadAll
        Online = False
        Exit Function
    End If

    '--- mngsvrutil ---
    oRun = oMngsvut & " -m "& oMngHost & ":28080 -u " & oUser & " -p " & oPas
s & _
        " -t " & Resource.IPAddress & " -k host -s start serve
r"
    Set oExec = WshShell.Exec(oRun)
    Do While oExec.Status = 0
        WshShell.Run oCjsleep & " 1", 0, True
    Loop
    If oExec.ExitCode <> 0 Then
        Resource.LogInformation "### Logical Servers start failed. ###"
" & oResult
        Resource.LogInformation oExec.StdErr.ReadAll
        Online = False
        Exit Function
    End If
    Online = True
End Function

'==== LooksAlive ====
Function LooksAlive( )
    Resource.LogInformation "===== Entering LooksAlive. ====="

    '--- tasklist ---
    Set oExec = WshShell.Exec("tasklist /NH /FI ""IMAGENAME eq adminagent.ex
e""")
    Do While oExec.Status = 0
        WshShell.Run oCjsleep & " 1", 0, True

```

```

Loop

If InStr(1, oExec.Stdout.ReadAll, "adminagent.exe", 1) <> 0 Then
    LooksAlive = True
Else
    LooksAlive = False
End If
End Function

'==== IsAlive ====
Function IsAlive( )
    Resource.LogInformation "==== Entering IsAlive. ====="

    '--- mngsvrutil ---
    oRun = oMngsvut & " -m " & oMngHost & ":28080 -u " & oUser & " -p " & oP
ass & _
        " -t " & Resource.IPAddress & " -k host check adminAge
nt"
    Set oExec = WshShell.Exec(oRun)
    Do While oExec.Status = 0
        WshShell.Run oCjsleep & " 1", 0, True
    Loop
    If oExec.ExitCode <> 0 Then
        Resource.LogInformation "### Administration Agent command failed. ##
# rtn=" & oExec.ExitCode
        Resource.LogInformation oExec.Stderr.ReadAll
        IsAlive = False
    Else
        IsAlive = True
    End If
End Function

'==== Offline ====
Function Offline( )
    Resource.LogInformation "==== Entering Offline. ====="

    '--- mngsvrutil ---
    oRun = oMngsvut & " -m " & oMngHost & ":28080 -u " & oUser & " -p " & oP
ass & _
        " -t " & Resource.IPAddress & " -k host -s stop serve
r"
    Set oExec = WshShell.Exec(oRun)
    Do While oExec.Status = 0
        WshShell.Run oCjsleep & " 1", 0, True
    Loop
    If oExec.ExitCode <> 0 Then
        Resource.LogInformation "### Logical Server stopping failed. ### rtn
=" & oResult
        Resource.LogInformation oExec.Stderr.ReadAll
        Offline = False
        Exit Function
    End If

    '--- adminagentctl ---
    Set oExec = WshShell.Exec(oAdminac & " stop")
    Do While oExec.Status = 0
        WshShell.Run oCjsleep & " 1", 0, True
    Loop

```

```

    If oExec.ExitCode <> 0 Then
        Resource.LogInformation "### Administration Agent stopping failed. #
## rtn=" & oResult
        Resource.LogInformation oExec.StdErr.ReadAll
        Offline = False
        Exit Function
    End If
    Offline = True
End Function

'==== Close ====
Function Close( )
    Resource.LogInformation "==== Entering Close. =====
    Close = True
End Function

'==== Terminate ====
Function Terminate( )
    Resource.LogInformation "==== Entering Terminate. =====
    Terminate = True
End Function

```

With the general-purpose script file of the executing node, you change variable values as and when required. For details on the variables defined in the general-purpose script file, see (a) *Variables of the general-purpose script file*.

Also, with the general-purpose script file of the N-to-1 recovery system, one script file is referenced from multiple virtual servers. Because of this, use a private property so that the general-purpose script file can reference the settings of each server. For details on the private property specified for each virtual server, see (3) *Settings of private property*.

(a) Variables of the general-purpose script file

With the general-purpose script file of the executing node, change variable values as and when required. Change the values of the variables described in the following table.

Table 19–2: Variables of the general-purpose script file of the executing node (In N-to-1 recovery system)

Variable name	Contents
oMngHost	Host being operated by Management Server
oUser	Login ID for logging to Management Server
oPass	Password for logging to Management Server

(3) Settings of private property

Set up a private property for each virtual server. The following table describes the private properties required for the general-purpose script file.

Table 19–3: Private properties required for the general-purpose script file

Private property	Contents (Example)
ServerName	J2EE server name Specification example: MyServer
StatusPath	Status path of the in-process OTS

Private property	Contents (Example)
	Specification example: I:\otsstatus
IPAddress	IP address of the virtual server Specification example: 172.16.12.31
HostName	Virtual server name Specification example: Server1

The private properties are set up in the `cluster` command. For details on the `cluster` command, see the documentation of the OS that is being used. The following example describes the settings of a private property that uses the `cluster` command:

```
cluster res Script1 /priv ServerName=MyServer
cluster res Script1 /priv StatusPath="I:\otsstatus"
cluster res Script1 /priv IPAddress=172.16.12.31
cluster res Script1 /priv HostName=Server1

cluster res Script2 /priv ServerName=MyServer
cluster res Script2 /priv StatusPath="J:\otsstatus"
cluster res Script2 /priv IPAddress=172.16.12.32
cluster res Script2 /priv HostName=Server2

cluster res Script3 /priv ServerName=MyServer
cluster res Script3 /priv StatusPath="K:\otsstatus"
cluster res Script3 /priv IPAddress=172.16.12.33
cluster res Script3 /priv HostName=Server3

cluster res Script4 /priv ServerName=MyServer2
cluster res Script4 /priv StatusPath="L:\otsstatus"
cluster res Script4 /priv IPAddress=172.16.12.34
cluster res Script4 /priv HostName=Server4
```

(4) Environment settings of the cluster software

For the environment settings of Windows Server Failover Cluster, see the documentation of the cluster software that is being used.

19.4.5 Setting Application Server

When setting Application Server, deploy Application Server in a cluster configuration using the management portal and cluster administrator. Also, set up the logical servers and distribute the configuration information, and then import J2EE applications and resource adapters in the J2EE servers. The following is the procedure to set Application Server:

1. In each executing node, set each resource group to online.
Set Group1 of Node1, Group2 of Node2, Group3 of Node3, and Group4 of Node4 to online.
2. Using the management portal, set up the J2EE server of each executing node.
Set up the J2EE server with the Setup screen of 'Configuration definition of management domain'. For details on the performing operations with the management portal and various screens, see the *uCosminexus Application Server Management Portal User Guide*.

3. Using the management portal, distribute the information set in the J2EE servers to the executing nodes.
Distribute the information set in the J2EE servers to each executing node with the Distribution of Setup Information screen of 'Environment setup of a logical server'.
4. Use the server management commands to import J2EE applications and resource adapters in the J2EE server of Application Server of each executing node. Set up and start the imported J2EE applications and resource adapters. For details on the settings of the J2EE applications, see *4.1.29 Setting and starting the business application (when using CUI)* in the *uCosminexus Application Server System Setup and Operation Guide* and for the settings of the resource adapters, see the following subsections in the *uCosminexus Application Server System Setup and Operation Guide*:
 - *4.1.26 Setting DB Connector (when using CUI)*
 - *4.1.27 Setting resource adapters other than DB Connector (when using CUI)*
 - *4.1.28 Starting the resource adapter (when using CUI)*For operations of the server management commands, see *3. Basic Operations of Server Management Commands* in the *uCosminexus Application Server Application Setup Guide*.
Execute the server management commands after starting the J2EE server and its prerequisite processes. Once you have completed the operations with server management commands, stop the server management commands.
5. With the cluster administrator, execute 'Move Group' for each resource group to switch the nodes, and then change the resource owner to the standby node (NodeR).
6. Using the server management commands, import the resource adapter in the standby node Application Server.
In the J2EE server of the standby node, import a resource adapter with the same settings as the resource adapter being used in each J2EE server of the executing node.
However, you need not use the connection pooling functionality in the resource adapter to be defined in the J2EE server of the standby node. Specify 0 as the minimum and maximum value of the connections to be set with the server management commands.
7. With the cluster administrator, execute 'Move Group' for each resource group to switch the nodes, and then return the resource owner to each executing node.

19.5 Settings for the N-to-1 recovery system (In UNIX)

The N-to-1 recovery system has one configuration in which you prepare one standby node (**recovery server**) for N number of executing nodes. When using a global transaction in a configuration with redundant J2EE servers, use the N-to-1 recovery system to terminate the transaction when an error occurs in a particular J2EE server. Use the management portal to build this system. For details on the management portal, see the *uCosminexus Application Server Management Portal User Guide*. This section describes the settings of the N-to-1 recovery system.

Execute applications in N number of executing node Application Servers. If a failure occurs in Application Server in which applications are running, the HA monitor software detects this failure and the recovery server terminates the transaction running on Application Server in which the failure occurred. Following this, application continues processing in the remaining executing node Application Servers that are operating.

Note that you can use the operations of the N-to-1 recovery system only in AIX or Linux.

For details on the functionality, see [19.2 Overview of the N-to-1 recovery system](#). For system configuration, see [3.11.5 Configuration using server exclusive for recovery \(N-to-1 recovery system\)](#) in the *uCosminexus Application Server System Design Guide*. For HA monitor, see the manual *High-reliability System Monitoring Functionality HA Monitor*.

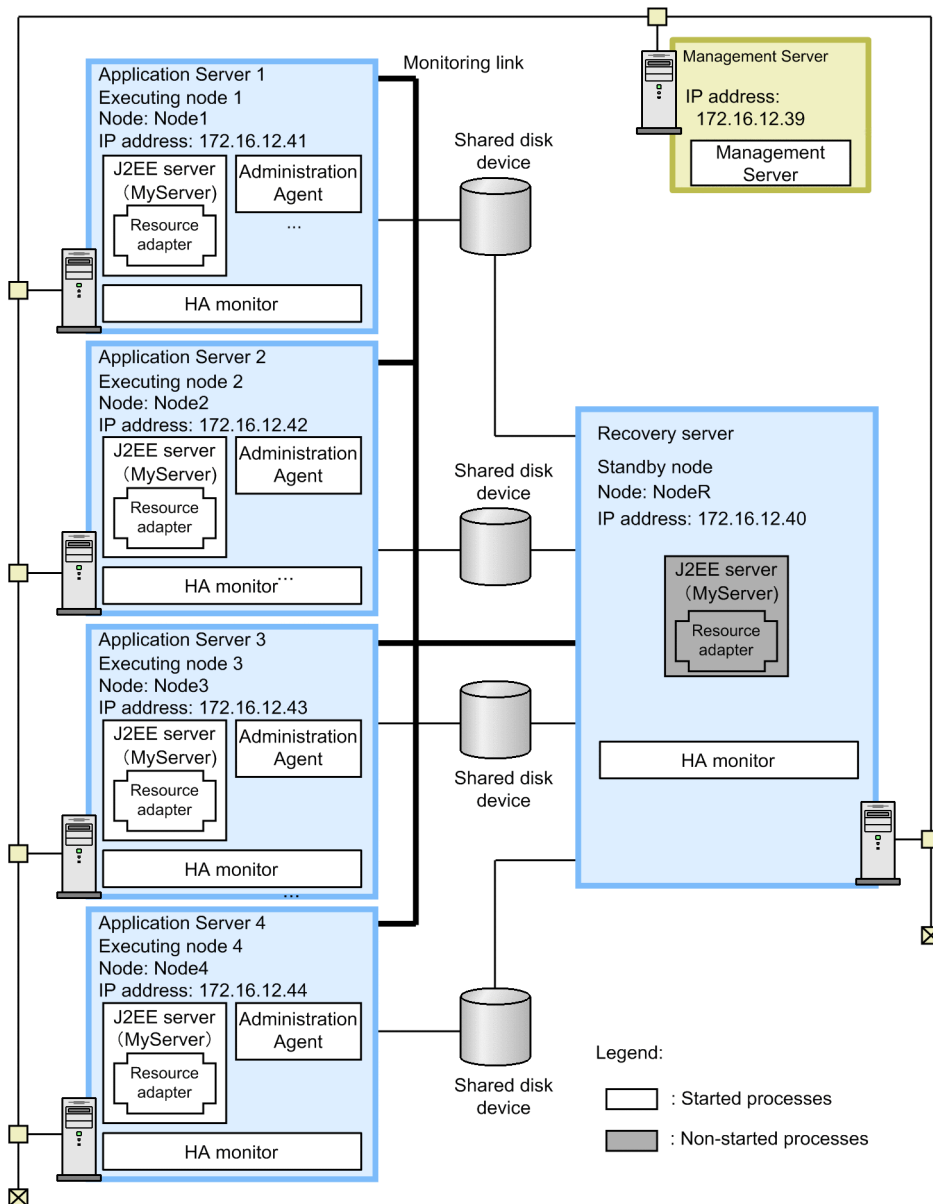
19.5.1 Procedure for setting the N-to-1 recovery system

This subsection describes an example of system configuration and the procedure for setting the system. This subsection also describes the setup procedure for adding or deleting an executing node.

(1) Example of system configuration

The following figure shows an example of system configuration of the N-to-1 recovery system. Note that the subsequent points describe examples of building the system using this example of system configuration.

Figure 19–6: Configuration example of the N-to-1 recovery system (In UNIX)



An overview of this example is described as follows:

- Deploy four executing node servers and one recovery server. The J2EE servers in the four executing nodes are configured so that the settings of the J2EE applications and resource adapters are the same. Only the alias IP address and the shared disk device to be mounted are different.
- Set the same name for the J2EE server in the executing node and the standby node. In this case, as you have deployed a J2EE server called *MyServer* in the executing nodes, also deploy a J2EE server called *MyServer* in the standby node.
- Build and perform operation management of the system using the management portal of Management Server.
- Operate Administration Agent in the executing node and not in the standby node.

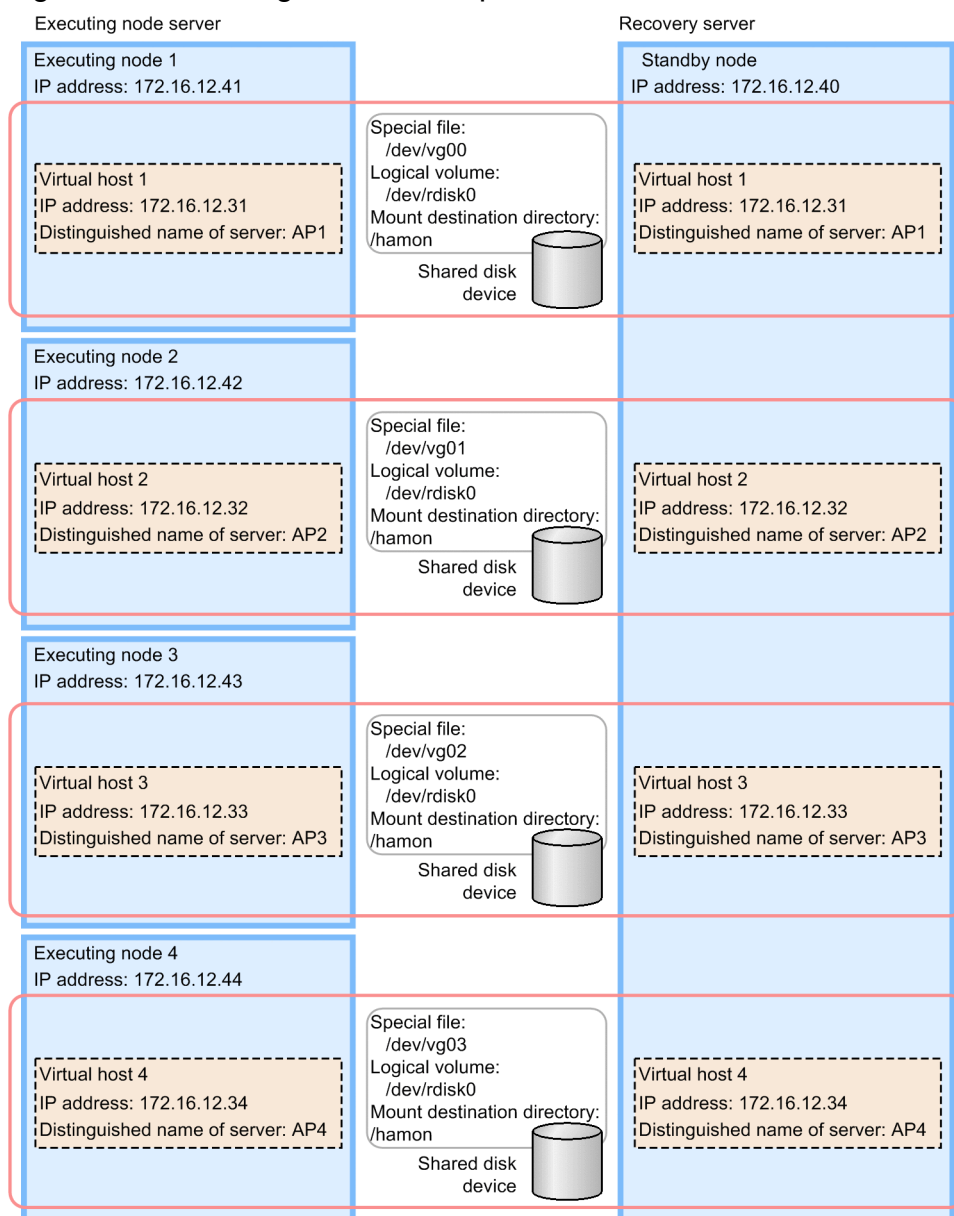
When operating the N-to-1 recovery system, build an Application Server that satisfies the following conditions:

- Use a global transaction that uses OTS. Save the status file of the transaction in the shared disk device.
- Operate the Naming Service as an in-process.

- In the J2EE server of the standby node, import and deploy a resource adapter with the same settings as all resource adapters that you are using in the J2EE servers of the executing node.
- In the N-to-1 recovery configuration, build multiple virtual hosts within one cluster. The executing node and recovery server configure the virtual host with the J2EE server deployed in a cluster configuration. In the N-to-1 recovery configuration, the J2EE server operates within the virtual host rather than the cluster. Therefore, set the IP address and host name of the virtual host.

To deploy Application Server in a cluster configuration, set the alias IP address, and specify settings so that the running node inherits the alias IP address and the client does not consider the nodes in the cluster. In the N-to-1 recovery configuration, the alias IP address becomes the IP address of the virtual host. A configuration example of the virtual host in the N-to-1 recovery configuration is as follows:

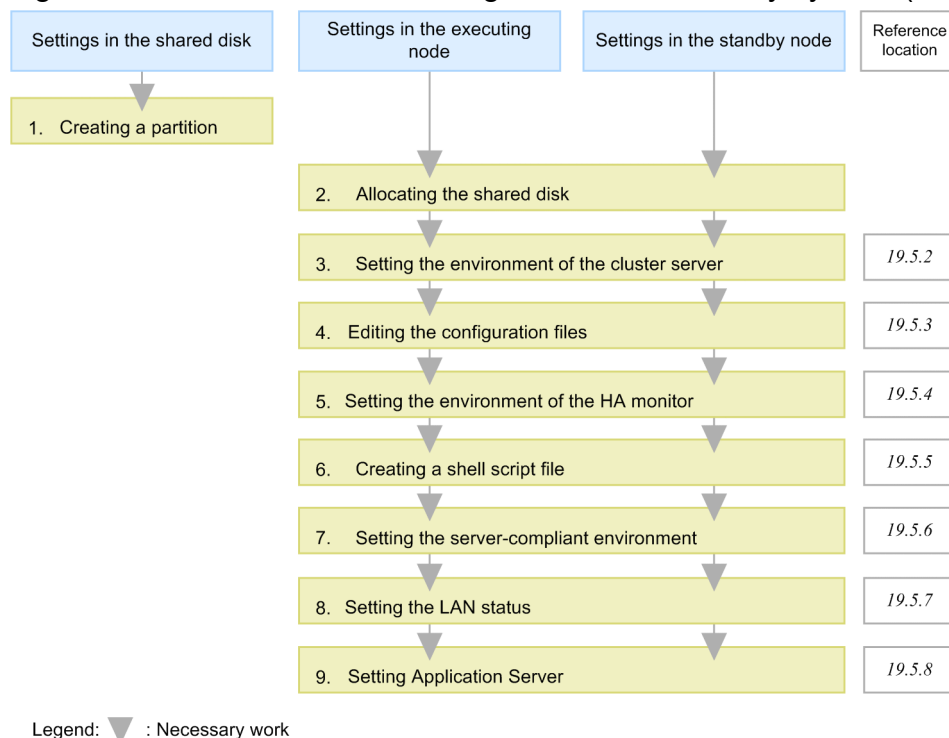
Figure 19–7: Configuration example of the virtual host in the N-to-1 recovery configuration



(2) Procedure for system setup

When integrating with the HA monitor, you must set the files of the management portal and HA monitor. The following figure shows the procedure for setting the N-to-1 recovery system:

Figure 19–8: Procedure for setting the N-to-1 recovery system (In UNIX)



Steps 1 to 9 of the figure are described as follows:

1. Create N number of partitions in the shared disk to build a file system.
2. Allocate the shared disk to the system.
When allocating the shared disk to the system, specify the same mount destination directory in the active node and spare node.
3. Set the environment of the cluster server with the management portal.
Specify the settings for using the HA monitor in 'Settings of Cosminexus Management Server', 'Configuration definition of management domain' and 'Environment setup of a logical server' of the management portal. For details, see [19.5.2 Setting the environment of the cluster server](#).
4. Edit the configuration files.
Specify various definition files of Administration Agent, Management Server, and Cosminexus HTTP Server. For details, see [19.5.3 Editing the configuration files](#).
5. Set the environment of the HA monitor.
Define the environment of the HA monitor with the `sysdef` file. For details, see [19.5.4 Setting the environment of the HA monitor](#).
6. Create the shell script file.
Create the shell script file for monitoring the Administration Agent, and for starting and stopping the logical servers on Application Server. For details, see [19.5.5 Creating a shell script](#).
7. Set the server-compliant environment.
Using the `servers` file of the HA monitor, define the environment of the active-node server and spare-node server to be operated on the nodes. For details, see [19.5.6 Setting the server-compliant environment](#).
8. Set the status of LAN.

Using the LAN status setup file of the HA monitor, specify the IP address of the LAN adapter, and define switching of LAN in the HA monitor. For details, see [19.5.7 Setting the LAN status](#).

9. Using the management portal and commands of the HA monitor, specify the settings for Application Server.

Using the management portal and commands of the HA monitor, deploy Application Server in a cluster configuration, and set the J2EE applications and resource adapters. For details, see [19.5.8 Setting Application Server](#).

Reference note

- In the N-to-1 recovery system, if there is timeout for the recovery process in the standby node due to a failure (when the server is down or a deadlock has occurred) in the database server, you must execute the recovery process manually.

After finding the cause for timeout, execute the `monbegin` command corresponding to the executing node host that has stopped, and then execute the `monact` command corresponding to the executing node host that has stopped, in the standby node host.

When an error occurs in the N-to-1 recovery system, see [2.5.8 If a problem occurs in N-to-1 recovery systems](#) in the *uCosminexus Application Server Maintenance and Migration Guide*.

The files that you must set in the N-to-1 recovery system are different for the executing node and the standby node. The following table describes the necessity of setting files in the N-to-1 recovery system:

Table 19–4: Necessity of setting files in the N-to-1 recovery system

Category		File type	Necessity of setting the file		Reference
			(N number of) Executing nodes	(One) Standby node (dedicated for recovery)	
Setting Application Server	Setting the Administration Agent	<code>adminagent.properties</code>	Necessary (Individual settings in N nodes)	N	19.5.3
	Setting the Smart Agent	<code>localaddr</code>			
	Setting Cosminexus HTTP Server	<code>httpsd.conf#</code>			
	Setting the properties of the J2EE server	<code>usrconf.properties#</code>	Necessary (Common settings in N nodes)	Necessary (Almost same as the executing node, except for alias IP address)	
Setting the HA monitor	Setting the environment of the HA monitor	<code>sysdef</code>	Necessary (Individual settings in each host)	Necessary (Settings for each of the N nodes)	19.5.4
	Creating the shell script file	<code>monitor.sh</code> <code>start.sh</code> <code>stop.sh</code>	Necessary (Usage of a common command for N nodes)	N	19.5.5
	Setting the server-compliant environment	<code>servers</code>	Necessary (Individual settings in N nodes)	Necessary (Settings for each of the N nodes)	19.5.6
		<code>recover.sh</code>	N	Necessary (Usage of a common command for N number of	

Category		File type	Necessity of setting the file		Reference
			(N number of) Executing nodes	(One) Standby node (dedicated for recovery)	
				executing nodes. Differences are passed with arguments)	
	Setting the LAN status	<i>server-identification-name</i> .up <i>server-identification-name</i> .down	Necessary (Individual settings in each host)	Necessary (Copy the configuration file of the executing node)	19.5.7

Legend:

N: Not required.

#

You can set these files when you build the execution environment of Application Server using the management portal.

(3) Setup procedure for adding or deleting an executing node

The setup procedure for adding or deleting (degenerating) an executing node in the N-to-1 recovery system is described as follows.

Adding an executing node

1. Create a mount directory in the executing node to be added.
2. In the `servers` file of the standby node, add an entry corresponding to the executing node to be added.
You do not require this step if the entry corresponding to the executing node to be added already exists.
3. In the standby node, execute the `monbegin` command corresponding to the executing node that you have added.
4. In the executing node, execute the `monbegin` command.
5. Using the management portal, build the execution environment of Application Server in the executing node to be added.
6. Using the management portal, start the executing node Application Servers as a batch.

Deleting an executing node

1. Execute the `monend` command in the executing node.

19.5.2 Setting the environment of the cluster server

For details on the points to be considered when specifying the settings with the management portal during integration with HA monitor, see [17.6.2 Setting the environment of the cluster server](#). In the N-to-1 recovery configuration, specify the IP address of the virtual host instead of the alias IP address in the host name.

19.5.3 Editing the configuration files

Specify various definition files of Administration Agent, Management Server, and Cosminexus HTTP Server. This subsection describes the file settings that you must be careful about when integrating with the HA monitor.

(1) Files to be set

The files that you must set are different for the executing node and standby node.

In the (N number of) executing nodes

The settings that you must be careful about in the (N number of) executing nodes are as follows:

Files that you can set by direct editing

You can set the following files by editing them directly:

- Setting the Administration Agent (`adminagent.properties`)
- Setting the Smart Agent (`localaddr`)

Files that you can set with the management portal

You can set the following files when building the execution environment of Application Server using the management portal:

- Setting Cosminexus HTTP Server (`httpsd.conf`)
- Setting the properties of the J2EE server (`usrconf.properties`)

For details on setting these files, see [17.6.3 Editing the configuration files](#). In the N-to-1 recovery configuration, set the IP address of the virtual host instead of the alias IP address in the host name.

In the standby node (One node for recovery)

The configuration file that you must be careful about in the standby node (one node for recovery) is described as follows. Note that you can set this file by editing it directly.

- Setting the properties of the J2EE server (`usrconf.properties`)

(2) Setting the properties of the J2EE server in the standby node

The key that you must pay attention to when setting properties in `usrconf.properties` in the J2EE server of the standby node is as follows:

- `ejbserver.distributedtx.XATransaction.enabled`
Set `true` to disable the light transaction.

19.5.4 Setting the environment of the HA monitor

Depending on the system you are using, define the environment of the HA monitor in the definition file `sysdef`. You must specify the settings individually in each host including the N number of executing nodes and the single standby node for recovery.

Set a unique name (host name of the HA monitor), `address` (unique value for resetting the CPU), `lan` (host name of LAN) according to the host environment. You must set unique values even in the N number of executing nodes and the single standby node for recovery. Also, you must set the same value for `lanport` (service name corresponding to the host name specified in the `lan` operand) even in the N number of executing nodes and the single standby node for recovery.

An example of the `sysdef` file is as follows. N in the sample indicates the Nth value.

