ExpressCluster® X Database Agent 1.0 *for Linux*

Administrator's Guide

8/31/2007
Fourth Edition

Revision History

Edition	Revised Date	Description
First	2006/09/08	New manual
Second	2006/12/12	EXPRESSCLUSTER logo has been changed.
Third	2007/6/30	Descriptions recommending using monitor resources have been added.
Fourth	2007/08/31	Added information about setup in "Preface".

Disclaimer

Trademark Information

Table of Contents

Preface ..... **vii**

Who Should Use This Guide vii
How This Guide is Organized vii
ExpressCluster X Documentation Set viii
Conventions ix
Contacting NEC x

Chapter 1 Overview of Database Agent **13**

What is ExpressCluster X Database Agent? 14
Overview of monitoring with ExpressCluster X Database Agent 14
Mechanism of monitoring 15
Target monitoring application 16
Using the Database Agent 17
System requirements for the Database Agent 17
Before using the Database Agent 17
License registration for the Database Agent 18
Registering the license interactively 18
Registering the license by specifying a license file 19

Chapter 2 Database Agent command reference **21**

List of database monitoring commands 22
Monitoring chart 22
Writing scripts to exec resource 23
Suspending and resuming monitoring 23
Database Agent command reference 24
DB2 V9 monitoring command 24
Oracle10g R2 monitoring command 28
PostgreSQL 8.1 monitoring command 32
MySQL5.0 monitoring command 37
Sybase 12.5.2 monitoring command 41

Chapter 3 How to check monitor status **47**

Verifying monitor information with monitoring commands 48
Verifying alert messages on the WebManager 48
Log collection when an error has occurred 48
Alert messages 49
Messages being output by clp_db2mon 49
Messages indicating normal operation 49
Messages due to setting error 49
Messages being output when an error is detected in database monitoring 50
Messages due to system error 51
Messages being output by clp_ora10mon 51
Messages indicating normal operation 51
Messages due to setting error 51
Messages being output when error is detected in database monitoring 52
Messages due to system error 53
Messages being output by clp_psql81mon 53
Messages indicating normal operation 53
Messages due to setting error 53
Messages being output when an error is detected in database monitoring 54
Messages due to system error 55
Messages being output by clp_mysql50mon 55
Messages indicating normal operation 55
Messages due to setting error 56
Messages being output when an error is detected in database monitoring 56
Messages due to system error 57

Messages being output by clp_sybmon 57
Messages indicating normal operation 57
Messages due to setting error 58
Messages being output when an error is detected in database monitoring 59
Messages due to system error 59

Chapter 4 Settings for ExpressCluster X Database Agent 61

Setting up ExpressCluster X Database Agent 62
Step 1 Creating a failover group 63
Step 1-1 Adding a group 63
Step 1-2 Adding a group resource (floating IP address) 63
Step 1-3 Adding a group resource (disk resource) 64
Step 2 Adding the exec resource (exec 1) for starting the target monitoring application 65
Step 3 Confirmation test for target monitoring application startup 67
Saving the cluster configuration data on a floppy disk (Windows) 67
Creating a cluster using the data in a floppy disk 68
Verifying the behavior of the target monitoring application 69
(Verification 1) Starting up a group 69
(Verification 2) Stopping a group 69
(Verification 3) Moving a group 69
(Verification 4) Failing over a group 70
Step 4 Adding the exec resource (exec 2) for starting monitoring command 71
Step 5 Adding pid monitor resource 74
Failover soon after detecting an error 76
Executing failover without stopping a server 76
Step 6 Verifying the settings for monitoring command 77
(Verification 1) Starting up a group 77
(Verification 2) Stopping a group 77
(Verification 3) Moving a group 78
(Verification 4) Failing over a group 78

Appendix A. Script templates 79

Setting up script templates 79
Installing the script templates on Windows machine 79
Installing script templates on Linux machine 80
Uninstalling the script templates 80
Uninstalling the script templates for Windows 80
Uninstalling the script templates for Linux 80
Details on script templates 81
For DB2 81
start.sh 81
stop.sh 83
For DB2 monitoring 84
start.sh 84
stop.sh 86
For Oracle10g 87
start.sh 87
stop.sh 88
startup.sql/shutdown.sql 89
For Oracle10g monitoring 90
start.sh 90
stop.sh 92
For PostgreSQL 93
start.sh 93
stop.sh 95
For PostgreSQL monitoring 96
start.sh 96
stop.sh 98
For MySQL 99
start.sh 99
stop.sh 101
my.cnf 102
For MySQL monitoring 103

start.sh 103
stop.sh 105
For Sybase Adaptive Server Enterprise (ASE) 106
start.sh 106
stop.sh 108
Start file and shutdown file 109
For ASE monitoring 110
start.sh 110
stop.sh 112

Appendix B. Glossary 113

Appendix C. Index 115

Preface

If you newly set up Agents as monitor resource with installation of version 1.1.0-1 or later rpm, please refer to the following manuals.

Getting Started Guide
- "Supported distributions and kernel versions"
- "Applications supported by monitoring options"

Reference Guide
- Chapter 6 Monitor resource details

Please refer to this manual in case that
- you set up Agents with installation of version 1.0.X-X rpm.
- you use a shell scripts to control Agents that created with version 1.0.X-X rpm for another cluster system with version 1.1.0-1 or later rpm.

Who Should Use This Guide

The *ExpressCluster X Databaser Agent Administrator's Guide* is intended for system engineers and administrators who want to build, operate, and maintain a cluster system. Instructions for setting up a cluster system with ExpressCluster X Database Agent is covered in this guide.

How This Guide is Organized

Chapter 1 **Overview of Database Agent**
This chapter provides information on a product overview of the Database Agent and how to set it up.

Chapter 2 **Database Agent command reference**
This chapter provides detailed information on the script commands.

Chapter 3 **How to check monitor status**
This chapter describes the monitoring method and the messages.

Chapter 4 **Settings for ExpressCluster X Database Agent**
This chapter describes how to create cluster configuration data using the Database Agent and how to create a cluster.

Appendix A **Script templates**
The script template used in the ExpressCluster is introduced.

Appendix B **Glossary**

Appendix C **Index**

ExpressCluster X Documentation Set

The ExpressCluster X manuals consist of the following four guides. The title and purpose of each guide is described below:

Getting Started Guide
This guide is intended for all users. The guide covers topics such as product overview, system requirements, and known problems.

Installation and Configuration Guide
This guide is intended for system engineers and administrators who want to build, operate, and maintain a cluster system. Instructions for designing, installing, and configuring a cluster system with ExpressCluster are covered in this guide.

Reference Guide
This guide is intended for system administrators. The guide covers topics such as how to operate ExpressCluster, function of each module, maintenance-related information, and troubleshooting. The guide is supplement to the *Installation and Configuration Guide*.

Administrator's Guide (Add-on product)
This guide is intended for system administrators. The detailed information on each product package is described in this guide. There are five guides for each optional product and topics such as product overview, instruction for setting up are covered:

Alert Service Administrator's Guide
Application Server Agent Administrator's Guide
Database Agent Administrator's Guide
File Server Agent Administrator's Guide
Internet Server Agent Administrator's Guide

Conventions

In this guide, **Note**, **Important**, **Related Information** are used as follows:

Note:
Used when the information given is important, but not related to the data loss and damage to the system and machine.

Important:
Used when the information given is necessary to avoid the data loss and damage to the system and machine.

Related Information:
Used to describe the location of the information given at the reference destination.

The following conventions are used in this guide.

Convention	Usage	Example
Bold	Indicates graphical objects, such as fields, list boxes, menu selections, buttons, labels, icons, etc.	In **User Name**, type your name. On the **File** menu, click **Open Database**.
Angled bracket within the command line	Indicates that the value specified inside of the angled bracket can be omitted.	`clpstat –s[-h host_name]`
#	Prompt to indicate that a Linux user has logged on as root user.	`# clpcl -s -a`
Monospace (courier)	Indicates path names, commands, system output (message, prompt, etc), directory, file names, functions and parameters.	`/Linux/1.0/eng/server/`
Monospace bold (courier)	Indicates the value that a user actually enters from a command line.	Enter the following: **`# clpcl -s -a`**
Monospace italic (courier)	Indicates that users should replace italicized part with values that they are actually working with.	`rpm –i expressclsbuilder -<version_number>-<release_number>.i686.rpm`

Contacting NEC

For the latest product information, visit our website below:

http://www.ace.comp.nec.co.jp/CLUSTERPRO/clp/global-link.html

Chapter 1 Overview of Database Agent

This chapter provides an overview of ExpressCluster X Database Agent and instructions for how to register the license.

This chapter covers:

- What is ExpressCluster X Database Agent? ····· 14
- Overview of monitoring with ExpressCluster X Database Agent····· 14
- Using the Database Agent ····· 17
- License registration for the Database Agent ····· 18

What is ExpressCluster X Database Agent?

When ExpressCluster monitors applications, failing over the application when detecting an error at application startup is possible; however, an application stalling and errors cannot be detected once application is started up.

With ExpressCluster X Database Agent, failover can be performed when an application is stalled or an error has occurred as well as at application startup. Thus you will be able to monitor applications on the cluster system.

ExpressCluster X Database Agent achieves the monitoring of an application as described above by using monitor resources[1] and monitoring commands.

For details on the function of this product of monitoring applications using monitor resources, see Chapter 6, “Monitor resource details” in the *Reference Guide*.

The remainder of this guide explains the monitoring function using monitoring commands.

Important:

It is recommended to perform monitoring using monitor resources since the WebManager and the – clpstat command enables you to monitor the status and to avoid describing dedicated scripts.

Overview of monitoring with ExpressCluster X Database Agent

ExpressCluster X Database Agent configures the following three settings to monitor the database:

1. the exec resource for starting the target monitoring application (exec 1)
2. the exec resource for starting the monitoring command (exec 2)
3. the pid monitor resource that monitors exec 2

[1] Supported from ExpressCluster X version 1.1.0-1.

Mechanism of monitoring

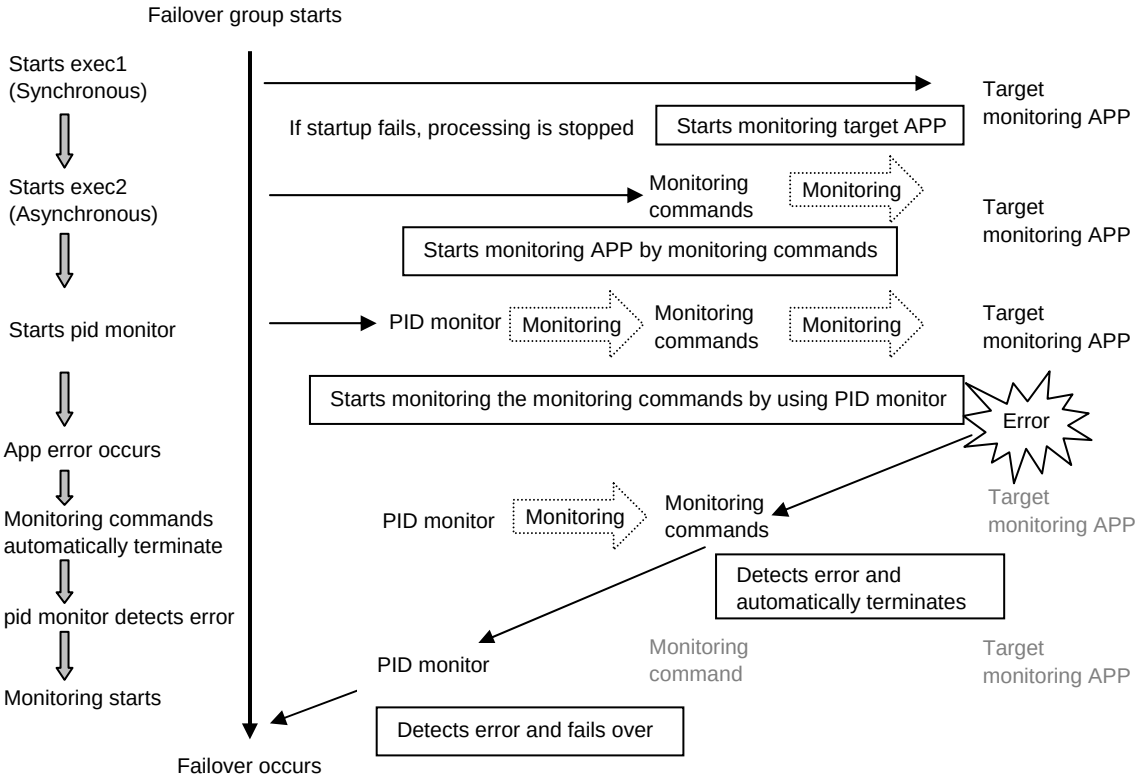
Start the target monitoring application in exec 1. Create an exec resource (exec 2) that differs from this exec 1, and in this exec 2 script, write the monitoring command that monitors the application started in exec 1. Two exec resources are created to distinguish whether an error has occurred at an application startup or after it has been started.

Configure the settings for the pid monitor resource to monitor exec 2. The monitoring commands specified in exec 2 are provided by ExpressCluster X Database Agent to monitor applications. These monitoring commands monitor the database at the intervals specified by the parameter, and if they detect a database error, they stop their own processes.

If these commands stop when a database error is detected, configure the settings to execute failover or server shutdown by the pid monitor that monitors them.

Using the mechanism above, you can detect a stalling problem and an error of the target monitoring application that cannot be detected by the ordinary settings and failover and shutdown become possible. The following figure explains this mechanism:

See below for an overview of the application (APP) monitoring using the monitoring command:



Important:

- These commands can detect an error which does not cause the target monitoring application to end abnormally (mainly stalling problem). This is indirectly achieved, by monitoring not the process ending of target monitoring application, but the monitoring operations such as attempting to access to the application server.
- The purpose of using these commands is to monitor the operation of the target monitoring application, and not to investigate or diagnose the cause when an error occurs on the application. When an error occurs, you need to use other means, such as application logs to see the details of its cause.
- Depending on the target monitoring application, access logs are output to syslog and other places, and logs are output to the local directory of target monitoring application every time you execute the monitoring processes. Since the settings cannot be controlled by these monitoring commands, configure these settings by the target monitoring application, if necessary. However, note that it may become difficult to investigate the cause because logs are not output when an error occurs, if the target monitoring application is set not to output logs.

Run the stop command (**`clp_`*`XXXXXX XXXXXX`*` --stop`**) to stop these monitoring commands. If a process is stopped by using the kill command of Linux, the monitoring command may not be restarted because management information of the monitoring command will not be initialized.

Related Information:
For details, see Note 4 at “DB2 V9 monitoring command” on page 24, and notes for other monitoring commands.

Target monitoring application

ExpressCluster X Database Agent monitors database working under the ExpressCluster environment. The following chart shows the version of ExpressCluster X Database Agent and database application that can be monitored.

For Linux IA-32

Database	ExpressCluster Database Agent 1.0-1
DB2 Universal Database V9	Yes
Oracle Database 10g Release 2	Yes
PostgreSQL 8.1	Yes
PowerGres Plus 2.0	Yes
MySQL 5.0	Yes
Sybase Adaptive Server Enterprise 12.5.2	Yes

Yes: Supported, No: Not supported

For Linux x86-64

Database	ExpressCluster Database Agent 1.0-1
DB2 Universal Database V9	Yes
Oracle Database 10g Release 2	Yes
PostgreSQL 8.1	Yes
PowerGres Plus 2.0	No

MySQL 5.0	Yes
Sybase Adaptive Server Enterprise 12.5.2	Yes

Yes: Supported, No: Not supported

To monitor databases, monitoring commands for each database are provided. Refer to Chapter 2, "Database Agent command reference" for more information on commands.

See ExpressCluster website, *http://www.ace.comp.nec.co.jp/CLUSTERPRO/global-link.html*, for supported versions of PostgreSQL and MySQL.

Using the Database Agent

In this section, how to use ExpressCluster X Database Agent is introduced. The module for the Database Agent is installed together with the ExpressCluster X. You only need to register the license to use the Database Agent.

System requirements for the Database Agent

Check each item on all servers where the Database Agent will be installed. The system requirements for the Database Agent are the following.

System requirements for the Database Agent (monitoring module)	
Hardware	IA-32 server, x86-64 server
OS	Same system requirements for the ExpressCluster Server. Database system to be monitored can function.
ExpressCluster	ExpressCluster X 1.0 or later
Memory space	7 MB (per command)

Obtain the latest update of the Database Agent. Refer to the update instructions for how to update the Database Server Agent.

Note:
Since these monitoring commands run as a client application of the database system, it is necessary to make settings for client application to run on a server. Refer to the manuals for each database system for details.

Before using the Database Agent

The following tasks need to be completed before registering the license for the Database Agent. If the tasks are not completed, refer to the Section II, "Installing and configuring ExpressCluster X" in the *Installation and Configuration Guide* for procedures and complete the tasks.

1. Installation of the ExpressCluster Server and ExpressCluster X Builder
2. Execution of the cluster creation command
3. License registration of the ExpressCluster Server

Upon completing 1 through 3 above, you can start operating the Database Agent by following the procedures below:

1. License registration of the Database Agent
   Register the license by following the procedure introduced in "License registration for the Database Agent" on the next page.

License registration for the Database Agent

License registration is necessary to use the Database Agent.

To register the license, log on as root user from the server which constitutes a cluster. You have to register a unique license key for each server.

Note:
When your ExpressCluster version does not support ExpressCluster X Database Agent, you may not be able to register the license properly. In such a case, you have to update your ExpressCluster.

Registering the license interactively

The explanation below is given using a product version as an example.

Check the following before installing a license:

- Verify that the ExpressCluster Server is installed, cluster creation command is executed, and the license for the ExpressCluster Server is registered.
- Prepare the license sheet for the Database Agent obtained from your sales agent. You will need to enter the value indicated in the license sheet.

1. Run the following command on a server:

   # clplcnsc –i –p DBAG10

2. The following is displayed. Enter 1 for the product version:

   ```
   Software license
     1   Product version
     2   Trial version
   Select the license version [1 or 2]…1
   ```

3. You will be prompted to enter the product serial number. Specify the number indicated in the license sheet:

   ```
   Enter serial number [Ex. XXX0000000]… xxxnnnnnnn
   ```

4. You will be prompted to enter the product license key. Specify the number indicated in the license sheet.

   Note:
   Enter the license sheet information exactly as indicated because the license key is case-sensitive. To avoid any confusion with other letters and numbers, “I” and “O” in capital letter are not used in the ExpressCluster license key.

   ```
   Enter license key
   [Ex. XXXXXXXX-XXXXXXXX-XXXXXXXX-XXXXXXXX]…
   xxxxxxxx-xxxxxxxx-xxxxxxxx-xxxxxxxx
   ```

   After running the command, the message “Command succeeded” is displayed in the console to indicate that the command is successfully completed. If other completion messages are displayed, refer to “License management command” in Chapter 4, “ExpressCluster command reference” in the *Reference Guide*.

Registering the license by specifying a license file

For a trial version, you may register the license using a license file in stead of a license sheet. License registration for trial version allows you to register the license for the entire cluster by registering to only one server in a cluster.

- Run the following command on a server:

  # clplcnsc –i *filepath* –p DBAG10

Assign the file path to the license file specified by option –i.

After running the command, the message “Command succeeded” is displayed in the console to indicate that the command is successfully completed. If other completion messages are displayed, refer to “License management command” in Chapter 4, “ExpressCluster command reference” in the *Reference Guide*.

This completes the setup of ExpressCluster X Database Agent. How to use the monitoring commands is introduced in the next chapter.

Chapter 2 Database Agent command reference

This chapter provides information regarding how to set up and operate ExpressCluster X Database Agent. Database monitoring commands and ExpressCluster X Database Agent commands are explained.

This chapter covers:

- List of database monitoring commands ····· 22
- Database Agent command reference····· 24

List of database monitoring commands

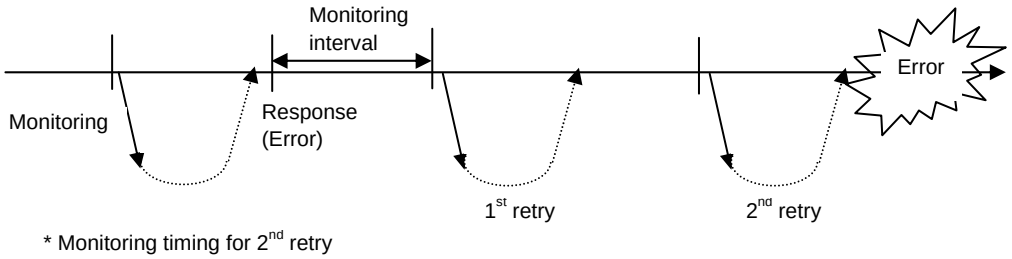
The Database Agent provides database monitoring commands to be written in the script.

Command	Used for	Refer to
clp_db2mon	Monitors DB2. Supports DB2 Universal Database V9.	Page 24
clp_ora10mon	Monitors Oracle. Supports Oracle Database 10g Release 2.	Page 28
clp_psql81mon	Supports PostgreSQL 8.1.	Page 32
clp_mysql50mon	Monitors MySQL. Supports MySQL5.0.	Page 37
clp_sybmon	Monitors Sybase Adaptive Server Enterprise. Supports Sybase Adaptive Server Enterprise 12.5.2.	Page 41

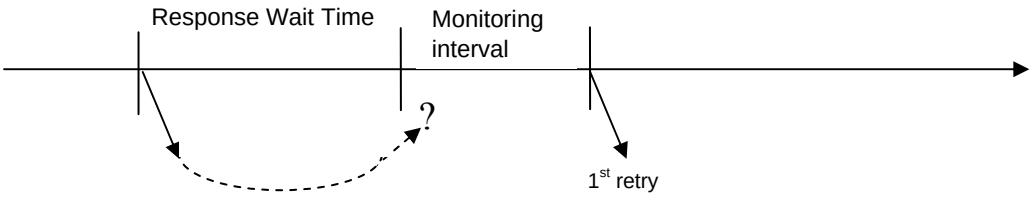
Note:
These commands should be run as root user. If a database monitoring command is run by a user other than root user, you may not be able to acquire the license information and cannot run the command. When running these commands, /usr/sbin must be added to a path although it is normally added.

Monitoring chart

The Database Agent detects an error in the following timing:



If the Agent does not get a response within the specified wait time, it is considered as an error immediately.



Writing scripts to exec resource

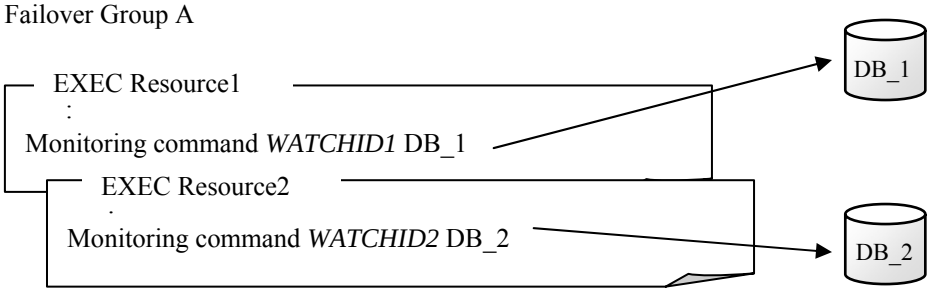
Note the following when you write a script to an exec resource:

- Before creating an exec resource in which you will write start/end of monitoring commands, complete the exec resource in which start/end of a target monitoring application is written. Then, check if a failover group successfully starts, stops, moves, and fails over. If you create an exec resource where the start/end of the monitoring commands are written without checking these issues, it would be difficult to determine the cause of an error is detected by the monitoring commands at failover group startup. For example, if an error has actually occurred, various environments of the target monitoring application were set properly, or a parameter value of the monitoring commands is appropriate, is hard to distinguish.
- Configure the dependency of the resources so that the exec resource for start/end of the Agent starts up after the exec resource in which start/end of a target monitoring application to be written starts up. If you make incorrect settings, the Agent may consider that the target monitoring application has an error.

These monitoring commands can also be written to monitor multiple databases in one failover group.

Example:

Failover Group A



See Appendix A “Script templates” for example of s writing a script.

Suspending and resuming monitoring

The Database Agent can suspend and resume monitoring. While monitoring is suspended, database maintenance can be performed. Follow the procedure below:

1. Run the monitoring command to start monitoring.
2. To suspend monitoring, run the following command from a server console whenever you wish to suspend:
   Monitoring Command watchid `--pause`
3. The following message is displayed on the ExpressCluster WebManager to indicate that monitoring has been suspended:
   `clp_xxxxmon will stop monitoring..` [ID`:`***`watchid`***]
4. To resume monitoring, run the following command from the server console whenever you wish to resume:
   Monitoring Command watchid `--resume`
5. The following message is displayed on the ExpressCluster WebManager to indicate that monitoring is suspended:
   clp_xxxxmon will start monitoring.. [ID***:watchid***]

Database Agent command reference

This section provides information on command functions. Note the following to understand the command functions:

- **Command line**

  Gives an idea what the user actually enters:

  "[ ]" indicates that the enclosed parameter may be skipped.

  "|" indicates one of the parameters separated by this symbol should be chosen.

- **Description**

  Provides information on function.

- **Option**

  Provides information on parameters used in the command line.

- **Remarks**

  Gives supplementary information such as details on parameter settings.

- **Monitoring method**

  Gives explanation on how to monitor.

- **Note**

  Gives information on what should be noted when using the command.

- **Example of command entry**

  Provides example of writing script when using the command.

DB2 V9 monitoring command

clp_db2mon: the clp_db2mon monitors DB2 V9.

Command line:

Start Monitoring clp_db2mon	*Identifier* -d *Database_name* [-m *Instance_name*] [-u *User_name*] [-p *Password*] [-t *Table_name*] [-i *Monitoring_interval*] [-c *Retry_count*] [-r *Response_wait_time*]
Stop Monitoring clp_db2mon	*Identifier* --stop [*Stop_wait_time*]
Interrupt Monitoring clp_db2mon	*Identifier* --pause
Resume Monitoring clp_db2mon	*Identifier* --resume
Display Information clp_db2mon	*Identifier* --disp
Delete Information clp_db2mon	*Identifier* --del

Description	Specifies the database name, and monitors DB2 per each database. This monitoring command stops when a DB2 error is detected. The command also stops, interrupts, and resumes monitoring. To specify --pause/--resume/--disp/--del, run from the root privileged console.

Option	*Identifier*	Specifies an identifier to uniquely identify the monitoring command. You must set an identifier.
	-d *Database_name*	Specifies the name of a database to be monitored. You must set a database name.
	-m *Instance_name*	Specifies an instance (database manager) name of the database to be monitored. Default value: db2inst1
	-u *User_name*	Specifies the user name for logging into the database. Default value: db2inst1
	-p *Password*	Specifies the password for logging into the database. Default value: ibmdb2
	-t *Table_name*	Specify the table name for monitoring to be created on the database. Default value: db2watch
	-i *Monitoring_interval*	Specifies the database monitoring interval in seconds (between 1 and 10000). Default value: 60
	-c *Retry_count*	Specifies how many retries will be made after an error is detected when monitoring the database (between 1 and 10000). Default Value: 2
	-*r* *Response_wait_time*	Specifies a response wait time (between 1 and 10000) in database monitoring in seconds. Default Value: 120
	--stop	Stops the monitoring command
	Stop_wait_time	Specifies time to wait for the monitoring command to stop normally (between 1 and 10000). Default value: 60
	--pause	Temporarily interrupts monitoring.
	--resume	Resumes monitoring.
	--disp	Displays a process ID (pid) of the monitoring command identifier managed by the monitoring command. Not for ordinary use. (See Note 4)
	--del	Deletes information about the monitoring command identifier managed by the monitoring command. Not for ordinary use. (See Note 4)

Remarks	About identifier: You need to specify a unique identifier on the system to control the monitoring command. You cannot use an identifier that is already used by a monitoring command for another monitoring command. Specify an identifier using alphanumeric characters. An identifier should be up to 255 bytes and is case sensitive. It is required to specify an identifier as the first argument of a monitoring command. About –u and –p parameters: When accessing the database by the user name with the same value as the instance name, it is not required to specify –u and -p parameters. Specify these parameters, when accessing the database as a user different from the instance name. About –t parameter: Create a table with the value specified by –t parameter on the database specified by –d parameter. The name of the table of -t parameter and the one used for operation should not overlap. About -d,-m,-u,-p and –t parameters: The actual effective length of text varies by each parameter. No verification for the effective length is performed with this monitoring command. The effective length complies with the DB2 specification.

Monitoring Method	This monitoring command monitors the following: Creates a table for monitoring on the database, and reads and writes the numeric value up to 5 digits by issuing the SQL statement. The command determines the following results as an error: (1) No response to the database connection or the issued SQL statement within a response wait time (-r parameter value) (2) An error is reported in a response to the database connection or the issued SQL statement (3) Written data and read data do not match The SQL statement to be used is "create/drop/insert/update/select.”

Note 1	This monitoring command monitors DB2, using the CLI library of DB2. For this reason, it is required to execute "source *instance user home*/sqllib/db2profile" as root user. Write this in a start script.
Note 2	If the code page of the database and the one for root user who executes this monitoring command differ, this monitoring command cannot access to the DB2 database. Write "export LANG=ja_JP.eucJP"as necessary in a start script. To check the code page of database, execute "db2 get db cfg for *Database_name*." For details, see DB2 manual.
Note 3	If values of database name, instance name, user name and password specified by a parameter differ from the DB2 environment for monitoring, DB2 cannot be monitored. Error message is displayed. Check the environment.
Note 4	Management information is not initialized properly when the monitoring command process is stopped by a kill command during the monitoring, and the monitoring command with the same identifier may fail to start. If this happens, run the "clp_db2mon identifier --disp" command and you will see the process ID corresponding to the specified identifier. Use the ps command to check the process ID of the executable file and if the process is other than the monitoring command, remove management information by using the "clp_db2mon identifier --del" command. A monitoring command malfunctions if you run the "clp_db2mon identifier --del" command by specifying a normally operating identifier.

Example of command entry	[start.sh] **export LANG=ja_JP.eucJP** **source /home/db2inst1/sqllib/db2profile** **clp_db2mon db2watch -d *Database_name*** [stop.sh] **source /home/db2inst1/sqllib/db2profile** **clp_db2mon db2watch --stop** This command starts up from an exec resource.

Oracle10g R2 monitoring command

clp_ora10mon: the clp_ora10mon monitors the Oracle10g.

Command line:

Start Monitoring clp_ora10mon	*Identifier* -d *Connection_string* [-u *User_name*] [-p *Password*] [-t *Table_name*] [-i *Monitoring_interval*] [-c *Retry_count*] [*Response_wait_time*]
Stop Monitoring clp_ora10mon	*Identifier* --stop [*Stop_wait_time*]
Interrupt Monitoring clp_ora10mon	*Identifier* --pause
Resume Monitoring clp_ora10mon	*Identifier* --resume
Display Information clp_ora10mon	*Identifier* --disp
Delete Information clp_ora10mon	*Identifier* --del

Description	Specifies the connection string, and monitors Oracle for each database. This monitoring command stops when an Oracle error is detected. The command also stops, interrupts and resumes monitoring. To specify --pause/--resume/--disp/--del, run from the root privileged console.

Option	*Identifier*	Specifies an identifier to uniquely identify the monitoring command. You must set an identifier.
	-d *Connection_string*	Specifies the connection string corresponding to the database to be monitored. You must specify the connection string.
	-*u* *User_name*	Specifies the user name to log in to the database. Default value: sys
	-p *Password*	Specifies the password to log in to the database. Default value: change_on_install
	-t *Table_name*	Specifies the table name for monitoring created on the database. Default value: orawatch
	-i *Monitoring_interval*	Specifies the database monitoring interval in seconds (between 1 and 10000). Default value: 60
	-c *Retry_count*	Specifies how many retries will be made after an error is detected when monitoring the database (between 1 and 10000). Default value: 2
	-r *Response_wait_time*	Specifies a response wait time (between 1 and 10000) in database monitoring in seconds. Default value: 120
	--stop	Stops the monitoring command
	Stop_wait_time	Specifies time to wait for the monitoring command to stop normally in seconds (between 1 and 10000). Default value: 60
	--pause	Temporarily interrupts monitoring.
	--resume	Resumes monitoring.
	--disp	Displays a process ID (pid) of the monitoring command identifier managed by the monitoring command. Not for ordinary use. (See Note 4)
	--del	Deletes information about the monitoring command identifier managed by the monitoring command. Not for ordinary use. (See Note 4)

Remarks	About identifier: You need to specify a unique identifier on the system to control the monitoring command. You cannot use an identifier that is already used by a monitoring command for another monitoring command. Specify an identifier using alphanumeric characters. An identifier should be up to 255 bytes and is case sensitive. You need to describe an identifier as the first argument of a monitoring command. About -u and –p parameters: Specify the Oracle user who can access to the database specified by `-d` parameter to the user specified by `-u` and `-p` parameters. About –t parameter: Create a table with the value specified by `-t` parameter on the database specified by `-d` parameter. The name of the table of -t parameter and the one used for operation should not overlap. About -d,-u,-p and –t: A character string that can be specified by the parameters above is up to 255 bytes. The valid length of a character string differs for each parameter, but you cannot check it with this monitoring command. The valid length follows the Oracle specification.

Monitoring Method	This monitoring command monitors the following: Creates a table for monitoring on the database, and reads and writes the numeric value up to 5 digits by issuing the SQL statement. The command determines the following results as an error: (1) No response to the database connection or the issued SQL statement within a response wait time (-r parameter value) (2) An error is informed in a response to the database connection or the issued SQL statement (3) Written data and read data do not match The SQL statement to be used is “create/drop/insert/update/select.”