```
environment name      $host_N,      #Host name of the own node
                  address $uniq_N,    #Unique value for resetting the CPU
```

```

patrol      5,          #Time period for judging node failure
lan         $host_N     #Host name of LAN
lanport     $service_N  #Service name corresponding to the host nam
e of lan operand

```

For details on setting the environment of the HA monitor, see the manual *High reliability System Monitoring Functionality HA Monitor*.

19.5.5 Creating a shell script

Create the following shell script files to monitor the processes of the Administration Agent and to start and stop the Administration Agent and logical servers:

In the (N number of) executing nodes

The shell script files that you must set in the executing nodes are described as follows. Note that you must use a common command in the N number of nodes.

- Shell script file for monitoring the processes of the Administration Agent
- Shell script file for starting the Administration Agent
- Shell script file for stopping the Administration Agent and logical servers

In the standby node (One node for recovery)

The shell script file that you must set in the standby node is described as follows. Note that you use a common command for the N number of executing nodes and pass the differences with the arguments of the command.

- Shell script file for executing the recovery process

Each script is described as follows:

(1) Shell script file for monitoring the processes of the Administration Agent

An example of the shell script file for monitoring the processes of the Administration Agent (`monitor.sh`) is described as follows:

```

#!/bin/sh

LOGDIR=/home/manager/hamon/log
AA=/opt/Cosminexus/manager/bin/adminagent

logg()
{
    echo `date '+[%Y/%m/%d %H:%M:%S]`'["$${}": $1" \
    >> ${LOGDIR}/adminagent.log 2>&1
}

logg "### $0: started. ###"
while true
do
    CHECK=`ps -ef | grep $AA | grep -v grep`

    if [ "$CHECK" = "" ]
    then

```



```

        logg "### $0: stop. ###"
        exit 0
    fi
    sleep 10
done

```

In this example, every 10 seconds, the shell script file monitors whether the processes of the Administration Agent exist.

(2) Shell script file for starting the Administration Agent

An example of the shell script file for starting the Administration Agent (`start.sh`) is described as follows:

```

#!/bin/sh

LOGDIR=/home/manager/hamon/log
MNGDIR=/opt/Cosminexus/manager

logg()
{
    echo `date '+[%Y/%m/%d %H:%M:%S]` `[$$]: $1" \
    >> ${LOGDIR}/adminagent.log 2>&1
}

# start Administration Agent
logg "### $0: starting Administration Agent. ###"
$MNGDIR/bin/adminagentctl start
if [ $? -eq 0 ] ; then
    logg "### $0: Administration Agent start normally. ###"
else
    logg "### $0: Administration Agent cannot start. ###"
    exit 1
fi

exit 0

```

(3) Shell script file for stopping the Administration Agent and logical servers

An example of the shell script file for stopping the Administration Agent and logical servers (`stop.sh`) is described as follows:

```

#!/bin/sh

LOGDIR=/home/manager/hamon/log
MNGDIR=/opt/Cosminexus/manager

logg()
{
    echo `date '+[%Y/%m/%d %H:%M:%S]` `[$$]: $1" \
    >> ${LOGDIR}/adminagent.log 2>&1
}

# stop logical server
logg "### $0: stop logical servers. ###"
$MNGDIR/bin/mngsvrutil -m mnghost:28080 -u admin -p admin \

```

```

-t 172.16.12.31 -k host -s stop server

# stop Administration Agent
logg "### $0: stopping Administration Agent. ###"
$MNGDIR/bin/adminagentctl stop

exit 0

```

The settings to be specified in the shell script file are described as follows.

- **Stopping the logical servers**

Specify the settings for batch stop of the logical servers running on the same host. Use the `mngsvrutil` command for batch stop of logical servers.

The user name and password are passed as arguments. Describe these values in the client-machine definition file (`.mngsvrutilrc`) of the `mngsvrutil` command, and set appropriate access privileges to manage these values.

Specify the alias IP address in the `-t` option of the `mngsvrutil` command.

For the `mngsvrutil` command, see *mngsvrutil (Management Server management command)* in the *uCosminexus Application Server Command Reference Guide*. For the client-machine definition file (`.mngsvrutilrc`), see 8.2.14 *.mngsvrutilrc (Client-side definition file of the mngsvrutil command)* in the *uCosminexus Application Server Definition Reference Guide*.

- **Stopping the Administration Agent**

Specify the settings for stopping the Administration Agent. Use `adminagentctl stop` to stop the Administration Agent.

For the `adminagentctl` command, see *adminagentctl (start or stop Administration Agent)* in the *uCosminexus Application Server Command Reference Guide*.

(4) Shell script file for executing the recovery process

An example of the shell script file for executing the recovery process for each J2EE server (`recover.sh`) is described as follows:

```

#!/bin/sh

LOGDIR=/home/manager/hamon/log
PATH=/opt/Cosminexus/CC/server/bin:/bin:/usr/bin:/home/manager/hamon/bin
LD_LIBRARY_PATH=/opt/DABroker/lib:/opt/Cosminexus/jdk/lib:/opt/Cosminexus/TP
B/lib:/opt/Cosminexus/PRF/lib:/opt/Cosminexus/CTM/lib:/bin:/opt/HiRDB/client
/lib:/opt/oracle/app/oracle/product/10.1.0/client_1/lib:/opt/oracle/app/orac
le/product/10.1.0/client_1/lib/libclntsh.so.10.1.0:/opt/oracle/app/oracle/prod
uct/10.1.0/client_1/lib/libclntsh.so
export LD_LIBRARY_PATH

LOCKFILE=/var/lock/kosmi_recover.lock
SLEEP_TIME=60
RETRIES=30
STATUS_PATH=$3/otsstatus

cjlockfile()
{
    counter=52
    TMP_LOCK_FILE=`dirname $3`/$$$.lock
    echo $$ > $TMP_LOCK_FILE
}

```

```

if [ -f $3 ]
then
    kill -0 `cat $3` 2>/dev/null || rm -f $3
fi

until ln $TMP_LOCK_FILE $3 2>/dev/null
do
    counter=`expr $counter - 1`
    if [ $counter -le 0 ]
    then
        rm -f $TMP_LOCK_FILE
        return 1
    fi
    sleep $1
done

rm -f $TMP_LOCK_FILE
return 0
}

logg()
{
    echo `date '+[%Y/%m/%d %H:%M:%S]`'"[$$]: $1" \
    >> ${LOGDIR}/adminagent.log 2>&1
}

if cjlockfile ${SLEEP_TIME} ${RETRIES} ${LOCKFILE}
then
    logg "### $0: started for $1 ###"
    cjstartrecover MyServer -p vbroker.se.iioptp.host=$2 \
        -p ejbserver.distributedtx.ots.status.directory1=$STATUS_P
ATH \
        -t 600
    logg "### $0: ended. $? ###"
fi

rm -f $LOCKFILE

exit 0

```

The settings to be specified in the shell script file are described as follows.

- The information that is different for each executing node (alias name, alias IP address, and mount directory) and is required for the recovery process is passed with `actcommand` of the standby node of `servers` file, as an argument of the command.
- You must explicitly specify environment variables such as `LD_LIBRARY_PATH`. For the environment variables, see *4.1.10 Environment variables* in the *uCosminexus Application Server System Setup and Operation Guide*.
- Use the `cjlockfile` function to execute the recovery process exclusively. This function provides a semaphore based on file lock. Specify the sleep time (in seconds) in the first argument, retry frequency in the second argument, and the lock file in the third argument. In this function, if the file specified in the third argument is locked, the system retries the acquisition of the file lock after waiting for the time period specified in the first argument. If the file lock cannot be acquired even after retrying for the number of times specified in the second argument, the recovery process fails. If the process for locking the file does not exist, the lock is cancelled.
- Execute the recovery process for the J2EE server (MyServer) with `jstartrecover`.

- In this sample, the timeout period is set to 600 seconds.
- With the `-p` option of the `cjstartrecover` command, specify the status file directory of the executing node that has stopped, in the `ejbserver.distributedtx.ots.status.directory1` key.
- With the `-p` option of the `cjstartrecover` command, specify the alias IP address after node switching, in the `vbroker.se.iiop_tp.host` key.
- With the `-p` option of the `cjstartrecover` command, overwrite a different property in each executing node that has stopped.
- In case of a configuration with multiple J2EE servers, execute the `cjstartrecover` command only as many times as the number of J2EE servers. In such cases, properly modify the J2EE server name (`MyServer`) and the path of the OTS status file directory.

For the `cjstartrecover` command, see *cjstartrecover (recover J2EE server transaction)* in the *uCosminexus Application Server Command Reference Guide*. For the `ejbserver.distributedtx.ots.status.directory1` and `vbroker.se.iiop_tp.host` keys, see *2.2.3 usrconf.properties (User property file for J2EE servers)* in the *uCosminexus Application Server Definition Reference Guide*.

19.5.6 Setting the server-compliant environment

When setting the server-compliant environment with the HA monitor, define the environment of the executing server and standby server to be operated on the nodes.

In the (N number of) executing nodes

You must specify individual settings in the N number of nodes.

In the standby node (One node for recovery)

You must specify the settings for N number of executing nodes.

In the `servers` definition file, define the server-compliant environment for N-to-1 recovery system. The following table describes the settings for the server-compliant environment:

Table 19–5: Settings for the server-compliant environment (In the N-to-1 recovery system)

Operand	Settings
<code>name</code>	<ul style="list-style-type: none"> • In the executing nodes To generalize the startup command in the N nodes, specify a unique name in <code>name</code>, and describe the shell script file for starting the Administration Agent in <code>actcommand</code>. • In the standby node Specify the same value as the corresponding executing node.
<code>alias</code>	<ul style="list-style-type: none"> • In the executing nodes This operand specifies the identification name of the server. Specification example: <code>AP1</code> • In the standby node Specify the same value as the corresponding executing node.
<code>acttype</code>	This operand specifies the startup method of the server. Here, you specify <code>monitor</code> because you start the server with the HA monitor command.
<code>termcommand</code>	<ul style="list-style-type: none"> • In the executing nodes This operand specifies the shell script file for stopping the Administration Agent and logical servers of the executing nodes.

Operand	Settings
	<p>Specification example: <code>/home/manager/hamon/bin/stop.sh</code></p> <ul style="list-style-type: none"> In the standby node <p>This operand specifies the shell script file for stopping the Administration Agent and logical servers of the standby node.</p>
<code>initial</code>	<p>This operand specifies the status when you start the server.</p> <ul style="list-style-type: none"> In the executing nodes <p>Specify <code>online</code>.</p> <ul style="list-style-type: none"> In the standby node <p>Specify <code>standby</code>.</p>
<code>disk</code>	<p>This operand specifies the character-type special file name of the shared disk device.</p> <ul style="list-style-type: none"> In the executing nodes <p>Specify in accordance with the allocated shared disk.</p> <ul style="list-style-type: none"> In the standby node <p>Specify the same value as the corresponding executing node.</p>
<code>lan_updown</code>	<p>This operand specifies whether or not to use the status setup file of LAN. Specify <code>use</code> because you are using the status setup file of LAN.</p>
<code>fs_name</code>	<p>This operand specifies the absolute path name of the logical volume corresponding to the file system to be switched to.</p> <ul style="list-style-type: none"> In the executing nodes <p>Specify in accordance with the allocated shared disk.</p> <ul style="list-style-type: none"> In the standby node <p>Specify the same value as the corresponding executing node.</p>
<code>fs_mount_dir</code>	<p>This operand specifies the absolute path name of the mount-destination directory of the file system to be switched to.</p> <ul style="list-style-type: none"> In the executing nodes <p>Specify in accordance with the allocated shared disk.</p> <ul style="list-style-type: none"> In the standby node <p>The mount directory is different for each corresponding executing node. You must create the mount directory beforehand when building the standby node because only an existent directory must be specified.</p>
<code>patrolcommand</code>	<ul style="list-style-type: none"> In the executing nodes <p>This operand specifies the shell script file for monitoring the processes of the Administration Agent. Specification example: <code>/home/manager/hamon/bin/monitor.sh</code></p> <ul style="list-style-type: none"> In the standby node <p>Specify the <code>monend</code> command, and once the recovery process with <code>actcommand</code> has terminated, you terminate the HA monitor.</p>
<code>servexec_retry</code>	<p>This operand specifies the restart frequency when the system detects a failure in an executing node. Specify the same value in the executing nodes and the standby node.</p>
<code>waitserv_exec</code>	<p>Specify <code>yes</code> to wait for completion of execution of the startup command when the startup completion process is executed.</p>
<code>actcommand</code>	<ul style="list-style-type: none"> In the executing nodes <p>This operand specifies the shell script file for starting the Administration Agent. Specification example: <code>/home/manager/hamon/bin/start.sh</code></p> <ul style="list-style-type: none"> In the standby node <p>This operand specifies the shell script file for executing the recovery process. Specification example: <code>/home/manager/hamon/bin/recover.sh</code></p>

A sample of the `servers` file is described as follows. N and $N + 1$ in the sample indicate the N^{th} value and $N + 1^{\text{th}}$ value, respectively.

Sample of the servers file (In the executing node)

The underlined part in this sample indicates that the contents are different for each executing node.

```
server name          $name_N,
  alias              $alias_N,
  acttype            monitor,
  initial             online,
  termcommand        /home/manager/hamon/bin/stop.sh,
  disk               $disk_N,
  lan_updown         use,
  fs_name            $fs_name_N,
  fs_mount_dir       /hamon,
  patrolcommand      /home/manager/hamon/bin/monitor.sh,
  servexec_retry     3,
  waitserv_exec      yes,
  actcommand         /home/manager/hamon/bin/start.sh;
```

Sample of the servers file (In the standby node)

In this sample, to set a single command for executing the recovery process, different information (alias name, alias IP address, and mount directory) is passed as arguments for each executing node. The underlined part in this sample indicates that the contents are different from the `servers` file of the executing node.

```
server name          $name_N,
  alias              $alias_N,
  acttype            monitor,
  initial             standby,
  termcommand        /home/manager/hamon/bin/stop.sh,
  disk               $disk_N,
  lan_updown         use,
  fs_name            $fs_name_N,
  fs_mount_dir       $mnt_N,
  patrolcommand      "/opt/hitachi/HAMon/bin/monend $alias_N",
  servexec_retry     3,
  waitserv_exec      yes,
  actcommand         "/home/manager/hamon/bin/recover.sh $alias_N $ip_
N $mnt_N";

server name          $name_N+1,
  alias              $alias_N+1,
  acttype            monitor,
  initial             standby,
  termcommand        /home/manager/hamon/bin/stop.sh,
  lan_updown         use,
  disk               $disk_N+1,
  fs_name            $fs_name_N+1,
  fs_mount_dir       $mnt_N+1,
  patrolcommand      "/opt/hitachi/HAMon/bin/monend $alias_N+1",
  servexec_retry     3,
  waitserv_exec      yes,
  actcommand         "/home/manager/hamon/bin/recover.sh $alias_N+1 $ip_
N+1 $mnt_N+1";
```

The settings to be specified in the `servers` file of the standby node are described as follows.

- Describe the settings for the N number of executing nodes in the `servers` file of the standby node. Also describe entries corresponding to the executing nodes.
- To set a single command for executing the recovery process, we recommend that you pass different information (alias name, alias IP address, and mount directory) as arguments for each executing node.

19.5.7 Setting the LAN status

This subsection defines switching of LAN in the HA monitor by specifying the IP address of the LAN adapter in the LAN status setup file of the HA monitor.

Specify the alias IP address in the following files:

- `server-identification-name.up` **file**

You use this file when connecting LAN. Specify the alias IP address to be added to the LAN adapter.

- `server-identification-name.down` **file**

You use this file when switching LAN. Specify the alias IP address to be deleted from the LAN adapter.

In `server-identification-name`, specify the value of `alias` of server-compliant environment settings (`servers` file).

Create a LAN status setup file for each of the executing nodes and the standby node.

(N number of) Executing nodes

Create the script for setting and deleting the alias IP address. Specify the settings in accordance with the alias IP address in each host.

Standby node (One node for recovery)

Prepare the `.up` file and `.down` file for the N number of executing node hosts. Specify a unique value for the logical network device in each of the N number of executing node hosts.

For details on setting the LAN status, see the manual *High-reliability System Monitoring Functionality HA Monitor*.

The following is an example of creating the file in Linux. For details on creating the file in other operating systems, see the manual *High-reliability System Monitoring Functionality HA Monitor*.

Example of the `server-identification-name.up` file

```
#!/bin/sh
/sbin/ifconfig eth0:1 inet 172.16.12.31 \
                        netmask 255.255.255.0 broadcast 10.209.112.255 up
/sbin/arping -U -c 2 -I eth0 172.16.12.31
```

Example of the `server-identification-name.down` file

```
#!/bin/sh
/sbin/ifconfig eth0:1 down
```

19.5.8 Setting Application Server

When setting Application Server, deploy Application Server in a cluster configuration using the management portal and commands of the HA monitor. Furthermore, set up the logical servers, distribute the configuration information, and then import J2EE applications and resource adapters. The following is the procedure to set Application Server:

1. In the standby node, start the J2EE server for recovery.

Using the `cjstartsv` command, start the J2EE server for recovery in the standby node (NodeR).

2. In the J2EE for recovery in the standby node, set the resource adapter.

In the standby node, import, deploy, and start a resource adapter with the same settings as all resource adapters that you are using in the executing nodes. However, you need not use the connection pooling functionality in the resource adapter to be defined in the J2EE server of the standby node. Specify 0 as the minimum and maximum value of the connection to be set with the server management commands.

Use server management commands to import, deploy and start the resource adapter. For the operations of the server management commands, see 3. *Basic Operations of Server Management Commands* in the *uCosminexus Application Server Application Setup Guide*.

3. When the setting of the resource adapter is complete, stop the J2EE server for recovery in the standby node.

Using the `cjstopsv` command, stop the J2EE server for recovery in the standby node (NodeR).

4. In the standby node, execute the `monbegin` command of the HA monitor, and set the spare-node Application Server to the standby state.

- **Command execution example in the standby node**

In the standby node (NodeR), execute `# monbegin server-identification-name` corresponding to the N number of executing nodes (Node1 to Node4), and set the standby node corresponding to the N number of executing nodes to the standby state.

5. Start Management Server.

Start the Management Server, and then start the Administration Agent, Management Server, and logical server (CORBA Naming Service for monitoring) on the Management Server. For details on how to start Management Server, see 4.1.1 *Procedure for starting a system* and 4.1.2 *Methods of starting a system* in the *uCosminexus Application Server Management Portal User Guide*.

6. In the executing nodes (Node1 to Node4), execute the `monbegin` command of the HA monitor, and start the executing node Application Servers as the executing nodes.

- **Command execution example in executing nodes**

In Node1 to Node4, execute `# monbegin server-identification-name`, and start each of the executing nodes. In this example, you execute the shell script file (`start.sh`), and start the Administration Agent of each executing node from Node1 to Node4.

7. With the management portal, set up the J2EE server of each executing node from Node1 to Node4.

Set up the J2EE server with the Setup screen of 'Configuration definition of management domain'. For details on the operating procedures and screens when using the management portal, see the *uCosminexus Application Server Management Portal User Guide*.

8. Using the management portal, distribute the information set in the J2EE servers to each executing node from Node1 to Node4.

Distribute the information set in the J2EE servers to each executing node from Node1 to Node4 with the Distribution of Setup Information screen of 'Environment setup of a logical server'.

9. With the management portal, start the logical server of each executing node.

In this example, as you do not start the logical server of the executing nodes in the shell script file (`start.sh`), start the logical servers using the management portal.

10. Use the server management commands to import the J2EE application and resource adapter in the J2EE server of each executing node from `Node1` to `Node4`. Set and start the imported J2EE application and resource adapter.

For the operations of the server management commands, see 3. *Basic Operations of Server Management Commands* in the *uCosminexus Application Server Application Setup Guide*.

Execute the server management commands after starting the J2EE server and its prerequisite processes. Once you have completed the operations with server management commands, stop the server management commands.

19.6 Starting and stopping the N-to-1 recovery system (In Windows)

This section describes the procedure for starting and stopping the system when you are using an N-to-1 recovery system.

When performing the operation using N-to-1 recovery system, you must specify beforehand the required environment settings beforehand, such as preparing the two hosts as the active node and spare node, and registering the script for monitoring, starting, and stopping the Administration Agent in which you run the cluster service. For details on how to specify the settings, see *19.4 Settings for the N-to-1 recovery system (In Windows)*.

Tip

When you use the N-to-1 recovery system, Management Server starts. In each Application-Server host, do not specify the settings for starting the Administration Agent together with the OS.

Reference note

When differentiating the nodes during operation, divide them into the 'executing node' and 'standby node'.

- **Executing node**
This is the node containing N number of Application Servers for executing the business processing.
- **Standby node**
This is the node containing a single recovery server.

19.6.1 Starting the N-to-1 recovery system

This subsection describes how to start the system when you are using the N-to-1 recovery system.

To use the N-to-1 recovery system, you must start the Management Server beforehand. Start the Management Server if you have not yet started. For details on how to start Management Server, see *4.1.1 Procedure for starting a system* and *4.1.2 Methods of starting a system* in the *uCosminexus Application Server Management Portal User Guide*.

Furthermore, in the N-to-1 recovery system, the node that you start first operates as the executing node and the node that you start later operates as the standby node. Always start the active node before the spare node.

The following is the procedure to start the N-to-1 recovery system:

1. From the **Start** menu, select **Control Panel - Performance and Maintenance - Management Tools**, and then select **Cluster Administrator**.
The Cluster Administrator starts.
2. In the console tree (left pane), select an active node, and then from the **File** menu, choose **Start Cluster Service**.
The cluster service of the selected active node starts. Execute this step for all active nodes.
3. In the console tree (left pane), select the spare node, and then from the **File** menu, choose **Start Cluster Service**.
The cluster service of the spare node starts.
4. In the console tree (left pane), select a resource group that contains an active node and the spare node, and then from the **File** menu, choose **Set to Online**.

The selected resource group goes online. Execute this step for all resource groups.

5. Start each logical server of the executing node host.

For details on how to start the logical servers, see *4.1.2 Methods of starting a system* in the *uCosminexus Application Server Management Portal User Guide*.

19.6.2 Stopping the N-to-1 recovery system

The following is the procedure to stop the N-to-1 recovery system:

1. From the **Start** menu, select **Control Panel - Performance and Maintenance - Management Tools**, and then select **Cluster Administrator**.

The Cluster Administrator starts.

2. In the console tree (left pane), select a resource group that contains an executing node and the standby node, and then from the **File** menu, choose **Set to Offline**.

The selected resource group goes offline. Execute this step for all resource groups.

3. In the console tree (left pane), select the spare node, and then from the **File** menu, choose **Stop Cluster Service**.

The cluster service of the spare node stops.

4. In the console tree (left pane), select an active node, and then from the **File** menu, choose **Stop Cluster Service**.

The cluster service of the selected active node stops. Execute this step for all active nodes.

Stop the Management Server manually at the end. For details on how to stop Management Server, see *4.1.3 Procedure for stopping a system* and *4.1.4 Methods of stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.

19.7 Starting and stopping the N-to-1 recovery system (In UNIX)

This section describes the procedure for starting and stopping a system when you are using an N-to-1 recovery system based on the HA monitor. Note that you can use the HA monitor only in AIX or Linux.

When performing the operation using the N-to-1 recovery system based on the HA monitor, you must specify the required environment settings beforehand, such as preparing the two hosts as the active node and spare node, and registering the script for monitoring, starting, and stopping the Administration Agent in which you run the cluster service. For details on how to specify the settings, see [19.5 Settings for the N-to-1 recovery system \(In UNIX\)](#).

Tip

When you use the N-to-1 recovery system based on the HA monitor, start Management Server, and start only the Administration Agent in each Application-Server host. In each Application-Server host, do not specify the settings for starting the Administration Agent together with the OS.

The following commands are the commands provided by the HA monitor. For details, see the manual *High-reliability System Monitoring Functionality HA Monitor*.

- `monbegin` (Starting a server that does not have an interface with the HA monitor)
- `monend` (Communication to stop a running server that does not have an interface with the HA monitor)

Reference note

When differentiating the nodes during operation of the HA monitor, divide them into the 'executing node' and 'standby node'.

- **Executing node**

This is the node containing N number of Application Servers for executing the business processing.

- **Standby node**

This is the node containing a single recovery server.

19.7.1 Starting the N-to-1 recovery system

This subsection describes the points to be considered during system startup when you are using the N-to-1 recovery system based on the HA monitor, and also describes the procedure for starting the system.

(1) Points to consider when starting the system

The points to be considered during system startup when you are using the N-to-1 recovery system based on the HA monitor are as follows:

- The HA monitor deployed on the executing node and standby node host starts simultaneously with the OS.
- Start Management Server if you have not yet started. For details on how to start Management Server, see [4.1.1 Procedure for starting a system](#) and [4.1.2 Methods of starting a system](#) in the *uCosminexus Application Server Management Portal User Guide*.

(2) Procedure for starting the system

The following is the procedure to start the system when you are using the N-to-1 recovery system based on the HA monitor:

1. In the standby node host, execute the `monbegin` command corresponding to each executing node N number of times.

```
# monbegin server-identification-name
```

In the underlined part, specify the identification name of the executing node server specified in the operand `alias` in the `servers` file.

By doing this, the standby node recovery server starts monitoring the executing nodes.

2. In each of the N number of executing node hosts, execute the `monbegin` command, and start the Administration Agent of each executing node host.

```
# monbegin server-identification-name
```

In the underlined part, specify the identification name of the server specified in the operand `alias` in the `servers` file.

By doing this, the Administration Agent of each executing node host starts.

Important note

When you have not specified the settings for batch start of the logical servers in the management domain during startup of Management Server, start the logical server of each executing node host. When you have specified the settings for batch start of the logical servers in the management domain during startup of Management Server, you need not start the logical servers manually. For starting the logical servers, see *4.1.2 Methods of starting a system* in the *uCosminexus Application Server Management Portal User Guide*.

Reference note

For details on defining the `servers` file (setting the server environment with the HA monitor), see *19.5.6 Setting the server-compliant environment*.

19.7.2 Stopping the N-to-1 recovery system

The following is the procedure to stop the system when you are using N-to-1 recovery system based on the HA monitor:

1. When you have not specified the settings for stopping the logical servers of the management domain in the script for stopping the Administration Agent, stop each logical server in the executing node host.

This step is not required when you have specified the settings for stopping the logical servers in the script for stopping the Administration Agent. For details on stopping the logical servers, see *4.1.4 Methods of stopping a system* in the *uCosminexus Application Server Management Portal User Guide*.

2. In each of the N number of executing node hosts, execute the `monend` command, and stop the Administration Agent of each executing node host.

```
# monend server-identification-name
```

In the underlined part, specify the identification name of the server specified in the operand `alias` in the `servers` file.

By doing this, the Administration Agent of each executing node host stops.



Reference note

For details on defining the `servers` file (setting the server environment with the HA monitor), see [19.5.6 Setting the server-compliant environment](#).

Stop Management Server manually at the end. For details on stopping Management Server, see [4.1.3 Procedure for stopping a system](#) and [4.1.4 Methods of stopping a system](#) in the *uCosminexus Application Server Management Portal User Guide*.



Appendixes

A. Settings of Cosminexus TPBroker for Integrating Cluster Software (in Windows)

You can use the cluster software (Windows Server Failover Cluster) to operate Cosminexus TPBroker in a cluster configuration.

With Cosminexus TPBroker, the following are the causes of node switching:

- Detection of system failure such as hardware failure by cluster services
- Planned node switching by the user

A.1 Starting and stopping a system

Determine the Cosminexus TPBroker functionality and environment settings required for configuring the node switching system based on the system requirements.

First, determine whether you want to use the following functionality:

- ORB functionality
- OTS functionality

To use the OTS functionality, you use a shared disk.

(1) ORB functionality

To operate Smart Agent and the processes connected to Smart Agent with the cluster software (Windows Server Failover Cluster), you must set up various IP addresses related to the ORB functionality. For details, see [A.2 Settings for using the ORB functionality](#).

The processes connected to the Smart Agent are the CORBA Naming Service, CTM daemon, and the J2EE server[#] that uses the Smart Agent.

Indicates that `true` is specified in the `vbroker.agent.enableLocator` key of the `usrconf.properties` file, and the J2EE server is started.

(2) OTS functionality

Start the transaction service in the in-process.

A.2 Settings for using the ORB functionality

This section describes the settings for using the ORB functionality.

You specify these settings on the host where Windows Server Failover Cluster will run when you execute the following processes:

- Smart Agent

- CORBA Naming Service

To set up Smart Agent as a failover target, register the Smart Agent as a 'General-purpose application' in Windows Server Failover Cluster.

(1) Settings for localaddr file and htc.clienthandleraddr file of the Smart Agent

For starting the Smart Agent on cluster software, you set up the `localaddr` file and `htc.clienthandleraddr` file of each node. Note that the settings are different for the multi-homed host environment and for the non multi-homed host environment.

The following table describes the settings for the `localaddr` file of the Smart Agent when a cluster environment is built on a multi-homed host environment or a multi-homed host environment.

Table A–1: Settings for the `localaddr` file of the Smart Agent

Host environment on which the cluster service and Smart Agent are running	Settings for the <code>localaddr</code> file of the Smart Agent	
	When Smart Agent is set as failover target	When Smart Agent is not set as failover target
Multi-homed host environment	<ul style="list-style-type: none"> • Fixed IP address (primary IP address) • IP address that you want to explicitly specify in the Smart Agent • Cluster IP address 	<ul style="list-style-type: none"> • Fixed IP address (primary IP address) • IP address that you want to explicitly specify in the Smart Agent
Non multi-homed host environment	<ul style="list-style-type: none"> • Fixed IP address (primary IP address) • Cluster IP address 	<ul style="list-style-type: none"> • Fixed IP address (primary IP address)

The following points describe the detailed settings for each environment. The details on the `htc.clienthandleraddr` file are also described.

(a) Multi-homed host environment

- **When Smart Agent is set as a target for failover**

Make sure that you specify settings for the fixed IP address (primary IP address), IP address that you want to explicitly specify in the Smart Agent[#], and the cluster IP address used by the Smart Agent in the `localaddr` file.

Also, you define the `htc.clienthandleraddr` file in such a manner so that the cluster IP address is returned for the processes connected to the Smart Agent.

- **When Smart Agent is not set as a target for failover**

Make sure that you specify settings for the fixed IP address (primary IP address) and the IP address that you want to explicitly specify in the Smart Agent[#] in the `localaddr` file. You do not specify the cluster IP address.

Also, you define the `htc.clienthandleraddr` file in such a manner so that the fixed IP address is returned for the processes connected to the Smart Agent.

#

If you specify the `localaddr` file, the Smart Agent will only recognize the IP address specified in the `localaddr` file. If you do not specify the `localaddr` file, the Smart Agent will recognize the IP addresses that can be acquired from `gethostbyname()` by default.

(b) Non multi-homed host environment

- **When Smart Agent is set as a target for failover**

Make sure that you specify settings for the fixed IP address (primary IP address) and the cluster IP address used by the Smart Agent in the `localaddr` file.

Also, you define the `htc.clienthandleraddr` file in such a manner so that the cluster IP address is returned for the processes connected to the Smart Agent.

- **When Smart Agent is not set as a target for failover**

Make sure that you specify settings for the fixed IP address (primary IP address) in the `localaddr` file. You do not specify the cluster IP address.

Also, you define the `htc.clienthandleraddr` file in such a manner so that the fixed IP address is returned for the processes connected to the Smart Agent.

(2) Settings for the `agentaddr` file, `vbroker.agent.addr` property, and environment variable `OSAGENT_ADDR` of the Smart Agent or processes connected to the Smart Agent

For communication between Smart Agents exist on different network domains or for communication between the Smart Agent and the processes connected to the Smart Agent present on different network domains, you must specify the settings for the Smart Agents or for the processes connected to the Smart Agents. Specify the settings in the `agentaddr` file, `vbroker.agent.addr` property, and the environment variable `OSAGENT_ADDR` of each node.

(a) Settings of the Smart Agents

For communications between the Smart Agents exist on different network domains and for setting up the Smart Agent as a failover target, you make sure to specify the cluster IP address in the `agentaddr` file of the Smart Agent that communicates with the target Smart Agent.

(b) Settings in the processes connected to Smart Agents

For starting a Smart Agent and the processes connected to the Smart Agent on different network domains and for setting a Smart Agent as a failover target, you make sure to specify the cluster IP address of the Smart Agent that becomes the failover target in either of the `agentaddr` file of the processes connected to the Smart Agent, `vbroker.agent.addr` property, or the environment variable `OSAGENT_ADDR`.

(3) Settings for the CORBA Naming Service

For setting the CORBA Naming Service as the failover target, you specify the start property of the CORBA Naming Service.

You set up the following start property in the `nameserv` command, and start the CORBA Naming Service:

```
-J-Dvbroker.se.iiop_tp.host=cluster-IP-address-or-cluster-network-name
```

Also, you apply the cluster IP address or cluster network specified in this property in the following key settings of the `usrconf.properties` (user property file for J2EE server) file:

- IP address or host name specified in the `ejbserver.naming.host` key
- Provider URL specified in the `ejbserver.jndi.namingservice.group.specify-group-name.providerurls` key

For details on the `usrconf.properties` file, see 2.2.3 *usrconf.properties (User property file for J2EE servers)* in the *uCosminexus Application Server Definition Reference Guide*. For starting the CORBA Naming Service using

the `nameserv` command, see *Appendix C.2 How to start a system* in the *uCosminexus Application Server Command Reference Guide*.

(4) Settings for the `vbroker.se.iiop_tp.host` key and `vbroker.se.iiop_tp.scm.iiop_tp.listener.port` key of J2EE server

To set up the J2EE server as the failover target, specify the cluster IP address in the `vbroker.se.iiop_tp.host` key, and start the J2EE server. Furthermore, use and manage the `vbroker.se.iiop_tp.scm.iiop_tp.listener.port` key of the `usrconf.properties` file in such a manner so that the same port number can be used for the port number that the J2EE server uses in all the nodes.

(5) Settings for the environment variables and options

To start the Smart Agent in the cluster environment, the following environment variable and option settings must be specified in the Smart Agent.

Specify the options only when you want to set the Smart Agent as a failover target.

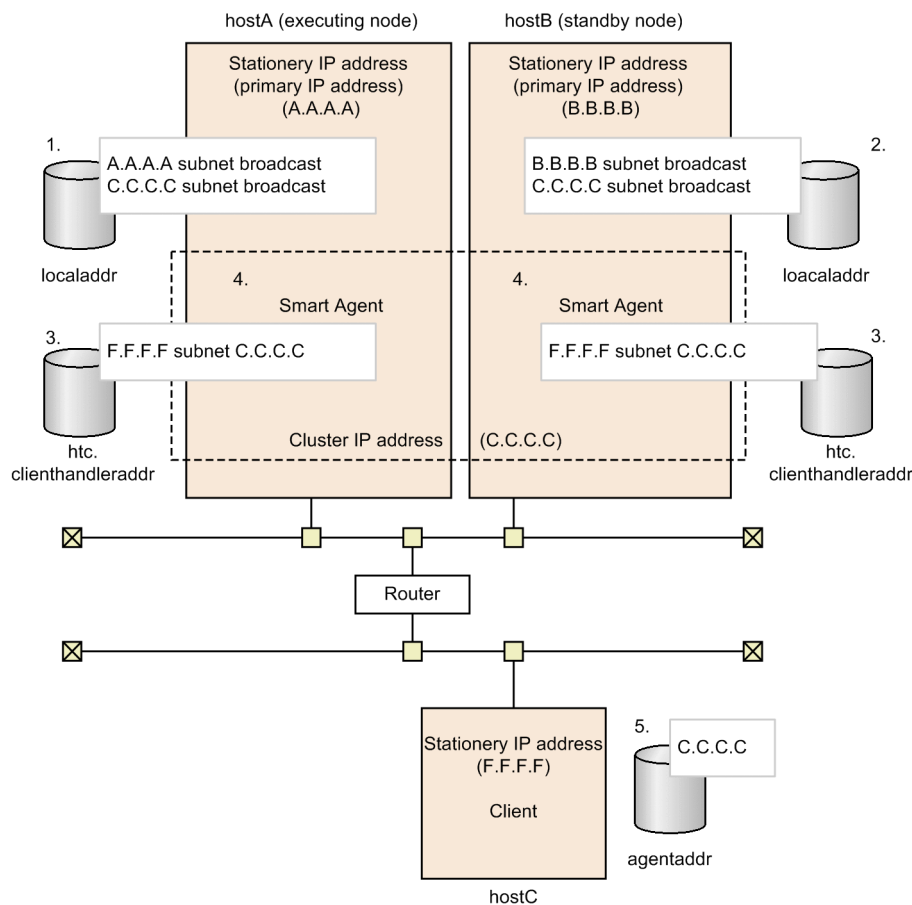
- Environment variable name: `OSAGENT_CLIENT_HANDLER_PORT`
- Option: `-m`

(6) Setting examples

The following figure shows an example of the IP address settings:

Setting example 1

Figure A–1: Example of the IP address settings when the Smart Agent is set as a target for failover

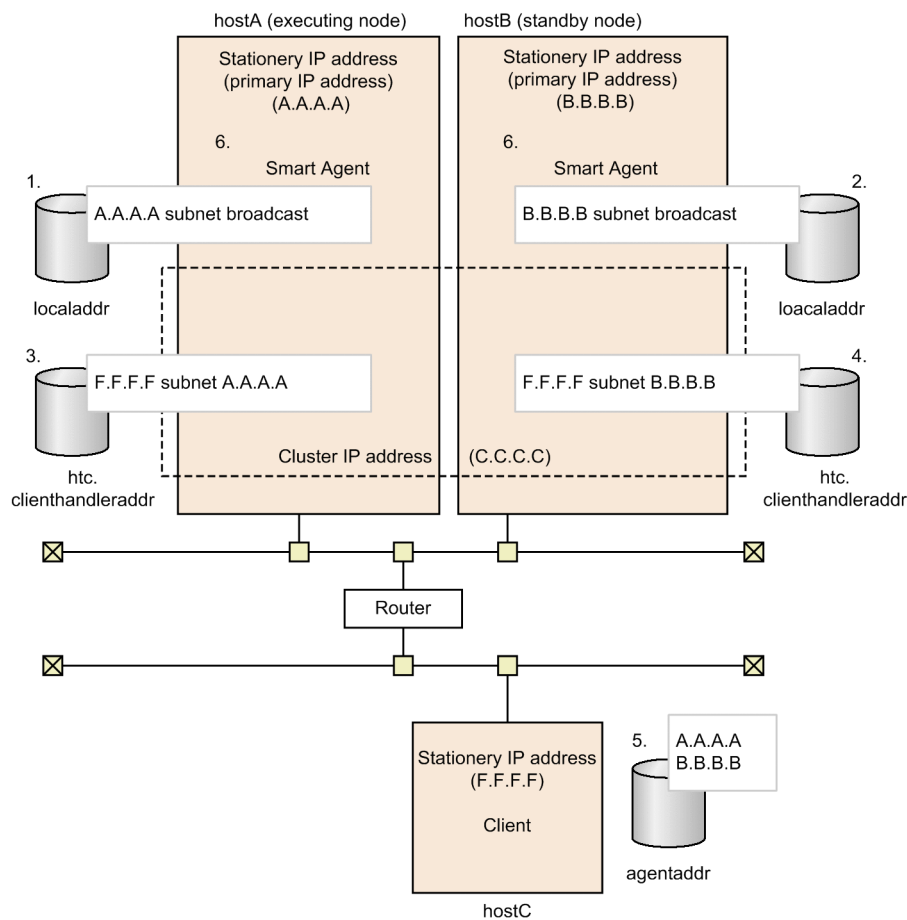


The points from 1 to 5 in the figure are as follows:

1. Specify 'hostA fixed IP address (A . A . A . A) subnet broadcast' and 'cluster IP address (C . C . C . C) subnet broadcast' in the `localaddr` file of hostA.
2. Specify ' hostB fixed IP address (B . B . B . B) subnet broadcast' and 'cluster IP address (C . C . C . C) subnet broadcast' in the `localaddr` file of hostB.
3. Specify 'Fixed IP address of client (F . F . F . F) subnet cluster IP address (C . C . C . C)' in the `htc.clienthandleraddr` file of hostA and hostB.
4. Specify the environment variable 'OSAGENT_CLIENT_HANDLER_PORT' and `-m` option in the Smart Agent started on hostA and hostB.
5. Specify the cluster IP address (C . C . C . C) in the `agentaddr` file of hostC.

Setting example 2

Figure A–2: Example of the IP address settings when the Smart Agent is not set as a target for failover



The points 1 to 5 in the figure are as follows:

1. Specify 'hostA fixed IP address (A . A . A . A) subnet broadcast' in the `localaddr` file of hostA.
2. Specify 'hostB fixed IP address (B . B . B . B) subnet broadcast' in the `localaddr` file of hostB.
3. Specify 'Client's fixed IP address (F . F . F . F) subnet hostA stationary IP address (A . A . A . A)' in the `htc.clienthandleraddr` file of hostA.
4. Specify 'Client's fixed IP address (F . F . F . F) subnet hostB stationary IP address (B . B . B . B)' in the `htc.clienthandleraddr` file of hostB.
5. Specify A . A . A . A and B . B . B . B in the `agentaddr` file of hostC.
6. Specify the environment variable `OSAGENT_CLIENT_HANDLER_PORT` in the Smart Agent running on hostA and hostB.

B. Settings of Cosminexus TPBroker for Integrating Cluster Software (In UNIX)

You can use the cluster software (HA monitor) to operate Cosminexus TPBroker in a cluster configuration.

With Cosminexus TPBroker, the following are the causes for node switching:

- Detection of system failure such as hardware failure by cluster services
- Planned node switching by user

B.1 Starting and stopping a system

You determine the Cosminexus TPBroker functionality and environment settings required for building the node switching based on the system requirements.

First, determine whether you want to use the following functionality:

- ORB functionality
- OTS functionality

To use the OTS functionality, you use a shared disk.

(1) ORB functionality

For operating a Smart Agent and the processes connected to Smart Agent with the cluster software (HA monitor), you must set up various IP addresses related to the ORB functionality. For details, see [B.2 Settings for using the ORB functionality](#).

The processes connected to the Smart Agent are the CORBA Naming Service, CTM daemon, and the J2EE server[#] that uses the Smart Agent.

#

Indicates that the `true` is specified in the `vbroker.agent.enableLocator` key of `usrconf.properties`, and J2EE server is started.

(2) OTS functionality

Start the transaction service in the in process.

(3) Notes for operations in the cluster configuration

Note the following points for operating Cosminexus TPBroker in the cluster configuration:

- Store the program suite containing Cosminexus TPBroker on the local disk of each node.
- Unify the Cosminexus TPBroker versions on all the nodes.
- Unify the system environment definition of Cosminexus TPBroker on all the nodes.
- If the port number used by the J2EE application is fixed, you make sure that the same port number can be used in all the nodes.

B.2 Settings for using the ORB functionality

This section describes the settings for using the ORB functionality.

You specify these settings on the host where the HA monitor will run when you execute the following processes:

- Smart Agent
- CORBA Naming Service

(1) Settings for the `localaddr` file and `htc.clienthandleraddr` file of the Smart Agent

For starting the Smart Agent on cluster software, you set up the `localaddr` file and `htc.clienthandleraddr` file of each node. Note that the settings are different for the multi-homed host environment and for non multi-homed host environment.

The following table describes the settings for the `localaddr` file of the Smart Agent when a cluster environment is built on a multi-homed host environment or a non multi-homed host environment.

Table B–1: Settings for the `localaddr` file of the Smart Agent

Host environment on which the cluster service and Smart Agent are running	Settings for the <code>localaddr</code> file of the Smart Agent	
	When Smart Agent is set as the target for node switching	When Smart Agent is not set as the target for node switching
Multi-homed host environment	<ul style="list-style-type: none">• Fixed IP address (primary IP address)• IP address that you want to explicitly specify in the Smart Agent• Alias IP address	<ul style="list-style-type: none">• Fixed IP address (primary IP address)• IP address that you want to explicitly specify in the Smart Agent
Non multi-homed host environment	<ul style="list-style-type: none">• IP address (primary IP address)• Alias IP address	<ul style="list-style-type: none">• IP address (primary IP address)

The following points describe the detailed settings for each environment. The details on the `htc.clienthandleraddr` file are also described.

(a) Multi-homed host environment

• When Smart Agent is set as the target for node switching

Make sure that you specify settings for the fixed IP address (primary IP address), IP address that you want to explicitly specify in the Smart Agent[#], and the Alias IP address used by the Smart Agent in the `localaddr` file.

Also, you define the `htc.clienthandleraddr` file in such a manner so that the Alias IP address is returned for the processes connected to the Smart Agent.

• When Smart Agent is not set as the target for node switching

Make sure that you specify settings for the fixed IP address (primary IP address) and the IP address that you want to explicitly specify in the Smart Agent[#] in the `localaddr` file. You do not specify the Alias IP address.

Also, you define the `htc.clienthandleraddr` file in such a manner so that the fixed IP address is returned for the processes connected to the Smart Agent.

#

If you specify the `localaddr` file, the Smart Agent will only recognize the IP address specified in the `localaddr` file. If you do not specify the `localaddr` file, the Smart Agent will recognize the IP addresses that can be acquired from `gethostbyname()` by default. Note that you can specify the `-v` option in the

`osagent` command, set the `verbose` mode to `ON`, and then start the Smart Agent for checking the IP address recognized by the Smart Agent

(b) Non multi-homed host environment

- **When Smart Agent is set as the target for node switching**

Make sure that you specify settings for the fixed IP address (primary IP address) and the Alias IP address used by the Smart Agent in the `localaddr` file.

Also, you define the `htc.clienthandleraddr` file in such a manner so that the Alias IP address is returned for the processes connected to the Smart Agent.

- **When Smart Agent is not set as the target for node switching**

Make sure that you specify settings for the fixed IP address (primary IP address) in the `localaddr` file. You do not specify the Alias IP address.

Also, you define the `htc.clienthandleraddr` file in such a manner so that the stationary IP address is returned for the processes connected to the Smart Agent.

(2) Settings for the `agentaddr` file, `vbroker.agent.addr` property, and environment variable `OSAGENT_ADDR` of the Smart Agent or processes connected to the Smart Agent

For communication between Smart Agents exist on different network domains or for communication between the Smart Agent and the processes connected to the Smart Agent present on different network domains, you must specify the settings for the Smart Agents or for the processes connected to the Smart Agents. Specify the settings in the `agentaddr` file, `vbroker.agent.addr` property, and the environment variable `OSAGENT_ADDR` of each node.

(a) Settings of the Smart Agents

For communications between the Smart Agents exist on different network domains and for setting up the Smart Agent as a target for node switching, you make sure to specify the Alias IP address in the `agentaddr` file of the Smart Agent that communicates with the target Smart Agent.

(b) Settings in the processes connected to Smart Agents

For starting a Smart Agent and the processes connected to the Smart Agent on different network domains and for setting a Smart Agent as a target for node switching, you make sure to specify the Alias IP address of the Smart Agent that becomes the target of node switching in either of the `agentaddr` file of the processes connected to the Smart Agent, `vbroker.agent.addr` property, or the environment variable `OSAGENT_ADDR`.

(3) Settings for the CORBA Naming Service

For setting the CORBA Naming Service as the target for node switching, you specify the start property of the CORBA Naming Service.

You set up the following start property in the `nameserv` command, and start the CORBA Naming Service:

```
-J-Dvbroker.se.iiop_tp.host = Alias-IP-address or Alias-host-name
```

Also, you apply the Alias IP address or Alias host name specified in this property in the following key settings of the `usrconf.properties` (user property file for J2EE server) file:

- IP address or host name specified in the `ejbserver.naming.host` key

- Provider URL specified in the `ejbserver.jndi.namingservice.group.Specify-group-name.providerurls` key.

For details on the `usrconf.properties` file, see 2.2.3 *usrconf.properties (User property file for J2EE servers)* in the *uCosminexus Application Server Definition Reference Guide*. For starting the CORBA Naming Service using the `nameserv` command, see *Appendix C.2 How to start a system* in the *uCosminexus Application Server Command Reference Guide*.

(4) Settings for the `vbroker.se.iiop_tp.host` key and `vbroker.se.iiop_tp.scm.iiop_tp.listener.port` key of J2EE server

To set up the J2EE server as the failover target, specify the cluster IP address in the `vbroker.se.iiop_tp.host` key and start the J2EE server. Furthermore, use and manage the `vbroker.se.iiop_tp.scm.iiop_tp.listener.port` key of the `usrconf.properties` file in such a manner so that the same port number can be used for the port number that the J2EE server uses in all the nodes.

(5) Settings for the environment variables

To start the Smart Agent in the cluster environment, the following environment variable settings must be specified in the Smart Agent:

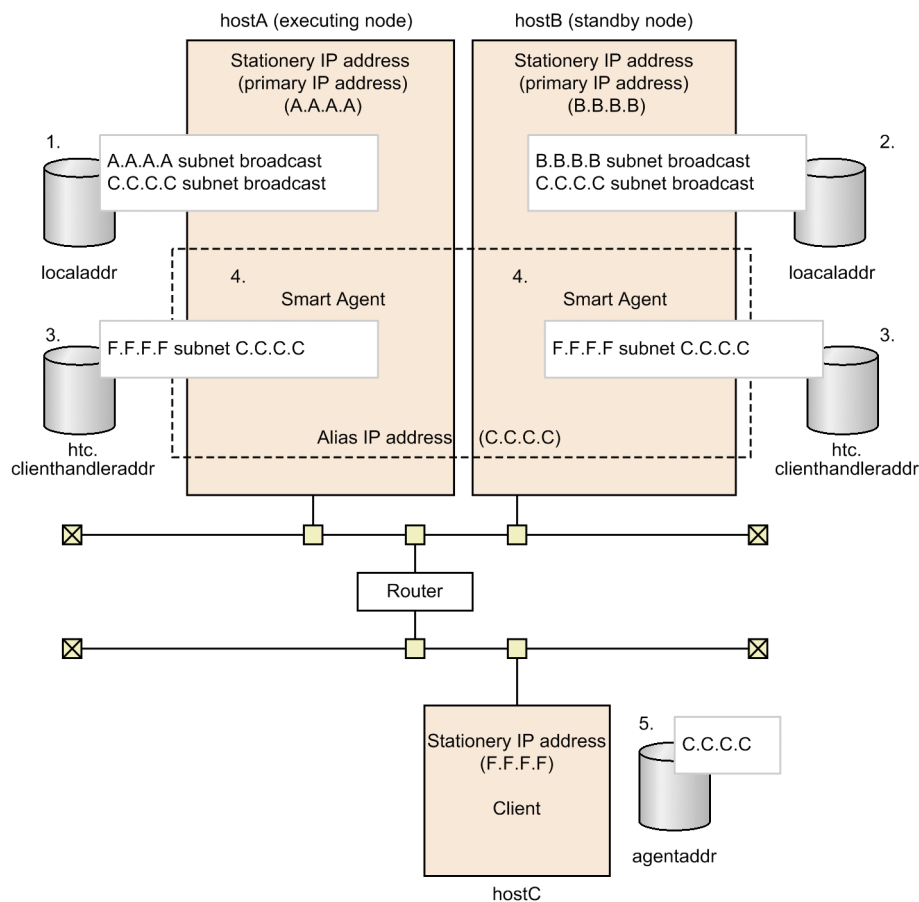
- Environment variable name: `OSAGENT_CLIENT_HANDLER_PORT`

(6) Setting examples

The following figure shows an example of the IP address settings:

Setting examples 1

Figure B–1: Example of the IP address settings when the Smart Agent is set as a target for node switching

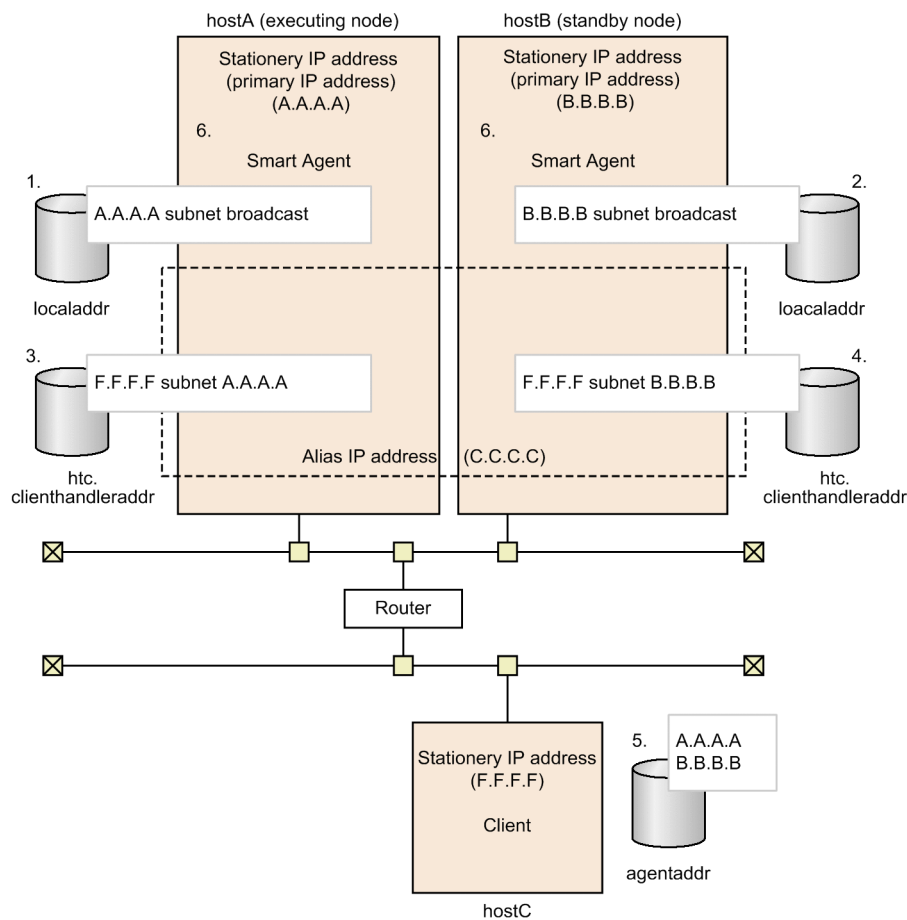


The points 1 to 5 in the figure are as follows:

1. Specify 'hostA fixed IP address (A . A . A . A) subnet broadcast' and Alias IP address (C . C . C . C) subnet broadcast' in the localaddr file of hostA.
2. Specify 'hostB fixed IP address (B . B . B . B) subnet broadcast' and Alias IP address (C . C . C . C) subnet broadcast' in the localaddr file of hostB.
3. Specify 'Fixed IP address of client (F . F . F . F) subnet Alias IP address (C . C . C . C)' in the htc.clienthandleraddr file of hostA and hostB.
4. Specify the environment variable 'OSAGENT_CLIENT_HANDLER_PORT' in the Smart Agent started on hostA and hostB.
5. Specify the Alias IP address (C . C . C . C) in the agentaddr file of hostC.

Setting example 2

Figure B–2: Example of the IP address settings when Smart Agent is not set as a target for node switching



The points 1 to 6 in the figure are as follows:

1. Specify 'hostA fixed IP address (A . A . A . A) subnet broadcast' in the `localaddr` file of hostA.
2. Specify 'hostB fixed IP address (B . B . B . B) subnet broadcast' in the `localaddr` file of hostB.
3. Specify 'Fixed IP address of client (F . F . F . F) subnet hostA fixed IP address (A . A . A . A)' in the `htc.clienthandleraddr` file of hostA.
4. Specify 'Fixed IP address of client (F . F . F . F) subnet hostB fixed IP address (B . B . B . B)' in the `htc.clienthandleraddr` file of hostB.
5. Specify A . A . A . A and B . B . B . B in the `agentaddr` file of hostC.
6. Specify the environment variable `OSAGENT_CLIENT_HANDLER_PORT` in the Smart Agent that is running on hostA and hostB.

C. Contents of the Protected Area List

The following are the contents of the protected area list:

```
#
# DO NOT EDIT THIS FILE.
#
# All Rights Reserved. Copyright (C) 2004, 2015, Hitachi, Ltd.

#-----
# Cosminexus Component Container
#-----
com.hitachi.software.auditlog.*
com.hitachi.software.ejb.*
com.hitachi.software.jpa.*
com.sun.ejb.*
com.sun.enterprise.*
com.sun.web.*
com.sun.activation.*
com.sun.mail.*

com.hitachi.software.javamail.*
com.hitachi.software.web.*
com.hitachi.software.was.logger.*
com.hitachi.software.was.tracer.*
com.hitachi.software.was.web.*
com.hitachi.software.was.sfo.*
org.apache.ajp.*
org.apache.catalina.*
org.apache.coyote.*
org.apache.jasper.*
org.apache.jk.*
org.apache.naming.*
org.apache.tomcat.*

com.cosminexus.cc.lib.*

#-----
# HiRDB Type4 JDBC Driver
#-----
JP.co.Hitachi.soft.HiRDB.JDBC.*

#-----
# Oracle JDBC Thin Driver
#-----
oracle.*

#-----
# DB2 Universal JDBC Driver
#-----
com.ibm.db2.*
COM.ibm.db2.*
COM.ibm.db2os390.*
sqlj.runtime.*

#-----
# Microsoft SQL Server JDBC Driver
```

```

#-----
com.microsoft.sqlserver.jdbc.*
microsoft.sql.*

#-----
# PostgreSQL JDBC Driver
#-----
org.postgresql.*

#-----
# MySQL Connector/J
#-----
com.mysql.*
org.gjt.mm.mysql.*

#-----
# HNTRLib
#-----
jp.co.hitachi.soft.hntrlib2.*
jp.co.hitachi.soft.hntrlibM.*

#-----
# Cosminexus TPBroker (ORB)
#-----
com.inprise.vbroker.*
com.borland.security.*
com.borland.vbroker.*
org.omg.BiDirPolicy.*
org.omg.CORBA.*
org.omg.CORBA_2_3.*
org.omg.CosEventChannelAdmin.*
org.omg.CosEventComm.*
org.omg.CosNaming.*
org.omg.CosNotification.*
org.omg.CosNotifyChannelAdmin.*
org.omg.CosNotifyComm.*
org.omg.CosNotifyFilter.*
org.omg.CosTransactions.*
org.omg.CosTypedEventChannelAdmin.*
org.omg.CosTypedEventComm.*
org.omg.CosTypedNotifyChannelAdmin.*
org.omg.CosTypedNotifyComm.*
org.omg.Dynamic.*
org.omg.DynamicAny.*
org.omg.Firewall.*
org.omg.IOP.*
org.omg.MessageRouting.*
org.omg.Messaging.*
org.omg.PortableInterceptor.*
org.omg.PortableServer.*
org.omg.SendingContext.*
org.omg.TimeBase.*
javax.rmi.*

#-----
# Cosminexus TPBroker (OTS)
#-----
COM.Hitachi.software.TPBroker.OTS.*

```

```
COM.Hitachi.software.TPBroker.otsinprocess.*
org.omg.CosTransactions.*
org.omg.CosTSInteroperation.*
```

```
#-----
# Cosminexus Performance Tracer
#-----
```

```
com.hitachi.software.ejb.prf.*
com.hitachi.software.ejb.ctm.*
JP.co.Hitachi.soft.CPRF.*
```

```
#-----
# Cosminexus Manager
#-----
```

```
com.cosminexus.mngsvr.*
com.cosminexus.admin.*
com.cosminexus.manager.*
```

```
#-----
# Cosminexus DABroker Library
#-----
```

```
JP.co.Hitachi.soft.DBPSV_Driver.*
```

```
#-----
# Cosminexus Component Library
#-----
```

```
com.cosminexus.cwc.*
```

```
#-----
# TP1/Message Queue - Access
#-----
```

```
jp.co.Hitachi.soft.mqadaptor.*
```

```
#-----
# Cosminexus Reliable Messaging
#-----
```

```
jp.co.Hitachi.soft.reliablemessaging.*
jp.co.Hitachi.soft.reliablemessaging_np.*
```

```
#-----
# Cosminexus TP1 Connector
#-----
```

```
jp.co.hitachi_system.tplconnector.*
JP.co.Hitachi.soft.OpenTP1.*
```

```
#-----
# Cosminexus Service Coordinator
#-----
```

```
jp.co.Hitachi.soft.csc.*
jp.co.Hitachi.soft.csciw.*
com.cosminexus.ftp.*
com.cosminexus.csc.monitor.*
```

```
#-----
# Cosminexus SOAP
#-----
```

```
com.cosminexus.cws.*
com.cosminexus.c4web.*
```

```
org.apache.axis.types.*
org.apache.commons.discovery.tools.*
javax.xml.soap.*
javax.xml.rpc.*
```

```
#-----
# Cosminexus Web Services - Security
#-----
com.cosminexus.wss.*
```

```
#-----
# Cosminexus JAX-WS (CJW)
#-----
com.cosminexus.xml.ws.*
com.cosminexus.tools.ws.*
com.cosminexus.istack.ws.*
com.cosminexus.org.jvnet.fastinfoset.*
com.cosminexus.xml.fastinfoset.*
com.cosminexus.org.apache.xml.internal.resolver.*
com.cosminexus.xml.messaging.saa.j.*
com.cosminexus.xml.stream.buffer.*
org.jvnet.mimepull.*
com.cosminexus.org.glassfish.external.*
com.cosminexus.org.glassfish.gmbal.*
javax.xml.ws.*
com.sun.xml.ws.*
com.cosminexus.wsrn.*
com.cosminexus.wsit.*
```

```
#-----
# Cosminexus JAX-RS (CJR)
#-----
com.cosminexus.jersey.*
com.cosminexus.ws.rs.*
com.sun.research.ws.wadl.*
com.cosminexus.org.codehaus.jackson.*
com.cosminexus.org.codehaus.jettison.*
javax.ws.rs.*
```

```
#-----
# uCosminexus Hitachi Code Conversion
#-----
JP.co.Hitachi.soft.codeconv.*
```

```
#-----
# Cosminexus JMS Provider
#-----
com.cosminexus.jmsprovider.*
```

```
#-----
# Cosminexus JSF/JSTL/BV
#-----
javax.faces.*
com.sun.faces.*
com.hitachi.software.faces.*
org.apache.taglibs.*
org.hibernate.validator.*
```

```

#-----
# uCosminexus Elastic Application Data Store
#-----
com.hitachi.software.eads.*

#-----
# Cosminexus CDI
#-----
com.hitachi.software.cdi.*

#-----
# J2SE
#-----
JP.co.Hitachi.soft.jvm.*
com.sun.corba.se.*
com.sun.crypto.provider.*
com.sun.image.*
com.sun.imageio.*
com.sun.inputmethods.internal.indicim.DevanagariInputMethodDescriptor
com.sun.inputmethods.internal.thaiim.ThaiInputMethodDescriptor
com.sun.java.browser.dom.DOMService
com.sun.java.swing.*
com.sun.java.util.jar.pack.Attribute
com.sun.java.util.jar.pack.Coding
com.sun.java.util.jar.pack.ConstantPool
com.sun.java.util.jar.pack.Constants
com.sun.java.util.jar.pack.Instruction
com.sun.java.util.jar.pack.NativeUnpack
com.sun.jdi.Bootstrap
com.sun.jlex.internal.Main
com.sun.jmx.interceptor.DefaultMBeanServerInterceptor
com.sun.jmx.mbeanserver.BaseMetaDataImpl
com.sun.jmx.mbeanserver.ClassLoaderRepositorySupport
com.sun.jmx.mbeanserver.DynamicMetaDataImpl
com.sun.jmx.mbeanserver.JmxMBeanServer
com.sun.jmx.mbeanserver.MBeanInstantiatorImpl
com.sun.jmx.mbeanserver.MetaDataImpl
com.sun.jmx.mbeanserver.RepositorySupport
com.sun.jmx.mbeanserver.StandardMBeanMetaDataImpl
com.sun.jmx.mbeanserver.StandardMetaDataImpl
com.sun.jmx.remote.internal.ArrayNotificationBuffer
com.sun.jmx.remote.internal.ClientCommunicatorAdmin
com.sun.jmx.remote.internal.ClientCommunicatorAdmin$Checker
com.sun.jmx.remote.internal.ClientNotifForwarder
com.sun.jmx.remote.internal.ClientNotifForwarder$LinearExecutor
com.sun.jmx.remote.internal.ClientNotifForwarder$LinearExecutor$1
com.sun.jmx.remote.internal.ClientNotifForwarder$NotifFetcher
com.sun.jmx.remote.internal.ServerCommunicatorAdmin
com.sun.jmx.remote.internal.ServerCommunicatorAdmin$Timeout
com.sun.jmx.remote.internal.ServerNotifForwarder
com.sun.jmx.remote.security.MBeanServerFileAccessController
com.sun.jmx.remote.security.SubjectDelegator
com.sun.jmx.remote.util.ClassLogger
com.sun.jmx.remote.util.EnvHelp
com.sun.jmx.snmp.IPAcl.ASCII_CharStream
com.sun.jmx.snmp.IPAcl.Parser
com.sun.jmx.snmp.IPAcl.SnmpAcl
com.sun.jmx.snmp.SnmpCounter64

```



```

com.sun.jmx.snmp.SnmpInt
com.sun.jmx.snmp.SnmpNull
com.sun.jmx.snmp.SnmpOid
com.sun.jmx.snmp.SnmpOidTableSupport
com.sun.jmx.snmp.SnmpParameters
com.sun.jmx.snmp.SnmpPeer
com.sun.jmx.snmp.SnmpString
com.sun.jmx.snmp.SnmpVarBind
com.sun.jmx.snmp.SnmpVarBindList
com.sun.jmx.snmp.Timestamp
com.sun.jmx.snmp.agent.SnmpMibTable
com.sun.jmx.snmp.agent.SnmpTableSupport
com.sun.jmx.snmp.daemon.CommunicatorServer
com.sun.jmx.snmp.daemon.SendQ
com.sun.jmx.snmp.daemon.SnmpAdaptorServer
com.sun.jmx.snmp.daemon.SnmpInformRequest
com.sun.jmx.snmp.daemon.SnmpRequestCounter
com.sun.jmx.snmp.daemon.SnmpResponseHandler
com.sun.jmx.snmp.daemon.SnmpSendServer
com.sun.jmx.snmp.daemon.SnmpSession
com.sun.jmx.snmp.daemon.SnmpSocket
com.sun.jmx.snmp.daemon.SnmpTimerServer
com.sun.jmx.snmp.daemon.WaitQ
com.sun.jmx.snmp.internal.SnmpEngineImpl
com.sun.jmx.snmp.tasks.ThreadService
com.sun.jmx.snmp.tasks.ThreadService$ExecutorThread
com.sun.jmx.trace.Trace
com.sun.jndi.dns.DnsClient
com.sun.jndi.dns.DnsContext
com.sun.jndi.dns.ZoneNode
com.sun.jndi.ldap.Connection
com.sun.jndi.ldap.EventQueue
com.sun.jndi.ldap.EventSupport
com.sun.jndi.ldap.LdapClient
com.sun.jndi.ldap.LdapCtx
com.sun.jndi.ldap.LdapRequest
com.sun.jndi.ldap.LdapSchemaCtx$SchemaInfo
com.sun.jndi.ldap.pool.ConnectionDesc
com.sun.jndi.ldap.pool.Connections
com.sun.jndi.ldap.pool.Pool
com.sun.jndi.ldap.pool.PoolCleaner
com.sun.jndi.toolkit.corba.CorbaUtils
com.sun.management.OSMBeanFactory
com.sun.management.OperatingSystem
com.sun.management.jmx.Introspector
com.sun.media.*
com.sun.naming.internal.FactoryEnumeration
com.sun.naming.internal.ResourceManager
com.sun.org.apache.bcel.internal.Constants
com.sun.org.apache.bcel.internal.util.ClassLoader
com.sun.org.apache.bcel.internal.util.InstructionFinder
com.sun.org.apache.bcel.internal.verifier.NativeVerifier
com.sun.org.apache.bcel.internal.verifier.VerifierAppFrame
com.sun.org.apache.bcel.internal.verifier.VerifierFactoryListModel
com.sun.org.apache.bcel.internal.verifier.VerifyDialog
com.sun.org.apache.bcel.internal.verifier.statics.Pass3aVerifier$InstOperand
ConstraintVisitor
com.sun.org.apache.html.internal.dom.HTMLBuilder

```