Note 1	This monitoring command monitors Oracle with the Oracle interface (Oracle Call Interface). For this reason, the library for interface (libclntsh.so) needs to be installed on the server for monitoring.
Note 2	If values of a connection string, user name and password specified by a parameter are different from the Oracle environment for monitoring, Oracle monitoring cannot be done. Error message is displayed. Check the environment.
Note 3	If DBA user authentication method is only the OS authentication: If DBA user authentication method is only the OS authentication and "NONE" is specified to "REMOTE_LOGIN_PASSWORDFIL" in the Oracle initialization parameter file, specify a database user without DBA authority to –u and –p parameters of "clp_ora10mon". In case of specifying a database user with DBA authority, an ORA-01031 error occurs and monitoring cannot be executed when starting clp_ora10mon.
Note 4	Management information is not initialized properly when the monitoring command process is stopped by a kill command during the monitoring, and the monitoring command with the same identifier may fail to start. If this happens, run the "clp_ora10mon identifier --disp" command and you will see the process ID corresponding to the specified identifier. Use the ps command to check the process ID of the executable file and if the process is other than the monitoring command, remove management information by using the "clp_ora10mon identifier --del" command. A monitoring command malfunctions if you run the "clp_ora10mon identifier --del" command by specifying a normally operating identifier.
Note 5	Use the character set supported by OS when creating a database. If Japanese is set to NLS_LANGUAGE in the Oracle initialization parameter file, specify English by NLS_LANG (environment variable of Oracle.) Specify the character set corresponds to the database. If it is not specified, an alert message of Event ID (0) is not displayed to the alert view correctly. However, as for an error of when connecting to the database such as incorrect user name and alert message may not be displayed correctly. For the NLS parameter and NLS_LANG settings, see the *Globalization Support Guide* by Oracle Corporation. For alert messages, refer to Alert messages on page 49.

Example of command entry	[start.sh] **export ORACLE_HOME=/opt/oracle/product/10.2.0/db_1** **export LD_LIBRARY_PATH=$ORACLE_HOME/lib** **export NLS_LANG=AMERICAN_AMERICA.JA16EUC** **clp_ora10mon orawatch -d *Connection_string*** [stop.sh] **export ORACLE_HOME=/opt/oracle/product/10.2.0/db_1** **export LD_LIBRARY_PATH=$ORACLE_HOME/lib** **clp_ora10mon orawatchsmtpmon smtpwatch --stop** This command starts up from an exec resource. Specify environment variables ORACLE_HOME and LD_LIBRARY_PATH to execute this monitoring command.

PostgreSQL 8.1 monitoring command

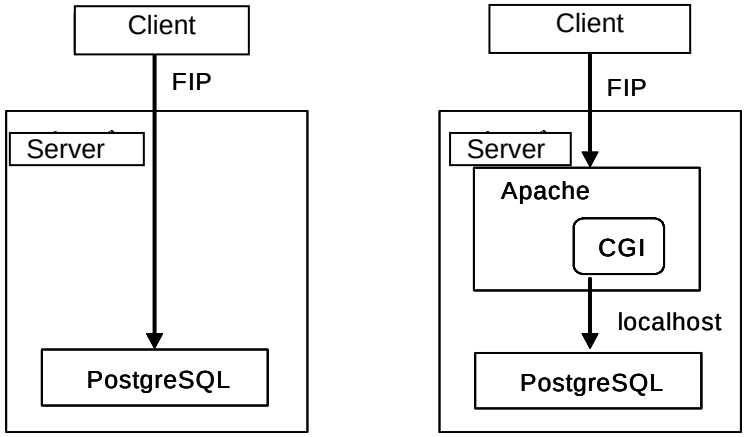
clp_psql81mon: the clp_psql81mon monitors PostgreSQL 8.1 and PowerGres Plus (see Note 1).

Command line

Start Monitoring clp_psql81mon	*Identifier* -d *Database_name* [-a *IP_address*] [-n *Port_number*] [-u *User_name*] [-p *Password*] [-t *Table_name*] [-i *Monitoring_interval*] [-c *Retry_count*] [-r *Response_wait_time*]
Stop Monitoring clp_psql81mon	*Identifier* --stop [*Stop_wait_time*]
Interrupt Monitoring clp_psql81mon	*Identifier* --pause
Resume Monitoring clp_psql81mon	*Identifier* --resume
Display Information clp_psql81mon	*Identifier* --disp
Delete Information clp_psql81mon	*Identifier* --del

Description	Specifies a database name and monitors PostgreSQL for each database. If an error of PostgreSQL is detected, this monitoring command is terminated. The command also stops, interrupts and resumes monitoring. Specify "--pause/--resume/--disp/--del" from the root authorized console.

Option	*Identifier*	Specifies an identifier to uniquely identify the monitoring command. You must set an identifier.
	-d *Database_name*	Specifies the name of a database to be monitored. You must set a database name.
	-a *IP_address*	Specifies an IP address for making a connection from the PostgreSQL client to PostgreSQL. Default value: 127.0.0.1
	-n *Port_number*	Specifies the port number of PostgreSQL. Default value: 5432 (If the PGPORT environment variable is set, the value of PG PORT).
	-u *User_name*	Specifies the user name for logging into the database. Default value: postgres
	-p *Password*	Specifies the password for logging into the database. Default value: none
	-t *Table_name*	Specify the table name for monitoring created on the database. Default value: psqlwatch
	-i *Monitoring_interval*	Specifies the database monitoring interval in seconds (between 1 and 10000). Default value: 60
	-c *Retry_count*	Specifies how many retries will be made after an error is detected when monitoring the database (between 1 and 10000). Default value: 2
	-r *Response_wait_time*	Specifies a response wait time (between 1 and 10000) in database monitoring in seconds. Default value: 120
	--stop	Stops the monitoring command
	Stop_wait_time	Specifies time to wait for the monitoring command to stop normally in seconds (between 1 and 10000). Default value: 60
	--pause	Temporarily interrupts monitoring.
	--resume	Resumes monitoring.
	--disp	Displays a process ID (pid) of the monitoring command identifier managed by the monitoring command. Not for ordinary use. (See Note 3)
	--del	Deletes information about the monitoring command identifier managed by the monitoring command. Not for ordinary use. (See Note 3)

Remarks	About identifier: You need to specify a unique identifier on the system to control the monitoring command. You cannot use an identifier that is already used by a monitoring command as another monitoring command. Specify an identifier using alphanumeric characters. An identifier should be up to 255 bytes and is case sensitive. An identifier needs to be written as the first argument of a monitoring command. About –a parameter: Specify FIP when accessing Postgre with FIP. This parameter does not have to be specified for localhost connection. Example:  The IP address specified by this parameter needs to be permitted the connection in the “pg_hba.conf” file. About –n parameter: If the port number is specified when PostgreSQL is started, specify this parameter. If the port number is not specified when PostgreSQL is started, 5432 is used. About -u and –p parameters: Specify the user name and password set by PostgreSQL (These are not the user name and password on Linux). Specify these parameters to monitor a database that limits access permitted users in “pg_hba.conf.” About –t parameter: Create a table with the value specified by –t parameter on the database specified by –d parameter. The name of the table of -t parameter and the one used for operation should not overlap. About -d,-a,-u,-p and –t parameters: A text can be specified up to 255 bytes with these parameters. The actual effective length varies by each parameter. No verification for the effective length is performed with this monitoring command. The effective length complies with the PostgreSQL specification.

Monitoring Method	This monitoring command monitors the following: Creates a table for monitoring on the database, and reads and writes the numeric value up to 5 digits by issuing the SQL statement. The command determines the following results as an error: (1) No response to the database connection or the issued SQL statement within a response wait time (-r parameter value) (2) An error is informed in a response to the database connection or the issued SQL statement (3) Written data and read data do not match The SQL statement to be used is create/drop/insert/update/select.

Note 1	This monitoring command uses the libpq library of PostgreSQL to monitor PostgreSQL. If this monitoring command cannot be executed ("Monitor xxx failed.(1:Process does not exist.(pid:xx)" is displayed on the WebManager alert view), set the application library path to the path where the libpq library of PostgreSQL exists. This monitoring command requires the PostgreSQL library of the following version. clp_psql81mon libpq.so.4 Example export LD_LIBRARY_PATH=/usr/local/pgsql/lib A library name is displayed in the regular error output of the server, for example, "clp_psql81mon: error while loading shared libraries: libpq.so.xx: cannot open shared object file: No such file or directory". If a library of the appropriate file name does not exist, soft link to the executable file of the PostgreSQL library. This is required especially when monitoring PowerGres Plus. Example ln –s libpq.so.xx libpq.so.xx
Note 2	If a value specified by a parameter differs from the PostgreSQL environment for monitoring, a message of indicating an error is displayed on the alert view of the WebManager. Check the environment.
Note 3	Management information is not initialized properly when the monitoring command process is stopped by a kill command during monitoring and the monitoring command with the same identifier may fail to start. If this happens, run the "clp_psql81mon Identifier –disp" command and you will see the process ID corresponding to the specified identifier. Use the ps command to check the process ID of the executable file and if the process is other than the monitoring command, remove management information by using the "clp_psql81mon Identifier –del" command. If you specify the identifier running normally and execute the "clp_psql81mon Identifier –del" command, a monitoring command malfunctions, so do not perform this operation.
Note 4	About client authentication: In this monitoring command, the following authentication method that can be set to the "pg_hba.conf" file has been checked its operation. trust, md5, password

Example of command entry	[start.sh] **export LD_LIBRARY_PATH=/usr/local/pgsql/lib** **clp_psql81mon psqlwatch -d *Database_name*** [stop.sh] **clp_psql81mon psqlwatch --stop** This monitoring command can be started from the exec resource.

MySQL5.0 monitoring command

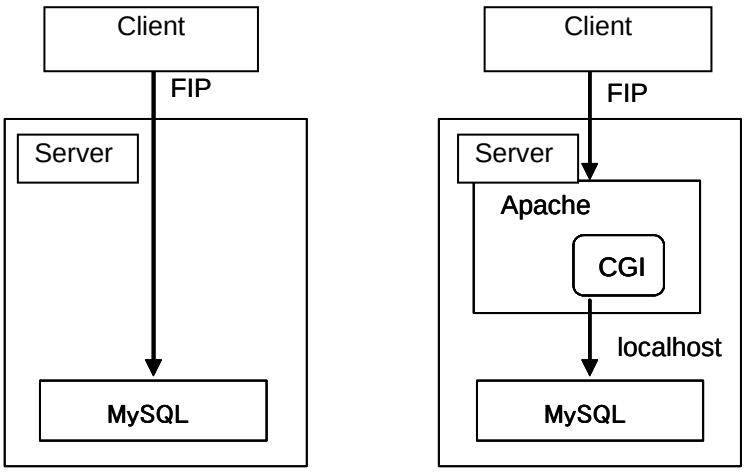
clp_mysql50mon: the clp_mysql50mon monitors MySQL5.0.

Command line

Start Monitoring clp_mysql50mon	*Identifier* -d *Database_name* [-a *IP_address*] [-n *Port_number*] [-u *User_name*] [-p *Password*] [-t *Table_name*] [-i *Monitoring_interval*] [-c *Retry_count*] [-r *Response_wait_time*]
Stop Monitoring clp_mysql50mon	*Identifier* --stop [*Stop_wait_time*]
Interrupt Monitoring clp_mysql50mon	*Identifier* --pause
Resume Monitoring clp_mysql50mon	*Identifier* --resume
Display Information clp_mysql50mon	*Identifier* --disp
Delete Information clp_mysql50mon	*Identifier* --del

Description	Specifies the database name and monitors MySQL for each database. If an error of MySQL is detected, this monitoring command is terminated. The command also stops, interrupts and resumes monitoring. Specify "--pause/--resume/--disp/--del" from the root authorized console.

Option	*Identifier*	Specifies an identifier to uniquely identify the monitoring command. You must set an identifier.
	-d *Database_name*	Specify the database name to be monitored. You must set a database name.
	-a *IP_adderss*	Specifies an IP address for making a connection from the MySQL client to MySQL. Default value: localhost
	-n *Port_number*	Specifies the port number of MySQL. Default value: 3306
	-u *User_name*	Specifies the user name to log in to the database. Default value: none
	-p *Password*	Specifies the password to log in to the database. Default value: none
	-t *Table_name*	Specifies the table name for monitoring to be created on the database. Default value: mysqlwatch
	-i *Monitoring_interval*	Specifies the database monitoring interval in seconds (between 1 and 10000). Default value: 60
	-c *Retry_count*	Specifies how many retries will be made after an error is detected when monitoring the database (between 1 and 10000). Default value: 2
	-r *Response_wait_time*	Specifies a response wait time (between 1 and 10000) in database monitoring in seconds. Default value: 120
	--stop	Stops the monitoring command
	Stop_wait_time	Specifies time to wait for the monitoring command to stop normally in seconds (between 1 and 10000). Default value: 60
	--pause	Temporarily interrupts monitoring.
	--resume	Resumes monitoring.
	--disp	Displays a process ID (pid) of the monitoring command identifier managed by the monitoring command. Not for ordinary use. (See Note 3)
	--del	Deletes information about the monitoring command identifier managed by the monitoring command. Not for ordinary use. (See Note 3)

Remarks	About identifier: You need to specify a unique identifier on the system to control the monitoring command. You cannot use an identifier that is already used by a monitoring command as another monitoring command. Specify an identifier using alphanumeric characters. An identifier should be up to 255 bytes and is case sensitive. You need to describe an identifier as the first argument of a monitoring command. About –a parameter: If this parameter value is localhost (excluding when 127.0.0.1 is specified), TCP/IP communication is not used for the connection to MySQL. Set a file for a socket specified by "my.cnf" in a script for starting this monitoring command, for example "export MYSQL_UNIX_PORT=/var/lib/mysql/mysql.sock." If you specify the IP address (including 127.0.0.1) with this parameter, the specified IP address needs to be permitted the connection in the grant statement. When accessing MySQL with the floating IP address, specify the floating IP address. As for the localhost connection, do not specify the address or specify 127.0.0.1. Example:  About –n parameter: Specify this parameter when the port number to connect to MySQL is not default value of MySQL (3306). About –t parameter: Create a table with the value specified by –t parameter on the database specified by –d parameter. The name of the table of -t parameter and the one used for operation should not overlap. About -d,-a,-u,-p and –t parameters: The actual effective length of text varies by each parameter. No verification for the effective length is performed with this monitoring command. The effective length complies with the MySQL specification.

Monitoring Method	This monitoring command monitors the following: Creates a table for monitoring on the database, and reads and writes the numeric value up to 5 digits by issuing the SQL statement. The command determines the following results as an error: (1) No response to the database connection or the issued SQL statement within a response wait time (-r parameter value) (2) An error is informed in a response to the database connection or the issued SQL statement (3) Written data and read data do not match The SQL statement to be used is create/drop/insert/update/select.

Note 1	This monitoring command monitors MySQL, using the libmysqlclient library of MySQL. If this monitoring command cannot be executed ("Monitor xxx failed.(1:Process does not exist.(pid:xx))"is displayed on the WebManager alert view), check that "libmysqlclient.so.xx" exists in the installation directory of the MySQL library. A library name also is displayed in the regular error output of the server, for example, "clp_mysql50mon: error while loading shared libraries: libmysqlclient.so.xx: cannot open shared object file: No such file or directory". This monitoring command requires the library of the following version of MySQL. clp_mysql50mon libmysqlclient.so.15
Note 2	If a value specified by a parameter differs from the MySQL environment for monitoring, an error message is displayed on the WebManager alert view. Check the environment.
Note 3	Management information is not initialized properly when the monitoring command process is stopped by a kill command during monitoring, and the monitoring command with the same identifier may fail to start. If this happens, run the "clp_mysql50mon identifier –disp" command and you will see the process ID corresponding to the specified identifier. Use the ps command to check the process ID of the executable file and if the process is other than the monitoring command, remove management information by using the "clp_mysql50mon identifier –disp" command. If you specify the identifier running normally and execute the "clp_mysql50mon identifier –disp" command, a monitoring command malfunctions, so do not perform this operation.

Example of command entry	[start.sh] **`export MYSQL_UNIX_PORT=/var/lib/mysql/mysql.sock`** **`clp_mysql50mon mysqlwatch -d`** ***`Database_name`*** [stop.sh] **`clp_mysql50mon mysqlwatch --stop`** This monitoring command can be started from the exec resource.

Sybase 12.5.2 monitoring command

clp_sybmon: the clp_sybmon monitors Sybase 12.5.2.

Command line

Start Monitoring clp_sybmon	*Identifier* –d *Database_name* -s *Database_server_name* [-u *User_name*] [-p *Password*] [-t *Table_name*] [-i *Monitoring_interval*] [-c *Retry_count*] [-r *Response_wait_time*]
Stop Monitoring clp_sybmon	*Identifier* --stop [*Stop_wait_time*]
Interrupt Monitoring clp_sybmon	*Identifier* --pause
Resume Monitoring clp_sybmon	*Identifier* --resume
Display Information clp_sybmon	*Identifier* --disp
Delete Information clp_sybmon	*Identifier* --del

Description	Specifies the database name and database server name, and monitors ASE for each database. If an error of ASE is detected, this monitoring command is terminated. The command also stops, interrupts and resumes monitoring. Specify "--pause/--resume/--disp/--del" from the root authorized console.

Option	*Identifier*	Specify an identifier to uniquely identify the monitoring command. You must set an identifier.
	-d *Database_name*	Specifies the database name to be monitored. You must set a database name.
	-s *Database_server_name*	Specifies a database server name to be monitored. You must set a database server name.
	-u *User_name*	Specifies the user name when log in to the database. Default value: sa
	-p *Password*	Specifies the password when log in to the database. Default value: none
	-t *Table_name*	Specify the table name for monitoring to be created on the database. Default value: sybasemonitor
	-i *Monitoring_interval*	Specifies the database monitoring interval in seconds (between 1 and 10000). Default value: 60
	-c *Retry_count*	Specifies how many retries will be made after an error is detected when monitoring the database (between 1 and 10000). Default value: 2
	-r *Response_wait_time*	Specifies a response wait time (between 1 and 10000) in database monitoring in seconds. Default value: 120
	--stop	Stops the monitoring command.
	Stop_wait_time	Specifies time to wait for the monitoring command to stop normally in seconds (between 1 and 10000). Default value: 60
	--pause	Temporarily interrupts monitoring.
	--resume	Resumes monitoring.
	--disp	Displays a process ID (pid) of the monitoring command identifier managed by the monitoring command. Not for ordinary use. (See Note 3)
	--del	Deletes information about the monitoring command identifier managed by the monitoring command. Not for ordinary use. (See Note 3)

Remarks	About identifier: You need to specify a unique identifier on the system to control the monitoring command. You cannot use an identifier that is already used by a monitoring command as another monitoring command. Specify an identifier using alphanumeric characters. An identifier should be up to 255 bytes and is case sensitive. You need to describe an identifier as the first argument of a monitoring command. About –s parameter: Specify the database server name that has been set when installing ASE with this parameter. You can check the database server name with the interfaces in the ASE installation directory. About -u and –p parameters: Specifies the user name and password set by ASE (These are not the user name and password on Linux). If a user name and password are not specified, use "sa" user account and a blank password to connect to the database server. About –t parameter: Create a table with the value specified by –t parameter on the database specified by –d parameter. The name of the table of -t parameter and the one used for operation should not overlap. About -d, -s,-u,-p and –t parameters: A text can be specified up to 255 bytes. The actual effective length varies by each parameter. No verification for the effective length is performed with this monitoring command. The effective length complies with the ASE specification.

Monitoring Method	This monitoring command monitors the following: Creates a table for monitoring on the database, and reads and writes the numeric value up to 5 digits (decimal number) by issuing the SQL statement. The command determines the following results as an error: (1) No response to the database connection or the issued SQL statement within a response wait time (-r parameter value) (2) An error is informed in a response to the database connection or the issued SQL statement (3) The written data and read data do not match The SQL statement to be used is "create/drop/insert/update/select."

Note 1	This monitoring command monitors ASE using Open Client DB-Library/C of ASE. If this monitoring command cannot be executed (”Monitor XXX failed.(1:Process does not exist.(pid=XXX)” is displayed on the WebManager alert view), set the application library path to the path where `libsybdb.so` exists. Example export LD_LIBRARY_PATH=/opt/sybase/OCS-12_5/lib/ A library name is displayed in the regular error output of the server, for example, “clp_sybmon: error while loading shared libraries: libsybdb.so: cannot open shared object file: No such file or directory”.
Note 2	If a value specified by a parameter differs from the ASE environment for monitoring, an error message is displayed on the WebManager alert view. Check the environment.
Note 3	Management information is not initialized properly when the monitoring command process is stopped by a kill command during monitoring, and the monitoring command with the same identifier may fail to start. If this happens, run the “clp_sybmon identifier –disp” command and you will see the process ID corresponding to the specified identifier. Use the ps command to check the process ID of the executable file and if the process is other than the monitoring command, remove management information by using the “clp_sybmon identifier –disp” command. If you specify the identifier running normally and execute the “clp_sybmon identifier –disp” command, a monitoring command malfunctions, so do not perform this operation.

Example of command entry	[start.sh] **`export LD_LIBRARY_PATH=/opt/sybase/OCS-12_5/lib`** **`clp_sybmon sybwatch -d Database_name -s Database_server_name`** [stop.sh] **`clp_sybmon sybwatch --stop`** This monitoring command can be started from the exec resource.

For each of the monitoring commands, user names and passwords need to be specified. It is required to write each of them explicitly as parameters of monitoring commands in start scripts of failover groups. A user name and a password are important data for security reason and it is recommended not to specify them explicitly.

If you write a combination of user name and password to a password management file in advance, the password management function can notify a user name and password to a monitoring command only by explicitly specifying a user name. A user name and password are managed per monitoring command.

Each monitoring command operates according to the registration status of a user name and password to a password management file. It is not required to set a special application and environment to use the user name management function.

To use the password management function, it is required to create a new password management file.

* Create the following files to manage passwords if needed.

Monitoring command name	Password management file
clp_db2mon	/opt/nec/clusterpro/work/clp_db2mon
clp_ora10mon	/opt/nec/clusterpro/work/clp_ora10mon
clp_psql81mon	/opt/nec/clusterpro/work/clp_psql81mon
clp_mysql50mon	/opt/nec/clusterpro/work/clp_mysql50mon
clp_sybmon	/opt/nec/clusterpro/work/clp_sybmon

A password management file can be accessed only by root user.

Separate a user and a password by a comma (,) in the information of a password management file. When registering more than one user, add by starting a new line.

```
user1,password1
user2,password2
user3,password3
        :
        :
userN,passwordN
```

Note:
These user names and passwords can be specified up to 255 bytes.
Do not specify unnecessary space and tab codes.
Do not specify the same user name more than once.

When using a password management function, you have to write the -u parameter specified by a user name. Do not write the -p parameter specified by a password in the description of the monitoring command.

The parameter specifications and operations of the monitoring command are as follows.

	With –u parameter	**Without –u parameter**
With –p parameter	A value specified by each parameter is valid	Default user name and a password specified by a parameter are valid
Without –p parameter	If a password corresponding to a user name that is specified by –u parameter is written in a password management file, this parameter is valid. If a password is not written, a user name specified by –u parameter and default password are valid.	A default value of each command is valid

Chapter 3 How to check monitor status

This chapter provides instructions for how to check the monitor status by using ExpressCluster X Database Agent. This chapter also includes information on results of commands and messages on the screen or in the log.

This chapter covers:

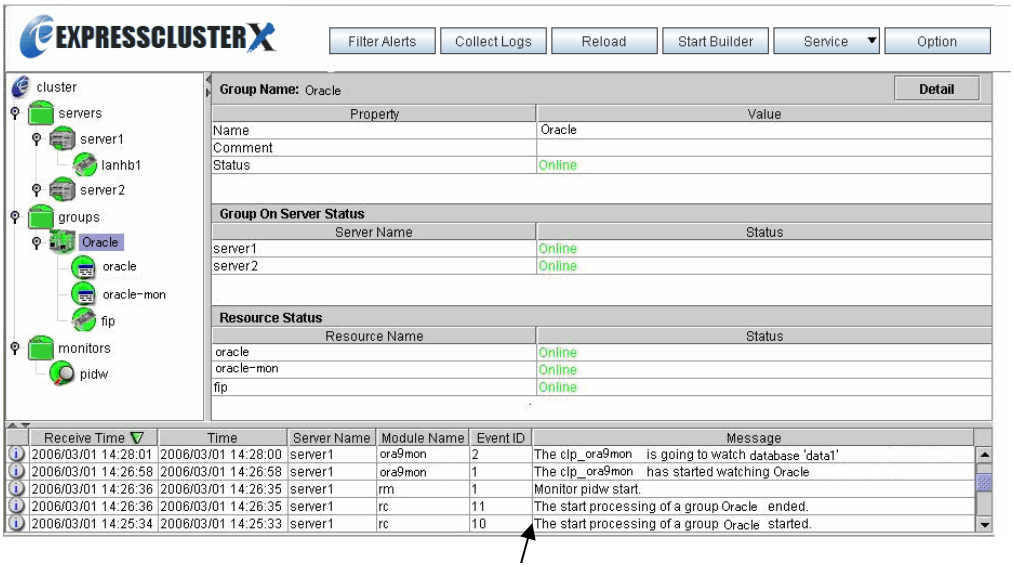
- Verifying monitor information with monitoring commands ····· 48
- Alert messages ····· 49
- Messages being output by clp_db2mon ····· 49
- Messages being output by clp_ora10mon ····· 51
- Messages being output by clp_psql81mon ····· 53
- Messages being output by clp_mysql50mon ····· 55
- Messages being output by clp_sybmon ····· 57

Verifying monitor information with monitoring commands

Verify the monitor status by using the monitoring commands on the ExpressCluster WebManager alert view. The message being output to the alert view is also output to the syslog of the server that executes the monitoring commands.

Verifying alert messages on the WebManager

You can view alert messages that ExpressCluster X displays on the WebManager.



Displayed on the ExpressCluster WebManager alert view.

As shown above, messages are displayed on the alert view of the lower part of the WebManager window. If the messages shown are long, they are displayed in multiple lines. In such a case, other message may interrupt a message shown in multiple lines depending on the timing.

The same message displayed in the alert message is logged to the syslog.

Refer to “Alert messages” on page 49 for details on alert messages.

Log collection when an error has occurred

Error logs of monitoring commands will be produced in the same directory where error logs of the ExpressCluster Server are produced. Logs are collected in the same way as how ExpressCluster logs are collected. For more information, see “Collecting logs by using the WebManager” in Chapter 1, “Functions of the WebManager” and “Collecting logs (clplogcc command)” in Chapter 4, “ExpressCluster command reference” in the *Reference Guide*.

Alert messages

Messages being output by clp_db2mon

Messages indicating normal operation

#	Message	Explanation	Remarks
1	clp_db2mon started monitoring the DB2UDB daemon. [ID:xx]	clp_db2mon has started up.	-
2	clp_db2mon will start monitoring the database ' xxx '. [ID:xx]	clp_db2mon has started monitoring the database xxx.	If this message is displayed immediately after the message above, an error may have occurred. Troubleshoot the problem according to the error message.
3	clp_db2mon will stop monitoring the DB2UDB database ' xxx '. [ID:xx]	clp_db2mon stops monitoring.	-
7	clp_db2mon will stop monitoring. [ID:xx]	clp_db2mon suspended monitoring.	Monitoring is suspended by using the --pause command.
8	clp_db2mon will start monitoring. [ID:xx]	clp_db2mon restarted monitoring.	Monitoring is restarted by using the --resume command.
9	clp_db2mon got the password. [ID:xx]	clp_db2mon acquired the password from the password management file.	-
52	clp_db2mon trial version is effective till nn/nn/nn (mm/dd/yyyy).	Running with a trial version license.	-

Messages due to setting error

#	Message	Explanation	Remarks
5	clp_db2mon will not watch DB2UDB database ' xxx '. [ID:xx]	Monitoring process is not performed due to a setting error.	Troubleshoot the problem by following the message displayed immediately before.
11	The parameter value of clp_db2mon is invalid. [ID:xx]	The clp_db2mon parameter value is invalid in its format.	Check the parameter value of the monitoring command.
12	'-d' parameter is not specified at the clp_db2mon command. [ID:xx]	`-d` parameter is not specified to the clp_db2mon command.	Check the parameter value of the monitoring command.
13	An identifier is not specified in the clp_db2mon command.	An identifier is not specified in the clp_db2mon command.	Check the parameter value of the monitoring command.
14	The specified identifier is already used. [ID:xx]	clp_db2mon cannot be restarted because the same identifier is already used.	Check the parameter value of the monitoring command.
15	clp_db2mon could not perform the end processing. [ID:xx]	End processing could not be performed with --stop parameter.	Check the parameter value of the monitoring command.

#	Message	Explanation	Remarks
21	Failed to connect to the database ' xxx '. [ID:xx]	An error has occurred in the database connection function of DB2.	The database name specified by `-d` parameter may be incorrect. Troubleshoot the problem by following the message (ID=0) displayed immediately before.
24	Specified user name ' xxx ' does not exist. The clp_db2mon will terminate. [ID:xx]	The user name specified by `-u` parameter does not exist.	Check that the specified user name exists.
25	The DB2 instance ' xxx ' has not been started. [ID:xx]	The instance specified by `-m` parameter has not been started.	Check that the specified instance has been started.
26	The code page of database ' xxx ' is not correct. [ID:xx]	The code page of database does not match the one of environment in which this monitoring command is executed.	Check the code page of database in “db2 get db cfg for database_name”, and specify “export LANG=ja_JP.eucJP.”
51	The license of clp_db2mon is not registered.	The license is not registered	Register the license.
53	The trial version license has expired in nn/nn/nn (mm/dd/yyyy).	The trial version license has expired.	-
55	The trial version license is effective from nn/nn/nn (mm/dd/yyyy).	The trial version license cannot be used yet.	-
56	The registration license overlaps.	The license key you have registered overlaps with another key.	Register a different license key for each server.

Messages being output when an error is detected in database monitoring

#	Message	Explanation	Remarks
6	clp_db2mon detected an error and will be terminated. [ID:xx]	Terminating because clp_db2mon detected an error.	Troubleshoot the problem by following the message displayed immediately before.
21	Failed to connect to the database ' xxx '. [ID:xx]	An error has occurred in the database connection function of DB2.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.
31	clp_db2mon has detected an error in DB2UDB database ' xxx ' (stall). [ID:xx]	There is no response when monitoring DB2.	Check that the database system has no error.
32	clp_db2mon has detected an error in DB2UDB database ' xxx ' (data access error). [ID:xx]	When reading the DB2 database, the read data and the one written immediately before differ.	Check that the database system has no error.
33	clp_db2mon has detected an error in DB2UDB database ' xxx '. [ID:xx]	An error has occurred in DB2.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.
34	Failed to execute SQL statement(xxx). [ID:xx]	Failed to execute SQL statement normally.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.
35	Error occurred in DB2UDB API xxx. [ID:xx]	An error has occurred in API for DB2.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.
0	SQLnnnn SQLSTATE=nnnnn xxxxxxxx	An error code output by DB2.	See the manual of the DB2 message.

Messages due to system error

#	Message	Explanation	Remarks
42	clp_db2mon has detected system error (xxx nn). [ID:xx]	A Linux system error has occurred. xxx indicates a function name and nn indicates an error code.	Check the system status based on error code.
54	Failed to check the license information of clp_db2mon.	Failed to verify the license information.	The license management module of ExpressCluster may be old. Check the module update.

Messages being output by clp_ora10mon

Messages indicating normal operation

#	Message	Explanation	Remarks
1	clp_ora10mon started monitoring Oracle.the SMTP daemon. [ID:xx]	clp_ora10mon has started.	-
2	clp_ora10mon will start monitoring database ' xxx '. [ID:xx]	clp_ora10mon has started monitoring the database xxx.	If this message is displayed immediately after the message above, an error may have occurred. Troubleshoot the problem according to the error message.
3	clp_ora10mon will stop monitoring Oracle database ' xxx '. [ID:xx]	clp_ora10mon stops monitoring.	-
7	clp_ora10mon will stop monitoring. [ID:xx]	clp_ora10mon suspended monitoring.	Monitoring is suspended by using the --pause command.
8	clp_ora10mon will start monitoring. [ID:xx]	clp_ora10mon restarted monitoring.	Monitoring is restarted by using the --resume command.
9	clp_ora10mon got the password. [ID:xx]	clp_ora10mon acquired the password from the password management file.	-
52	clp_ora10mon trial version is effective till nn/nn/nn (mm/dd/yyyy).	Running with a trial version license.	-

Messages due to setting error

#	Message	Explanation	Remarks
5	clp_ora10mon will not watch Oracle database ' xxx '. [ID:xx]	Monitoring process is not performed due to a setting error.	Troubleshoot the problem by following the message displayed immediately before.
11	The parameter value of clp_ora10mon is invalid. [ID:xx]	The clp_ora10mon parameter value is invalid in its format.	Check the parameter value of the monitoring command.
12	-d' parameter is not specified at the clp_ora10mon command. [ID:xx]	`-d` parameter is not specified to the clp_ora10mon command.	Check the parameter value of the monitoring command.
13	An identifier is not specified in the clp_ora10mon command.	The identifier is not specified in clp_ora10mon. End processing could not be performed with –stop parameter.	Check the parameter value of the monitoring command.

#	Message	Explanation	Remarks
14	The specified identifier is already used. [ID:xx]	clp_ora10mon cannot be restarted because the same identifier is already used.	Check the parameter value of the monitoring command
15	clp_ora10mon could not perform the end processing. [ID:xx]	End processing could not be performed with --stop parameter.	Check the parameter value of the monitoring command
22	Failed to connect with the server xxx. [ID:xx]	An error has occurred in the server connection function of Oracle.	A connection string specified by –d parameter may be incorrect. Troubleshoot the problem by following the message (ID=0) displayed immediately before.
23	Failed to start the user session xxx. [ID:xx]	An error has occurred in the session start function of Oracle.	Check that the specified user name and password are correct.
51	The license of clp_ora10mon is not registered.	The license is not registered	Register the license.
53	The trial version license has expired in nn/nn/nn (mm/dd/yyyy).	The trial version license has expired.	-
55	The trial version license is effective from nn/nn/nn (mm/dd/yyyy).	The trial version license cannot be used yet.	-
56	The registration license overlaps.	The license key you have registered overlaps with another key.	Register a different license key for each server.