```

com.sun.org.apache.html.internal.dom.HTMLCollectionImpl
com.sun.org.apache.html.internal.dom.HTMLDocumentImpl
com.sun.org.apache.html.internal.dom.HTMLTableElementImpl
com.sun.org.apache.html.internal.dom.ObjectFactory
com.sun.org.apache.regexp.internal.REDemo
com.sun.org.apache.regexp.internal.RETest
com.sun.org.apache.regexp.internal.recompile
com.sun.org.apache.xalan.internal.client.XSLTProcessorApplet
com.sun.org.apache.xalan.internal.client.XSLTProcessorApplet$TrustedAgent
com.sun.org.apache.xalan.internal.lib.ExsltStrings
com.sun.org.apache.xalan.internal.lib.Extensions
com.sun.org.apache.xalan.internal.lib.ObjectFactory
com.sun.org.apache.xalan.internal.xslt.ObjectFactory
com.sun.org.apache.xalan.internal.xslt.Process
com.sun.org.apache.xalan.internal.xsltc.cmdline.ObjectFactory
com.sun.org.apache.xalan.internal.xsltc.compiler.ObjectFactory
com.sun.org.apache.xalan.internal.xsltc.compiler.XSLTC
com.sun.org.apache.xalan.internal.xsltc.compiler.util.ObjectFactory
com.sun.org.apache.xalan.internal.xsltc.dom.DocumentCache
com.sun.org.apache.xalan.internal.xsltc.dom.ObjectFactory
com.sun.org.apache.xalan.internal.xsltc.runtime.ObjectFactory
com.sun.org.apache.xalan.internal.xsltc.trax.ObjectFactory
com.sun.org.apache.xalan.internal.xsltc.trax.TemplatesHandlerImpl
com.sun.org.apache.xalan.internal.xsltc.trax.TemplatesImpl
com.sun.org.apache.xerces.internal.dom.CoreDOMImplementationImpl
com.sun.org.apache.xerces.internal.dom.ObjectFactory
com.sun.org.apache.xerces.internal.impl.XMLEntityManager
com.sun.org.apache.xerces.internal.impl.dv.DTDDVFactory
com.sun.org.apache.xerces.internal.impl.dv.ObjectFactory
com.sun.org.apache.xerces.internal.impl.dv.SchemaDVFactory
com.sun.org.apache.xerces.internal.impl.dv.xs.AbstractDateTimeDV$DateTimeData
com.sun.org.apache.xerces.internal.impl.dv.xs.Base64BinaryDV$XBase64
com.sun.org.apache.xerces.internal.impl.dv.xs.DecimalDV$XDecimal
com.sun.org.apache.xerces.internal.impl.dv.xs.DoubleDV$XDouble
com.sun.org.apache.xerces.internal.impl.dv.xs.FloatDV$XFloat
com.sun.org.apache.xerces.internal.impl.dv.xs.HexBinaryDV$XHex
com.sun.org.apache.xerces.internal.impl.dv.xs.ListDV$XListData
com.sun.org.apache.xerces.internal.impl.dv.xs.QNameDV$XQName
com.sun.org.apache.xerces.internal.impl.dv.xs.XSSimpleTypeDecl
com.sun.org.apache.xerces.internal.impl.xpath.regex.Match
com.sun.org.apache.xerces.internal.impl.xpath.regex.ParserForXMLSchema
com.sun.org.apache.xerces.internal.impl.xpath.regex.REUtil
com.sun.org.apache.xerces.internal.impl.xpath.regex.RangeToken
com.sun.org.apache.xerces.internal.impl.xpath.regex.RegexParser
com.sun.org.apache.xerces.internal.impl.xpath.regex.RegularExpression
com.sun.org.apache.xerces.internal.impl.xpath.regex.Token
com.sun.org.apache.xerces.internal.impl.xs.SchemaGrammar
com.sun.org.apache.xerces.internal.impl.xs.SchemaGrammar$BuiltinSchemaGrammar
com.sun.org.apache.xerces.internal.impl.xs.XSAnnotationImpl
com.sun.org.apache.xerces.internal.impl.xs.XSComplexTypeDecl
com.sun.org.apache.xerces.internal.impl.xs.XSModelImpl
com.sun.org.apache.xerces.internal.impl.xs.util.XSNamedMap4Types
com.sun.org.apache.xerces.internal.impl.xs.util.XSNamedMapImpl
com.sun.org.apache.xerces.internal.parsers.CachingParserPool$SynchronizedGrammarPool
com.sun.org.apache.xerces.internal.parsers.JAXPConfiguration

```

```

com.sun.org.apache.xerces.internal.parsers.ObjectFactory
com.sun.org.apache.xerces.internal.util.SynchronizedSymbolTable
com.sun.org.apache.xerces.internal.util.XMLGrammarPoolImpl
com.sun.org.apache.xerces.internal.xinclude.ObjectFactory
com.sun.org.apache.xml.internal.dtm.DTMException
com.sun.org.apache.xml.internal.dtm.DTMManager
com.sun.org.apache.xml.internal.dtm.FactoryFinder
com.sun.org.apache.xml.internal.dtm.ObjectFactory
com.sun.org.apache.xml.internal.dtm.ref.CoroutineManager
com.sun.org.apache.xml.internal.dtm.ref.DTMManagerDefault
com.sun.org.apache.xml.internal.dtm.ref.DTMSafeStringPool
com.sun.org.apache.xml.internal.dtm.ref.ObjectFactory
com.sun.org.apache.xml.internal.serialize.ObjectFactory
com.sun.org.apache.xml.internal.serialize.OutputFormat
com.sun.org.apache.xml.internal.serialize.SerializerFactory
com.sun.org.apache.xml.internal.serializer.CharInfo
com.sun.org.apache.xml.internal.serializer.ObjectFactory
com.sun.org.apache.xml.internal.serializer.OutputPropertiesFactory
com.sun.org.apache.xml.internal.serializer.ToHTMLStream
com.sun.org.apache.xml.internal.serializer.ToStream
com.sun.org.apache.xml.internal.utils.ObjectFactory
com.sun.org.apache.xml.internal.utils.ObjectPool
com.sun.org.apache.xml.internal.utils.StringBufferPool
com.sun.org.apache.xml.internal.utils.XMLReaderManager
com.sun.org.apache.xpath.internal.VariableStack
com.sun.org.apache.xpath.internal.axes.IteratorPool
com.sun.org.apache.xpath.internal.compiler.Compiler
com.sun.org.apache.xpath.internal.compiler.ObjectFactory
com.sun.org.apache.xpath.internal.functions.ObjectFactory
com.sun.org.omg.CORBA.AttrDescriptionSeqHelper
com.sun.org.omg.CORBA.AttributeDescriptionHelper
com.sun.org.omg.CORBA.AttributeModeHelper
com.sun.org.omg.CORBA.ContextIdSeqHelper
com.sun.org.omg.CORBA.ContextIdentifierHelper
com.sun.org.omg.CORBA.DefinitionKindHelper
com.sun.org.omg.CORBA.ExcDescriptionSeqHelper
com.sun.org.omg.CORBA.ExceptionDescriptionHelper
com.sun.org.omg.CORBA.IDLTypeHelper
com.sun.org.omg.CORBA.IdentifierHelper
com.sun.org.omg.CORBA.InitializerHelper
com.sun.org.omg.CORBA.InitializerSeqHelper
com.sun.org.omg.CORBA.OpDescriptionSeqHelper
com.sun.org.omg.CORBA.OperationDescriptionHelper
com.sun.org.omg.CORBA.OperationModeHelper
com.sun.org.omg.CORBA.ParDescriptionSeqHelper
com.sun.org.omg.CORBA.ParameterDescriptionHelper
com.sun.org.omg.CORBA.ParameterModeHelper
com.sun.org.omg.CORBA.RepositoryHelper
com.sun.org.omg.CORBA.RepositoryIdHelper
com.sun.org.omg.CORBA.RepositoryIdSeqHelper
com.sun.org.omg.CORBA.StructMemberHelper
com.sun.org.omg.CORBA.StructMemberSeqHelper
com.sun.org.omg.CORBA.ValueDefPackage.FullValueDescriptionHelper
com.sun.org.omg.CORBA.ValueMemberHelper
com.sun.org.omg.CORBA.ValueMemberSeqHelper
com.sun.org.omg.CORBA.VersionSpecHelper
com.sun.org.omg.CORBA.VisibilityHelper
com.sun.org.omg.SendingContext.CodeBaseHelper

```

```

com.sun.org.omg.SendingContext.CodeBasePackage.URLHelper
com.sun.org.omg.SendingContext.CodeBasePackage.URLSeqHelper
com.sun.org.omg.SendingContext.CodeBasePackage.ValueDescSeqHelper
com.sun.rmi.rmid.ExecOptionPermission
com.sun.rmi.rmid.ExecPermission
com.sun.rowset.JdbcRowSetImpl
com.sun.rowset.JdbcRowSetResourceBundle
com.sun.rowset.internal.WebRowSetXmlReader
com.sun.security.*
com.sun.swing.*
com.sun.tools apt.comp.Apt
com.sun.tools apt.main.Main
com.sun.tools.corba.se.idl.som.cff.Messages
com.sun.tools.corba.se.idl.toJavaPortable.Factories
com.sun.tools.corba.se.idl.toJavaPortable.Helper
com.sun.tools.doclets.internal.toolkit.builders.LayoutParser
com.sun.tools.doclets.internal.toolkit.util.Util
com.sun.tools.example.debug.expr.ASCII_UCodeESC_CharStream
com.sun.tools.example.debug.expr.ExpressionParserConstants
com.sun.tools.example.debug.expr.ExpressionParserTokenManager
com.sun.tools.example.debug.expr.LValue
com.sun.tools.example.debug.tty.Commands
com.sun.tools.example.debug.tty.Env
com.sun.tools.example.debug.tty.EventHandler
com.sun.tools.example.debug.tty.EventRequestSpec
com.sun.tools.example.debug.tty.EventRequestSpecList
com.sun.tools.example.debug.tty.MessageOutput
com.sun.tools.example.debug.tty.TTYResources
com.sun.tools.example.debug.tty.TTYResources_ja
com.sun.tools.example.debug.tty.TTYResources_zh_CN
com.sun.tools.example.debug.tty.ThreadInfo
com.sun.tools.example.debug.tty.VMConnection
com.sun.tools.javac.code.Flags
com.sun.tools.javac.jvm.ByteCodes
com.sun.tools.javac.jvm.Code$Mneumonics
com.sun.tools.javac.main.Main
com.sun.tools.javac.parser.Tokens
com.sun.tools.javac.resources.compiler
com.sun.tools.javac.resources.compiler_ja
com.sun.tools.javac.tree.Pretty
com.sun.tools.javac.util.Name$Table
com.sun.tools.javadoc.Start
com.sun.tools.jdi.AbstractLauncher$Helper
com.sun.tools.jdi.EventQueueImpl
com.sun.tools.jdi.EventQueueImpl$TimerThread
com.sun.tools.jdi.EventRequestManagerImpl
com.sun.tools.jdi.EventRequestManagerImpl$ClassVisibleEventRequestImpl
com.sun.tools.jdi.EventRequestManagerImpl$EventRequestImpl
com.sun.tools.jdi.EventRequestManagerImpl$ThreadVisibleEventRequestImpl
com.sun.tools.jdi.EventSetImpl
com.sun.tools.jdi.JNITypeParser
com.sun.tools.jdi.ObjectReferenceImpl
com.sun.tools.jdi.Packet
com.sun.tools.jdi.ReferenceTypeImpl
com.sun.tools.jdi.SharedMemoryConnection
com.sun.tools.jdi.SharedMemoryTransportService
com.sun.tools.jdi.SocketConnection
com.sun.tools.jdi.SocketTransportService

```

```

com.sun.tools.jdi.StackFrameImpl
com.sun.tools.jdi.TargetVM
com.sun.tools.jdi.TargetVM$EventController
com.sun.tools.jdi.ThreadReferenceImpl
com.sun.tools.jdi.VMState
com.sun.tools.jdi.VirtualMachineImpl
com.sun.tools.jdi.VirtualMachineManagerImpl
java.applet.*
java.awt.*
java.beans.BeansAppletContext
java.beans.DefaultPersistenceDelegate
java.beans.EventHandler
java.beans.EventSetDescriptor
java.beans.IndexedPropertyDescriptor
java.beans.Introspector
java.beans.MetaData
java.beans.MethodDescriptor
java.beans.PropertyChangeSupport
java.beans.PropertyDescriptor
java.beans.PropertyEditorManager
java.beans.PropertyEditorSupport
java.beans.ReflectionUtils
java.beans.VetoableChangeSupport
java.beans.beancontext.BeanContextChildSupport
java.beans.beancontext.BeanContextEvent
java.beans.beancontext.BeanContextServicesSupport
java.beans.beancontext.BeanContextServicesSupport$BCSSChild
java.beans.beancontext.BeanContextSupport
java.io.BufferedInputStream
java.io.BufferedOutputStream
java.io.BufferedReader
java.io.BufferedWriter
java.io.ByteArrayInputStream
java.io.ByteArrayOutputStream
java.io.CharArrayReader
java.io.CharArrayWriter
java.io.DataOutputStream
java.io.ExpiringCache
java.io.File
java.io.FileDescriptor
java.io.FileInputStream
java.io.FileOutputStream
java.io.FilePermissionCollection
java.io.FileSystem
java.io.FilterInputStream
java.io.IOException
java.io.InputStream
java.io.LineNumberReader
java.io.ObjectInputStream
java.io.ObjectInputStream$Caches
java.io.ObjectInputStream$CallbackContext
java.io.ObjectOutputStream
java.io.ObjectOutputStream$Caches
java.io.ObjectStreamClass
java.io.ObjectStreamClass$Caches
java.io.ObjectStreamClass$EntryFuture
java.io.OutputStream
java.io.PipedInputStream

```

java.io.PipedOutputStream
java.io.PipedReader
java.io.PipedWriter
java.io.PrintStream
java.io.PrintWriter
java.io.PushbackInputStream
java.io.PushbackReader
java.io.RandomAccessFile
java.io.Reader
java.io.Serializable
java.io.StringBufferInputStream
java.io.StringReader
java.io.Win32FileSystem
java.io.WinNTFileSystem
java.io.Writer
java.lang.AbstractMethodError
java.lang.ArithmeticException
java.lang.ArrayIndexOutOfBoundsException
java.lang.ArrayStoreException
java.lang.AssertionStatusDirectives
java.lang.Boolean
java.lang.Byte
java.lang.CharSequence
java.lang.Character
java.lang.Class
java.lang.ClassCastException
java.lang.ClassCircularityError
java.lang.ClassFormatError
java.lang.ClassLoader
java.lang.ClassLoader\$NativeLibrary
java.lang.ClassNotFoundException
java.lang.CloneNotSupportedException
java.lang.Cloneable
java.lang.Compiler
java.lang.Double
java.lang.Error
java.lang.Exception
java.lang.ExceptionInInitializerError
java.lang.Float
java.lang.IllegalAccessError
java.lang.IllegalAccessException
java.lang.IllegalArgumentException
java.lang.IllegalMonitorStateException
java.lang.IllegalThreadStateException
java.lang.IncompatibleClassChangeError
java.lang.IndexOutOfBoundsException
java.lang.InstantiationError
java.lang.InstantiationException
java.lang.Integer
java.lang.InternalError
java.lang.InterruptedOperationException
java.lang.InvalidClassException
java.lang.LinkageError
java.lang.Long
java.lang.Math
java.lang.NegativeArraySizeException
java.lang.NoClassDefFoundError
java.lang.NoSuchFieldError

java.lang.NoSuchFieldException
java.lang.NoSuchMethodError
java.lang.NoSuchMethodException
java.lang.NullPointerException
java.lang.Object
java.lang.OutOfMemoryError
java.lang.Package
java.lang.ProcessEnvironment
java.lang.ProcessImpl
java.lang.Runtime
java.lang.RuntimeException
java.lang.SecurityManager
java.lang.Short
java.lang.Shutdown
java.lang.StackOverflowError
java.lang.StackTraceElement
java.lang.StrictMath
java.lang.String
java.lang.StringBuffer
java.lang.StringIndexOutOfBoundsException
java.lang.System
java.lang.Thread
java.lang.ThreadDeath
java.lang.ThreadGroup
java.lang.ThreadLocal
java.lang.Throwable
java.lang.UnsatisfiedLinkError
java.lang.UnsupportedClassVersionError
java.lang.VerifyError
java.lang.management.ManagementFactory
java.lang.ref.FinalReference
java.lang.ref.Finalizer
java.lang.ref.Finalizer\$3
java.lang.ref.PhantomReference
java.lang.ref.Reference
java.lang.ref.Reference\$ReferenceHandler
java.lang.ref.ReferenceQueue
java.lang.ref.SoftReference
java.lang.ref.WeakReference
java.lang.reflect.AccessibleObject
java.lang.reflect.Array
java.lang.reflect.Constructor
java.lang.reflect.Field
java.lang.reflect.InvocationTargetException
java.lang.reflect.Method
java.lang.reflect.Modifier
java.lang.reflect.Proxy
java.math.BigDecimal
java.math.MathContext
java.net.Authenticator
java.net.CookieHandler
java.net.DatagramPacket
java.net.DatagramSocket
java.net.FactoryURLConnectionLoader
java.net.Inet4Address
java.net.Inet4AddressImpl
java.net.Inet6Address
java.net.Inet6AddressImpl


```

java.net.InetAddress
java.net.InetAddressImplFactory
java.net.MulticastSocket
java.net.NetworkInterface
java.net.PlainDatagramSocketImpl
java.net.PlainSocketImpl
java.net.ResponseCache
java.net.ServerSocket
java.net.Socket
java.net.SocketInputStream
java.net.SocketOutputStream
java.net.SocketPermission
java.net.SocketPermissionCollection
java.net.SocksSocketImpl
java.net.URL
java.net.URLConnection
java.net.URLStreamHandler
java.nio.Bits
java.nio.MappedByteBuffer
java.nio.channels.Channels
java.nio.channels.Channels$1
java.nio.channels.Channels$ReadableByteChannelImpl
java.nio.channels.Channels$WritableByteChannelImpl
java.nio.channels.spi.AbstractInterruptibleChannel
java.nio.channels.spi.AbstractInterruptibleChannel$1
java.nio.channels.spi.AbstractSelectableChannel
java.nio.channels.spi.AbstractSelectionKey
java.nio.channels.spi.AbstractSelector
java.nio.channels.spi.SelectorProvider
java.nio.charset.Charset
java.nio.charset.CoderResult$Cache
java.rmi.activation.ActivationGroup
java.rmi.server.LogStream
java.rmi.server.ObjID$InsecureRandom
java.rmi.server.RMISocketFactory
java.rmi.server.RemoteObjectInvocationHandler$MethodToHash_Maps$1
java.rmi.server.UID
java.security.*
java.sql.DriverManager
java.sql.SQLException
java.sql.Date
java.sql.Time
java.sql.Timestamp
java.sql.BatchUpdateException
java.sql.SQLWarning
java.text.AttributedString
java.text.AttributedString$AttributeMap
java.text.AttributedString$AttributedStringIterator
java.text.Bidi
java.text.BreakIterator
java.text.Collator
java.text.DecimalFormat
java.text.RuleBasedCollator
java.util.AbstractList
java.util.Calendar
java.util.Collections
java.util.Collections$SynchronizedCollection
java.util.Collections$SynchronizedList

```



```

java.util.Collections$SynchronizedMap
java.util.Collections$SynchronizedRandomAccessList
java.util.Collections$SynchronizedSet
java.util.Collections$SynchronizedSortedMap
java.util.Collections$SynchronizedSortedSet
java.util.Currency
java.util.Date
java.util.GregorianCalendar
java.util.Hashtable
java.util.Hashtable$Enumerator
java.util.ListResourceBundle
java.util.Locale
java.util.Observable
java.util.Properties
java.util.PropertyPermission
java.util.PropertyPermissionCollection
java.util.Random
java.util.ResourceBundle
java.util.SimpleTimeZone
java.util.Stack
java.util.TimeZone
java.util.TimeZone$DisplayNames
java.util.Timer
java.util.Timer$1
java.util.TimerTask
java.util.TimerThread
java.util.Vector
java.util.Vector$1
java.util.concurrent.*
java.util.jar.JarFile
java.util.jar.JarVerifier
java.util.jar.Pack200
java.util.logging.ErrorManager
java.util.logging.FileHandler
java.util.logging.Formatter
java.util.logging.Handler
java.util.logging.Level
java.util.logging.LogManager
java.util.logging.LogManager$Cleaner
java.util.logging.LogRecord
java.util.logging.Logger
java.util.logging.MemoryHandler
java.util.logging.SimpleFormatter
java.util.logging.SocketHandler
java.util.logging.StreamHandler
java.util.prefs.AbstractPreferences
java.util.prefs.AbstractPreferences$EventDispatchThread
java.util.prefs.WindowsPreferences
java.util.prefs.XmlSupport
java.util.regex.Pattern
java.util.zip.*
javax.imageio.*
javax.management.AttributeChangeNotificationFilter
javax.management.MBeanInfo
javax.management.MBeanServerDelegate
javax.management.MBeanServerFactory
javax.management.MBeanServerPermission
javax.management.MBeanServerPermissionCollection

```

```

javax.management.NotificationBroadcasterSupport
javax.management.NotificationFilterSupport
javax.management.ObjectName
javax.management.StandardMBean
javax.management.loading.MLet
javax.management.modelmbean.DescriptorSupport
javax.management.modelmbean.RequiredModelMBean
javax.management.monitor.CounterMonitor
javax.management.monitor.GaugeMonitor
javax.management.monitor.Monitor
javax.management.monitor.StringMonitor
javax.management.relation.MBeanServerNotificationFilter
javax.management.relation.RelationService
javax.management.relation.RelationSupport
javax.management.remote.JMXConnectorServer
javax.management.remote.rmi.RMIConnectionImpl
javax.management.remote.rmi.RMIConnector
javax.management.remote.rmi.RMIConnector$RMIClientCommunicatorAdmin
javax.management.remote.rmi.RMIConnectorServer
javax.management.remote.rmi.RMIServerImpl
javax.management.timer.Timer
javax.naming.spi.NamingManager
javax.print.*
javax.rmi.ssl.SslRMIClientSocketFactory
javax.rmi.ssl.SslRMIServerSocketFactory
javax.security.*
javax.sound.*
javax.sql.ConnectionEvent
javax.sql.rowset.spi.SyncFactory
javax.swing.*
javax.xml.datatype.FactoryFinder
javax.xml.parsers.FactoryFinder
javax.xml.transform.FactoryFinder
javax.xml.transform.TransformerException
javax.xml.validation.SchemaFactoryFinder
javax.xml.xpath.XPathFactoryFinder
org.omg.CORBA.AnySeqHelper
org.omg.CORBA.BooleanSeqHelper
org.omg.CORBA.CharSeqHelper
org.omg.CORBA.CompletionStatusHelper
org.omg.CORBA.CurrentHelper
org.omg.CORBA.DefinitionKindHelper
org.omg.CORBA.DoubleSeqHelper
org.omg.CORBA.FieldNameHelper
org.omg.CORBA.FloatSeqHelper
org.omg.CORBA.IDLTypeHelper
org.omg.CORBA.IdentifierHelper
org.omg.CORBA.LongLongSeqHelper
org.omg.CORBA.LongSeqHelper
org.omg.CORBA.NameValuePairHelper
org.omg.CORBA.ORB
org.omg.CORBA.ObjectHelper
org.omg.CORBA.OctetSeqHelper
org.omg.CORBA.ParameterModeHelper
org.omg.CORBA.PolicyErrorCodeHelper
org.omg.CORBA.PolicyErrorHelper
org.omg.CORBA.PolicyHelper
org.omg.CORBA.PolicyListHelper

```

org.omg.CORBA.PolicyTypeHelper
org.omg.CORBA.RepositoryIdHelper
org.omg.CORBA.ServiceDetailHelper
org.omg.CORBA.ServiceInformationHelper
org.omg.CORBA.SetOverrideTypeHelper
org.omg.CORBA.ShortSeqHelper
org.omg.CORBA.StringSeqHelper
org.omg.CORBA.StringValueHelper
org.omg.CORBA.StructMemberHelper
org.omg.CORBA.ULongLongSeqHelper
org.omg.CORBA.ULongSeqHelper
org.omg.CORBA.UShortSeqHelper
org.omg.CORBA.UnionMemberHelper
org.omg.CORBA.UnknownUserExceptionHelper
org.omg.CORBA.ValueBaseHelper
org.omg.CORBA.ValueMemberHelper
org.omg.CORBA.VersionSpecHelper
org.omg.CORBA.VisibilityHelper
org.omg.CORBA.WCharSeqHelper
org.omg.CORBA.WStringSeqHelper
org.omg.CORBA.WStringValueHelper
org.omg.CORBA.WrongTransactionHelper
org.omg.CosNaming.BindingHelper
org.omg.CosNaming.BindingIteratorHelper
org.omg.CosNaming.BindingListHelper
org.omg.CosNaming.BindingTypeHelper
org.omg.CosNaming.IstringHelper
org.omg.CosNaming.NameComponentHelper
org.omg.CosNaming.NameHelper
org.omg.CosNaming.NamingContextExtHelper
org.omg.CosNaming.NamingContextExtPackage.AddressHelper
org.omg.CosNaming.NamingContextExtPackage.InvalidAddressHelper
org.omg.CosNaming.NamingContextExtPackage.StringNameHelper
org.omg.CosNaming.NamingContextExtPackage.URLStringHelper
org.omg.CosNaming.NamingContextHelper
org.omg.CosNaming.NamingContextPackage.AlreadyBoundHelper
org.omg.CosNaming.NamingContextPackage.CannotProceedHelper
org.omg.CosNaming.NamingContextPackage.InvalidNameHelper
org.omg.CosNaming.NamingContextPackage.NotEmptyHelper
org.omg.CosNaming.NamingContextPackage.NotFoundHelper
org.omg.CosNaming.NamingContextPackage.NotFoundReasonHelper
org.omg.DynamicAny.AnySeqHelper
org.omg.DynamicAny.DynAnyFactoryHelper
org.omg.DynamicAny.DynAnyFactoryPackage.InconsistentTypeCodeHelper
org.omg.DynamicAny.DynAnyHelper
org.omg.DynamicAny.DynAnyPackage.InvalidValueHelper
org.omg.DynamicAny.DynAnyPackage.TypeMismatchHelper
org.omg.DynamicAny.DynAnySeqHelper
org.omg.DynamicAny.DynArrayHelper
org.omg.DynamicAny.DynEnumHelper
org.omg.DynamicAny.DynFixedHelper
org.omg.DynamicAny.DynSequenceHelper
org.omg.DynamicAny.DynStructHelper
org.omg.DynamicAny.DynUnionHelper
org.omg.DynamicAny.DynValueHelper
org.omg.DynamicAny.FieldNameHelper
org.omg.DynamicAny.NameDynAnyPairHelper
org.omg.DynamicAny.NameDynAnyPairSeqHelper

```

org.omg.DynamicAny.NameValuePairHelper
org.omg.DynamicAny.NameValuePairSeqHelper
org.omg.IOP.CodecFactoryHelper
org.omg.IOP.CodecFactoryPackage.UnknownEncodingHelper
org.omg.IOP.CodecPackage.FormatMismatchHelper
org.omg.IOP.CodecPackage.InvalidTypeForEncodingHelper
org.omg.IOP.CodecPackage.TypeMismatchHelper
org.omg.IOP.ComponentIdHelper
org.omg.IOP.IORHelper
org.omg.IOP.MultipleComponentProfileHelper
org.omg.IOP.ProfileIdHelper
org.omg.IOP.ServiceContextHelper
org.omg.IOP.ServiceContextListHelper
org.omg.IOP.ServiceIdHelper
org.omg.IOP.TaggedComponentHelper
org.omg.IOP.TaggedProfileHelper
org.omg.Messaging.SyncScopeHelper
org.omg.PortableInterceptor.AdapterManagerIdHelper
org.omg.PortableInterceptor.AdapterNameHelper
org.omg.PortableInterceptor.AdapterStateHelper
org.omg.PortableInterceptor.CurrentHelper
org.omg.PortableInterceptor.ForwardRequestHelper
org.omg.PortableInterceptor.IORInterceptor_3_0Helper
org.omg.PortableInterceptor.InvalidSlotHelper
org.omg.PortableInterceptor.ORBIdHelper
org.omg.PortableInterceptor.ORBInitInfoPackage.DuplicateNameHelper
org.omg.PortableInterceptor.ORBInitInfoPackage.InvalidNameHelper
org.omg.PortableInterceptor.ORBInitInfoPackage.ObjectIdHelper
org.omg.PortableInterceptor.ObjectIdHelper
org.omg.PortableInterceptor.ObjectReferenceFactoryHelper
org.omg.PortableInterceptor.ObjectReferenceTemplateHelper
org.omg.PortableInterceptor.ObjectReferenceTemplateSeqHelper
org.omg.PortableInterceptor.ServerIdHelper
org.omg.PortableServer.CurrentHelper
org.omg.PortableServer.CurrentPackage.NoContextHelper
org.omg.PortableServer.ForwardRequestHelper
org.omg.PortableServer.POAHelper
org.omg.PortableServer.POAManagerPackage.AdapterInactiveHelper
org.omg.PortableServer.POAPackage.AdapterAlreadyExistsHelper
org.omg.PortableServer.POAPackage.AdapterNonExistentHelper
org.omg.PortableServer.POAPackage.InvalidPolicyHelper
org.omg.PortableServer.POAPackage.NoServantHelper
org.omg.PortableServer.POAPackage.ObjectAlreadyActiveHelper
org.omg.PortableServer.POAPackage.ObjectNotActiveHelper
org.omg.PortableServer.POAPackage.ServantAlreadyActiveHelper
org.omg.PortableServer.POAPackage.ServantNotActiveHelper
org.omg.PortableServer.POAPackage.WrongAdapterHelper
org.omg.PortableServer.POAPackage.WrongPolicyHelper
org.omg.PortableServer.ServantActivatorHelper
org.omg.PortableServer.ServantLocatorHelper
org.omg.stub.java.management.remote.rmi._RMICConnectionImpl_Tie
org.omg.stub.java.management.remote.rmi._RMICConnection_Stub
org.omg.stub.java.management.remote.rmi._RMIServerImpl_Tie
org.omg.stub.java.management.remote.rmi._RMIServer_Stub
sun.applet.*
sun.audio.*
sun.awt.*
sun.corba.Bridge

```