Messages being output when error is detected in database monitoring

#	Message	Explanation	Remarks
6	clp_ora10mon detected an error and will be terminated. [ID:xx]	Terminating because clp_ora10mon detected an error.	Troubleshoot the problem by following the message displayed immediately before.
22	Failed to connect with the server xxx. [ID:xx]	An error has occurred in the database connection function of Oracle.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.
31	clp_ora10mon detected an error in Oracle database ' xxx ' (stall). [ID:xx]	There is no response when monitoring Oracle.	Check that the database system has no error.
32	clp_ora10mon detected an error in Oracle database ' xxx ' (data access error). [ID:xx]	When reading the Oracle database, the read data and the one written immediately before differ.	Check that the database system has no error.
33	clp_ora10mon has detected an error in Oracle database ' xxx '. [ID:x]	An error has occurred in Oracle database.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.
34	Failed to execute SQL statement(xxx). [ID:xx]	Failed to execute SQL statement normally.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.
35	Error occurred in Oracle API xxx. [ID:xx]	An error has occurred in Oracle API.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.
0	ORA- nnnnn:xxxxxxxx	This is an error code output by Oracle.	See the manual of the Oracle message.

Messages due to system error

#	Message	Explanation	Remarks
42	clp_ora10mon has detected system error (xxx nn). [ID:xx]	A Linux system error has occurred. xxx indicates a function name and nn indicates an error code.	Check the system status based on error code.
54	Failed to check the license information of clp_ora10mon.	Failed to verify the license information.	The license management module of ExpressCluster may be old. Check the module update.

Messages being output by clp_psql81mon

Messages indicating normal operation

#	Message	Explanation	Remarks
1	clp_psql81mon started monitoring the PostgreSQL. daemon. [ID:xx]	clp_psql81mon has started up.	-
2	clp_psql81mon will start monitoring the database ' xxx '. [ID:xx]	clp_psql81mon has started monitoring the database xxx.	If this message is displayed immediately after the message above, an error may have occurred. Troubleshoot the problem according to the error message.
3	clp_psql81mon will stop monitoring the PostgreSQL database ' xxx '. [ID:xx]	clp_psql81mon stops monitoring.	-
7	clp_psql81mon will stop monitoring. [ID:xx]	clp_psql81mon suspended monitoring.	Monitoring is suspended by using the --pause command.
8	clp_psql81mon will start monitoring. [ID:xx]	clp_psql81mon restarted monitoring.	Monitoring is restarted by using the --resume command.
9	clp_psql81mon got the password. [ID:xx]	clp_psql81mon acquired the password from the password management file.	-
52	clp_psql81mon trial version is effective till nn/nn/nn (mm/dd/yyyy).	Running with a trial version license.	-

Messages due to setting error

#	Message	Explanation	Remarks
5	clp_psql81mon will not watch PostgreSQL database ' xxx '. [ID:xx]	Monitoring process is not performed due to a setting error.	Troubleshoot the problem by following the message displayed immediately before.
11	The parameter value of clp_psql81mon is invalid. [ID:xx]	The parameter value of clp_psql81mon is invalid.	Check the parameter value of the monitoring command.

#	Message	Explanation	Remarks
12	'-d' parameter is not specified at the clp_psql81mon command. [ID:xx]	`-d` parameter is not specified to the clp_psql81mon command.	Check the parameter value of the monitoring command.
13	An identifier is not specified in the clp_psql81mon command.	An identifier is not specified to the clp_psql81mon command.	Check the parameter value of the monitoring command.
14	The specified identifier is already used. [ID:xx]	clp_psql81mon cannot be started, because the same identifier has been already executed.	Check the parameter value of the monitoring command.
15	clp_psql81mon could not perform the end processing. [ID:xx]	End processing could not be performed with –stop parameter.	Check the parameter value of the monitoring command.
21	Failed to connect to the database ' xxx '. [ID:xx]	An error has occurred in the database connection function of PostgreSQL.	A database name specified by –d parameter or a user name/password may be incorrect. Troubleshoot the problem by following the message (ID=0) displayed immediately before.
51	The license of clp_psql81mon is not registered.	The license is not registered	Register the license.
53	The trial version license has expired in nn/nn/nn (mm/dd/yyyy).	The trial version license has expired.	-
55	The trial version license is effective from nn/nn/nn (mm/dd/yyyy).	The trial version license cannot be used yet.	-
56	The registration license overlaps.	The license key you have registered overlaps with another key.	Register a different license key for each server.

Messages being output when an error is detected in database monitoring

#	Message	Explanation	Remarks
6	clp_psql81mon detected an error and will be terminated. [ID:xx]	clp_psql81mon detected an error and will be terminated.	Troubleshoot the problem by following the message displayed immediately before.
21	Failed to connect to the database ' xxx '. [ID:xx]	An error has occurred in the database connection function of PostgreSQL.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.
31	clp_psql81mon has detected an error in PostgreSQL database ' xxx ' (stall). [ID:xx]	When monitoring PostgreSQL, there is no response.	Check that the database system has no error.
32	clp_psql81mon has detected an error in PostgreSQL database ' xxx ' (data access error). [ID:xx]	When reading the PostgreSQL database, the read data and the one written immediately before differ.	Check that the database system has no error.
33	clp_psql81mon has detected an error in PostgreSQL database ' xxx '. [ID:xx]	An error has occurred in PostgreSQL database.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.

#	Message	Explanation	Remarks
34	Failed to execute SQL statement(xxx). [ID:xx]	Failed to execute SQL statement normally.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.
35	Error occurred in PostgreSQL API xxx. [ID:xx]	An error has occurred in API for PostgreSQL.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.
0	xxxxxxxx	This is an error code output by PostgreSQL.	See the manual of PostgreSQL message.

Messages due to system error

#	Message	Explanation	Remarks
42	clp_psql81mon has detected system error (xxx nn). [ID:xx]	A Linux system error has occurred. xxx indicates a function name and nn indicates an error code.	Check the system status based on the error code.
54	Failed to check the license information of clp_psql81mon.	Failed to verify the license information.	The license management module of ExpressCluster may be old. Check the module update.

Messages being output by clp_mysql50mon

Messages indicating normal operation

#	Message	Explanation	Remarks
1	clp_mysql50mon started monitoring the MySQL daemon. [ID:xx]	clp_mysql50mon has started up.	-
2	clp_mysql50mon will start monitoring the database ' xxx '. [ID:xx]	clp_mysql50mon has started monitoring the database xxx.	If this message is displayed immediately after the message above, an error may have occurred. Troubleshoot the problem according to the error message.
3	clp_mysql50mon will stop monitoring the MySQL database ' xxx '. [ID:xx]	clp_mysql50mon stops monitoring.	-
7	clp_mysql50mon will stop monitoring. [ID:xx]	clp_mysql50mon suspended monitoring.	Monitoring is suspended by using the --pause command.
8	clp_mysql50mon will start monitoring. [ID:xx]	clp_mysql50mon restarted monitoring.	Monitoring is restarted by using the --resume command.
9	clp_mysql50mon got the password. [ID:xx]	clp_mysql50mon acquired the password from the password management file.	-
52	clp_mysql50mon trial version is effective till nn/nn/nn (mm/dd/yyyy).	Running with a trial version license.	-

Messages due to setting error

#	Message	Explanation	Remarks
5	clp_mysql50mon will not watch MySQL database ' xxx '. [ID:xx]	Monitoring process is not performed due to a setting error.	Troubleshoot the problem by following the message displayed immediately before.
11	The parameter value of clp_mysql50mon is invalid. [ID:xx]	The parameter value of clp_mysql50mon is invalid in its format.	Check the parameter value of the monitoring command.
12	'-d' parameter is not specified at the clp_mysql50mon command. [ID:xx]	`-d` parameter is not specified to the clp_mysql50mon command.	Check the parameter of the monitoring command.
13	An identifier is not specified in the clp_mysql50mon command.	An identifier is not specified to the clp_mysql50mon command.	Check the parameter of the monitoring command.
14	The specified identifier is already used. [ID:xx]	clp_mysql50mon cannot be started because the same identifier has been already executed.	Check the parameter of the monitoring command.
15	clp_mysql50mon could not perform the end processing. [ID:xx]	End processing could not be performed with –stop parameter.	Check the parameter of the monitoring command.
21	Failed to connect to the database ' xxx '. [ID:xx]	An error has occurred in the database connection function of MySQL.	A database name specified by –d parameter or a user name/password may be incorrect. Troubleshoot the problem by following the message (ID=0) displayed immediately before.
51	The license of clp_mysql50mon is not registered.	The license is not registered.	Register the license.
53	The trial version license has expired in nn/nn/nn (mm/dd/yyyy).	The trial version license has expired.	-
55	The trial version license is effective from nn/nn/nn (mm/dd/yyyy).	The trial version license cannot be used yet.	-
56	The registration license overlaps.	The license key you have registered overlaps with another key.	Register a different license key for each server.

Messages being output when an error is detected in database monitoring

#	Message	Explanation	Remarks
6	clp_mysql50mon detected an error and will be terminated. [ID:xx]	clp_mysql50mon detected an error and will be terminated.	Troubleshoot the problem by following the message displayed immediately before.
21	Failed to connect to the database ' xxx '. [ID:xx]	An error has occurred in the database connection function of MySQL.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.
31	clp_mysql50mon has detected an error in MySQL database ' xxx ' (stall). [ID:xx]	There is no response when monitoring MySQL.	Check that the database system has no error.

#	Message	Explanation	Remarks
32	clp_mysql50mon has detected an error in MySQL database ' xxx ' (data access error). [ID:xx]	When reading the MySQL database, the read data and the one written immediately before differ.	Check that the database system has no error.
33	clp_mysql50mon has detected an error in MySQL database ' xxx '. [ID:xx]	An error has occurred in MySQL database.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.
34	Failed to execute SQL statement(xxx). [ID:xx]	Failed to execute SQL statement normally.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.
35	Error occurred in MySQL API xxx. [ID:xx]	An error has occurred in API of MySQL.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.
0	xxxxxxxx	This is an error code output by MySQL.	See the manual of MySQL message.

Messages due to system error

#	Message	Explanation	Remarks
42	clp_mysql50mon has detected system error (xxx nn). [ID:xx]	A Linux system error has occurred. xxx indicates a function name and nn indicates an error code.	Check the system status based on the error code.
54	Failed to check the license information of clp_mysql50mon.	Failed to verify the license information.	The license management module of ExpressCluster may be old. Check the module update.

Messages being output by clp_sybmon

Messages indicating normal operation

#	Message	Explanation	Remarks
1	clp_sybmon started monitoring the ASE. daemon. [ID:xx]	clp_sybmon has started up.	-
2	clp_sybmon will start monitoring the database ' xxx '. [ID:xx]	clp_sybmon has started monitoring the database xxx.	If this message is displayed immediately after the message above, an error may have occurred. Troubleshoot the problem according to the error message.
3	clp_sybmon will stop monitoring the DB2UDB database ' xxx '. [ID:xx]	clp_sybmon stops monitoring.	-
7	clp_sybmon will stop monitoring. [ID:xx]	clp_sybmon suspended monitoring.	Monitoring is suspended by using the --pause command.
8	clp_sybmon will start monitoring. [ID:xx]	clp_sybmon restarted monitoring.	Monitoring is restarted by using the --resume command.

#	Message	Explanation	Remarks
9	clp_sybmon got the password. [ID:xx]	clp_sybmon acquired the password from the password management file.	-
52	clp_sybmon trial version is effective till nn/nn/nn (mm/dd/yyyy).	Running with a trial version license.	-

Messages due to setting error

#	Message	Explanation	Remarks
5	clp_sybmon will not watch ASE database ' xxx '.	Monitoring process is not performed due to a setting error.	Troubleshoot the problem by following the message displayed immediately before.
11	The parameter value of clp_sybmon is invalid. [ID:xx]	The parameter value of clp_sybmon is invalid in its format.	Check the parameter value of the monitoring command.
12	'-d' parameter is not specified at the clp_sybmon command. [ID:xx]	`-d` parameter is not specified to the clp_sybmon command.	Check the parameter value of the monitoring command.
13	An identifier is not specified in the clp_sybmon command.	An identifier is not specified to the clp_sybmon command.	Check the parameter value of the monitoring command.
14	The specified identifier is already used. [ID:xx]	clp_sybmon cannot be started, because the same identifier has been already executed.	Check the parameter of the monitoring command.
15	clp_sybmon could not perform the end processing. [ID:xx]	End processing could not be performed with –stop parameter.	Check the parameter of the monitoring command.
16	'-s' parameter is not specified at the clp_sybmon command.	`-s` parameter is not specified to the clp_sybmon command.	Check the parameter of the monitoring command.
21	Failed to connect to the database ' xxx '. [ID:xx]	An error has occurred in the database connection function of ASE.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.
51	The license of clp_sybmon is not registered.	The license is not registered	Register the license.
53	The trial version license has expired in nn/nn/nn (mm/dd/yyyy).	The trial version license has expired.	-
55	The trial version license is effective from nn/nn/nn (mm/dd/yyyy).	The trial version license cannot be used yet.	-
56	The registration license overlaps.	The license key you have registered overlaps with another key.	Register a different license key for each server.

Messages being output when an error is detected in database monitoring

#	Message	Explanation	Remarks
6	clp_sybmon detected an error and will be terminated. [ID:xx]	clp_sybmon detected an error and will be terminated.	Troubleshoot the problem by following the message displayed immediately before.
21	Failed to connect to the database ' xxx '.	An error has occurred in the database connection function of ASE.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.
31	clp_sybmon has detected an error in ASE database ' xxx ' (stall). [ID:xx]	There is no response when monitoring ASE,	Check that the database system has no error.
32	clp_sybmon has detected an error in ASE database ' xxx ' (data access error). [ID:xx]	When reading the ASE database, the read data and the one written immediately before differ.	Check that the database system has no error.
33	clp_sybmon has detected an error in ASE database ' xxx '. [ID:xx]	An error has occurred in ASE database.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.
34	Failed to execute SQL statement(xxx). [ID:xx]	Failed to execute SQL statement normally.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.
35	Error occurred in ASE API xxx. [ID:xx]	An error has occurred in ASE API.	Troubleshoot the problem by following the message (ID=0) displayed immediately before.

Messages due to system error

#	Message	Explanation	Remarks
42	clp_sybmon has detected system error (xxx nn). [ID:xx]	A Linux system error has occurred. xxx indicates a function name and nn indicates an error code.	Check the system status based on error code.
54	Failed to check the license information of clp_sybmon.	Failed to verify the license information.	The license management module of ExpressCluster may be old. Check the module update.

Chapter 4 Settings for ExpressCluster X Database Agent

This chapter provides an explanation on how to set ExpressCluster X Database Agent.

This chapter covers:

- Setting up ExpressCluster X Database Agent ····· 62
- Step 1 Creating a failover group ····· 63
- Step 2 Adding the exec resource (exec 1) for starting the target monitoring application ····· 65
- Step 3 Confirmation test for target monitoring application startup ····· 67
- Step 4 Adding the exec resource (exec 2) for starting monitoring command ····· 71
- Step 5 Adding pid monitor resource ····· 74
- Step 6 Verifying the settings for monitoring command ····· 77

Setting up ExpressCluster X Database Agent

To monitor the applications by using ExpressCluster X Database Agent, follow the steps below.

Step 1 Creating a failover group (for the target monitoring application)

Create a failover group for monitoring the target monitoring application and performing failover when an error occurs. Then add a group resource excluding exec resource, such as disk resource and IP resource.

If you have already created a failover group before setting up ExpressCluster X Database Agent and it is used for monitoring, use that group. In this case, you do not need to perform Step 1.

Step 2 Adding exec 1 (for starting the target monitoring application)

Add the exec resource for starting the target monitoring application to the failover group you have created in Step 1. In this guide, this exec resource is called exec 1.

Step 3 Confirmation test

If you have finished Step 2, reflect the settings information to the server, and check that the target monitoring application is started normally by exec 1.

Step 4 Adding exec 2 (for starting the monitoring command)

Add the exec resource for starting the monitoring command to the failover group. In this guide, this exec resource is called exec 2.

Step 5 Adding a pid monitor resource

Add a pid monitor resource for monitoring the exec 2 that has been added in Step 4.

Step 6 Verifying the settings for monitoring command and pid monitor resource

Reflect the settings information of the exec 2 and the pid monitor resource that you have added in Steps 4 and 5 to the server, and check that they can operate normally.

Step 1 Creating a failover group

Create a failover group for the target monitoring application. Add the group resource excluding exec resource such as this disk resource or IP resource.

Note:
If you have already created a failover group before setting up ExpressCluster X Database Agent, you can use this group. In this case, you do not have to perform Step 1-1 to 1-3.

Step 1-1 Adding a group

Add a failover group to an existing cluster system. The explanation is given with an example of creating a failover group that includes FIP resource and disk resource to add to a cluster system that consists of two servers (server1, 2). Replace the value when you actually operate.

Note:
When creating a new cluster, refer to Chapter 3, "Creating the cluster configuration data using the Builder" in the *Installation and Configuration Guide*.

1. Start up the Builder.
   (Default path: C:\Program Files\CLUSTERPRO\clpbuilder-l\clptrek.html)
2. Open the cluster configuration data where a group will be added.
3. Click **Groups** in the tree view, and select **Add** on the **Edit** menu.
4. **Group Definition** dialog box is displayed. Enter the group name **failover1** in the **Name** box, and click **Next**.
5. Click **server1** in the **Available Servers** and click **Add**. **server1** is added to the **Servers that can run the Group**. Likewise, add server2.
6. Check that **server1** and **server2** are set in this order to the **Servers that can run the Group,** and then click **Finish**.

Proceed to the following Step 1-2.

Step 1-2 Adding a group resource (floating IP address)

Add a floating IP address to the group you created in Step 1-1.