```

sun.dc.pr.*
sun.font.AdvanceCache
sun.font.CompositeFont
sun.font.FileFont
sun.font.FileFont$FileFontDisposer
sun.font.FileFontStrike
sun.font.Font2D
sun.font.FontFamily
sun.font.FontManager
sun.font.FontStrikeDisposer
sun.font.GlyphLayout
sun.font.GlyphList
sun.font.PhysicalStrike
sun.font.StrikeCache
sun.font.SunLayoutEngine
sun.font.TrueTypeFont
sun.font.TrueTypeFont$TTDisposerRecord
sun.font.Type1Font
sun.instrument.InstrumentationImpl
sun.instrument.TransformerManager
sun.io.CharacterEncoding
sun.io.Converters
sun.java2d.*
sun.jdbc.*
sun.jvmstat.monitor.MonitoredHost
sun.jvmstat.perfdata.monitor.PerfDataBufferImpl
sun.jvmstat.perfdata.monitor.protocol.local.LocalEventTimer
sun.jvmstat.perfdata.monitor.protocol.local.LocalMonitoredVm
sun.jvmstat.perfdata.monitor.protocol.local.LocalVmManager
sun.jvmstat.perfdata.monitor.protocol.local.MonitoredHostProvider
sun.jvmstat.perfdata.monitor.protocol.rmi.MonitoredHostProvider
sun.jvmstat.perfdata.monitor.protocol.rmi.PerfDataBuffer
sun.jvmstat.perfdata.monitor.protocol.rmi.RemoteMonitoredVm
sun.management.Agent
sun.management.ClassLoadingImpl
sun.management.FileSystem
sun.management.FileSystemImpl
sun.management.GarbageCollectorImpl
sun.management.GcInfoBuilder
sun.management.GcInfoCompositeData
sun.management.HotspotRuntime
sun.management.HotspotThread
sun.management.LazyCompositeData
sun.management.MXBeanSupport
sun.management.ManagementFactory
sun.management.MappedMXBeanType
sun.management.MemoryImpl
sun.management.MemoryManagerImpl
sun.management.MemoryPoolImpl
sun.management.NotificationEmitterSupport
sun.management.Sensor
sun.management.ThreadImpl
sun.management.VMManagementImpl
sun.management.counter.perf.PerfInstrumentation
sun.management.jmxremote.ConnectorBootstrap
sun.management.jmxremote.ConnectorBootstrap$PermanentExporter
sun.management.snmp.AdaptorBootstrap
sun.management.snmp.jvminstr.JVM_MANAGEMENT_MIB_IMPL

```

```
sun.management.snmp.jvminstr.JvmMemPoolEntryImpl
sun.management.snmp.jvminstr.JvmThreadingImpl
sun.management.snmp.jvmmib.JvmMemGCTableMeta
sun.management.snmp.jvmmib.JvmMemManagerTableMeta
sun.management.snmp.jvmmib.JvmMemMgrPoolRelTableMeta
sun.management.snmp.jvmmib.JvmMemPoolTableMeta
sun.management.snmp.jvmmib.JvmRTBootClassPathTableMeta
sun.management.snmp.jvmmib.JvmRTCClassPathTableMeta
sun.management.snmp.jvmmib.JvmRTInputArgsTableMeta
sun.management.snmp.jvmmib.JvmRTLlibraryPathTableMeta
sun.management.snmp.jvmmib.JvmThreadInstanceTableMeta
sun.management.snmp.util.JvmContextFactory
sun.management.snmp.util.SnmpTableCache
sun.misc.AtomicLong
sun.misc.AtomicLongCSImpl
sun.misc.AtomicLongLockImpl
sun.misc.Cache
sun.misc.ClassFileTransformer
sun.misc.Cleaner
sun.misc.ConditionLock
sun.misc.ExtensionDependency
sun.misc.FIFOQueueEnumerator
sun.misc.FloatingDecimal
sun.misc.FloatingDecimal$1
sun.misc.FormattedFloatingDecimal
sun.misc.FormattedFloatingDecimal$1
sun.misc.GC
sun.misc.GC$Daemon
sun.misc.GC$LatencyRequest
sun.misc.LIFOQueueEnumerator
sun.misc.Launcher$AppClassLoader
sun.misc.Lock
sun.misc.MessageUtils
sun.misc.NativeSignalHandler
sun.misc.PathPermissions
sun.misc.Perf
sun.misc.Perf$1
sun.misc.PerformanceLogger
sun.misc.Queue
sun.misc.Ref
sun.misc.RequestProcessor
sun.misc.Resource
sun.misc.Signal
sun.misc.Timer
sun.misc.TimerThread
sun.misc.TimerTickThread
sun.misc.URLClassPath
sun.misc.Unsafe
sun.misc.VM
sun.net.InetAddressCachePolicy
sun.net.ProgressMonitor
sun.net.dns.ResolverConfiguration
sun.net.dns.ResolverConfigurationImpl
sun.net.dns.ResolverConfigurationImpl$AddressChangeListener
sun.net.ftp.FtpClient
sun.net.spi.DefaultProxySelector
sun.net.spi.DefaultProxySelector$2
sun.net.www.HeaderParser
```

sun.net.www.MessageHeader
sun.net.www.MessageHeader\$HeaderIterator
sun.net.www.MeteredStream
sun.net.www.MimeEntry
sun.net.www.MimeTable
sun.net.www.http.ChunkedInputStream
sun.net.www.http.ChunkedOutputStream
sun.net.www.http.ClientVector
sun.net.www.http.HttpClient
sun.net.www.http.KeepAliveCache
sun.net.www.http.KeepAliveStream
sun.net.www.http.PostterOutputStream
sun.net.www.protocol.doc.DocURLConnection
sun.net.www.protocol.doc.Handler
sun.net.www.protocol.file.FileURLConnection
sun.net.www.protocol.file.Handler
sun.net.www.protocol.ftp.FtpURLConnection
sun.net.www.protocol.http.AuthCacheImpl
sun.net.www.protocol.http.AuthenticationInfo
sun.net.www.protocol.http.DigestAuthentication\$Parameters
sun.net.www.protocol.http.HttpURLConnection
sun.net.www.protocol.http.NTLMAuthSequence
sun.net.www.protocol.http.NTLMAuthentication
sun.net.www.protocol.jar.JarFileFactory
sun.net.www.protocol.jar.URLJarFile
sun.net.www.protocol.mailto.Handler
sun.net.www.protocol.mailto.MailToURLConnection
sun.net.www.protocol.netdoc.Handler
sun.nio.ch.AllocatedNativeObject
sun.nio.ch.ChannelInputStream
sun.nio.ch.DatagramChannelImpl
sun.nio.ch.DatagramDispatcher
sun.nio.ch.DatagramSocketAdaptor
sun.nio.ch.FileChannelImpl
sun.nio.ch.FileDispatcher
sun.nio.ch.FileLockImpl
sun.nio.ch.IOUtil
sun.nio.ch.NativeThreadSet
sun.nio.ch.Net
sun.nio.ch.SelectorImpl
sun.nio.ch.ServerSocketAdaptor
sun.nio.ch.ServerSocketChannelImpl
sun.nio.ch.SocketAdaptor
sun.nio.ch.SocketAdaptor\$SocketInputStream
sun.nio.ch.SocketChannelImpl
sun.nio.ch.SocketDispatcher
sun.nio.ch.Util
sun.nio.ch.Util\$SelectorWrapper\$Closer
sun.nio.ch.WindowsSelectorImpl
sun.nio.ch.WindowsSelectorImpl\$FinishLock
sun.nio.ch.WindowsSelectorImpl\$StartLock
sun.nio.ch.WindowsSelectorImpl\$SubSelector
sun.nio.cs.AbstractCharsetProvider
sun.nio.cs.FastCharsetProvider
sun.nio.cs.StreamDecoder
sun.nio.cs.StreamEncoder
sun.print.*
sun.reflect.ConstantPool


```

sun.reflect.MethodAccessorGenerator
sun.reflect.NativeConstructorAccessorImpl
sun.reflect.NativeMethodAccessorImpl
sun.reflect.Reflection
sun.reflect.annotation.AnnotationType
sun.reflect.misc.MethodUtil
sun.rmi.log.ReliableLog
sun.rmi.registry.RegistryImpl
sun.rmi.rmic.BatchEnvironment
sun.rmi.rmic.Main
sun.rmi.rmic.iiop.DirectoryLoader
sun.rmi.rmic.iiop.NameContext
sun.rmi.rmic.newrmic.Main
sun.rmi.runtime.Log$LogStreamLog
sun.rmi.runtime.Log$LoggerLog
sun.rmi.runtime.ThreadPool
sun.rmi.runtime.ThreadPool$Worker
sun.rmi.server.ActivatableRef
sun.rmi.server.Activation
sun.rmi.server.Activation$ActivationSystemImpl
sun.rmi.server.Activation$DelayedAcceptServerSocket
sun.rmi.server.Activation$GroupEntry
sun.rmi.server.Activation$GroupEntry$Watchdog
sun.rmi.server.Activation$ObjectEntry
sun.rmi.server.Activation$Shutdown
sun.rmi.server.Activation$ShutdownHook
sun.rmi.server.ActivationGroupImpl
sun.rmi.server.LoaderHandler
sun.rmi.server.MarshalInputStream
sun.rmi.server.PipeWriter
sun.rmi.server.UnicastRef
sun.rmi.server.UnicastServerRef
sun.rmi.server.Util
sun.rmi.server.WeakClassHashMap
sun.rmi.transport.DGCAckHandler
sun.rmi.transport.DGCClient
sun.rmi.transport.DGCClient$EndpointEntry
sun.rmi.transport.DGCClient$EndpointEntry$RenewCleanThread
sun.rmi.transport.DGCImpl
sun.rmi.transport.DGCImpl$LeaseInfo
sun.rmi.transport.ObjectTable
sun.rmi.transport.ObjectTable$Reaper
sun.rmi.transport.Target
sun.rmi.transport.WeakRef
sun.rmi.transport.proxy.HttpOutputStream
sun.rmi.transport.proxy.HttpReceiveSocket
sun.rmi.transport.proxy.HttpSendInputStream
sun.rmi.transport.proxy.HttpSendSocket
sun.rmi.transport.proxy.RMIMasterSocketFactory
sun.rmi.transport.proxy.RMIMasterSocketFactory$AsyncConnector
sun.rmi.transport.proxy.WrappedSocket
sun.rmi.transport.tcp.ConnectionAcceptor
sun.rmi.transport.tcp.ConnectionMultiplexer
sun.rmi.transport.tcp.MultiplexInputStream
sun.rmi.transport.tcp.MultiplexOutputStream
sun.rmi.transport.tcp.TCPChannel
sun.rmi.transport.tcp.TCPEndpoint
sun.rmi.transport.tcp.TCPEndpoint$FQDN

```



```

sun.rmi.transport.tcp.TCPTransport
sun.rmi.transport.tcp.TCPTransport$ConnectionHandler
sun.security.*
sun.swing.*
sun.text.resources.DateFormatZoneData
sun.text.resources.LocaleData
sun.tools.asm.SwitchData
sun.tools.hprof.Tracker
sun.tools.jar.JarVerifierStream
sun.tools.jar.Main
sun.tools.java.Constants
sun.tools.java.Identifier
sun.tools.java.Imports
sun.tools.java.MemberDefinition
sun.tools.java.RuntimeConstants
sun.tools.java.Type
sun.tools.javac.Main
sun.tools.javac.SourceClass
sun.tools.javap.ClassData
sun.tools.javap.Constants
sun.tools.javap.MethodData
sun.tools.javap.RuntimeConstants
sun.tools.javap.Tables
sun.tools.jstat.JStatLogger
sun.tools.jstat.Jstat$2
sun.tools.jstatd.RemoteHostImpl
sun.tools.native2ascii.Main
sun.tools.serialver.SerialVer
sun.tools.tree.SynchronizedStatement
sun.util.calendar.CalendarSystem
sun.util.calendar.ZoneInfo
sun.util.calendar.ZoneInfoFile
javax.crypto.*
javax.net.ServerSocketFactory
javax.net.DefaultServerSocketFactory
javax.net.SocketFactory
javax.net.DefaultSocketFactory
javax.net.ssl.*
sun.net.www.protocol.https.*
com.sun.net.ssl.*
com.sun.management.UnixOperatingSystem
java.io.UnixFileSystem
java.lang.UNIXProcess
java.lang.UNIXProcess$1$1
java.lang.UNIXProcess$2$1
java.lang.UNIXProcess$DeferredCloseInputStream
java.lang.UNIXProcess$Gate
java.util.prefs.FileSystemPreferences
sun.font.NativeStrikeDisposer
sun.font.XMap
sun.io.CharToByteCOMPOUND_TEXT
sun.nio.ch.DevPollArrayWrapper
sun.nio.ch.DevPollSelectorImpl
sun.nio.ch.InheritedChannel
sun.nio.ch.PollSelectorImpl
com.sun.java.util.jar.pack.PropMap
com.sun.java.util.jar.pack.UnpackerImpl
com.sun.org.apache.bcel.internal.verifier.statics.Pass2Verifier$CPESSC_Visit

```

```

or
com.sun.tools.corba.se.idl.Token
com.sun.tools.javac.comp.Attr
com.sun.tools.javac.resources.compiler_zh_CN
com.sun.tools.javah.oldjavah.OldHeaders
sun.font.NativeFont
sun.font.NativeStrike
sun.font.X11TextRenderer
sun.jvm.hotspot.HotSpotAgent
sun.jvm.hotspot.HotSpotAgent$1
sun.jvm.hotspot.bugspot.BugSpotAgent
sun.jvm.hotspot.bugspot.BugSpotAgent$1
sun.jvm.hotspot.debugger.PageCache
sun.jvm.hotspot.debugger.dbx.DbxDiagnosticLocal
sun.jvm.hotspot.debugger.linux.LinuxDebuggerLocal
sun.jvm.hotspot.debugger.linux.LinuxDebuggerLocal$LinuxDebuggerLocalWorkerTh
read
sun.jvm.hotspot.debugger.proc.ProcDebuggerLocal
sun.jvm.hotspot.debugger.win32.Win32DebuggerLocal
sun.jvm.hotspot.debugger.windbg.WindbgDebuggerLocal
sun.jvm.hotspot.interpreter.Bytecodes
sun.jvm.hotspot.interpreter.OopMapCacheEntry
sun.jvm.hotspot.interpreter.OopMapForCacheEntry
sun.jvm.hotspot.jdi.SAJDIDClassLoader
sun.jvm.hotspot.jdi.VirtualMachineImpl
sun.jvm.hotspot.oops.AccessFlags
sun.jvm.hotspot.tools.JStack
sun.jvm.hotspot.utilities.HeapGXLWriter
sun.jvm.hotspot.utilities.HeapHprofBinWriter
sun.jvm.hotspot.utilities.MessageQueueBackend$MessageQueueImpl
sun.jvm.hotspot.utilities.StreamMonitor
sun.jvm.hotspot.utilities.SystemDictionaryHelper
sun.nio.ch.NativeThread
sun.nio.ch.SinkChannelImpl
sun.nio.ch.SourceChannelImpl
sun.rmi.rmic.iiop.Constants
sun.tools.jconsole.ConnectDialog$ManagedVmTableModel
sun.tools.jconsole.ProxyClient
sun.tools.jconsole.Resources
sun.tools.jconsole.Tab
sun.tools.jconsole.VMPanel
sun.tools.jconsole.Worker
sun.tools.jconsole.inspector.Utills
sun.tools.jconsole.inspector.XMBeanAttributes
sun.tools.jconsole.inspector.XMBeanAttributes$AttributesListener$1
sun.tools.jconsole.inspector.XMBeanNotifications
sun.tools.jconsole.inspector.XMBeanNotifications$XMBeanNotificationsListener
sun.tools.jconsole.inspector.XMBeanTree
sun.tools.jconsole.inspector.XObject
sun.tools.jconsole.inspector.XOperations
sun.tools.jconsole.inspector.XOperations$1
sun.tools.jconsole.inspector.XSheet
sun.tools.jconsole.inspector.XSheet$1
sun.tools.jconsole.inspector.XSheet$3
sun.tools.jconsole.inspector.XSheet$4
sun.tools.jconsole.inspector.XSheet$XMBeanPane
sun.tools.jconsole.inspector.XTree
com.sun.jndi.cosnaming.CNCTX

```

```

com.sun.jndi.cosnaming.OrbReuseTracker
com.sun.org.apache.xml.internal.utils.ThreadControllerWrapper$ThreadControll
er$SafeThread
com.sun.servicetag.Installer
com.sun.servicetag.RegistrationData
com.sun.servicetag.SystemEnvironment
com.sun.servicetag.Util
sun.font.CreatedFontTracker
sun.font.Type1Font$T1DisposerRecord
sun.management.ConnectorAddressLink
sun.net.www.URLConnection
sun.nio.ch.EPollArrayWrapper
sun.nio.ch.EPollSelectorImpl
sun.net.ResourceManager

#-----
# J2SE (JDK6 or later)
#-----
com.sun.activation.registries.LogSupport
com.sun.activation.registries.MailcapFile
com.sun.codemodel.internal.CodeWriter
com.sun.codemodel.internal.util.EncoderFactory
com.sun.istack.internal.Pool$Impl
com.sun.istack.internal.tools.MaskingClassLoader
com.sun.jmx.mbeanserver.Introspector
com.sun.jmx.mbeanserver.MBeanInstantiator
com.sun.jmx.mbeanserver.MBeanIntrospector
com.sun.jmx.mbeanserver.MXBeanLookup
com.sun.jmx.mbeanserver.MXBeanSupport
com.sun.jmx.mbeanserver.OpenConverter
com.sun.jmx.mbeanserver.Repository
com.sun.jmx.mbeanserver.StandardMBeanIntrospector
com.sun.jmx.mbeanserver.Util
com.sun.jmx.remote.internal.ServerNotifForwarder$2
com.sun.net.httpserver.spi.HttpServerProvider
com.sun.org.apache.regexp.internal.RECompiler
com.sun.org.apache.regexp.internal.RETestCase
com.sun.org.apache.xalan.internal.xsltc.trax.SAX2DOM
com.sun.org.apache.xerces.internal.impl.dv.xs.PrecisionDecimalDV$XPrecisionD
ecimal
com.sun.org.apache.xerces.internal.impl.xs.ElementPSVImpl
com.sun.org.apache.xerces.internal.impl.xs.SchemaGrammar$Schema4Annotations
com.sun.org.apache.xerces.internal.jaxp.SAXParserImpl$JAXPSAXParser
com.sun.org.apache.xerces.internal.jaxp.validation.SoftReferenceGrammarPool
com.sun.org.apache.xerces.internal.jaxp.validation.WeakReferenceXMLSchema
com.sun.org.apache.xerces.internal.util.XMLCatalogResolver
com.sun.org.apache.xml.internal.resolver.Catalog
com.sun.org.apache.xml.internal.security.Init
com.sun.org.apache.xml.internal.security.exceptions.XMLSecurityException
com.sun.org.apache.xml.internal.security.exceptions.XMLSecurityRuntimeExcept
ion
com.sun.org.apache.xml.internal.security.keys.content.x509.XMLX509SKI
com.sun.org.apache.xml.internal.security.utils.I18n
com.sun.org.apache.xml.internal.security.utils.IdResolver
com.sun.tools.attach.spi.AttachProvider
com.sun.tools.hat.internal.model.Snapshot
com.sun.tools.hat.internal.oql.OQLEngine
com.sun.tools.hat.internal.parser.FileReadBuffer

```

```

com.sun.tools.hat.internal.parser.MappedReadBuffer
com.sun.tools.hat.internal.server.PlatformClasses
com.sun.tools.internal.ws.processor.model.AsyncOperation
com.sun.tools.internal.ws.processor.util.IndentingWriter
com.sun.tools.internal.ws.wscompile.Options
com.sun.tools.internal.ws.wsdl.framework.TWSDLParserContextImpl
com.sun.tools.internal.xjc.Driver$1
com.sun.tools.internal.xjc.SchemaCache
com.sun.tools.internal.xjc.addon.sync.SynchronizedMethodAddOn
com.sun.tools.internal.xjc.api.impl.s2j.SchemaCompilerImpl
com.sun.tools.internal.xjc.reader.xmlschema.bindinfo.BindInfo
com.sun.tools.javac.api.JavacTaskImpl
com.sun.tools.javac.api.JavacTaskImpl$1
com.sun.tools.javac.code.Symbol$VarSymbol
com.sun.tools.javac.processing.JavacFiler
com.sun.tools.javac.processing.JavacFiler$FilerOutputFileObject
com.sun.tools.javac.processing.JavacFiler$FilerOutputStream
com.sun.tools.javac.processing.JavacFiler$FilerWriter
com.sun.tools.javac.processing.JavacProcessingEnvironment$DiscoveredProcessors$ProcessorStateIterator
com.sun.tools.javac.processing.JavacProcessingEnvironment$NameProcessIterator
com.sun.tools.javac.processing.JavacProcessingEnvironment$ProcessorState
com.sun.tools.javac.sym.CreateSymbols
com.sun.tools.javac.util.DefaultFileManager
com.sun.tools.javac.util.Log
com.sun.tools.javac.util.Paths
com.sun.tools.javac.zip.ZipFileIndex
com.sun.tools.jconsole.JConsolePlugin
com.sun.tools.jdi.EventRequestManagerImpl$ClassPrepareRequestImpl
com.sun.tools.jdi.MonitorInfoImpl
com.sun.xml.internal.bind.AccessorFactoryImpl
com.sun.xml.internal.bind.v2.runtime.JAXBContextImpl
com.sun.xml.internal.bind.v2.runtime.MarshallerImpl
com.sun.xml.internal.bind.v2.runtime.reflect.Lister
com.sun.xml.internal.bind.v2.runtime.reflect.Lister$IDREFS$Pack
com.sun.xml.internal.bind.v2.runtime.reflect.TransducedAccessor$IDREFTransducedAccessorImpl
com.sun.xml.internal.bind.v2.runtime.reflect.TransducedAccessor$IDREFTransducedAccessorImpl$1
com.sun.xml.internal.bind.v2.runtime.reflect.opt.Injector
com.sun.xml.internal.bind.v2.runtime.unmarshaller.UnmarshallingContext
com.sun.xml.internal.fastinfoset.AbstractResourceBundle
com.sun.xml.internal.fastinfoset.CommonResourceBundle
com.sun.xml.internal.messaging.saa.j.SOAPExceptionImpl
com.sun.xml.internal.messaging.saa.j.packaging.mime.MessagingException
com.sun.xml.internal.messaging.saa.j.packaging.mime.internet.MimeMultipart
com.sun.xml.internal.messaging.saa.j.soap.AttachmentPartImpl
com.sun.xml.internal.messaging.saa.j.soap.MessageImpl
com.sun.xml.internal.messaging.saa.j.util.ParserPool
com.sun.xml.internal.messaging.saa.j.util.TeeInputStream
com.sun.xml.internal.rngom.parse.compact.JavaCharStream
com.sun.xml.internal.ws.api.pipe.Engine
com.sun.xml.internal.ws.api.pipe.Engine$DaemonThreadFactory
com.sun.xml.internal.ws.api.pipe.Fiber
com.sun.xml.internal.ws.api.streaming.XMLStreamReaderFactory$Default
com.sun.xml.internal.ws.api.streaming.XMLStreamWriterFactory$Default
com.sun.xml.internal.ws.client.AsyncResponseImpl

```

```

com.sun.xml.internal.ws.client.ResponseImpl
com.sun.xml.internal.ws.client.Stub
com.sun.xml.internal.ws.client.WSServiceDelegate
com.sun.xml.internal.ws.client.dispatch.DispatchImpl
com.sun.xml.internal.ws.client.dispatch.DispatchImpl$DispatchAsyncInvoker$1
com.sun.xml.internal.ws.client.sei.AsyncMethodHandler
com.sun.xml.internal.ws.client.sei.AsyncMethodHandler$Invoker
com.sun.xml.internal.ws.client.sei.CallbackMethodHandler
com.sun.xml.internal.ws.server.StatefulInstanceResolver
com.sun.xml.internal.ws.server.StatefulInstanceResolver$Instance
com.sun.xml.internal.ws.server.StatefulInstanceResolver$Instance$1
com.sun.xml.internal.ws.server.WSEndpointImpl
com.sun.xml.internal.ws.transport.http.HttpAdapter
com.sun.xml.internal.ws.transport.http.client.CookieJar
com.sun.xml.internal.ws.transport.http.client.HttpTransportPipe
com.sun.xml.internal.ws.transport.http.server.EndpointImpl
com.sun.xml.internal.ws.transport.http.server.HttpEndpoint
com.sun.xml.internal.ws.transport.http.server.ServerMgr
com.sun.xml.internal.ws.transport.http.server.WSHttpHandler
com.sun.xml.internal.ws.transport.http.server.WSHttpHandler$HttpHandlerRunna
ble
com.sun.xml.internal.ws.util.CompletedFuture
com.sun.xml.internal.ws.util.DOMUtil
com.sun.xml.internal.ws.util.Pool
com.sun.xml.internal.ws.util.pipe.StandalonePipeAssembler
com.sun.xml.internal.ws.util.pipe.StandaloneTubeAssembler
com.sun.xml.internal.xsom.impl.scd.SimpleCharStream
java.beans.java_util_Collections$SynchronizedCollection_PersistenceDelegate
java.beans.java_util_Collections$SynchronizedList_PersistenceDelegate
java.beans.java_util_Collections$SynchronizedMap_PersistenceDelegate
java.beans.java_util_Collections$SynchronizedRandomAccessList_PersistenceDel
egate
java.beans.java_util_Collections$SynchronizedSet_PersistenceDelegate
java.beans.java_util_Collections$SynchronizedSortedMap_PersistenceDelegate
java.beans.java_util_Collections$SynchronizedSortedSet_PersistenceDelegate
java.io.Console
java.io.Console$LineReader
java.io.DeleteOnExitHook
java.lang.ApplicationShutdownHooks
java.lang.Byte$ByteCache
java.lang.Character$CharacterCache
java.lang.Integer$IntegerCache
java.lang.Long$LongCache
java.lang.SecurityException
java.lang.Short$ShortCache
java.lang.VirtualMachineError
java.lang.management.GarbageCollectorMXBean
java.lang.management.MemoryManagerMXBean
java.lang.management.MemoryPoolMXBean
java.rmi.server.ObjID
java.sql.DriverService
java.text.NumberFormat
java.util.Arrays
java.util.ServiceLoader$LazyIterator
javax.activation.DataHandler
javax.activation.MailcapCommandMap
javax.activation.MimetypesFileTypeMap
javax.activation.ObjectDataContentHandler

```

```

javax.annotation.processing.AbstractProcessor
javax.management.monitor.CounterMonitor$CounterMonitorObservedObject
javax.management.monitor.GaugeMonitor$GaugeMonitorObservedObject
javax.management.monitor.Monitor$DaemonThreadFactory
javax.management.monitor.Monitor$MonitorTask
javax.management.monitor.Monitor$ObservedObject
javax.management.monitor.Monitor$SchedulerTask
javax.management.monitor.StringMonitor$StringMonitorObservedObject
javax.management.openmbean.OpenType
javax.management.remote.rmi._RMICConnectionImpl_Tie
javax.management.remote.rmi._RMICConnection_Stub
javax.management.remote.rmi._RMIServerImpl_Tie
javax.management.remote.rmi._RMIServer_Stub
javax.tools.DiagnosticCollector
javax.tools.StandardLocation
javax.tools.ToolProvider
javax.tools.ToolProvider$Lazy
javax.xml.bind.ContextFinder
javax.xml.bind.JAXBException
javax.xml.bind.TypeConstraintException
javax.xml.soap.SOAPException
javax.xml.stream.FactoryFinder
javax.xml.ws.Service
sun.font.FontDesignMetrics
sun.font.FontDesignMetrics$KeyReference
sun.font.GlyphLayout$SDCache
sun.jkernel.BackgroundDownloader
sun.jkernel.Bundle
sun.jkernel.DownloadManager
sun.jkernel.Mutex
sun.management.Flag
sun.management.HotSpotDiagnostic
sun.misc.ClassLoaderUtil
sun.misc.Launcher
sun.misc.MetaIndex
sun.misc.VMSupport
sun.misc.Version
sun.net.httpserver.ContextList
sun.net.httpserver.HttpConnection
sun.net.httpserver.HttpServerImpl
sun.net.httpserver.HttpsServerImpl
sun.net.httpserver.LeftOverInputStream
sun.net.httpserver.Request$ReadStream
sun.net.httpserver.Request$WriteStream
sun.net.httpserver.SSLStreams
sun.net.httpserver.SSLStreams$EngineWrapper
sun.net.httpserver.SelectorCache
sun.net.httpserver.SelectorCache$CacheCleaner
sun.net.httpserver.ServerImpl
sun.net.httpserver.ServerImpl$Dispatcher
sun.net.httpserver.ServerImpl$ServerTimerTask
sun.net.www.http.KeepAliveStreamCleaner
sun.net.www.protocol.http.HttpURLConnection$HttpInputStream
sun.net.www.protocol.http.InMemoryCookieStore
sun.net.www.protocol.http.NegotiateAuthentication
sun.nio.ch.FileChannelImpl$SharedFileLockTable
sun.nio.ch.FileChannelImpl$SimpleFileLockTable
sun.rmi.runtime.RuntimeUtil

```

```

sun.rmi.runtime.RuntimeUtil$1
sun.tools.attach.HotSpotAttachProvider
sun.tools.attach.HotSpotVirtualMachine
sun.tools.attach.WindowsVirtualMachine
sun.tools.attach.WindowsVirtualMachine$PipedInputStream
sun.tools.jconsole.ClassTab$2
sun.tools.jconsole.HTMLPane
sun.tools.jconsole.InternalDialog$MastheadIcon
sun.tools.jconsole.JConsole
sun.tools.jconsole.MemoryTab$4
sun.tools.jconsole.Plotter
sun.tools.jconsole.ProxyClient$SnapshotInvocationHandler
sun.tools.jconsole.SummaryTab
sun.tools.jconsole.SummaryTab$1
sun.tools.jconsole.ThreadTab$1
sun.tools.jconsole.inspector.XMBean
sun.tools.jconsole.inspector.XMBeanNotifications$XMBeanNotificationsListener
$1
sun.tools.jconsole.resources.JConsoleResources
sun.tools.jconsole.resources.JConsoleResources_ja
sun.tools.jconsole.resources.JConsoleResources_zh_CN
sun.util.LocaleServiceProviderPool
sun.util.TimeZoneNameUtility
com.sun.codemodel.internal.JJavaName
com.sun.codemodel.internal.JMods
com.sun.tools.hat.internal.model.StackFrame
com.sun.tools.internal.ws.processor.generator.Names
com.sun.tools.javac.parser.Token
com.sun.xml.internal.bind.api.impl.NameUtil
java.lang.management.ThreadInfo
javax.lang.model.SourceVersion
sun.jvm.hotspot.memory.SystemDictionary
sun.nio.cs.ext.COMPOUND_TEXT_Encoder
sun.nio.cs.ext.CompoundTextSupport
sun.tools.attach.LinuxVirtualMachine
sun.tools.attach.LinuxVirtualMachine$SocketInputStream
com.sun.org.apache.xalan.internal.xsltc.runtime.BasisLibrary
com.sun.xml.internal.org.jvnet.mimepull.DataFile
com.sun.xml.internal.org.jvnet.mimepull.DataHead
com.sun.xml.internal.org.jvnet.mimepull.DataHead$ReadOnceStream
com.sun.xml.internal.org.jvnet.mimepull.MIMEMessage
com.sun.xml.internal.org.jvnet.mimepull.WeakDataFile
com.sun.xml.internal.ws.api.server.HttpEndpoint
com.sun.xml.internal.ws.client.AsyncInvoker
com.sun.xml.internal.ws.client.sei.AsyncMethodHandler$SEIAsyncInvoker$1
com.sun.xml.internal.ws.encoding.MimeCodec
com.sun.xml.internal.ws.model WrapperBeanGenerator
com.sun.xml.internal.ws.server.JMXAgent
sun.jvm.hotspot.CommandProcessor$9
sun.jvm.hotspot.CommandProcessor$9$1
sun.jvm.hotspot.SALauncherLoader
sun.jvm.hotspot.ui.AnnotatedMemoryPanel
sun.jvm.hotspot.ui.classbrowser.HTMLGenerator
sun.jvm.hotspot.ui.FindInCodeCachePanel$Visitor$1
sun.jvm.hotspot.ui.ProcessListPanel
sun.jvm.hotspot.utilities.soql.SOQLEngine
sun.net.spi.DefaultProxySelector$3

```



```

#-----
# Package, class added to "J2SE"
# (Did not exist in version 0870 of JDK5 or JDK6, but was included in versio
n 0900)
#-----
com.rsa.*
com.sun.tools.javac.util.CloseableURLClassLoader
java.text.DateFormatSymbols
java.text.SimpleDateFormat
sun.nio.cs.ext.Big5
sun.nio.cs.ext.Big5_Solaris
sun.nio.cs.ext.MS950

#-----
# Package, class added to "J2SE"
# (Did not exist in 0900, but was included in 0950)
#-----
com.sun.beans.AppContext
com.sun.org.apache.xalan.internal.utils.ObjectFactory

#-----
# J2SE (JDK 7 or later)
#-----
com.sun.beans.editors.EnumEditor
com.sun.beans.finder.PersistenceDelegateFinder
com.sun.beans.finder.PropertyEditorFinder
com.sun.beans.TypeResolver
com.sun.istack.internal.tools.ParallelWorldClassLoader
com.sun.java.accessibility.AccessBridge
com.sun.java.accessibility.util.AccessibilityListenerList
com.sun.java.accessibility.util.AWTEventMonitor
com.sun.java.accessibility.util.AWTEventMonitor$AWTEventsListener
com.sun.java.accessibility.util.ComponentEvtDispatchThread
com.sun.java.accessibility.util.EventQueueMonitor
com.sun.java.accessibility.util.GUIInitializedListener
com.sun.java.accessibility.util.GUIInitializedMulticaster
com.sun.java.accessibility.util.java.awt.ButtonTranslator
com.sun.java.accessibility.util.java.awt.CheckboxTranslator
com.sun.java.accessibility.util.java.awt.LabelTranslator
com.sun.java.accessibility.util.java.awt.ListTranslator
com.sun.java.accessibility.util.java.awt.TextComponentTranslator
com.sun.java.accessibility.util.SwingEventMonitor
com.sun.java.accessibility.util.SwingEventMonitor$SwingEventListener
com.sun.java.accessibility.util.TopLevelWindowListener
com.sun.java.accessibility.util.TopLevelWindowMulticaster
com.sun.java.accessibility.util.Translator
com.sun.java.util.jar.pack.Driver
com.sun.java.util.jar.pack.PackageReader
com.sun.java.util.jar.pack.PackageWriter
com.sun.java.util.jar.pack.PackerImpl
com.sun.java.util.jar.pack.Utills
com.sun.jmx.mbeanserver.DefaultMXBeanMappingFactory
com.sun.jmx.remote.internal.ServerNotifForwarder$NotifForwarderBufferFilter
com.sun.naming.internal.ResourceManager$AppletParameter
com.sun.nio.zipfs.ZipCoder
com.sun.nio.zipfs.ZipDirectoryStream
com.sun.nio.zipfs.ZipDirectoryStream$1
com.sun.nio.zipfs.ZipFileSystem

```



```

com.sun.nio.zipfs.ZipFileSystem$EntryInputStream
com.sun.nio.zipfs.ZipFileSystemProvider
com.sun.nio.zipfs.ZipUtils
com.sun.org.apache.xalan.internal.utils.SecuritySupport
com.sun.org.apache.xalan.internal.utils.XMLSecurityPropertyManager
com.sun.org.apache.xalan.internal.utils.XMLSecurityPropertyManager$Property
com.sun.org.apache.xalan.internal.utils.XMLSecurityPropertyManager$State
com.sun.org.apache.xerces.internal.impl.xpath.regex.CaseInsensitiveMap
com.sun.org.apache.xerces.internal.impl.xpath.regex.RegularExpression$Context
com.sun.org.apache.xerces.internal.utils.SecuritySupport
com.sun.org.apache.xerces.internal.utils.XMLSecurityPropertyManager
com.sun.org.apache.xerces.internal.utils.XMLSecurityPropertyManager$Property
com.sun.org.apache.xerces.internal.utils.XMLSecurityPropertyManager$State
com.sun.org.apache.xerces.internal.xinclude.SecuritySupport
com.sun.org.apache.xerces.internal.xinclude.SecuritySupport$1
com.sun.org.apache.xerces.internal.xinclude.SecuritySupport$2
com.sun.org.apache.xerces.internal.xinclude.SecuritySupport$3
com.sun.org.apache.xerces.internal.xinclude.SecuritySupport$4
com.sun.org.apache.xerces.internal.xinclude.SecuritySupport$5
com.sun.org.apache.xerces.internal.xinclude.SecuritySupport$6
com.sun.org.apache.xerces.internal.xinclude.SecuritySupport$7
com.sun.org.apache.xerces.internal.xinclude.SecuritySupport$8
com.sun.org.apache.xml.internal.security.algorithms.JCEMapper
com.sun.org.apache.xml.internal.security.algorithms.SignatureAlgorithm
com.sun.org.apache.xml.internal.security.cl4n.Canonicalizer
com.sun.org.apache.xml.internal.security.keys.keyresolver.KeyResolver
com.sun.org.apache.xml.internal.security.transforms.Transform
com.sun.org.apache.xml.internal.security.utils.ElementProxy
com.sun.org.apache.xml.internal.security.utils.resolver.ResourceResolver
com.sun.org.apache.xml.internal.security.utils.UnsyncBufferedOutputStream$1
com.sun.org.apache.xml.internal.security.utils.UnsyncByteArrayOutputStream$1
com.sun.org.apache.xml.internal.serialize.SecuritySupport
com.sun.org.apache.xml.internal.serialize.SecuritySupport$1
com.sun.org.apache.xml.internal.serialize.SecuritySupport$2
com.sun.org.apache.xml.internal.serialize.SecuritySupport$3
com.sun.org.apache.xml.internal.serialize.SecuritySupport$4
com.sun.org.apache.xml.internal.serialize.SecuritySupport$5
com.sun.org.apache.xml.internal.serialize.SecuritySupport$6
com.sun.org.apache.xml.internal.serialize.SecuritySupport$7
com.sun.org.apache.xml.internal.serialize.SecuritySupport$8
com.sun.org.glassfish.external.amx.AMXGlassfish$BootAMXCallback
com.sun.org.glassfish.external.amx.AMXGlassfish$WaitForDomainRootListenerCallback
com.sun.org.glassfish.external.amx.MBeanListener$CallbackImpl
com.sun.org.glassfish.external.statistics.impl.AverageRangeStatisticImpl
com.sun.org.glassfish.external.statistics.impl.BoundaryStatisticImpl
com.sun.org.glassfish.external.statistics.impl.BoundedRangeStatisticImpl
com.sun.org.glassfish.external.statistics.impl.CountStatisticImpl
com.sun.org.glassfish.external.statistics.impl.RangeStatisticImpl
com.sun.org.glassfish.external.statistics.impl.StatisticImpl
com.sun.org.glassfish.external.statistics.impl.StatsImpl
com.sun.org.glassfish.external.statistics.impl.StringStatisticImpl
com.sun.org.glassfish.external.statistics.impl.TimeStatisticImpl
com.sun.org.glassfish.gmbal.util.GenericConstructor
com.sun.org.glassfish.gmbal.util.GenericConstructor$1
com.sun.tools.classfile.Attribute$Factory
com.sun.tools.internal.xjc.reader.Ring

```

```

com.sun.tools.javac.api.ClientCodeWrapper$WrappedDiagnosticListener
com.sun.tools.javac.api.ClientCodeWrapper$WrappedFileObject
com.sun.tools.javac.api.ClientCodeWrapper$WrappedJavaFileManager
com.sun.tools.javac.api.ClientCodeWrapper$WrappedJavaFileObject
com.sun.tools.javac.api.ClientCodeWrapper$WrappedTaskListener
com.sun.tools.javac.code.Flags$Flag
com.sun.tools.javac.file.CacheFSInfo
com.sun.tools.javac.file.ZipFileIndex
com.sun.tools.javac.file.ZipFileIndexCache
com.sun.tools.javac.processing.JavacProcessingEnvironment$ServiceIterator
com.sun.tools.javac.Server
com.sun.tools.javac.util.BaseFileManager
com.sun.tools.javac.util.SharedNameTable
com.sun.tools.javap.JavapTask
com.sun.xml.internal.bind.v2.runtime.reflect.opt.AccessorInjector
com.sun.xml.internal.ws.model.AbstractWrapperBeanGenerator
com.sun.xml.internal.ws.model.Injector
com.sun.xml.internal.ws.policy.sourcemodel.AssertionData
com.sun.xml.internal.ws.policy.sourcemodel.PolicySourceModel
com.sun.xml.internal.ws.transport.http.server.PortableHttpHandler
com.sun.xml.internal.ws.transport.http.server.PortableHttpHandler$HttpHandle
rRunnable
java.beans.ChangeListenerMap
java.beans.MetaData$java_awt_AWTKeyStroke_PersistenceDelegate
java.beans.MetaData$java_awt_BorderLayout_PersistenceDelegate
java.beans.MetaData$java_awt_CardLayout_PersistenceDelegate
java.beans.MetaData$java_awt_Choice_PersistenceDelegate
java.beans.MetaData$java_awt_Component_PersistenceDelegate
java.beans.MetaData$java_awt_Container_PersistenceDelegate
java.beans.MetaData$java_awt_Font_PersistenceDelegate
java.beans.MetaData$java_awt_font_TextAttribute_PersistenceDelegate
java.beans.MetaData$java_awt_GridBagLayout_PersistenceDelegate
java.beans.MetaData$java_awt_Insets_PersistenceDelegate
java.beans.MetaData$java_awt_List_PersistenceDelegate
java.beans.MetaData$java_awt_Menu_PersistenceDelegate
java.beans.MetaData$java_awt_MenuBar_PersistenceDelegate
java.beans.MetaData$java_awt_MenuShortcut_PersistenceDelegate
java.beans.MetaData$java_awt_SystemColor_PersistenceDelegate
java.beans.MetaData$javax_swing_border_MatteBorder_PersistenceDelegate
java.beans.MetaData$javax_swing_Box_PersistenceDelegate
java.beans.MetaData$javax_swing_DefaultComboBoxModel_PersistenceDelegate
java.beans.MetaData$javax_swing_DefaultListModel_PersistenceDelegate
java.beans.MetaData$javax_swing_JFrame_PersistenceDelegate
java.beans.MetaData$javax_swing_JMenu_PersistenceDelegate
java.beans.MetaData$javax_swing_JTabbedPane_PersistenceDelegate
java.beans.MetaData$javax_swing_ToolTipManager_PersistenceDelegate
java.beans.MetaData$javax_swing_tree_DefaultMutableTreeNode_PersistenceDeleg
ate
java.beans.MetaData$sun_swing_PrintColorUIResource_PersistenceDelegate
java.beans.ThreadGroupContext
java.lang.BootstrapMethodError
java.lang.ClassLoader$ParallelLoaders
java.lang.ClassValue
java.lang.ClassValue$ClassValueMap
java.lang.IllegalStateException
java.lang.invoke.BoundMethodHandle
java.lang.invoke.BoundMethodHandle$SpeciesData
java.lang.invoke.CallSite

```

```

java.lang.invoke.LambdaForm
java.lang.invoke.LambdaForm$NamedFunction
java.lang.invoke.MethodHandle
java.lang.invoke.MethodHandleImpl$BindCaller
java.lang.invoke.MethodHandleImpl$BindCaller$2
java.lang.invoke.MethodHandleNatives
java.lang.invoke.MethodType$WeakInternSet
java.lang.invoke.MutableCallSite
java.lang.StringBuilder
java.lang.StringValue
java.lang.StringValue$StringCache
java.lang.Thread$Caches
java.lang.UNIXProcess$1
java.lang.UNIXProcess$ProcessPipeInputStream
java.lang.UNIXProcess$ProcessPipeOutputStream
java.lang.UnsupportedOperationException
java.net.AbstractPlainDatagramSocketImpl
java.net.AbstractPlainSocketImpl
java.net.DualStackPlainDatagramSocketImpl
java.net.DualStackPlainSocketImpl
java.net.InMemoryCookieStore
java.net.TwoStacksPlainDatagramSocketImpl
java.net.TwoStacksPlainSocketImpl
java.net.URLClassLoader
java.nio.channels.AsynchronousChannelGroup
java.nio.channels.AsynchronousFileChannel
java.nio.channels.AsynchronousSocketChannel
java.nio.channels.Channels$2
java.nio.channels.Channels$3
java.nio.channels.SelectionKey
java.nio.channels.SocketChannel
java.nio.file.attribute.AclEntry
java.nio.file.attribute.AclEntry$1
java.nio.file.attribute.AclEntry$Builder
java.nio.file.attribute.AclEntryFlag
java.nio.file.attribute.AclEntryPermission
java.nio.file.attribute.AclEntryType
java.nio.file.attribute.AclFileAttributeView
java.nio.file.attribute.AttributeView
java.nio.file.attribute.BasicFileAttributes
java.nio.file.attribute.BasicFileAttributeView
java.nio.file.attribute.DosFileAttributes
java.nio.file.attribute.DosFileAttributeView
java.nio.file.attribute.FileAttribute
java.nio.file.attribute.FileAttributeView
java.nio.file.attribute.FileOwnerAttributeView
java.nio.file.attribute.FileStoreAttributeView
java.nio.file.attribute.FileTime
java.nio.file.attribute.FileTime$1
java.nio.file.attribute.FileTime$DaysAndNanos
java.nio.file.attribute.GroupPrincipal
java.nio.file.attribute.PosixFileAttributes
java.nio.file.attribute.PosixFileAttributeView
java.nio.file.attribute.PosixFilePermission
java.nio.file.attribute.PosixFilePermissions
java.nio.file.attribute.PosixFilePermissions$1
java.nio.file.attribute.UserDefinedFileAttributeView
java.nio.file.attribute.UserPrincipal

```

```

java.nio.file.attribute.UserPrincipalLookupService
java.nio.file.attribute.UserPrincipalNotFoundException
java.nio.file.CopyMoveHelper
java.nio.file.Files
java.nio.file.SecureDirectoryStream
java.nio.file.spi.FileSystemProvider
java.sql.DriverManager$2
java.text.DecimalFormatSymbols
java.util.Currency$1
java.util.HashMap
java.util.logging.Level$KnownLevel
java.util.logging.LogManager$LoggerContext
java.util.prefs.FileSystemPreferences$6
java.util.prefs.FileSystemPreferences$7
java.util.Vector$Itr
java.util.Vector$ListItr
java.util.XMLUtils
javax.sql.rowset.serial.SerialClob
sun.dc.DuctusRenderingEngine
sun.dc.DuctusRenderingEngine$FillAdapter
sun.font.CreatedFontTracker$TempFileDeletionHook
sun.font.FcFontConfiguration
sun.font.FontAccess
sun.font.FontConfigManager
sun.font.FontConfigManager$FcCompFont
sun.font.FontConfigManager$FontConfigFont
sun.font.FontConfigManager$FontConfigInfo
sun.font.FontManagerFactory
sun.font.FontManagerFactory$1
sun.font.FontManagerForSGE
sun.font.FontManagerNativeLibrary
sun.font.FontManagerNativeLibrary$1
sun.font.FontScaler
sun.font.FontScalerException
sun.font.FontUtilities
sun.font.FontUtilities$1
sun.font.FreetypeFontScaler
sun.font.GlyphDisposedListener
sun.font.NullFontScaler
sun.font.SunFontManager
sun.font.SunFontManager$1
sun.font.SunFontManager$10
sun.font.SunFontManager$11
sun.font.SunFontManager$12
sun.font.SunFontManager$13
sun.font.SunFontManager$14
sun.font.SunFontManager$2
sun.font.SunFontManager$3
sun.font.SunFontManager$4
sun.font.SunFontManager$5
sun.font.SunFontManager$6
sun.font.SunFontManager$7
sun.font.SunFontManager$8
sun.font.SunFontManager$8$1
sun.font.SunFontManager$9
sun.font.SunFontManager$FamilyDescription
sun.font.SunFontManager$FontRegistrationInfo
sun.font.SunFontManager$TlFilter

```

```

sun.font.SunFontManager$TTFilter
sun.font.SunFontManager$TTorT1Filter
sun.font.T2KFontScaler
sun.font.T2KFontScaler$1
sun.font.Underline
sun.font.XRGlyphCache
sun.font.XRGlyphCache$1
sun.font.XRGlyphCacheEntry
sun.font.XRTextRenderer
sun.invoke.util.ValueConversions
sun.jvm.hotspot.CommandProcessor$12
sun.jvm.hotspot.CommandProcessor$12$1
sun.jvm.hotspot.debugger.bsd.BsdDebuggerLocal
sun.jvm.hotspot.debugger.bsd.BsdDebuggerLocal$BsdDebuggerLocalWorkerThread
sun.jvm.hotspot.runtime.CompilerThread
sun.launcher.LauncherHelper
sun.launcher.resources.launcher
sun.management.jdp.JdpController
sun.management.ManagementFactoryHelper
sun.management.ManagementFactoryHelper$1
sun.management.ManagementFactoryHelper$PlatformLoggingImpl
sun.misc.Hashing
sun.misc.JavaxSecurityAuthKerberosAccess
sun.misc.Launcher$ExtClassLoader
sun.misc.PerfCounter
sun.misc.PostVMInitHook
sun.misc.Service$LazyIterator
sun.net.ftp.FtpClientProvider
sun.net.httpserver.ServerImpl$ServerTimerTask1
sun.net.www.http.HttpCapture
sun.net.www.protocol.http.ntlm.NTLMAuthentication
sun.net.www.protocol.jar.URLJarFile$1
sun.nio.ch.AbstractPollSelectorImpl
sun.nio.ch.AsynchronousChannelGroupImpl
sun.nio.ch.AsynchronousChannelGroupImpl$2
sun.nio.ch.AsynchronousChannelGroupImpl$3
sun.nio.ch.AsynchronousFileChannelImpl
sun.nio.ch.AsynchronousServerSocketChannelImpl
sun.nio.ch.AsynchronousSocketChannelImpl
sun.nio.ch.BsdAsynchronousChannelProvider
sun.nio.ch.CompletedFuture
sun.nio.ch.EPoll
sun.nio.ch.EPollPort
sun.nio.ch.EPollPort$EventHandlerTask
sun.nio.ch.FileChannelImpl$Unmapper
sun.nio.ch.FileDispatcherImpl
sun.nio.ch.Invoker
sun.nio.ch.Iocp
sun.nio.ch.Iocp$EventHandlerTask
sun.nio.ch.KQueue
sun.nio.ch.KQueuePort
sun.nio.ch.KQueuePort$EventHandlerTask
sun.nio.ch.LinuxAsynchronousChannelProvider
sun.nio.ch.MembershipKeyImpl
sun.nio.ch.PendingFuture
sun.nio.ch.PendingIoCache
sun.nio.ch.Port
sun.nio.ch.SctpChannelImpl

```

```

sun.nio.ch.SctpMultiChannelImpl
sun.nio.ch.SctpNet
sun.nio.ch.SctpServerChannelImpl
sun.nio.ch.SharedFileLockTable
sun.nio.ch.SimpleAsynchronousFileChannelImpl
sun.nio.ch.SimpleAsynchronousFileChannelImpl$DefaultExecutorHolder
sun.nio.ch.SolarisAsynchronousChannelProvider
sun.nio.ch.SolarisEventPort
sun.nio.ch.SolarisEventPort$EventHandlerTask
sun.nio.ch.ThreadPool
sun.nio.ch.UnixAsynchronousServerSocketChannelImpl
sun.nio.ch.UnixAsynchronousSocketChannelImpl
sun.nio.ch.UnixAsynchronousSocketChannelImpl$1
sun.nio.ch.UnixAsynchronousSocketChannelImpl$2
sun.nio.ch.WindowsAsynchronousChannelProvider
sun.nio.ch.WindowsAsynchronousFileChannelImpl
sun.nio.ch.WindowsAsynchronousFileChannelImpl$LockTask
sun.nio.ch.WindowsAsynchronousFileChannelImpl$ReadTask
sun.nio.ch.WindowsAsynchronousFileChannelImpl$WriteTask
sun.nio.ch.WindowsAsynchronousServerSocketChannelImpl
sun.nio.ch.WindowsAsynchronousServerSocketChannelImpl$AcceptTask
sun.nio.ch.WindowsAsynchronousSocketChannelImpl
sun.nio.ch.WindowsAsynchronousSocketChannelImpl$ConnectTask
sun.nio.ch.WindowsAsynchronousSocketChannelImpl$ReadTask
sun.nio.ch.WindowsAsynchronousSocketChannelImpl$WriteTask
sun.nio.cs.AbstractCharsetProvider$1
sun.nio.cs.ext.EUC_CN
sun.nio.cs.ext.EUC_KR
sun.nio.cs.ext.GBK
sun.nio.cs.ext.IBM1364
sun.nio.cs.ext.IBM1381
sun.nio.cs.ext.IBM1383
sun.nio.cs.ext.IBM930
sun.nio.cs.ext.IBM933
sun.nio.cs.ext.IBM935
sun.nio.cs.ext.IBM937
sun.nio.cs.ext.IBM939
sun.nio.cs.ext.IBM942
sun.nio.cs.ext.IBM943
sun.nio.cs.ext.IBM948
sun.nio.cs.ext.IBM949
sun.nio.cs.ext.IBM950
sun.nio.cs.ext.IBM970
sun.nio.cs.ext.Johab
sun.nio.cs.ext.MS932
sun.nio.cs.ext.MS936
sun.nio.cs.ext.MS949
sun.nio.fs.AbstractPoller
sun.nio.fs.AbstractPoller$Request
sun.nio.fs.AbstractWatchKey
sun.nio.fs.AbstractWatchService
sun.nio.fs.Cancellable
sun.nio.fs.LinuxNativeDispatcher
sun.nio.fs.PollingWatchService
sun.nio.fs.PollingWatchService$3
sun.nio.fs.PollingWatchService$PollingWatchKey
sun.nio.fs.UnixCopyFile
sun.nio.fs.UnixDirectoryStream

```



```

sun.nio.fs.UnixDirectoryStream$UnixDirectoryIterator
sun.nio.fs.UnixFileAttributes
sun.nio.fs.UnixFileAttributeViews$Basic
sun.nio.fs.UnixFileStore
sun.nio.fs.UnixFilesystem$FileStoreIterator
sun.nio.fs.UnixFilesystemProvider
sun.nio.fs.UnixNativeDispatcher
sun.nio.fs.UnixSecureDirectoryStream
sun.nio.fs.UnixSecureDirectoryStream$BasicFileAttributeViewImpl
sun.nio.fs.UnixSecureDirectoryStream$PosixFileAttributeViewImpl
sun.nio.fs.WindowsDirectoryStream
sun.nio.fs.WindowsDirectoryStream$WindowsDirectoryIterator
sun.nio.fs.WindowsFileAttributes
sun.nio.fs.WindowsFileCopy
sun.nio.fs.WindowsFilesystem$FileStoreIterator
sun.nio.fs.WindowsFilesystemProvider
sun.nio.fs.WindowsNativeDispatcher
sun.nio.fs.WindowsPath
sun.nio.fs.WindowsSecurity
sun.nio.fs.WindowsSecurity$1
sun.nio.fs.WindowsSecurity$Privilege
sun.nio.fs.WindowsSecurityDescriptor
sun.text.normalizer.UBiDiProps
sun.tools.jconsole.inspector.XMBeanAttributes$1
sun.tools.jconsole.inspector.XMBeanAttributes$2
sun.tools.jconsole.inspector.XMBeanAttributes$AttributesListener
sun.util.calendar.LocalGregorianCalendar$1
sun.util.locale.LocaleObjectCache
sun.util.logging.PlatformLogger

#-----
# J2SE (JDK 8 or later)
#-----
com.sun.beans.util.Cache
com.sun.istack.internal.tools.DefaultAuthenticator
com.sun.java.accessibility.AccessBridge$102
com.sun.java.accessibility.AccessBridge$152
com.sun.java.accessibility.AccessBridge$DefaultNativeWindowHandler
com.sun.java.accessibility.AccessBridge$InvocationUtils
com.sun.java.accessibility.AccessBridge$InvocationUtils$1
com.sun.java.accessibility.AccessBridge$InvocationUtils$CallableWrapper
com.sun.java.accessibility.AccessBridge$ObjectReferences
com.sun.org.apache.xalan.internal.utils.XMLSecurityManager
com.sun.org.apache.xalan.internal.utils.XMLSecurityManager$Limit
com.sun.org.apache.xalan.internal.utils.XMLSecurityManager$NameMap
com.sun.org.apache.xalan.internal.utils.XMLSecurityManager$State
com.sun.org.apache.xerces.internal.utils.XMLSecurityManager
com.sun.org.apache.xerces.internal.utils.XMLSecurityManager$Limit
com.sun.org.apache.xerces.internal.utils.XMLSecurityManager$NameMap
com.sun.org.apache.xerces.internal.utils.XMLSecurityManager$State
com.sun.org.apache.xml.internal.security.algorithms.implementations.SignatureECDSA$SignatureECDSASHA256
com.sun.org.apache.xml.internal.security.algorithms.implementations.SignatureECDSA$SignatureECDSASHA384
com.sun.org.apache.xml.internal.security.algorithms.implementations.SignatureECDSA$SignatureECDSASHA512
com.sun.org.apache.xml.internal.security.c14n.implementations.Canonicalizer1
1$1

```

```

com.sun.org.apache.xml.internal.security.c14n.implementations.Canonicalizer2
0010315$1
com.sun.org.apache.xml.internal.security.c14n.implementations.CanonicalizerP
hysical
com.sun.org.apache.xml.internal.security.encryption.AbstractSerializer
com.sun.org.apache.xml.internal.security.encryption.DocumentSerializer
com.sun.org.apache.xml.internal.security.encryption.Serializer
com.sun.org.apache.xml.internal.security.keys.content.DEREncodedKeyValue
com.sun.org.apache.xml.internal.security.keys.content.KeyInfoReference
com.sun.org.apache.xml.internal.security.keys.content.x509.XMLX509Digest
com.sun.org.apache.xml.internal.security.keys.keyresolver.implementations.DE
REncodedKeyValueResolver
com.sun.org.apache.xml.internal.security.keys.keyresolver.implementations.Ke
yInfoReferenceResolver
com.sun.org.apache.xml.internal.security.keys.keyresolver.implementations.Pr
ivateKeyResolver
com.sun.org.apache.xml.internal.security.keys.keyresolver.implementations.Se
cretKeyResolver
com.sun.org.apache.xml.internal.security.keys.keyresolver.implementations.Si
ngleKeyResolver
com.sun.org.apache.xml.internal.security.keys.keyresolver.implementations.X5
09DigestResolver
com.sun.org.apache.xml.internal.security.keys.storage.implementations.KeySto
reResolver$KeyStoreIterator$1
com.sun.org.apache.xml.internal.security.signature.Reference$2
com.sun.org.apache.xml.internal.security.signature.Reference$2$1
com.sun.org.apache.xml.internal.security.signature.reference.ReferenceData
com.sun.org.apache.xml.internal.security.signature.reference.ReferenceNodeSe
tData
com.sun.org.apache.xml.internal.security.signature.reference.ReferenceOctetS
treamData
com.sun.org.apache.xml.internal.security.signature.reference.ReferenceSubTre
eData
com.sun.org.apache.xml.internal.security.signature.reference.ReferenceSubTre
eData$DelayedNodeIterator
com.sun.org.apache.xml.internal.security.utils.ClassLoaderUtils
com.sun.org.apache.xml.internal.security.utils.ClassLoaderUtils$1
com.sun.org.apache.xml.internal.security.utils.DOMNamespaceContext
com.sun.org.apache.xml.internal.security.utils.JDKXPathAPI
com.sun.org.apache.xml.internal.security.utils.JDKXPathFactory
com.sun.org.apache.xml.internal.security.utils.resolver.ResourceResolverCont
ext
com.sun.org.apache.xml.internal.security.utils.SignatureElementProxy
com.sun.org.apache.xml.internal.security.utils.XalanXPathAPI
com.sun.org.apache.xml.internal.security.utils.XalanXPathFactory
com.sun.org.apache.xml.internal.security.utils.XPathAPI
com.sun.org.apache.xml.internal.security.utils.XPathFactory
com.sun.org.apache.xml.internal.serializer.utils.SerializerMessages_pt_BR
com.sun.source.util.Trees
com.sun.tools.classfile.ClassFile
com.sun.tools.doclets.internal.toolkit.util.DocFileFactory
com.sun.tools.doclets.internal.toolkit.util.PathDocFileFactory$StandardDocFi
le
com.sun.tools.internal.ws.processor.modeler.annotation.WebServiceAp
com.sun.tools.internal.ws.wscompile.WsimportOptions$RereadInputStream
com.sun.tools.javac.code.Lint
com.sun.tools.javac.jvm.Gen
com.sun.tools.javac.parser.Tokens$TokenKind

```



```

com.sun.tools.javac.processing.JavacProcessingEnvironment$Round
com.sun.tools.javac.resources.javac
com.sun.tools.javac.util.ServiceLoader$LazyIterator
com.sun.tools.javadoc.api.JavadocTaskImpl
com.sun.xml.internal.bind.DatatypeConverterImpl
com.sun.xml.internal.ws.api.message.Packet
com.sun.xml.internal.ws.api.server.Container
com.sun.xml.internal.ws.api.server.MethodUtil
com.sun.xml.internal.ws.api.server.ThreadLocalContainerResolver
com.sun.xml.internal.ws.api.server.ThreadLocalContainerResolver$2
com.sun.xml.internal.ws.api.WSService
com.sun.xml.internal.ws.binding.BindingImpl
com.sun.xml.internal.ws.client.sei.MethodUtil
com.sun.xml.internal.ws.commons.xmlutil.Converter
com.sun.xml.internal.ws.db.DatabindingImpl
com.sun.xml.internal.ws.dump.LoggingDumpTube
com.sun.xml.internal.ws.dump.MessageDumpingFeature
com.sun.xml.internal.ws.dump.MessageDumpingTube
com.sun.xml.internal.ws.policy.privateutil.MethodUtil
com.sun.xml.internal.ws.server.provider.AsyncProviderInvokerTube
com.sun.xml.internal.ws.server.provider.AsyncProviderInvokerTube$AsyncProvid
erCallbackImpl
com.sun.xml.internal.ws.server.WSEndpointMOMProxy
com.sun.xml.internal.ws.transport.http.HttpAdapter$HttpToolkit
com.sun.xml.internal.ws.util.InjectionPlan
com.sun.xml.internal.ws.util.InjectionPlan$Compositor
com.sun.xml.internal.ws.util.ServiceFinder$LazyIterator
java.beans.WeakIdentityMap
java.io.FilterOutputStream
java.lang.invoke.InnerClassLambdaMetafactory
java.lang.invoke.MemberName
java.lang.invoke.MethodHandleImpl
java.lang.invoke.MethodHandles$Lookup
java.lang.invoke.MethodType$ConcurrentWeakInternSet
java.lang.invoke.MethodTypeForm
java.lang.Process
java.lang.reflect.Executable
java.lang.reflect.Parameter
java.lang.reflect.Proxy$ProxyClassFactory
java.lang.reflect.WeakCache
java.lang.reflect.WeakCache$CacheKey
java.lang.reflect.WeakCache$Factory
java.lang.UNIXProcess$2
java.net.URLPermission
java.net.URLPermission$Authority
java.sql.DriverAction
java.sql.JDBCType
java.sql.SQLType
java.time.chrono.AbstractChronology
java.time.format.DateTimeFormatterBuilder$LocalizedPrinterParser
java.time.format.DateTimeFormatterBuilder$ZoneIdPrinterParser
java.time.format.DateTimeFormatterBuilder$ZoneTextPrinterParser
java.time.format.DateTimeTextProvider
java.time.format.DecimalStyle
java.time.temporal.WeekFields
java.time.zone.TzdbZoneRulesProvider
java.time.zone.ZoneRules
java.time.zone.ZoneRulesProvider

```