1. Click **failover1** in the tree view, and click **Add** on the **Edit** menu.
2. The **Resource Definition** dialog box is displayed. Select the group resource type "**floating ip resource**" in the **Type** box, and enter the group name (floating ip address) in the **Name** box. Click **Next**.
3. Enter the IP address (ex. **10.0.0.12**) in the **IP Address** box. Click **Next**.
4. **Recovery Operation at Activation Failure Detection** and **Recovery Operation at Deactivation Failure Detection** are shown. Ignore them and click **Next**.
5. Click **Finish** in the screen shown.

Proceed to the following Step 1-3.

Step 1-3 Adding a group resource (disk resource)

Add a disk resource to the group you added a FIP resource in Step 1-2.

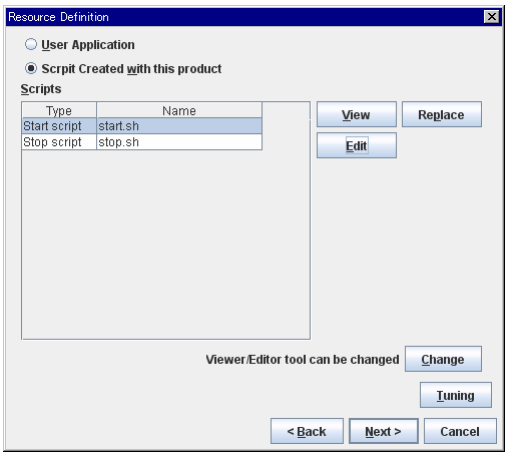
1. Click **failover1** in the tree view, and click **Add** on the **Edit** menu.
2. Enter the 2nd group resource information in the **Resource Definition** dialog box. Select the group resource type (**disk resource**) in the **Type** box, and enter the group name **disk1** in the **Name** box. Click **Next**.
3. Enter the device name (ex. **/dev/sdb2**), mount point (ex. **/mnt/sdb2**) to their corresponding boxes. Select the file system (ex. **ext3**) on the **File System** box and the disk type **Disk** from the **Disk Type** box. Click **Next**.
4. **Recovery Operation at Activation Failure Detection** and **Recovery Operation at Deactivation Failure Detection** are shown. Ignore them and click **Next**.
5. Click **Finish** in the screen shown.

Proceed to the following Step 2.

Step 2 Adding the exec resource (exec 1) for starting the target monitoring application

Add the exec resource for starting the target monitoring application to the failover group that you have created in Step 1. In this guide, this exec resource is called exec 1. You can use the script templates for starting the target monitoring application. Here, the settings for starting the DB2 daemon are given as an example.

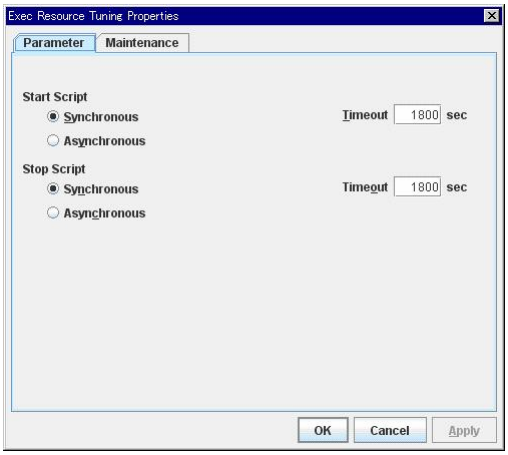
1. If the Builder has not been started yet, start it.
   (Default path: C:\Program Files\CLUSTERPRO\clpbuilder-l\clptrek.html)
2. Right-click the target failover group, and click **Add Resource**.
3. The **Resource Definition** dialog box is displayed. Select **execute resource** in **Type**. Enter any name you like (enter DB2 in this example), and click **Next**.
4. The following window is displayed. Confirm that **Script Created with this product** is selected. Select **Start script** in **Type** in the **Scripts** table, and click **Replace**.



5. Select the script to start the target monitoring application. Move to the folder of the script templates, and specify the script, “start.sh” in the “db2” folder. Click **Open**.

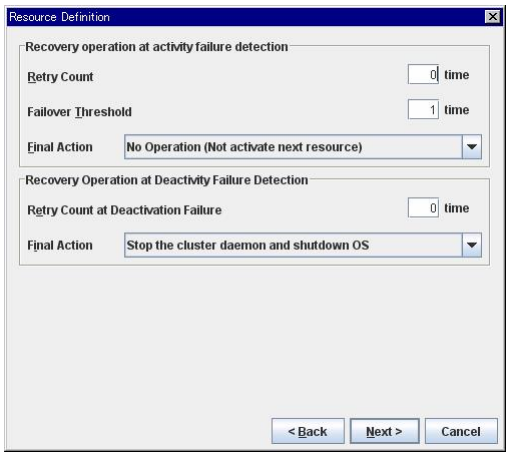
Note:
The following is the default installation folder of the script templates.
C:\Program Files\CLUSTERPRO\clpbuilder-l\scripts\linux

6. The replacement confirmation dialog box is displayed. Specify the original file to be replaced and the path, and select **Yes**.
7. In the same way, replace the end script. Select **Stop Script** in **Type** from the **Scripts** table and click **Replace**. Specify the script, “stop.sh” in the “db2” folder, and click **Open**.
8. Modify the script. If you click **Edit**, the editor will open. You can modify the script according to the environment. See Appendix A “Script templates”, for what to be modified.
9. Click **Tuning**, and check that **Synchronous** is set for **Start Script** and **Stop Script** on the **Parameter** tab. (**Synchronous** is set by default.)

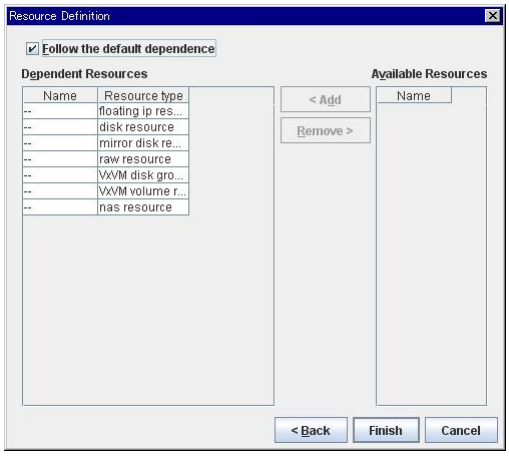


Note:
If you select **Synchronous**, the script is executed with the return value. Here, you need to set **Synchronous** to **Start Scrip** and **Stop Script** because it is required to confirm that the application has been started successfully.

10. In the following screen, you do not need to change the settings in particular. If necessary, configure the settings according to your environment. Click **Next**.



11. In the following screen, you do not need to change the settings. Make sure that the disk resource and IP resource exist on **Dependent Resources**, and click **Finish**.



12. Exec resource for the target monitoring application (exec 1) has been created. Move to the Step 3.

Step 3 Confirmation test for target monitoring application startup

After completing the Steps 1 and 2, convert the configuration information to the server, and check that the monitored application is started normally by exec 1 after that.

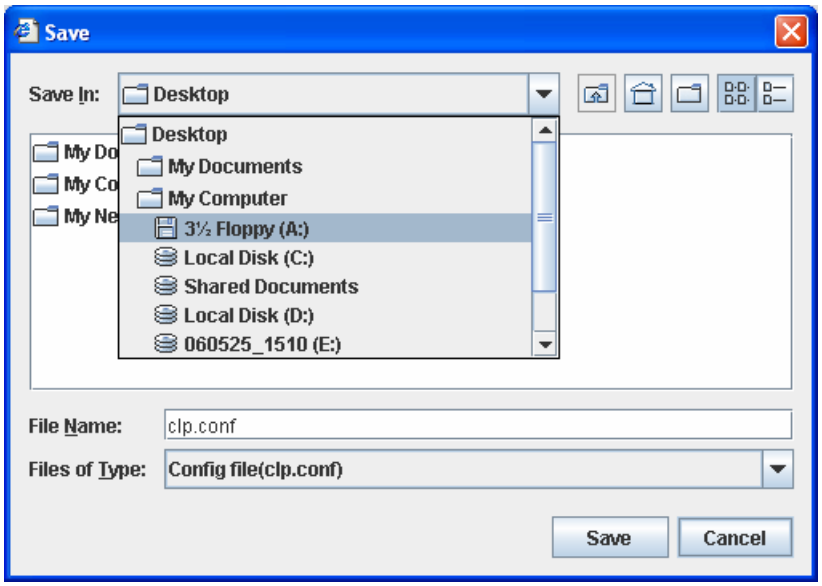
The following describes how to modify the settings to the server using an example of editing the cluster configuration data with the Builder installed on the Windows machine and saving it to the floppy disk.

Related Information:
For using the Builder installed on the Linux machine or saving the cluster configuration data on the file system, refer to "Saving the cluster configuration data" in Chapter 3 "Creating the cluster configuration data using the Builder" in the *Installation and Configuration Guide*.

Saving the cluster configuration data on a floppy disk (Windows)

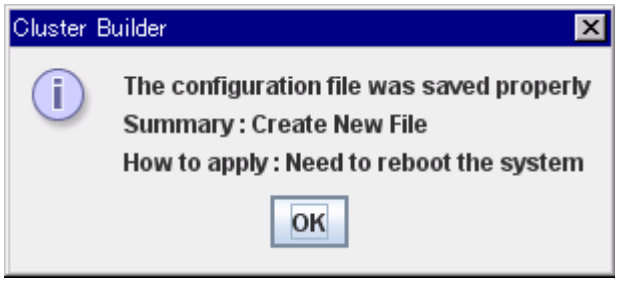
To save the cluster configuration data created with the Builder on Windows machine, follow the procedures below:

1. Insert the floppy disk into the floppy disk drive. Click **Save the configuration file** on the **File** menu. You can directly save it in the floppy disk without creating a directory in the floppy disk.
2. In the **Save** dialog box, select the floppy disk drive and click **Save**.



Note:
- Two files (clp.conf and clp.conf.rep) and the directory (scripts) are saved. If any of these are missing, the command to create a cluster does not run successfully. Make sure to treat these three as a set. If you have changed a newly created configuration data, clp.conf.bak is also created as well as those three files.

When the cluster configuration data is saved, the following message is displayed:



3. Check if the two files (clp.conf and clp.conf.rep) and one directory (scripts) are located in the floppy disk. If they are in a directory in the floppy disk, you need to specify the directory when you run the command to create a cluster.

Creating a cluster using the data in a floppy disk

To create a cluster using the cluster configuration data saved on a floppy disk, follow the procedures below. If the data is saved in a file system, refer to Chapter 4, "Installing ExpressCluster" in the *Installation and Configuration Guide*.

Note:
- You have to restart all servers after installing the server RPM.
- The server used to create a cluster is the one specified as a master server at the time of creating the cluster configuration data.

1. Insert the floppy disk that contains the cluster configuration data created with the Builder into the floppy disk drive of the server specified as a master server.

Note:
In the clpcfctrl command, /dev/fd0 is used as the floppy disk device, and /mnt/floppy as the mount point by default. If your environment is different from these default values, specify them using the option. For details on option, refer to the *Reference Guide*.

2. Distribute the configuration data saved in the floppy disk to the servers.
   - To use the floppy disk that contains the data saved on Linux, run the following command:
     clpcfctrl --push -l
   - To use the floppy disk (1.44-MB formatted) that contains the data saved on Windows, or to use the Windows-formatted floppy disk that contains the data on Linux, run the following command:
     clpcfctrl --push -w

Note:
If the two files (clp.conf and clp.conf.rep) and one directory (scripts) that are requisite to create a cluster are saved directly in the floppy disk, you do not need to specify any directory.

The message below is displayed:

```
Need to shutdown system and reboot
please shutdown system after push. (hit return) :
```

3. Press **ENTER**.
   When the cluster is successfully created, the message below is displayed:

```
success.(code:0)
```

Related Information:
For troubleshooting of the clpcfctrl command, refer to the *Reference Guide*.

Verifying the behavior of the target monitoring application

Execute the operations below from the WebManager or the command line to verify that the target monitoring application is running normally.

Note:
Be sure to execute Steps 1 to 4 below before you move to the next step (adding exec 2). If you execute these steps after adding exec 2, it becomes difficult to distinguish whether an error is caused when the application is being started or after it has been started.

Execute the following steps to confirm the monitoring target operation with the WebManager. Start the WebManager (http://*FIP_address_for_the WebManager_group*: *port_number* (default value is 29003)), and execute the following four verification steps.

(Verification 1) Starting up a group

1. Right-click the icon of the failover group that you want to start in the WebManager tree view, and click **Start**.
2. Check that the group icon has turned green and the application is running normally.

(Verification 2) Stopping a group

1. Right-click the icon of the failover group that you want to start in the WebManager tree view, and click **Stop**.
2. Check that the group icon has turned gray and the application has stopped.

Note:
Check that the group has been started and/or stopped among all the servers that start the failover group.

(Verification 3) Moving a group

Move a failover group among servers. Check that the started group properly moves by moving the group to other servers in order.

1. Right-click the icon of the failover group that you want to move in the WebManager tree view.
2. Click the group icon, and check that the group has been started on the destination server in the table view. Check that the application is running normally.

Note:
Depending on the application startup/stop process described in the script, it may take a few minutes to complete moving a group. See the tree view on the main screen to check that the group has been moved successfully.

(Verification 4) Failing over a group

Shut down the server on which the failover group is running, and check that the group fails over on the destination server.

1. In the WebManager tree view, right-click the server on which the group you want to fail over is running, and click **Reboot OS**.
2. Restart the server, and check that the failover group fails over to the destination server. Check that the application is running normally on the destination server.

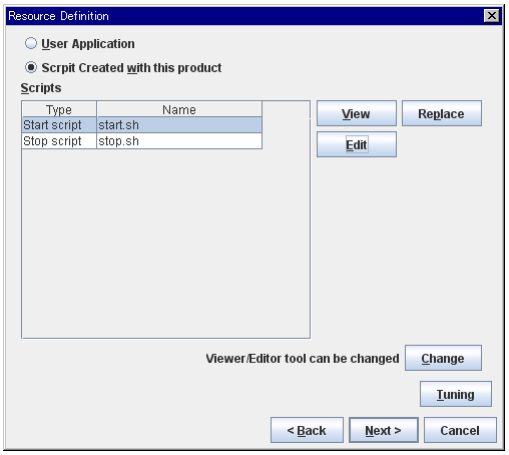
You have completed verifying that the application is running normally by exec 1. Move to Step 4.

Note:
To verify the operation of the application from the command line, run the following commands.
(Verification 1) clpgrp -s *group_name*
(Verification 2) clpgrp -t *group_name*
(Verification 3) clpgrp -m *group_name*
Run the clpstat command to confirm the cluster status.

Step 4 Adding the exec resource (exec 2) for starting monitoring command

Add the exec resource for starting the monitoring command to the failover group. In this guide, this exec resource is called exec 2.

1. If the Builder has not been started, start it.
   (Default path: C:\Program Files\CLUSTERPRO\clpbuilder-l\clptrek.html)
2. Right-click the target failover group, and click **Add Resource**.
3. The **Resource Definition** dialog box is displayed. Select **execute resource** in **Type**. Enter any name you like (enter "DB2MON" in this example), and click **Next**.
4. The following window is displayed. Confirm that **Script Created with this product** is selected. Select **Start script** in **Type** in the **Scripts** table, and click **Replace**.



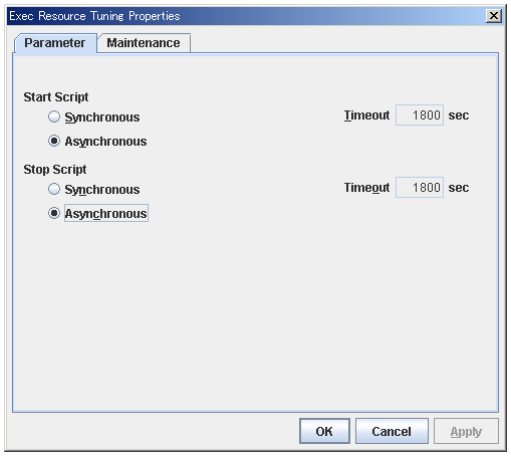
5. Specify the script to start the monitoring command. Move to the folder of the script templates, and specify the script "start.sh" in the "`db2-mon`" folder. Then, click **Open**.

Note:

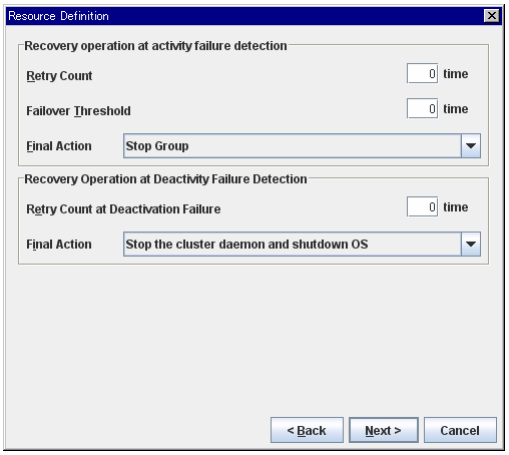
- The following is the default installation folder of script template:
  C:\Program Files\CLUSTERPRO\clpbuilder-l\scripts\linux
- The target monitoring application name is given to the folder of the script templates for starting up the target monitoring application. "-mon" is added after the target monitoring application name in the folder of the script templates for starting up the target monitoring application.