```

java.time.ZoneOffset
java.util.ArrayPrefixHelpers$CumulateTask
java.util.ArrayPrefixHelpers$DoubleCumulateTask
java.util.ArrayPrefixHelpers$IntCumulateTask
java.util.ArrayPrefixHelpers$LongCumulateTask
java.util.ArraysParallelSortHelpers$EmptyCompleter
java.util.ArraysParallelSortHelpers$FJByte$Merger
java.util.ArraysParallelSortHelpers$FJByte$Sorter
java.util.ArraysParallelSortHelpers$FJChar$Merger
java.util.ArraysParallelSortHelpers$FJChar$Sorter
java.util.ArraysParallelSortHelpers$FJDouble$Merger
java.util.ArraysParallelSortHelpers$FJDouble$Sorter
java.util.ArraysParallelSortHelpers$FJFloat$Merger
java.util.ArraysParallelSortHelpers$FJFloat$Sorter
java.util.ArraysParallelSortHelpers$FJInt$Merger
java.util.ArraysParallelSortHelpers$FJInt$Sorter
java.util.ArraysParallelSortHelpers$FJLong$Merger
java.util.ArraysParallelSortHelpers$FJLong$Sorter
java.util.ArraysParallelSortHelpers$FJObject$Merger
java.util.ArraysParallelSortHelpers$FJObject$Sorter
java.util.ArraysParallelSortHelpers$FJShort$Merger
java.util.ArraysParallelSortHelpers$FJShort$Sorter
java.util.ArraysParallelSortHelpers$Relay
java.util.Collections$SynchronizedNavigableMap
java.util.Collections$SynchronizedNavigableSet
java.util.logging.LogManager$LoggerWeakRef
java.util.SplittableRandom
java.util.stream.AbstractShortCircuitTask
java.util.stream.AbstractTask
java.util.stream.DistinctOps$1
java.util.stream.DoublePipeline$5$1
java.util.stream.FindOps$FindTask
java.util.stream.ForEachOps$ForEachOrderedTask
java.util.stream.ForEachOps$ForEachTask
java.util.stream.IntPipeline$7$1
java.util.stream.LongPipeline$6$1
java.util.stream.Nodes$CollectorTask
java.util.stream.Nodes$SizedCollectorTask
java.util.stream.Nodes$ToArrayTask
java.util.stream.ReduceOps$ReduceTask
java.util.stream.ReferencePipeline$10$1
java.util.stream.ReferencePipeline$7$1
java.util.stream.ReferencePipeline$8$1
java.util.stream.ReferencePipeline$9$1
java.util.stream.SliceOps$SliceTask
java.util.stream.Streams$1
java.util.stream.Streams$2
java.util.stream.StreamSpliterators$DistinctSpliterator
java.util.stream.StreamSpliterators$UnorderedSliceSpliterator
java.util.Vector$VectorSpliterator
javax.activation.CommandMap
javax.activation.FileTypeMap
javax.sql.rowset.spi.SyncFactory$2
jdk.internal.dynalink.beans.OverloadedMethod
jdk.internal.dynalink.ChainedCallSite
jdk.internal.dynalink.support.CallSiteDescriptorFactory
jdk.internal.dynalink.support.ClassMap
jdk.internal.org.objectweb.asm.commons.InstructionAdapter

```

```

jdk.internal.org.objectweb.asm.commons.SerialVersionUIDAdder
jdk.internal.org.objectweb.asm.util.Textifier
jdk.internal.util.xml.PropertiesDefaultHandler
jdk.nashorn.api.scripting.ScriptObjectMirror
jdk.nashorn.api.scripting.ScriptUtils
jdk.nashorn.internal.codegen.ClassEmitter
jdk.nashorn.internal.codegen.CodeGenerator
jdk.nashorn.internal.codegen.CompilationPhase$8
jdk.nashorn.internal.codegen.types.Type
jdk.nashorn.internal.ir.debug.ObjectSizeCalculator
jdk.nashorn.internal.objects.Global
jdk.nashorn.internal.objects.NativeArray
jdk.nashorn.internal.objects.NativeArray$6
jdk.nashorn.internal.objects.NativeDate
jdk.nashorn.internal.objects.NativeObject
jdk.nashorn.internal.runtime.AccessorProperty
jdk.nashorn.internal.runtime.arrays.ArrayData
jdk.nashorn.internal.runtime.arrays.IteratorAction
jdk.nashorn.internal.runtime.CodeStore$2
jdk.nashorn.internal.runtime.CodeStore$3
jdk.nashorn.internal.runtime.Context
jdk.nashorn.internal.runtime.DebuggerSupport
jdk.nashorn.internal.runtime.JSType
jdk.nashorn.internal.runtime.linker.JavaAdapterFactory$AdapterInfo
jdk.nashorn.internal.runtime.linker.LinkerCallSite
jdk.nashorn.internal.runtime.linker.LinkerCallSite$1
jdk.nashorn.internal.runtime.linker.NashornCallSiteDescriptor$1
jdk.nashorn.internal.runtime.ListAdapter
jdk.nashorn.internal.runtime.PropertyListeners
jdk.nashorn.internal.runtime.PropertyMap
jdk.nashorn.internal.runtime.RecompilableScriptFunctionData
jdk.nashorn.internal.runtime.regex.joni.Regex
jdk.nashorn.internal.runtime.ScriptFunctionData
jdk.nashorn.internal.runtime.ScriptingFunctions
jdk.nashorn.internal.runtime.ScriptingFunctions$1
jdk.nashorn.internal.runtime.ScriptingFunctions$2
jdk.nashorn.internal.runtime.ScriptLoader
jdk.nashorn.internal.runtime.ScriptObject
jdk.nashorn.internal.runtime.ScriptRuntime
jdk.nashorn.internal.runtime.Source
jdk.nashorn.internal.runtime.Source$URLData
jdk.nashorn.internal.runtime.UserAccessorProperty
sun.jvm.hotspot.CommandProcessor$17
sun.jvm.hotspot.CommandProcessor$17$1
sun.management.DiagnosticCommandImpl
sun.management.OperatingSystemImpl
sun.misc.GThreadHelper
sun.net.ExtendedOptionsImpl
sun.net.PortConfig
sun.nio.ch.sctp.SctpChannelImpl
sun.nio.ch.sctp.SctpMultiChannelImpl
sun.nio.ch.sctp.SctpNet
sun.nio.ch.sctp.SctpServerChannelImpl
sun.nio.cs.ext.IBM300
sun.nio.cs.ext.JIS_X_0208
sun.nio.cs.ext.JIS_X_0208_MS5022X
sun.nio.cs.ext.JIS_X_0208_MS932
sun.nio.cs.ext.JIS_X_0208_Solaris

```

```

sun.nio.cs.ext.JIS_X_0212
sun.nio.cs.ext.JIS_X_0212_MS5022X
sun.nio.cs.ext.JIS_X_0212_Solaris
sun.nio.cs.ext.PCK
sun.nio.cs.ext.SJIS
sun.util.calendar.CalendarSystem$1
sun.util.calendar.ZoneInfoFile$1
sun.util.locale.provider.AuxLocaleProviderAdapter
sun.util.locale.provider.HostLocaleProviderAdapterImpl$2
sun.util.locale.provider.HostLocaleProviderAdapterImpl$3
sun.util.locale.provider.HostLocaleProviderAdapterImpl$4
sun.util.locale.provider.JRELocaleProviderAdapter
sun.util.locale.provider.LocaleProviderAdapter
sun.util.locale.provider.LocaleResources
sun.util.locale.provider.LocaleServiceProviderPool
sun.util.locale.provider.SPILocaleProviderAdapter$BreakIteratorProviderDeleg
ate
sun.util.locale.provider.SPILocaleProviderAdapter$CalendarDataProviderDelega
te
sun.util.locale.provider.SPILocaleProviderAdapter$CalendarNameProviderDelega
te
sun.util.locale.provider.SPILocaleProviderAdapter$CollatorProviderDelegate
sun.util.locale.provider.SPILocaleProviderAdapter$CurrencyNameProviderDelega
te
sun.util.locale.provider.SPILocaleProviderAdapter$DateFormatProviderDelegate
sun.util.locale.provider.SPILocaleProviderAdapter$DateFormatSymbolsProviderD
elegante
sun.util.locale.provider.SPILocaleProviderAdapter$DecimalFormatSymbolsProvid
erDelegate
sun.util.locale.provider.SPILocaleProviderAdapter$LocaleNameProviderDelegate
sun.util.locale.provider.SPILocaleProviderAdapter$NumberFormatProviderDelega
te
sun.util.locale.provider.SPILocaleProviderAdapter$TimeZoneNameProviderDelega
te
sun.util.locale.provider.TimeZoneNameUtility
sun.util.resources.OpenListResourceBundle
sun.util.resources.ParallelListResourceBundle
sun.util.xml.PlatformXmlPropertiesProvider

#-----
# J2EE
#-----
javax.activation.*
javax.annotation.*
javax.ejb.*
javax.el.*
javax.jms.*
javax.mail.*
javax.management.*
javax.persistence.*
javax.resource.*
javax.servlet.*
javax.transaction.*

```

D. Main Functionality Changes in Each Version

This appendix describes the main updates in the functionality of Application Server versions prior to 11-00. The updates are described here with the respective purpose. For the main updates in the functionality of 11-00, see [1.5 Main updates in the functionality of Application Server 11-00](#).

The description contents are as follows:

- This section gives an overview and describes the main updates in the functionality of Application Server. For details on the functionality, check the description in the *Section* column of the *Reference manual* column in the following table. The *Reference manual* column and *Section* column show the main locations for the descriptions of the corresponding functionality in the manuals for 11-00.
- *uCosminexus Application Server* is omitted from the manual names mentioned in the *Reference manual* column.

D.1 Main functional changes in version 09-87

(1) Supporting standard and existing functionality

The following table lists the items that are changed to support standard and existing functionality:

Table D–1: Changes made for supporting standard and existing functionality

Item	Overview of changes	Reference manual	Reference location
Java SE 11 support	The Java SE 11 functionality can now be used.	<i>Maintenance and Migration Guide</i>	<i>Chapter 9</i>

D.2 Main functional changes in version 09-80

(1) Supporting standard and existing functionality

The following table lists the items that are changed to support standard and existing functionality:

Table D–2: Changes made for supporting standard and existing functionality

Item	Overview of changes	Reference manual	Reference location
Lambda expression support in the JAX-RS function	Lambda expressions can now be used with the package specified for the servlet initialization parameter in the <code>web.xml</code> file and with the classes included in the subpackages of that package.	<i>Web Service Development Guide</i>	<i>11.2</i>
Java SE 9 support	The Java SE 9 functionality can now be used.	<i>Maintenance and Migration Guide</i>	<i>Chapter 9</i>

(2) Maintaining and improving reliability

The following table lists the items that are changed for maintaining and improving reliability:

Table D–3: Changes made for maintaining and improving the reliability

Item	Overview of changes	Reference manual	Reference location
Apache 2.4 support for the web server	Apache 2.4 is supported as the base component of the web server.	<i>HTTP Server User Guide</i>	<i>Chapter 6, Appendix G</i>
Elliptic curve cryptography support in SSL communication	Elliptic curve cryptography can now be used in SSL communication.	<i>HTTP Server User Guide</i>	<i>Chapter 5, Appendix G</i>
Change of the SSL library	The SSL library that provides SSL functions was changed to OpenSSL.	<i>HTTP Server User Guide</i>	<i>Chapter 5, Appendix G</i>

D.3 Main functional changes in version 09-70

(1) Supporting standard and existing functionality

The following table lists the items that are changed to support standard and existing functionality:

Table D–4: Changes made for supporting standard and existing functionality

Item	Overview of changes	Reference manual	Reference location
Addition of JSP compilation method versions in the management portal	The compilation methods compliant with JDK 1.7 specifications and with JDK 7 specifications were additionally supported for servlets generated from JSP files on the J2EE server.	<i>Management Portal User Guide</i>	<i>10.8.4</i>
		<i>Definition Reference Guide</i>	<i>4.11.2</i>
Metaspace support in JDK 8	The options for the Permanent area used for Java VM startup were changed to the options for the Metaspace area.	<i>System Setup and Operation Guide</i>	<i>Appendix A.2</i>
		<i>Management Portal User Guide</i>	<i>10.8.7</i>
		<i>Definition Reference Guide</i>	<i>5.2.1, 5.2.2, 8.2.3</i>
SHA-2 support for user authentication in the integrated user management	The following hash algorithms for user authentication in the integrated user management were additionally supported: SHA-224, SHA-256, SHA-384, SHA-512	<i>Security Management Guide</i>	<i>5.3.1, 5.3.9, 5.10.7, 11.4.3, 12.4.3, 12.5.3, 13.2, 14.2.2</i>
Addition of automatic startup, automatic restart, and automatic termination to Red Hat Enterprise Linux Server 7	Automatic start, restart, and stop were supported by Management Server and Administration Agent that operate on Red Hat Enterprise Linux Server 7.	This manual	<i>2.6.3, 2.6.4, 2.6.5</i>
		<i>Command Reference Guide</i>	<i>7.2</i>

(2) Maintaining and improving the operational performance

The following table lists the items that are changed for maintaining and improving operational performance:

Table D–5: Changes made for maintaining and improving operational performance

Item	Overview of changes	Reference manual	Reference location
Upgrade to version 9.7	A procedure for changing the options for the Permanent area used for Java VM startup to the options for the Metaspace area was added.	<i>Maintenance and Migration Guide</i>	10.3.1, 10.3.2, 10.3.4

(3) Other purpose

The following table lists the items that are changed for other purpose:

Table D–6: Changes made for other purposes

Item	Overview of changes	Reference manual	Reference location
Collection-target data for the snapshot log	Java VM event log data and Management Server thread dumps were added to the collection targets for the snapshot log.	<i>Maintenance and Migration Guide</i>	<i>Appendix A.2</i>

D.4 Main functional changes in version 09-60

(1) Supporting standard and existing functionality

The following table lists the items that are changed to support standard and existing functionality:

Table D–7: Changes made for supporting standard and existing functionality

Item	Overview of changes	Reference manual	Reference location
G1 GC support	G1 GC can now be selected.	<i>System Design Guide</i>	7.15
		<i>Definition Reference Guide</i>	14.5
Support for the object-pointer compression function	The object-pointer compression function can now be used.	<i>Maintenance and Migration Guide</i>	9.16

(2) Maintaining and improving reliability

The following table lists the items that are changed for maintaining and improving reliability:

Table D–8: Changes made for maintaining and improving the reliability

Item	Overview of changes	Reference manual	Reference location
Addition of the finalize-retention resolution function	A function that resolves a problem that delays execution of many finalize processes was added to prevent occurrence of a delay in releasing OS resources.	<i>Maintenance and Migration Guide</i>	9.17

(3) Other purpose

The following table lists the items that are changed for other purpose:

Table D–9: Changes made for other purposes

Item	Overview of changes	Reference manual	Reference location
Addition of the log file asynchronous output functionality	Log files can now be output asynchronously.	<i>Definition Reference Guide</i>	14.2

D.5 Main functional changes in version 09-50

(1) Improving development productivity

The following table lists the items for which changes were made to improve development productivity.

Table D–10: Changes made for improving development productivity

Item	Overview of changes	Reference manual	Reference location
Simplification of Eclipse setup	An Eclipse environment can now be set up from the GUI.	<i>Application Development Guide</i>	1.1.5, 2.4
Aid for debugging with user-extended performance analysis traces	A user-extended performance analysis trace configuration file can now be created in the development environment.	<i>Application Development Guide</i>	1.1.3, 6.4

(2) Simplifying implementation and setup

The following table lists the items that are changed to simplify implementation and setup:

Table D–11: Changes made for simplifying implementation and setup

Item	Overview of changes	Reference manual	Reference location
Addition of system configurations in a virtualization environment	Additional tier types (<code>http-tier</code> , <code>j2ee-tier</code> , and <code>ctm-tier</code>) can now be used in a virtualization environment. As a result, systems can now be built in the following configurations: <ul style="list-style-type: none"> Configuration in which a web server and J2EE server are deployed on different hosts Configuration in which the front-end (servlet and JSP) and back-end (EJB) are separately deployed Configuration that uses CTM 	<i>Virtual System Setup and Operation Guide</i>	1.1.2

(3) Supporting standard and existing functionality

The following table lists the items that are changed to support standard and existing functionality:

Table D–12: Changes made for supporting standard and existing functionality

Item	Overview of changes	Reference manual	Reference location
Support for the JDBC 4.0 specifications	DB Connector now supports HiRDB Type4 JDBC Driver compliant with JDBC 4.0 and the JDBC driver for SQL Server.	<i>Common Container Functionality Guide</i>	3.6.3

Item	Overview of changes	Reference manual	Reference location
Relaxation of the naming rules for Portable Global JNDI names	Characters that can be used in Portable Global JNDI names were added.	<i>Common Container Functionality Guide</i>	2.4.3
Support for the Servlet 3.0 specifications	HTTP Cookie names and URL path parameter names can now be changed in Servlet 2.5 or earlier as they can be changed in Servlet 3.0.	<i>Web Container Functionality Guide</i>	2.7
Expansion of applications that can link with Bean Validation	Bean Validation can now be used with the CDI or a user application to perform validation.	<i>Common Container Functionality Guide</i>	Chapter 9
JavaMail support	An email sending and receiving function that uses an API compliant with JavaMail 1.4 can now be used.	<i>Common Container Functionality Guide</i>	Chapter 7
OSs supported by the <code>javacore</code> command	The <code>javacore</code> command can now be used to obtain thread dumps in Windows.	<i>Command Reference Guide</i>	<i>javacore (Acquiring the thread dump/in Windows)</i>

(4) Maintaining and improving reliability

The following table lists the items that are changed for maintaining and improving reliability:

Table D–13: Changes made for maintaining and improving the reliability

Item	Overview of changes	Reference manual	Reference location
Prevention of depletion of the code cache area	Depletion of the code cache area can now be prevented by checking the size of the area used by the system and changing the threshold value before depletion occurs.	<i>Maintenance and Migration Guide</i>	5.7.2, 5.7.3
		<i>Definition Reference Guide</i>	14.1, 14.2, 14.4
Efficient use of the Explicit Memory Management functionality	The following functions were added that can control the objects that move to the Explicit heap so that the time required for automatic release is reduced to efficiently apply the Explicit Memory Management functionality. <ul style="list-style-type: none"> Function that controls object movement to Explicit memory blocks Function that specifies classes to which the Explicit Memory Management functionality is not applied 	<i>Expansion Guide</i>	7.2.2, 7.6.5, 7.10, 7.13.1, 7.13.3
		<i>Maintenance and Migration Guide</i>	5.5
Expansion of the output range of class-wise statistical information	An extended thread dump containing class-wise statistical information can now include reference relationships based on static fields.	<i>Maintenance and Migration Guide</i>	9.6

(5) Maintaining and improving the operational performance

The following table lists the items that are changed for maintaining and improving operational performance:

Table D–14: Changes made for maintaining and improving operational performance

Item	Overview of changes	Reference manual	Reference location
Support for the EADs session failover functionality	The EADs session failover functionality, which enables session failover in collaboration with EADs, is supported.	<i>Expansion Guide</i>	Chapter 5

Item	Overview of changes	Reference manual	Reference location
Operations using WAR files	A WAR application that consists of only WAR files can now be deployed on the J2EE server.	<i>Web Container Functionality Guide</i>	2.2.1
		<i>Common Container Functionality Guide</i>	15.9
		<i>Command Reference Guide</i>	<i>cjimportwar</i> (Import a WAR application)
Synchronous start and stop of the management functionality	An option that synchronously starts and stops the management functionality (Management Server and Administration Agent) was added.	This manual	2.6.1, 2.6.2, 2.6.3, 2.6.4
		<i>Command Reference Guide</i>	<i>adminagentctl</i> (start or stop Administration Agent), <i>mngautorun</i> (Set up/canceling the set up of autostart and autorestart), and <i>mngsvrctl</i> (start, stop, or setup Management Server)
Forced release of Explicit memory blocks by the Explicit Memory Management functionality	The <code>javagc</code> command can now be used to release Explicit memory blocks at any time.	<i>Expansion Guide</i>	7.6.1, 7.9
		<i>Command Reference Guide</i>	<i>javagc</i> (forcibly perform GC)

(6) Other purpose

The following table lists the items that are changed for other purpose:

Table D–15: Changes made for other purposes

Item	Overview of changes	Reference manual	Reference location
Acquisition of definition information	The <code>snapshotlog</code> command (collect snapshot logs) can now be used to collect definition files only.	<i>Maintenance and Migration Guide</i>	2.3
		<i>Command Reference Guide</i>	<i>snapshotlog</i> (collect snapshot logs)
Output of log data for the <code>cjenvsetup</code> command	The information about setup operations (using the <code>cjenvsetup</code> command) performed by the Component Container administrator is now output to the message log.	<i>System Setup and Operation Guide</i>	4.1.4
		<i>Maintenance and Migration Guide</i>	4.20
		<i>Command Reference Guide</i>	<i>cjenvsetup</i> (set up Component Container Administrator)
BIG-IP v11 support	BIG-IP v11 was added as a type of load balancer that can be used.	<i>System Setup and Operation Guide</i>	4.7.2

Item	Overview of changes	Reference manual	Reference location
		<i>Virtual System Setup and Operation Guide</i>	2.1
Output of the CPU time to the event log for the Explicit Memory Management functionality	The CPU time required to release Explicit memory blocks is now output to the event log for the Explicit Memory Management functionality.	<i>Maintenance and Migration Guide</i>	5.11.3
Extension of the user-extended performance analysis trace functionality	<p>The following changes were made to the user-extended performance analysis trace functionality:</p> <ul style="list-style-type: none"> Although the trace target could be specified by only method, it can now also be specified by package or class. The range of applicable event IDs was expanded. The maximum number of lines that can be contained in the user-extended performance analysis trace configuration file was increased. The user-extended performance analysis trace configuration file can now be used to specify the trace collection level. 	<i>Maintenance and Migration Guide</i>	7.5.2, 7.5.3, 8.23.1
Improvement of information analysis in cases where asynchronous Session Bean invocations are used	The requests of invocation source and destination can now be matched by using the root application information of the PRF trace.	<i>EJB Container Functionality Guide</i>	2.17.3

D.6 Main updates in the functionality of 09-00

(1) Simplifying implementation and setup

The following table lists the items that are changed for simplifying the implementation and setup.

Table D–16: Changes for simplifying implementation and setup

Item	Overview of changes	Reference manual	Reference location
Changing the units to be set up and operated in the virtual environment	The units to be operated when you set up and operate the virtual environment were changed from the virtual server to the virtual server group. You can now use the file in which the virtual server group is defined, and register multiple virtual servers to a management unit in a batch.	<i>Virtual System Setup and Operation Guide</i>	1.1.2
Canceling the restrictions on the environment setup by using Setup Wizard	The restrictions on the environments that can be set up with Setup Wizard were removed. Even if an environment has been set up with another functionality, you can now unset up the environment and use Setup Wizard for setup.	<i>System Setup and Operation Guide</i>	2.2.7
Simplifying the procedure for deleting the setup environment	The deletion procedure has now been simplified by adding the functionality to delete a system environment set up by using Management Server (mngunsetup command).	<i>System Setup and Operation Guide</i>	4.1.37
		<i>Management Portal User Guide</i>	3.6, 5.4
		<i>Command Reference Guide</i>	mngunsetup (Deleting the setup environment of

Item	Overview of changes	Reference manual	Reference location
			<i>Management Server)</i>

(2) Supporting standard and existing functionality

The following table lists the items that are changed to support the standard and existing functionality.

Table D–17: Changes for supporting the standard and the existing functionality

Item	Overview of changes	Reference manual	Reference location
Supporting Servlet 3.0	Servlet 3.0 is now supported.	<i>Web Container Functionality Guide</i>	<i>Chapter 7</i>
Supporting EJB 3.1	EJB 3.1 is now supported.	<i>EJB Container Functionality Guide</i>	<i>Chapter 2</i>
Supporting JSF 2.1	JSF 2.1 is now supported.	<i>Web Container Functionality Guide</i>	<i>Chapter 3</i>
Supporting JSTL 1.2	JSTL 1.2 is now supported.	<i>Web Container Functionality Guide</i>	<i>Chapter 3</i>
Supporting CDI 1.0	CDI 1.0 is now supported.	<i>Common Container Functionality Guide</i>	<i>Chapter 8</i>
Using the Portable Global JNDI name	You can now look up objects for which Portable Global JNDI names are used.	<i>Common Container Functionality Guide</i>	<i>2.4</i>
Supporting JAX-WS 2.2	JAX-WS 2.2 is now supported.	<i>Web Service Development Guide</i>	<i>1.1, 16.1.5, 16.1.7, 16.2.1, 16.2.6, 16.2.10, 16.2.12, 16.2.13, 16.2.14, 16.2.16, 16.2.17, 16.2.18, 16.2.20, 16.2.22, 19.1, 19.2.3, 37.2, 37.6.1, 37.6.2, 37.6.3</i>
Supporting JAX-RS 1.1	JAX-RS 1.1 is now supported.	<i>Web Service Development Guide</i>	<i>1.1, 1.2.2, 1.3.2, 1.4.2, 1.5.1, 1.6, 2.3, Chapter 11, Chapter 12, Chapter 13, Chapter 17, Chapter 24, Chapter 39</i>

(3) Maintaining and improving reliability

The following table lists the items that are changed for maintaining and improving reliability.

Table D–18: Changes for maintaining and improving the reliability

Item	Overview of changes	Reference manual	Reference location
Using TLSv1.2 in the SSL/TLS communication	You can now use RSA BSAFE SSL-J to execute the SSL/TLS communication with a security protocol containing TLSv 1.2.	--	--

Legend:

--: Functionality deleted in 09-70

(4) Maintaining and improving operational performance

The following table lists the items that are changed for maintaining and improving operational performance.

Table D–19: Changes for maintaining and improving operational performance

Item	Overview of changes	Reference manual	Reference location
Monitoring the total pending queues of the entire Web container	You can now output the total pending queues of the entire Web container in the operation information and monitor the number of queues.	This manual	Chapter 3
Output of performance analysis trace for applications (user-extended trace)	The performance analysis trace used for analyzing the processing performance of user-developed applications can now be output without changing the applications.	<i>Maintenance and Migration Guide</i>	Chapter 7
Operations performed by using the user script in a virtual environment	The user-created script (user script) can now be executed on a virtual server at any timing	<i>Virtual System Setup and Operation Guide</i>	7.8
Improving the management portal	Changes were made so that the messages describing the procedure are displayed on the following management portal windows: <ul style="list-style-type: none"> Distribution of setup information window Start windows for the Web server, J2EE server, and SFO server Batch start, batch restart, and start windows for Web server cluster and J2EE server cluster 	<i>Management Portal User Guide</i>	10.10.1, 11.9.2, 11.10.2, 11.10.4, 11.10.6, 11.11.2, 11.12.2, 11.12.4, 11.12.6
Adding the restart functionality of the management functionality	You can now specify the settings for automatic restart by the management functionality (Management Server and Administration Agent), and continue operations even when an error occurs in the management functionality. The procedure for automatic start settings was also changed.	This manual	2.4.1 , 2.4.2 , 2.6.3 , 2.6.4
		<i>Command Reference Guide</i>	<i>mngautorun (Set up and unset up of automatic start and automatic restart)</i>

(5) Other purposes

The following table lists the items that are changed for other purposes.

Table D–20: Changes for other purposes

Item	Overview of changes	Reference manual	Reference location
Changing the file switching units when log is output	The output destination files are now switched by the date when the log is output.	<i>Maintenance and Migration Guide</i>	3.2.1

Item	Overview of changes	Reference manual	Reference location
Changing the Web server name	The name of the Web server included in Application Server is changed to HTTP Server.	<i>HTTP Server User Guide</i>	--
Supporting the direct connection that uses BIG-IP APIs (SOAP architecture)	The direct connection that uses APIs (SOAP architecture) in BIG-IP (load balancer) is now supported. Also, the procedure for setting up the connection environment of the load balancer was changed for using the direct connection that uses APIs.	<i>System Setup and Operation Guide</i>	4.7.3, Appendix J
		<i>Virtual System Setup and Operation Guide</i>	2.1, Appendix C
		<i>Security Management Guide</i>	8.2, 8.4, 8.5, 8.6, 18.2.1, 18.2.2, 18.2.3

Legend:

--: Reference the entire manual

D.7 Main updates in the functionality of 08-70

(1) Simplifying implementation and setup

The following table lists the items that are changed for simplifying the implementation and setup.

Table D–21: Changes for simplifying implementation and setup

Item	Overview of changes	Reference manual	Section
Improving the management portal	Changes have been made to enable the user to set up the property (settings of the Connector property file) for defining resource adapter attributes and performing the connection test, using the management portal window. Also, J2EE applications (ear file and zip file) can now be uploaded on Management Server using the Management portal window.	<i>First Step Guide</i>	3.5
		<i>Management Portal User Guide</i>	--
Adding the <code>import</code> attribute implicit import functionality of the <code>page/tag</code> directive	The <code>import</code> attribute implicit import functionality of the <code>page/tag</code> directive can now be used.	<i>Web Container Functionality Guide</i>	2.3.7
Supporting the automatic environment settings for the JP1 products in a virtual environment	Changes have been made so that when Application Server is set up on a virtual server, the environment settings of the JP1 products can be automatically set up for the virtual server by using the hook script.	<i>Virtual System Setup and Operation Guide</i>	7.7.2
Improving the Integrated user management functionality	When using a database in a user information repository, the database can now be connected by using the JDBC driver of the database products. The database connection through the JDBC driver of Cosminexus DABroker Library is not supported anymore. You can now set up the integrated user management functionality using the Easy Setup definition file and the management portal window. Active Directory now supports double byte characters such as Japanese language in DN.	<i>Security Management Guide</i>	Chapter 5, 14.2.2
		<i>Management Portal User Guide</i>	3.5, 10.8.1
Enhancing HTTP Server settings	You can now directly set up the directives (settings of <code>httpsd.conf</code>) that define the operation environment of HTTP Server using the Easy Setup definition file and the management portal window.	<i>System Setup and Operation Guide</i>	4.1.21
		<i>Management Portal User Guide</i>	10.9.1

Item	Overview of changes	Reference manual	Section
		<i>Definition Reference Guide</i>	<i>4.10</i>

Legend:

--: Reference the entire manual.

(2) Supporting standard and existing functionality

The following table lists the items that are changed to support the standard and existing functionality.

Table D–22: Changes for supporting the standard and the existing functionality

Item	Overview of changes	Reference manual	Section
Adding items to be specified in <code>ejb-jar.xml</code>	The class level and method level interceptors can now be specified in <code>ejb-jar.xml</code> .	<i>EJB Container Functionality Guide</i>	<i>2.15</i>
Supporting the parallel copy garbage collection	The parallel copy garbage collection can now be selected.	<i>Definition Reference Guide</i>	<i>14.5</i>
Supporting the global transaction of the Inbound resource adapters conforming to the Connector 1.5 specifications	<code>Transacted Delivery</code> can now be used with the resource adapters conforming to the Connector 1.5 specifications. This enables participation of EIS invoking Message-driven Beans in the global transaction.	<i>Common Container Functionality Guide</i>	<i>3.16.3</i>
Supporting MHP of TP1 inbound adapters	MHP can now be used as an <code>OpenTP1client</code> that invokes Application Server by using TP1 inbound adapters.	<i>Common Container Functionality Guide</i>	<i>Chapter 4</i>
Supporting the FTP inbound adapter of the <code>cjrarupdate</code> command	An FTP inbound adapter has been added to the resource adapters that can be upgraded by using the <code>cjrarupdate</code> command.	<i>Command Reference Guide</i>	<i>2.2</i>

(3) Maintaining and improving reliability

The following table lists the items that are changed for maintaining and improving reliability.

Table D–23: Changes for maintaining and improving the reliability

Item	Overview of changes	Reference manual	Section
Improving the database session failover functionality	The user can now select a mode that does not acquire the lock of the database in which the global session information is stored in a performance-centric system. Also, exclusive requests for references can now be defined without updating the database.	<i>Expansion Guide</i>	<i>Chapter 6</i>
Expanding a process for the <code>OutOfMemory</code> handling functionality	A process for the <code>OutOfMemory</code> handling functionality has been added.	<i>Maintenance and Migration Guide</i>	<i>2.5.4</i>
		<i>Definition Reference Guide</i>	<i>14.2</i>
Adding the memory saving functionality for the Explicit heap used in an HTTP session	The functionality to minimize the amount of the Explicit heap memory used in an HTTP session has been added.	<i>Expansion Guide</i>	<i>7.11</i>

(4) Maintaining and improving the operational performance

The following table lists the items that are changed for maintaining and improving the operational performance.

Table D–24: Changes for maintaining and improving the operational performance

Item	Overview of changes	Reference manual	Section
Supporting user authentication using JP1 products in a virtual environment (handling cloud operations)	The administration and the authentication of the users using a virtual server manager can now be performed by using the authentication server of the JP1 products when integrating JP1.	<i>Virtual System Setup and Operation Guide</i>	1.2.2, Chapter 3, Chapter 4, Chapter 5, Chapter 6, 7.9

(5) Other purposes

The following table lists the items that are changed for other purposes.

Table D–25: Changes for other purposes

Item	Overview of changes	Reference manual	Section
Supporting the direct connection using APIs (REST architecture) to the load balancing functionality	The direct connection using APIs (REST architecture) is now supported as a method to connect to the load balancing functionality. ACOS (AX2500) is added to the types of the available load balancing functions.	<i>System Setup and Operation Guide</i>	4.7.2, 4.7.3
		<i>Virtual System Setup and Operation Guide</i>	2.1
		<i>Definition Reference Guide</i>	4.2.4
Improving the response timeout when collecting the snapshot logs and the collection targets	You can now stop the snapshot log collection (timeout) at a specified time. The contents collected as the primary delivery data have been changed.	<i>Maintenance and Migration Guide</i>	Appendix A

D.8 Main updates in the functionality of 08-53

(1) Simplifying implementation and setup

The following table lists the items that are changed to simplify implementation and setup:

Table D–26: Changes made for simplifying implementation and setup

Item	Overview of changes	Reference manual	Section
Setting up a virtual environment corresponding to various hypervisors	Application Server can now be set up on a virtual server implemented by using various hypervisors. Also, an environment with multiple co-existing hypervisors is also supported now.	<i>Virtual System Setup and Operation Guide</i>	Chapter 2, Chapter 3, Chapter 5

(2) Supporting standard and existing functionality

The following table lists the items that are changed to support standard and existing functionality:

Table D–27: Changes made for supporting standard and existing functionality

Item	Overview of changes	Reference manual	Section
Invoking from OpenTP1 that supports transaction integration	Transactions can now be integrated when a Message-driven Bean operating on the Application Server is invoked from OpenTP1.	<i>Common Container Functionality Guide</i>	Chapter 4

Item	Overview of changes	Reference manual	Section
JavaMail	A mail receiving function using APIs compliant with JavaMail 1.3 is now available by integrating with a mail server compliant with POP3.	<i>Common Container Functionality Guide</i>	<i>Chapter 7</i>

(3) Maintaining and improving reliability

The following table lists the items that are changed for maintaining and improving reliability:

Table D–28: Changes made for maintaining and improving the reliability

Items	Overview of changes	Reference manual	Section
Improving the JavaVM troubleshooting functionality	<p>The following functionality is now available as the JavaVM troubleshooting functionality:</p> <ul style="list-style-type: none"> The operations when <code>OutOfMemoryError</code> occurs can now be changed. The maximum value for C heap to be allocated during JIT compilation can now be set up. The maximum number of threads can now be set up. The items output to the extension verbosegc information have been extended. 	<i>Maintenance and Migration Guide</i>	<i>Chapter 4, Chapter 5, Chapter 9</i>

(4) Maintaining and improving the operational performance

The following table lists the items that are changed for maintaining and improving operational performance:

Table D–29: Changes made for maintaining and improving operational performance

Item	Overview of changes	Reference manual	Section
Supporting JP1/ITRM	JP1/ITRM, a product that uniformly manages IT resources, is now supported.	<i>Virtual System Setup and Operation Guide</i>	<i>1.3, 2.1</i>

(5) Other purpose

The following table lists the items that are changed for other purpose:

Table D–30: Changes made for other purposes

Item	Overview of changes	Reference manual	Section
Supporting Microsoft IIS 7.0 and Microsoft IIS 7.5	Microsoft IIS 7.0 and Microsoft IIS 7.5 are now supported as Web servers.	--	--
Supporting HiRDB Version 9 and SQL Server 2008	<p>The following products are now supported as databases:</p> <ul style="list-style-type: none"> HiRDB Server Version 9 HiRDB/Developer's Kit Version 9 HiRDB/Run Time Version 9 SQL Server 2008 <p>Also, SQL Server JDBC Driver is supported as the JDBC driver for SQL Server 2008.</p>	<i>Common Container Functionality Guide</i>	<i>Chapter 3</i>

Legend:

--: Not applicable.

D.9 Main updates in the functionality of 08-50

(1) Simplifying implementation and setup

The following table lists the items that are changed to simplify implementation and setup:

Table D–31: Changes made for simplifying implementation and setup

Item	Overview of changes	Reference manual	Section
Changing the specification of required tags of <code>web.xml</code> at the Web service provider machine	Specifications of the <code>listener</code> , <code>servlet</code> , and <code>servlet-mapping</code> tags have been changed from 'Required' to 'Optional' in the <code>web.xml</code> on the Web service provider machine.	<i>Definition Reference Guide</i>	2.2.3
Using network resources of logical servers	Functionality has been added for accessing network resources or a network drive on another host from a J2EE application.	This manual	1.2.3, 5.2, 5.7
Simplifying the execution procedure of sample programs	Sample programs have been provided in the EAR format to simplify the execution procedure.	<i>First Step Guide</i>	3.5
		<i>System Setup and Operation Guide</i>	Appendix L
Improving the operations of the Management Portal window	The default window update interval has been changed from 'Do not update' to '3 seconds'.	<i>Management Portal User Guide</i>	7.4.1
Improving the completion window of the Setup Wizard	Changes have been made to enable the display of the Easy Setup definition file and the HITACHI Connector Property file, used during the setup, on the screen that is displayed when the Setup Wizard completes.	<i>System Setup and Operation Guide</i>	2.2.6
Setting up the virtual environment	A procedure has been added for setting up Application Server on the virtual server, implemented by using a hypervisor. #	<i>Virtual System Setup and Operation Guide</i>	Chapter 3, Chapter 5

#

To set up a virtual environment in the 08-50 mode, see *Appendix D. Settings for Using Virtual Server Manager in the 08-50 Mode* in the *uCosminexus Application Server Virtual System Setup and Operation Guide*.

(2) Supporting standard and existing functionality

The following table lists the items that are changed to support standard and existing functionality:

Table D–32: Changes made for supporting standard and existing functionality

Item	Overview of changes	Reference manual	Section
Supporting invocation from OpenTP1	Changes have been made to enable the invocation of Message-driven Beans operating on Application Server from OpenTP1.	<i>Common Container Functionality Guide</i>	Chapter 4
Supporting JMS	Changes have been made to enable the use of the Cosminexus JMS provider functionality corresponding to the JMS1.1 specifications.	<i>Common Container Functionality Guide</i>	Chapter 6
Supporting Java SE 6	Changes have been made to enable the use of the functionality provided with Java SE 6.	<i>Maintenance and Migration Guide</i>	5.5, 5.8.1
Supporting the use of Generics	Changes have been made to enable the usage of Generics in EJBs.	<i>EJB Container Functionality Guide</i>	4.2.18

(3) Maintaining and improving reliability

The following table lists the items that are changed for maintaining and improving reliability:

Table D–33: Changes made for maintaining and improving the reliability

Item	Overview of changes	Reference manual	Section
Improving usability of the Explicit Memory Management functionality	Changes have been made to enable easy usage of the Explicit Memory Management functionality with the automatic arrangement setup file.	<i>System Design Guide</i>	7.2, 7.7.3, 7.11.4, 7.12.1
		<i>Expansion Guide</i>	Chapter 7
Controlling URI units of the database session failover functionality	Changes have been made to enable the specification of requests that are outside the scope of the functionality in URI units, when using the database session failover functionality.	<i>Expansion Guide</i>	5.6.1
Monitoring errors in a virtual environment	A virtual server in a virtual system can now be monitored and the occurrence of errors can be detected.	<i>Virtual System Setup and Operation Guide</i>	Appendix D

(4) Maintaining and improving the operational performance

The following table lists the items that are changed for maintaining and improving operational performance:

Table D–34: Changes made for maintaining and improving operational performance

Item	Overview of changes	Reference manual	Section
Omitting the management user account	Changes have been made to enable the omission of entering user login IDs and passwords in commands of Management Server and commands of the Smart Composer functionality.	<i>System Setup and Operation Guide</i>	4.1.15
		<i>Management Portal User Guide</i>	2.2, 7.1.1, 7.1.2, 7.1.3, 8.1, 8.2.1, Appendix F.2
		<i>Command Reference Guide</i>	1.4, mngsvrctl (Starting/Stopping/Setting up Management Server), mngsvrutil (Management Server management command), 8.3, cmx_admin_passwd (Setting up the management user account for Management Server)
Virtual environment operations	A procedure has been added for executing batch start or batch stop and scalein or scaleout for multiple virtual servers in a virtual system. [#]	<i>Virtual System Setup and Operation Guide</i>	Chapter 4, Chapter 6

#

To operate the virtual environment in the 08-50 mode, see *Appendix D. Settings for Using Virtual Server Manager in the 08-50 Mode* in the *uCosminexus Application Server Virtual System Setup and Operation Guide*.

(5) Other purpose

The following table lists the items that are changed for other purpose:

Table D–35: Changes made for other purposes

Item	Overview of changes	Reference manual	Section
Unused objects statistical functionality in the Tenured area	Changes have been made to enable identification of only the required objects in the Tenured area	<i>Maintenance and Migration Guide</i>	9.8

Item	Overview of changes	Reference manual	Section
Base object list output functionality for Tenured augmentation factors	Changes have been made to enable the information output about objects acting as the base of the unused objects that are identified using the unused object statistics functionality in the Tenured area.		9.9
Class-wise statistics analysis functionality	Changes have been made to enable the output of class-wise statistics in the CSV format.		9.10
Cluster node switching based on detection of the excessive automatic restart count of logical servers	Changes have been made to enable node switching when a logical server stops abnormally (when the automatic restart count exceeds or when a failure is detected while the automatic restart is set to 0) for the cluster configuration in which node switching is monitored using Management Server.	This manual	16.2.2, 16.3.3, 16.3.4, 18.4.3, 18.5.3
Node switching system for the host management model	Changes have been made to enable node switching for host unit management models while operating the systems integrated with cluster software.		Chapter 16
Supporting ACOS (AX2000, BS320)	ACOS (AX2000, BS320) was added as a load balancer type that can be used.	System Setup and Operation Guide	4.7.2, 4.7.3, 4.7.5, 4.7.6, Appendix J, Appendix J.2
		Definition Reference Guide	4.2.4, 4.3.2, 4.3.4, 4.7.1
Adding transaction attributes that can be specified in the Stateful Session Bean (SessionSynchronization) when performing transaction management with CMT	Changes have been made to enable the specification of Supports, NotSupported, and Never as transaction attributes in the Stateful Session Bean (SessionSynchronization), when performing transaction management with CMT.	EJB Container Functionality Guide	2.7.3
Forced termination of Administration Agent when OutOfMemoryError occurs	Changes have been made to enable the forced termination of Administration Agent when OutOfMemoryError occurs in Java VM.	Maintenance and Migration Guide	2.5.5
Asynchronous parallel processing of threads	Changes have been made to enable the implementation of asynchronous timer processing and asynchronous thread processing using TimerManager and WorkManager.	Expansion Guide	--

D.10 Main functionality changes in 08-00

(1) Improving the development productivity

The following table lists the items that are changed for improving the development productivity:

Table D–36: Changes made for improving the development productivity

Item	Overview of changes	Reference manual	Section
Easy migration from other Application Server products	The use of the following functionality was enabled for smooth migration from other Application Server products:	Web Container Functionality Guide	2.3, 2.7.5

Item	Overview of changes	Reference manual	Section
	<ul style="list-style-type: none"> The upper limit of the HTTP sessions can now be determined using exceptions. If the JavaBeans ID is repeated and if the upper case or lower case in the custom tag property name and TLD definition are different, the occurrence of translation errors can now be controlled. 		
Provision of <code>cosminexus.xml</code>	By specifying the Cosminexus Application Server independent property in <code>cosminexus.xml</code> , J2EE applications can be started without specifying the property settings after being imported into the J2EE server.	<i>Common Container Functionality Guide</i>	13.3

(2) Supporting standard functionality

The following table lists the items that are changed to support the standard functionality:

Table D–37: Changes made for supporting standard functionality

Item	Overview of changes	Reference manual	Section
Support for Servlet 2.5	Servlet 2.5 is supported.	<i>Web Container Functionality Guide</i>	2.2, 2.5.4, 2.6, Chapter 7
Support for JSP 2.1	JSP 2.1 is supported.	<i>Web Container Functionality Guide</i>	2.3.1, 2.3.3, 2.5, 2.6, Chapter 7
JSP debugging	JSP debugging can now be performed in the development environment using MyEclipse. #	<i>Web Container Functionality Guide</i>	2.4
Storing tag library in the library JAR and TLD mapping	If the tag library is stored in the library JAR, the following processing occurs: When the Web application starts, the TLD file in the library JAR is searched by the Web container and can be automatically mapped.	<i>Web Container Functionality Guide</i>	2.3.4
Omitting <code>application.xml</code>	<code>application.xml</code> can now be omitted in J2EE applications.	<i>Common Container Functionality Guide</i>	13.4
Combined use of annotation and DD	Annotation and DD can now be used together and the contents specified in annotation can be updated in DD.	<i>Common Container Functionality Guide</i>	14.5
Conforming annotation to Java EE 5 standards (default interceptor)	The default interceptor can now be stored in the library JAR. Also, DI can be performed from the default interceptor.	<i>Common Container Functionality Guide</i>	13.4
Resolving the <code>@Resource</code> references	The resource references can now be resolved with <code>@Resource</code> .	<i>Common Container Functionality Guide</i>	14.4
Support for JPA	JPA specifications are supported.	<i>Common Container Functionality Guide</i>	Chapter 5

#

With version 09-00 or later, you can use the JSP debug functionality in the development environment using WTP.

(3) Maintaining and improving reliability

The following table lists the items that are changed for maintaining and improving reliability:

Table D–38: Changes made for maintaining and improving the reliability

Item	Overview of changes	Reference manual	Section
Persistence of session information	The session information of the HTTP session can now be saved and inherited in the database.	<i>Expansion Guide</i>	<i>Chapter 5, Chapter 6</i>
Suppression of Full GC	Occurrence of Full GC can now be suppressed by placing objects that can trigger Full GC outside the Java heap.	<i>Expansion Guide</i>	<i>Chapter 7</i>
Client performance monitor	The time required for client processing can now be checked or analyzed.	--	--

Legend:

--: This functionality has been deleted in version 09-00.

(4) Maintaining and improving the operational performance

The following table lists the items that are changed for maintaining and improving operational performance:

Table D–39: Changes made for maintaining and improving operational performance

Item	Overview of changes	Reference manual	Section
Improving the operational performance of applications on the management portal	The server management commands and management portal can now be interoperated for the application and resource operations.	<i>Management Portal User Guide</i>	<i>1.1.3</i>

(5) Other purpose

The following table lists the items that are changed for other purpose:

Table D–40: Changes made for other purposes

Item	Overview of changes	Reference manual	Section
Deleting invalid HTTP Cookie	Invalid HTTP Cookies can now be deleted.	<i>Web Container Functionality Guide</i>	<i>2.7.4</i>
Detecting Naming Service failure	When a naming service failure occurs, the EJB client can now detect the error faster.	<i>Common Container Functionality Guide</i>	<i>2.9</i>
Timeout in detecting connection failure	A timeout value can now be specified for the connection failure detection timeout.	<i>Common Container Functionality Guide</i>	<i>3.15.1</i>
Support for Oracle11g	Oracle11g can now be used as the database.	<i>Common Container Functionality Guide</i>	<i>Chapter 3</i>
Scheduling of batch processing	The execution of batch applications can now be scheduled using CTM.	<i>Expansion Guide</i>	<i>Chapter 4</i>
Batch processing log	Changes have been made so that the size and number of log files of batch execution commands, and the retry frequency and retry interval when a failure occurs in the log exclusion processing, can be specified.	<i>Definition Reference Guide</i>	<i>3.2.5</i>

Item	Overview of changes	Reference manual	Section
Snapshot log	The collected contents of the snapshot log are changed.	<i>Maintenance and Migration Guide</i>	<i>Appendix A.1, Appendix A.2</i>
Releasing the protected area of method cancellation	The contents of the protected area list that are not the target for method cancellation were released.	This manual	<i>Appendix C</i>
Selecting whether to perform GC before outputting statistics	Whether to perform GC can now be selected before outputting class-wise statistical information.	<i>Maintenance and Migration Guide</i>	9.7
Age distribution information output functionality for survivor area	The age distribution information of the Java objects in the Survivor area can now be output in the JavaVM log file.	<i>Maintenance and Migration Guide</i>	9.11
Finalize retention cancellation functionality	The JavaVM finalize processing status can now be monitored and processing retention can be cancelled.	--	--
Changing the maximum heap size of the server management command	The maximum heap size used by the server management command has changed.	<i>Definition Reference Guide</i>	5.2.1, 5.2.2
Support when non-recommended display name has specified.	When an unrecommended display name has specified for a J2EE application, a message will now be displayed.	<i>Messages</i>	<i>KDJE42 374-W</i>

Legend:

--: Functionality deleted in 09-00.

E. Glossary

Terminology used in this manual

For the terms used in the manual, see the *uCosminexus Application Server and BPM/ESB Platform Terminology Guide*.

Index

Symbols

.mngsvrutilrc 273

Numerics

1-to-1 node switching system 225, 247
1-to-1 node switching system for the Management Server 295
1-to-1 node switching system of application server 290

A

accessing network resources from j2ee applications 112, 170
actcommand 379
active node 225
adminagent.adapter.bind_host 258, 272, 318, 328
adminagent.cluster.localaddress.check 258, 272
adminagent.j2ee.process.console_log.enabled 215
adminagent.process.consolelog.enabled 215
adminagent.process.consolelog.fileenum 215
adminagent.process.consolelog.filesize 215
adminagent.properties 215, 258, 327
adminagent.userserver.process.console_event.enabled 215
adminagent.userserver.process.console_log.enabled 215
adminagentcheck 181
agentaddr 394, 400
alias IP address 253, 255, 310
application of issuing Management events 199
Application Server 15
archive format 161
automatic node switching 250, 312
automatic restart of Administration Agent 36
automatic restart of logical server 36
automatic restart of Management Server 36

B

batch operation 39

C

canceling
 timeout request 111
cjjspc 167
cjlistthread 132

cjstopthread 133
cluster 367
cluster ip address 255
cmx_stop_target command 143
cold standby 229
Component Container administrator 27
Configuration example of 1-to-1 node switching system for the Management Server (In UNIX) 281
Configuration example of 1-to-1 node switching system for the Management Server (in Windows) 262
configuration example of 1-to-1 node switching system of application server (in Unix) 269
configuration example of 1-to-1 node switching system of application server (in Windows) 255
Configuration example of mutual node switching system 309, 340
Configuration example of mutual node switching system (in UNIX) 324
Configuration example of mutual node switching system (in Windows) 315
Configuration example of mutual standby system 337
configuration example of n-to-1 recovery system (in UNIX) 370
configuration example of virtual host in n-to-1 recovery configuration 371
Configuration pattern and locking methods of J2EE applications 148
confirming
 execution status of J2EE application 132
confirming settings of node switching system 305
console log 213
console output information 213
Contents to be set up in the server-compliant environment settings (in 1-to-1 node switching of the Management Server) 287
Contents to be set up in the server-compliant environment settings (in the mutual node switching system) 331
copying the Management Server settings from the active node to the spare node 265, 284

D

Definition for event issuing in Easy Setup definition file 91
Definition of event issuing functionality in cosminexus.xml 90
Definition of the resource depletion monitoring functionality in cosminexus.xml 97

Definition of the resource depletion monitoring functionality in the Easy Setup definition file 98

Definition required for functionality for J2EE application operations 152

Definition required for statistics file collection in the Easy Setup definition file 70

determining timeout and operations after timeout 117

E

ejbserver.distributedtx.ots.status.directory1 318, 327, 380

ejbserver.distributedtx.xatransaction.enabled 362

ejbserver.distributedtx.XATransaction.enabled 318, 327

ejbserver.manager.agent.MEventAgent.enabled 199

environment variables that can be used in command files of management action execution commands 203

Event issuing functionality 86

example of IP address settings when Smart Agent is not set as target for failover 397

example of IP address settings when Smart Agent is set as target for failover 396

Example of setting timeout value and the range of setting values 121

example of system configuration (in n-to-1 recovery system) 359

example of system configuration of n-to-1 recovery system 354

Example of system configuration of the mutual node switching system 309

Example of the IP address settings when Smart Agent is not set as a target for node switching 403

Example of the IP address settings when the Smart Agent is set as a target for node switching 402

example system configuration for a 1-to-1 node switching system 248

example system configuration for operating the Management Server in the 1-to-1 node switching system 248

executing node 225, 386, 388

expanded archive format 161

F

Flow for starting the mutual node switching system 342

Flow for stopping the mutual node switching system 344

Flow of automatic node switching process 251, 312

Flow of automatic node switching process in the 1-to-1 node switching system of the Management Server 251

Flow of automatic node switching process in the mutual node switching system 313

flow of monitoring and canceling execution time of J2EE application 131

Flow of planned node switching process 251, 313

Flow of planned node switching process in the 1-to-1 node switching system of the Management Server 251

Flow of planned node switching process in the mutual node switching system 313

flow of recovery process 355

Flow of the node switching process 312

Flow of the node switching process in the 1-to-1 node switching system 249

Flow of the node switching process in the mutual node switching system 311

forced termination command 116

forced termination of J2EE applications 111, 152

forced termination process 152

front 141

front EJB 143, 152

functionality

- method cancellation 115
- method timeout 115
- supporting daily system operations 22
- supporting system audit 23
- supporting system maintenance 23

Functionality for compatibility with products of older version (compatibility functionality) 19

Functionality for executing the Enterprise Beans (EJB containers) 17

Functionality for executing the Web applications (Web containers) 17

Functionality for migrating from products of older versions (migration functionality) 19

Functionality for monitoring system usage (watch functionality) 18

Functionality for operating the system by linking with other products (linkage functionality) 18

Functionality for troubleshooting (maintenance functionality) 18

Functionality used for daily operations, such as starting and stopping the systems (operation functionality) 18

Functionality used in both Web applications and Enterprise Beans (Container common function) 17

H

HA monitor 224

HA monitor-based 1-to-1 node switching system 297

HA monitor environment settings 285

header file 67

How to issue an event 87
htc.clienthandleraddr 393, 399

I

inheriting information during node switching 253, 356
Issuing definition for management events in the Easy Setup definition file (J2EE server) 199
items that can be displayed using statistics monitoring (in batch servers) 188
items that can be monitored as the statistics 183

J

J2EE application
canceling 114
canceling request for which timeout has occurred 133
forced termination after normal termination by setting any timeout period (server management command) 157
forced termination after normal termination in default timeout period (server management command) 156
forced termination automatically after normal termination by setting any timeout period (server management command) 158
monitoring 114
normal service lock (using load balancer) 143
normal termination in default timeout period (server management command) 154
normal termination setting any timeout period (server management command) 155
operation 111
operation functionality 23
partial service lock (using load balancer) 144
renaming 168
replacing (redeploying) 164
replacing (reloading) 166
replacing and maintaining 163
service lock (using load balancer) 143
service lock (Web front system) 142
structure 145
J2EE application termination
forced termination 146
normal termination 145
J2EE application termination process flow 146
J2EE server
functionality 17
JSP pre-compile 166
replacing J2EE application by reloading 167
starting J2EE application 167

JSP pre-compile functionality 161
JSP pre-compile when replacing a J2EE application by redeploying 167
jstartrecover 379

L

list display command 116
localaddr 258, 273, 393, 399
locking a J2EE application 112
lock process 147
CTM schedule 148
front-end EJB 148
Web application 148
logical server 35
log information that is output when J2EE server starts 134
log information that is output when method cancellation is executed by extension of timeout 135
log information that is output when method cancellation is executed by using command 136
log information that is output while monitoring execution time of J2EE applications 134

M

maction.properties 201
main functional changes in version 09-70 446
main functional changes in version 09-80 445
main functional changes in version 09-87 445
maintaining J2EE application 113
management action 40, 94, 193
controlling 195
management event 40, 94, 193
management functionality based on JP1 integration 23
manager.mevent.message_id.list 199
manager.mevent.retry.interval 199
manager.mevent.retry.limit 199
manager.mevent.send.max 199
manager.mevent.send.timeout 199
manager.mevent.sender.bind.host 199
manage the J2EE application generations 168
mapping
header files with statistics files 67
method cancellation 118
operation 119
processing 118
timing 120
method cancellation command 116

- Method for checking the operations of logical servers 34
- Method of switching a J2EE application 160
- method timeout 116
- Method timeout time for each J2EE application 117
- Method to output server statistics 182
- mngagent.connector.host 317, 327
- mngsvr.myhost.name 264, 284
- monbegin 297, 341, 388
- monend 297, 341, 388
- monitoring
 - J2EE server 183
 - logical server 41
 - status 41
 - status item 42
- monitoring and canceling the execution time of J2EE application 112
- monitoring base 115
- monitoring connection pool 97
- monitoring execution time of J2EE application 115
- monitoring file descriptor 95
- monitoring for each resource type 98
- monitoring HTTP request queue 96
- monitoring memory 95
- monitoring number of sessions 96
- monitoring system operations 39
- monitoring target for which threshold value can be set 86
- monitoring the execution time of J2EE application 153
- monitoring the status of service units 41
- monitoring thread 96
- monitoring thread dump 96
- monsbystp 297, 341
- monswap 297, 341
- mserver.properties 264
- mstrexport 265, 266, 284
- mstrimport 265
- mutual node switching system 226
- mutual standby configuration 314, 323

N

- necessity of setting files in n-to-1 recovery system 373
- node 225
- node switching functionality based on cluster software linkage 24
- node switching functionality based on linkage with cluster software 225
- node switching operations 311

- node switching operations in the 1-to-1 node switching system 250
- node switching operations in the mutual node switching system 312
- non-protected area 118
- N seconds 190
- n-to-1 recovery system 226, 353
- Number of entire pending requests for Web applications 87
- number of entire pending requests for Web containers 86
- number of Full GC occurrences 86
- Number of pending requests for Web applications 87
- Number of pending requests for Web containers 86

O

- operations
 - Management action execution command 208
- Operations that can be executed in the 1-to-1 node switching system 297
- Option message settings when method timeout occurs 130
- ORB functionality 392, 398
- OTS functionality 392, 398
- Output destination and number of statistics files 68
- Output format
 - resource depletion monitoring information 102
- Output statistics 63
- Output target of the console log 213
- overview of accessing network resources 170

P

- partial service lock 144
- planned node switching 250, 312
- precautions during application development using method cancellation 130
- prerequisites for executing node 229
- prerequisites for standby node 229
- problems during operations 173
- Procedure for copying the Management Server settings 265
- Procedure for replacing a normal J2EE application 159
- Procedure for replacing the application using the re-deploy functionality 161
- procedure for starting and stopping systems 44
- property file for executing management action 201
- protected area 114, 118, 133
- protected area list file 118

R

- recovery server 358, 369
- redeploy 160, 164
- reload 161, 166
- replacing J2EE application 111, 113, 159, 163
- replacing J2EE application by partial locking of the service 160
- replacing J2EE application using the re-deploy functionality 160
- replacing J2EE application using the reload functionality 161
- resource depletion monitoring functionality 95, 98
- resource depletion monitoring information 94, 101
 - connection pool depletion 107
 - file descriptor depletion 105
 - HTTP request pending queue depletion 106
 - memory depletion 103
 - session depletion 107
 - thread depletion 106
 - thread dump depletion 106
- resource depletion monitoring log file 94
- restarting
 - when failure occurs 36
- runtime attributes 165

S

- sample to restart the server that issues Management events 204
- sampling time 189
- server for which nodes are to be switched 228
- service lock 111, 138, 145
 - displayed error page 140
 - method 138
 - releasing 141
 - when using load balancer 139
- Service lock of back-end system (for system using CTM) 142
- service lock partially
 - method 162
 - system configuration 162
- service lock that can be executed in back-end system (when CTM is not used) 143
- service lock that can be executed in back-end system (when CTM is used) 142
- Service lock that can be executed in Web front system 142
- service unit 32
- setting

- acquiring console log 215
- automatic execution of process by Management event 198
- each function (using management event issuing function) 201
- Management action execution command 199, 203
- message ID list file for issuing management event 198, 200
- monitoring resource depletion 201
- operation for publishing Management event 199
- procedure of automatic execution of process by Management event 198
- property file used to execute Management action 199, 201
- setting LAN status 277
- setting of Cosminexus TPBroker for integrating cluster software (in UNIX) 398
- settings for accessing network resources 170
- Settings for automatic restart 50
- Settings for automatic start 48
- settings for automatic stop 53
- Settings for Hitachi Connector Property file 100
- Settings for monitoring the J2EE application execution time 201
- Settings for mutual node switching system (In UNIX) 323
- settings for n-to-1 recovery system (in UNIX) 369
- settings for n-to-1 recovery system (in Windows) 358
- settings for server-compliant environment (in 1-to-1 node switching of application server) 276
- settings for server-compliant environment (in n-to-1 recovery system) 380
- settings for starting and stopping the system 44
- settings for the localaddr file of the Smart Agent 393, 399
- Settings for the mutual node switching system (in Windows) 314
- settings for using the ORB functionality 392, 399
- settings of 1-to-1 node switching system of application server (in UNIX) 268
- settings of 1-to-1 node switching system of application server (in Windows) 254
- settings of Cosminexus TPBroker for integrating cluster software (in Windows) 392
- settings of private property 366
- Settings of the 1-to-1 node switching system for the Management Server (In UNIX) 280
- Settings of the 1-to-1 node switching system in the Management Server (In Windows) 261

- Setting timeout value in method invocation process of Enterprise Bean 128
- Setting timeout value in processing of Web application requests 127
- spare node 225
- standby node 225, 386, 388
- starting and checking
 - logical server 33
- starting and stopping
 - logical server 33
- starting and stopping 1-to-1 node switching system in application server (in Windows) 290
- Starting and stopping a 1-to-1 node switching system for the Management Server (In UNIX) 297
- Starting and stopping a 1-to-1 node switching system for the Management Server (In Windows) 295
- starting and stopping n-to-1 recovery system (in UNIX) 388
- starting and stopping n-to-1 recovery system (in Windows) 386
- Starting and stopping system during daily operations 32
- Starting and stopping the mutual node switching system (in UNIX) 340
- Starting and stopping the mutual node switching system (In Windows) 337
- Starting and stopping the mutual node switching system for maintenance 347
- starting system 44
- statement cancel 146
- stationary IP address 253, 310
- statistics collection functionality 61
- statistics file
 - output functionality 62
- statistics file 61, 65, 67
 - file name 68
 - output destination 68
 - output format and output contents 70
- statistics file (output information)
 - DB Connector 78
 - JavaVM 73
 - JCA resource 80
 - Message-driven Bean 77
 - process resource 75
 - Stateful Session Bean 76
 - Stateless Session Bean 77
 - transaction service 81
 - URL group 84
 - Web application 81
 - Web container 83

- Status of the J2EE application 155
- stopping
 - logical server 35
 - system 46
- stopping a J2EE application 111, 112, 145
- stop process 150
- switching intervals of and number of statistics files 68
- sysdef 273, 329, 375
- system operation
 - monitoring 39

T

- Target operations of the console log output 213
- Target processes of method timeout functionality 117
- Target processes of the console log output 213
- targets applicable for method timeout (Callback method) 129
- Targets applicable for method timeout (Enterprise Bean invocation) 128
- targets applicable for method timeout (Processing of Web application requests) 127
- thread status and possibility of executing method cancellation 133
- threshold event 61
- Time interval for detecting timeout 117
- Timeout detection interval and method timeout time 117
- Timing for node switching 249
- types and meaning of operational status 42
- Types of resources that can be monitored 95
- Types of statistics and J2EE server functionality to output the statistics 62

U

- un-deploy process 151
- URL group
 - number of pending requests 87
- Usage of light transaction functionality 228

V

- vbroker.agent.addr 394, 400
- vbroker.agent.enableLocator 398
- vbroker.se.iiop_tp.host 318, 327, 380

W

- Web application
 - switching 162
- webserver.connector.nio_http.bind_host 318, 327

Windows Server Failover Cluster [224](#)

working directory for the management action
execution commands [208](#)