6. The confirmation dialog box asking for replacement is displayed. Specify the original file to be replaced and the path, and select **Yes**.
7. In the same way, replace the end script. Select **Stop Script** in **Type** and click **Replace**. Specify the script "stop.sh" in the "`db2-mon`" folder, and click **Open**.
8. Modify the script. Open the editor by clicking **Edit** to modify the script according to your environment. See Appendix A "Script templates" for what to be modified.
9. Click **Tuning**, and check that **Asynchronous** is selected for **Start Script** and **Synchronous** for **Stop Script** on the **Parameter** tab.

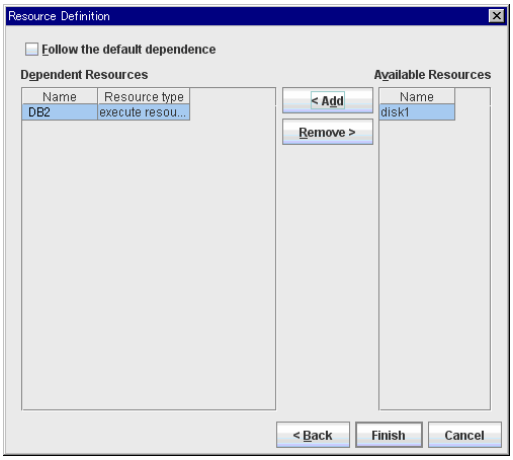
Important:
If you select **Asynchronous**, the script is not started synchronously by the return value, and it is run by itself. The monitoring command enters the monitoring mode and will not take the return value. Be sure to set **Start Script** of the exec resource for running monitoring commands to **Asynchronous**.



10. The following window is displayed. Set **0** to **Failover Threshold** of **Recovery Operation at Activation Failure Detection**, and set **Stop Group** to **Final Action**. Click **Next**.



11. In the following window, uncheck **Follow the default dependence** and set the exec resource (exec 1) created in Step 2 to **Dependent Resources**.

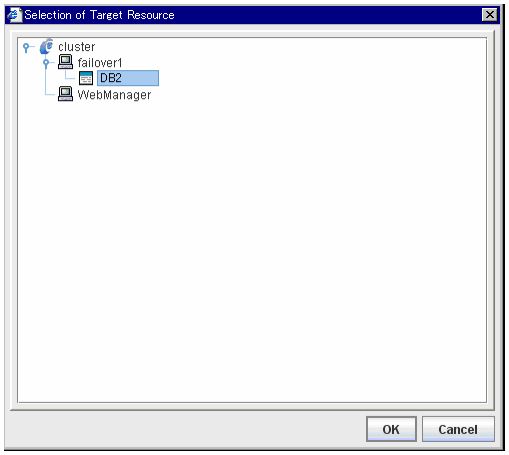


Click the **Finish** button. You have completed adding the exec resource (exec 2) for starting the monitoring command. Move to Step 5.

Step 5 Adding pid monitor resource

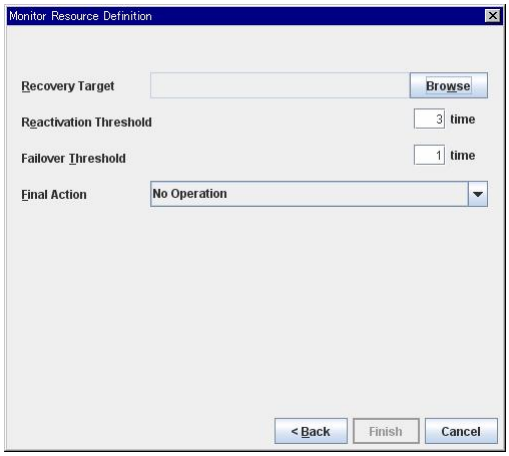
Add the pid monitor resource for monitoring the exec 2 that has been added in Step 4.

1. If the Builder has not been started yet, start it.
   (default path: C:\Program Files\CLUSTERPRO\clpbuilder-l\clptrek.html)
2. Right-click **Monitors** on the tree view, and click **Add Monitor Resource**.
3. The **Monitor Resource Definition** dialog box is displayed. Select the monitor resource type (pid monitor) and enter a monitor resource name in the **Name** box. Click **Next**.
4. Click **Browse** in the window displayed. The **Selection of Target Resource** window is displayed.



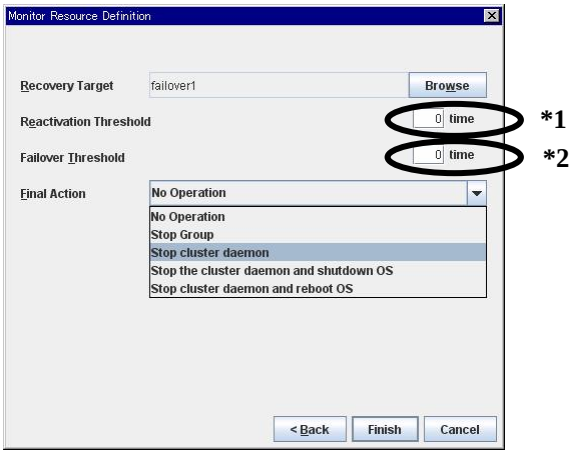
Note:
In the **Selection of Target Resource** window, only the exec resource set to **Asynchronous** in **Exec Resource Tuning Properties** is displayed as selectable target. If nothing can be selected, open the setting window for target exec resource and set the exec resource to **Asynchronous**.

5. Specify the exec resource (exec 2) that starts the monitoring command and click **OK.**
6. Click **Next** to move to the next window and then click **Browse** of **Recovery Target**.



7. The **Selection of Recovery Target** dialog box is displayed. Select the failover group in which the target monitoring application exists.

8. Select an action to be taken for **Final Action**. This determines the operation of the ExpressCluster X when the monitoring command detects an error in the monitored application.



9. When you finish configuring the settings, click **Finish**. You have finished creating the monitor resource. Move to Step 6.

Failover soon after detecting an error

If you want to execute failover when an error is detected, set **0** to **Reactivation Threshold**. If you want to reactivate the group when an error is detected, specify a value other than **0** to **Reactivation Threshold**.

When you set a value other than 0 to **Reactivation Threshold**, you need to set a value longer than the time below to **Wait Time to Start Monitoring**.

From when a monitoring command is started to when it is terminated due to an error:

The time to terminate the monitoring command is determined by the values of -i parameter (monitoring interval), -c parameter (retry time) and -r parameter (response wait time).

Example: When the monitoring command is terminated because an error is detected in the database server:

- -i (monitoring interval)
- : 60 (sec.)
- -c (retry time)
- : 2 (times)
- -r (response wait time)
- : 120 (sec.)

Approximate time to finish the monitoring command:

60(sec.) × 2(times) + monitoring period = 120 + α (sec.)

* Monitoring interval × retry count + monitoring period

Depending on the type of error, the time to end the monitoring command slightly differs. For the monitoring command operation depending on the error type, see “Alert messages” on page 49.

Executing failover without stopping a server

If you want to perform failover without stopping a server as a cluster when an error is detected, set **1** to **Failover Threshold**. In this case, other failover groups are operated continuously on the server that detected the error. When an error is also detected in the failover destination server, a failover occurs again and the failover group returns. If you want to perform failover by stopping a server as a cluster when an error is detected, set **0** to **Failover Threshold**.

In addition, specify either **Stop cluster daemon**, **Stop the cluster daemon and shut down OS,** or **Stop cluster daemon and reboot OS** to **Final Action**.

In this case, other failover group is terminated or a failover occurs because the server that detected an error will not be operated as a cluster.

Considering that the recovery will be done when an error occurs, it is recommended to specify as follows:

Failover Threshold 0

Final Action Stop cluster daemon

Step 6 Verifying the settings for monitoring command

In Step 2, you have added the exec resource for starting target monitoring application to the failover group created in Step 1. In Step 3, you have checked that the target monitoring application runs normally.

Then in Step 4, you have added the exec resource for starting monitoring commands. Finally, in Step 5, you have added the monitor resource for monitoring the exec resource for starting monitoring commands.

In Step 6, after updating a failover group, check that monitoring commands run normally by performing the following steps in the same way as you checked the operation of the target monitoring application.

Note:

You need to reflect the cluster configuration data edited in Steps 4 and 5 to the cluster system. For how to reflect the data, see "

Saving the cluster configuration data on a floppy disk (Windows)," "Creating a cluster using the data in a floppy disk" in Step 3.

Execute the following operation from the WebManager, and check that the target monitoring application runs normally.

Start the WebManager (http://*IP_address_for_the WebManager_group*: *port_number* (default value: 29003)), and execute the following four verification steps.

(Verification 1) Starting up a group

1. Right-click the icon of the failover group that you want to start in the WebManager tree view, and click **Start**.
2. Check that the group icon has turned green.
3. Check that the messages of the Event ID1 and ID2 of the monitoring command are displayed in the WebManager alert view.

Note:
If a monitoring command displays an error message in the ExpressCluster WebManager when the group is started, a parameter value of the monitoring command may not be appropriate. If an error message is displayed on a particular server, there may be an error in the environment settings of the target monitoring application.

(Verification 2) Stopping a group

1. In the WebManager tree view, right-click the icon of the failover group you want to stop, and click **Stop**.
2. Check that the group icon has turned gray.
3. Check that the message of the Event ID 3 of the monitoring command is displayed in the WebManager alert view.

Note:
Check that the group has been started and/or stopped among all the servers that start the failover group.

(Verification 3) Moving a group

Move a failover group among servers. Check that the started group properly moves by moving the group to other servers in order.

1. In the WebManager tree view, right-click the icon of the failover group that you want to move, and click **Move**.
2. Click the group icon, and check that the group has been started on the destination server in the table view.
3. Check that the monitoring command message is displayed in the WebManager alert view.

   Server name: *Name_of_the_server_from_which_the_group_is_moved*

   Event ID: 3

   Server name: *Name_of_the_server_to_which_the_group_is_moved*

   Event ID: 1, 2

Note:
Depending on the application startup/stop process described in the script, it may take a few minutes to complete moving a group. See the tree view on the main screen to check that the group has been moved.

(Verification 4) Failing over a group

Shut down the server on which the failover group is running, and verify that the group fails over to the destination server.

1. In the WebManager tree view, right-click the server in which the group that you want to fail over is running, and click **Reboot OS**.
2. Check that the server is restarted and the failover group fails over to the destination server.
3. Verify that the message of the monitoring command is displayed in the WebManager alert view.

   Server name: *Name_of_the_server_from_which_the_failover_group_fails_over*

   Event ID: 3

   Server Name: *Name_of_the_server_to_which_the_failover_group_fails_over*

   Event ID: 1, 2

Note:
If you set an extremely small value such as 1 (one) to the monitoring interval and response time parameters, you may not be able to perform monitoring normally. Check the operation carefully beforehand.

Related Information:
For more information on messages, see “Alert messages” on page 49.

You have successfully completed the settings for ExpressCluster X Database Agent.

Note:
Run the following commands to confirm the group failover from the command line.
(Verification 1) clpgrp -s group_name
(Verification 2) clpgrp -t group_name
(Verification 3) clpgrp -m group_name
Run the clpstat command to check the cluster status.

Appendix A. Script templates

Setting up script templates

ExpressCluster provides various script templates to monitor applications. Write the necessary information into templates to monitor applications effectively.

The script templates can be used from the ExpressCluster X Builder. There are Windows version and Linux version for the Builder and the templates are prepared for each version. Choose and set up the script templates according to your environment.

Obtain the latest version of the script templates and follow the instruction that comes with the product to update.

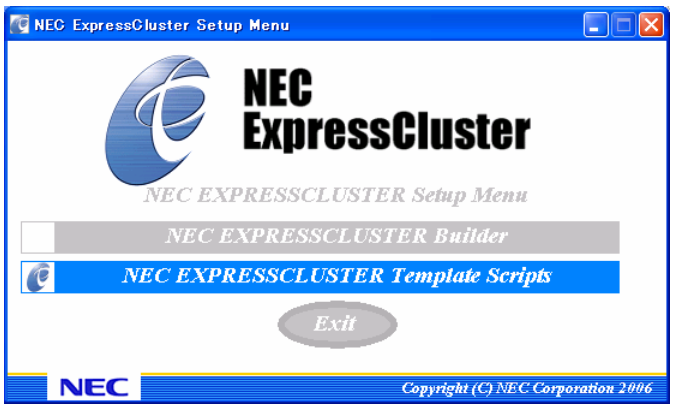
Installing the script templates on Windows machine

Log on as administrator when installing the script templates on Windows machine.

Follow the procedures below:

1. Insert the ExpressCluster X CD into a CD drive.
   The Setup Menu **window is displayed automatically. If the window is not displayed, execute the menu.exe on the CD drive directly.**
2. Click **ExpressCluster for Linux**.

3. Click **ExpressCluster Template Scripts**.



4. A dialog box to verify where to install the template scripts is displayed. Check the path and click **Extract**.

Installing script templates on Linux machine

Set up the script templates after the Builder for the Linux version has been set up. The script templates are installed in a folder having the scripts for the Builder. Refer to Chapter 3, "Creating the cluster configuration data using the Builder" in the *Installation and Configuration Guide* for how to install the Builder.

Log on as root user when installing the script templates on Linux machine.

Follow the procedures below:

1. Insert the ExpressCluster X CD into a CD drive.
2. Mount the CD.

   ```
   # mount /dev/cdrom
   ```

3. Move to the following directory:

   ```
   # cd /mnt/cdrom/Linux/1.0/en/script/Linux
   ```

4. Install the script templates using the rpm command.

   ```
   # rpm –i expresscls-script-1.0.0-1.linux.i686.rpm
   ```

   Note:
   The rpm file name may vary depending on the version of the Agent. The command description may also vary depending on the type of Linux.

Uninstalling the script templates

Uninstalling the script templates for Windows

To uninstall the script templates, log on as an administrator and delete all folders except defaultl and defaultw under C:\Program Files\CLUSTERPRO\clpbuilder-l\scripts\linux.

Note:
Check the directory because the underlined part may vary depending on the terminal environment.

Uninstalling the script templates for Linux

To uninstall the script templates, run the following command as root user:

```
# rpm -e expresscls-script
```

If you uninstall the Builder for Linux, the script templates (the Builder for Linux) will be uninstalled as well, but the rpm module information will be remained. Therefore, it is necessary to forcefully delete the package using the rpm command:

(Example): `rpm –e --force expresscls-script`

Details on script templates

What needs to be corrected is written as comments in a script template. Refer to them to make corrections. Comments are written in Japanese in the Windows version template and in English in the Linux version template.

Two templates, start.sh and stop.sh, are provided for the scripts for DB2. Modify according to the system requirements and use them. Modifications to the scripts need to be made are underlined boldface italics.

For DB2

start.sh

```
#! /bin/sh
#***************************************
#*            start.sh                 *
#***************************************

#! /bin/sh
#***************************************
#*            start.sh                 *
#***************************************

if [ "$CLP_EVENT" = "START" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
  echo "NORMAL1"
  if [ "$CLP_SERVER" = "HOME" ]
  then
    echo "NORMAL2"
  else
    echo "ON_OTHER1"
  fi
  date +"%Y/%m/%d %T"
  echo "DB2 start"
#
# Correct the instance name to an appropriate value.
#
  su - db2inst1 -c "DB2INSTANCE=db2inst1;db2start`"
else
  echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
  echo "FAILOVER1"
  if [ "$CLP_SERVER" = "HOME" ]
  then
    echo "FAILOVER2"
  else
    echo "ON_OTHER2"
  fi
  date +"%Y/%m/%d %T"
  echo "DB2 start"
#
# Correct the instance name to an appropriate value.
#
  su - db2inst1 -c "DB2INSTANCE=db2inst1;db2start`"
else
  echo "ERROR_DISK from FAILOVER"
```

```
fi
else
echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

stop.sh

```
#! /bin/sh
#****************************************
#*             stop.sh                  *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "NORMAL1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "NORMAL2"
   else
     echo "ON_OTHER1"
   fi
   date +"%Y/%m/%d %T"
   echo "DB2 stop"
#
# Correct the instance name to an appropriate value.
#
   su - db2inst1 -c "db2stop force"

else
   echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "FAILOVER2"
   else
     echo "ON_OTHER2"
   fi
   date +"%Y/%m/%d %T"
   echo "DB2 stop"
#
# Correct the instance name to an appropriate value.
#
   su - db2inst1 -c "db2stop force"

else
   echo "ERROR_DISK from FAILOVER"
fi
else
echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

For DB2 monitoring

start.sh

```
#! /bin/sh
#***************************************
#*            start.sh                 *
#***************************************

if [ "$CLP_EVENT" = "START" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "NORMAL1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "NORMAL2"
   else
     echo "ON_OTHER1"
   fi
   date +"%Y/%m/%d %T"
   echo "clp_db2mon start"
#
# Specify according to the code page of the database.
#
   export LANG=ja_JP.eucJP
#
# Correct the home name of instance to an appropriate value.
#
   source /home/db2inst1/sqllib/db2profile
#
# Correct the instance name or database name to an appropriate value.
#
   clp_db2mon db2watch -d XXXX -m db2inst1
   else
   echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "FAILOVER2"
   else
     echo "ON_OTHER2"
   fi
   date +"%Y/%m/%d %T"
   echo "clp_db2mon start"
#
# Specify according to the code page of the database.
#
   export LANG=ja_JP.eucJP
#
# Correct the home name of instance to an appropriate value.
#
   source /home/db2inst1/sqllib/db2profile
#
# Correct the instance name or database name to an appropriate value.
#
   clp_db2mon db2watch -d XXXX -m db2inst1
   else
   echo "ERROR_DISK from FAILOVER"
fi
```

```
else
echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

stop.sh

```
#! /bin/sh
#***************************************
#*             stop.sh                 *
#***************************************

if [ "$CLP_EVENT" = "START" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "NORMAL1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "NORMAL2"
   else
     echo "ON_OTHER1"
   fi
   date +"%Y/%m/%d %T"
   echo "clp_db2mon stop"
#
# Correct the home name of instance to an appropriate value.
#
   source /home/db2inst1/sqllib/db2profile
   clp_db2mon db2watch --stop
else
   echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "FAILOVER2"
   else
     echo "ON_OTHER2"
   fi
   date +"%Y/%m/%d %T"
   echo "clp_db2mon stop"
#
# Correct the home name of instance to an appropriate value.
#
   source /home/db2inst1/sqllib/db2profile
   clp_db2mon db2watch --stop
else
   echo "ERROR_DISK from FAILOVER"
fi
else
echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

For Oracle10g

start.sh

```
#! /bin/sh
#****************************************
#*            start.sh                  *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
  echo "NORMAL1"
  if [ "$CLP_SERVER" = "HOME" ]
  then
    echo "NORMAL2"
  else
    echo "ON_OTHER1"
  fi
  date +"%Y/%m/%d %T"
  echo "Oracle10g start"
#
# Correct the user name and the listener name to appropriate values.
#
  su - oracle -c "lsnrctl start LISTENER"
#
# Correct the user name, the SID name and the full path name of the
start script to appropriate values.
#
  su - oracle -c "export ORACLE_SID=orcl;sqlplus /nolog @/XXXX/startup.sql`"
else
  echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
  echo "FAILOVER1"
  if [ "$CLP_SERVER" = "HOME" ]
  then
    echo "FAILOVER2"
  else
    echo "ON_OTHER2"
  fi
  date +"%Y/%m/%d %T"
  echo "Oracle10g start"
#
# Correct the user name and the listener name to appropriate values.
#
  su - oracle -c "lsnrctl start LISTENER"
#
# Correct the user name, the SID name and the full path name of the
start script to appropriate values.
#
  su - oracle -c "export ORACLE_SID=orcl;sqlplus /nolog @/XXXX/startup.sql`"
else
  echo "ERROR_DISK from FAILOVER"
fi
else
echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

stop.sh

```
#! /bin/sh
#****************************************
#*             stop.sh                  *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "NORMAL1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "NORMAL2"
   else
     echo "ON_OTHER1"
   fi
   date +"%Y/%m/%d %T"
   echo "Oracle10g stop"
#
# Correct the user name, the SID name and the full path name of the
end script to appropriate values
#
   su - oracle -c "export ORACLE_SID=orcl;sqlplus /nolog @/XXXX/shutdown.sql"
#
# Correct the user name and the listener name to appropriate values.
#
   su - oracle -c "lsnrctl stop LISTENER`"
else
   echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "FAILOVER2"
   else
     echo "ON_OTHER2"
   fi
   date +"%Y/%m/%d %T"
   echo "Oracle10g stop"
#
# Correct the user name, the SID name and the full path name of the
end script to appropriate values.
#
   su - oracle -c "export ORACLE_SID=orcl;sqlplus /nolog @/XXXX/shutdown.sql"
#
# Correct the user name and the listener name to appropriate values.
#
   su - oracle -c "lsnrctl stop LISTENER`"
else
   echo "ERROR_DISK from FAILOVER"
fi
else
echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

startup.sql/shutdown.sql

To start Oracle10g, both “startup.sql” and “shutdown.sql” files are necessary to use sqlplus. You can assign any name for the path and file, but the name needs to be same as the full path specified by “start.sh/stop.sh.”

Write the following in each file. A template is not available.

startup.sql

```
connect / as sysdba
startup pfile=Initialization file name
exit
```

shutdown.sql

```
connect / as sysdba
shutdown immediate
exit
```

For Oracle10g monitoring

start.sh

```
#! /bin/sh
#***************************************
#*            start.sh                 *
#***************************************

if [ "$CLP_EVENT" = "START" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "NORMAL1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "NORMAL2"
   else
     echo "ON_OTHER1"
   fi
   date +"%Y/%m/%d %T"
   echo "clp_oramon start"
#
# Specify ORACLE_HOME to an appropriate value.
#
   export ORACLE_HOME=/opt/oracle/product/10.2.0/db_1
   export LD_LIBRARY_PATH=$ORACLE_HOME/lib
#
# Specify NLS_LANG parameter to an appropriate value.
#
#    export NLS_LANG=AMERICAN_AMERICA.JA16EUC
#
# Correct the database name to an appropriate value.
#
   clp_ora10mon orawatch -d XXXX
   else
   echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "FAILOVER2"
   else
     echo "ON_OTHER2"
   fi
   date +"%Y/%m/%d %T"
   echo "clp_oramon start"
#
# Specify ORACLE_HOME to an appropriate value.
#
   export ORACLE_HOME=/opt/oracle/product/10.2.0/db_1
   export LD_LIBRARY_PATH=$ORACLE_HOME/lib
#
# Specify NLS_LANG parameter to an appropriate value.
#
#    export NLS_LANG=AMERICAN_AMERICA.JA16EUC
#
# Correct the database name to an appropriate value.
#
   clp_ora10mon orawatch -d XXXX
else
```

```
  echo "ERROR_DISK from FAILOVER"
fi
else
echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

stop.sh

```
#! /bin/sh
#***************************************
#*             stop.sh                 *
#***************************************

if [ "$CLP_EVENT" = "START" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "NORMAL1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "NORMAL2"
   else
     echo "ON_OTHER1"
   fi
#
# Specify ORACLE_HOME to an appropriate value.
#
   export ORACLE_HOME=/opt/oracle/product/10.2.0/db_1
   export LD_LIBRARY_PATH=$ORACLE_HOME/lib

   date +"%Y/%m/%d %T"
   echo "clp_oramon stop"
   clp_ora10mon orawatch --stop
else
   echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "FAILOVER2"
   else
     echo "ON_OTHER2"
   fi
#
# Specify ORACLE_HOME to an appropriate value.

#
   export ORACLE_HOME=/opt/oracle/product/10.2.0/db_1
   export LD_LIBRARY_PATH=$ORACLE_HOME/lib

   date +"%Y/%m/%d %T"
   echo "clp_oramon stop"
   clp_ora10mon orawatch --stop
else
   echo "ERROR_DISK from FAILOVER"
fi
else
echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

For PostgreSQL

Two templates, start.sh and stop.sh, are provided for the scripts for PostgreSQL.

Modify according to the system requirements and use them. Modifications to the scripts need to be made are underlined boldface italics.

In case of starting more than two PostgreSQLs at one server, for example multi-directional stand-by, set the identifier, the database domain directory, and the port number not to overlap with others.

start.sh

```
#! /bin/sh
#****************************************
#*            start.sh                  *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "NORMAL1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
      echo "NORMAL2"
   else
      echo "ON_OTHER1"
   fi
   date +"%Y/%m/%d %T"
   echo "PostgreSQL start"
#
# Correct the PostgreSQL user name, the database domain directory,
and the port number to appropriate values.
#
   su - postgres -c "pg_ctl start -D /mnt/pgsql/data -l /dev/null -o
'-i -p 5432'"
else
   echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
      echo "FAILOVER2"
   else
      echo "ON_OTHER2"
   fi
   date +"%Y/%m/%d %T"
   echo "PostgreSQL start"
#
# Correct the PostgreSQL user name, the database domain directory,
and the port number to appropriate values.
#
   su - postgres -c "pg_ctl start -D /mnt/pgsql/data -l /dev/null -o
'-i -p 5432'"
else
   echo "ERROR_DISK from FAILOVER"
fi
else
```

```
echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

stop.sh

```
#! /bin/sh
#****************************************
#*             stop.sh                  *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "NORMAL1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "NORMAL2"
   else
     echo "ON_OTHER1"
   fi
   date +"%Y/%m/%d %T"
   echo "PostgreSQL stop"
#
# Correct the PostgreSQL user name and the database domain directory
to appropriate values.
#
   su - postgres -c 'pg_ctl stop -D /mnt/pgsql/data -m fast'
else
   echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "FAILOVER2"
   else
     echo "ON_OTHER2"
   fi
   date +"%Y/%m/%d %T"
   echo "PostgreSQL stop"
#
# Correct the PostgreSQL user name and the database domain directory
to appropriate values.
#
   su - postgres -c 'pg_ctl stop -D /mnt/pgsql/data -m fast'
else
   echo "ERROR_DISK from FAILOVER"
fi
else
echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

For PostgreSQL monitoring

Two templates, start.sh and stop.sh, are provided for the scripts for PostgreSQL. Modify according to the system requirements and use them. Modifications to the scripts need to be made are underlined boldface italics.

start.sh

```
#! /bin/sh
#****************************************
#*            start.sh                  *
#****************************************

if [ "$CLP_EVENT" = "START" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "NORMAL1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "NORMAL2"
   else
     echo "ON_OTHER1"
   fi
   date +"%Y/%m/%d %T"
   echo "clp_psqlmon start"
#
# Correct the path of the PostgreSQL library.
#
#    export LD_LIBRARY_PATH=/usr/local/pgsql/lib
#
# Correct the database name and the port number to appropriate values.
#
   clp_psql81mon psqlwatch -d XXXX -n 5432
else
   echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "FAILOVER2"
   else
     echo "ON_OTHER2"
   fi
   date +"%Y/%m/%d %T"
   echo "clp_psqlmon start"
#
# Correct the path of the PostgreSQL library.
#
#    export LD_LIBRARY_PATH=/usr/local/pgsql/lib
#
# Correct the database name and the port number to appropriate values.
#
   clp_psql81mon psqlwatch -d XXXX -n 5432
else
   echo "ERROR_DISK from FAILOVER"
fi
else
echo "NO_CLP"
```

```
fi
echo "EXIT"
exit 0
```

stop.sh

```
#! /bin/sh
#***************************************
#*             stop.sh                 *
#***************************************

if [ "$CLP_EVENT" = "START" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
  echo "NORMAL1"
  if [ "$CLP_SERVER" = "HOME" ]
  then
    echo "NORMAL2"
  else
    echo "ON_OTHER1"
  fi
  date +"%Y/%m/%d %T"
  echo "clp_psqlmon stop"
#
# Correct the path of the PostgreSQL library.
#
#    export LD_LIBRARY_PATH=/usr/local/pgsql/lib

  clp_psql81mon psqlwatch --stop
else
  echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
  echo "FAILOVER1"
  if [ "$CLP_SERVER" = "HOME" ]
  then
    echo "FAILOVER2"
  else
    echo "ON_OTHER2"
  fi
  date +"%Y/%m/%d %T"
  echo "clp_psqlmon stop"
#
# Correct the path of the PostgreSQL library.
#
#    export LD_LIBRARY_PATH=/usr/local/pgsql/lib

  clp_psql81mon psqlwatch --stop
else
  echo "ERROR_DISK from FAILOVER"
fi
else
echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

For MySQL

Two templates, start.sh and stop.sh, are provided for the scripts for MySQL. Modify according to the system requirements and use them. Modifications to the scripts need to be made are underlined boldface italics.

start.sh

```
#! /bin/sh
#***************************************
#*             start.sh                *
#***************************************

if [ "$CLP_EVENT" = "START" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "NORMAL1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "NORMAL2"
   else
     echo "ON_OTHER1"
   fi
   date +"%Y/%m/%d %T"
   echo "MySQL start"
#
# Correct the start option definition file to an appropriate value.
# To register this file as a script file of the ExpressCluster X, specify
the group name of the path.
#
   mysqld_safe
--defaults-file=/opt/nec/clusterpro/scripts/group/groupname/my.cn
f &
else
   echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "FAILOVER2"
   else
     echo "ON_OTHER2"
   fi
   date +"%Y/%m/%d %T"
   echo "MySQL start"
#
# Correct the start option definition file to an appropriate value.
# To register this file as a script file of the ExpressCluster X, specify
the group name of the path.
#
   mysqld_safe
--defaults-file=/opt/nec/clusterpro/scripts/group/groupname/my.cn
f &
else
   echo "ERROR_DISK from FAILOVER"
fi
else
```

```
echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

stop.sh

```
#! /bin/sh
#***************************************
#*             stop.sh                 *
#***************************************

if [ "$CLP_EVENT" = "START" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "NORMAL1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "NORMAL2"
   else
     echo "ON_OTHER1"
   fi
   date +"%Y/%m/%d %T"
   echo "MySQL stop"
#
# Correct the start option definition file to an appropriate value.
# To register this file as a script file of the ExpressCluster X, specify
the group name of the path.
#
   mysqladmin
--defaults-file=/opt/nec/clusterpro/scripts/group/groupname/my.cn
f shutdown
else
   echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "FAILOVER2"
   else
     echo "ON_OTHER2"
   fi
   date +"%Y/%m/%d %T"
   echo "MySQL stop"
#
# Correct the start option definition file to an appropriate value.
# To register this file as a script file of the ExpressCluster X, specify
the group name of the path.
#
   mysqladmin
--defaults-file=/opt/nec/clusterpro/scripts/group/groupname/my.cn
f shutdown
else
   echo "ERROR_DISK from FAILOVER"
fi
else
echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

my.cnf

When starting MySQL, creating a startup option definition file in advance makes it easier to specify a parameter. You can assign any name for the path and file, but the name needs to be same as the full path specified by “start.sh/stop.sh.” The file can be registered as a script of a failover group or it can be created on each server in addition to a script file.

* **In case of registering this file as a script, do not correct the content directly on each server. Make a correction only from the ExpressCluster X Builder.**

```
[client]
port            = 3306
socket      = /var/lib/mysql/mysql.sock

[mysqld]
port            = 3306
socket      = /var/lib/mysql/mysql.sock
```

Specify a database file storage directory in datadir. Usually, specify a directory on the shared disk.

In case of multi-directional stand-by, it is required to set different values to each failover group on all the items above.

Example

“my.cnf” for failover group 1

```
[client]
port            = 3306
socket      = /var/lib/mysql/mysql1.sock
[mysqld]
port            = 3306
socket      = /var/lib/mysql/mysql1.sock
pid-file        = /var/lib/mysql/mysql1.pid
datadir         = /mnt/mysql1/
```

“my.cnf” for failover group 2

```
[client]
port            = 3307
socket      = /var/lib/mysql/mysql2.sock
[mysqld]
port            = 3307
socket      = /var/lib/mysql/mysql2.sock
pid-file        = /var/lib/mysql/mysql2.pid
datadir         = /mnt/mysql2/
```

For MySQL monitoring

Two templates, start.sh and stop.sh, are provided for the scripts for MySQL. Modify according to the system requirements and use them. Modifications to the scripts need to be made are underlined boldface italics.

start.sh

```
#! /bin/sh
#***************************************
#*             start.sh                *
#***************************************

if [ "$CLP_EVENT" = "START" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "NORMAL1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "NORMAL2"
   else
     echo "ON_OTHER1"
   fi
   date +"%Y/%m/%d %T"
   echo "clp_mysqlmon start"
#
# In case of not specifying -a parameter, correct the file name for
socket to an appropriate value.

#
#    export MYSQL_UNIX_PORT=/var/lib/mysql/mysql.sock
#
# Correct the database name and the IP address to appropriate values.
#
   clp_mysql150mon mysqlwatch -d XXXX -a nnn.nnn.nnn.nnn
else
   echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "FAILOVER2"
   else
     echo "ON_OTHER2"
   fi
   date +"%Y/%m/%d %T"
   echo "clp_mysqlmon start"
#
# In -a parameter is not specified, correct the file name for socket
to an appropriate value.
#
#    export MYSQL_UNIX_PORT=/var/lib/mysql/mysql.sock
#
# Correct the database name and the IP address to appropriate values.
#
   clp_mysql150mon mysqlwatch -d XXXX -a nnn.nnn.nnn.nnn
else
   echo "ERROR_DISK from FAILOVER"
fi
```

```
else
echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

stop.sh

```
#! /bin/sh
#***************************************
#*             stop.sh                 *
#***************************************

if [ "$CLP_EVENT" = "START" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "NORMAL1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "NORMAL2"
   else
     echo "ON_OTHER1"
   fi
   date +"%Y/%m/%d %T"
   echo "clp_mysqlmon stop"
   clp_mysql50mon mysqlwatch --stop
else
   echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "FAILOVER2"
   else
     echo "ON_OTHER2"
   fi
   date +"%Y/%m/%d %T"
   echo "clp_mysqlmon stop"
   clp_mysql50mon mysqlwatch --stop
else
   echo "ERROR_DISK from FAILOVER"
fi
else
echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

For Sybase Adaptive Server Enterprise (ASE)

Two templates, start.sh and stop.sh, are provided for the scripts for ASE. Modify according to the system requirements and use them. Modifications to the scripts need to be made are underlined boldface italics.

start.sh

```
#! /bin/sh
#***************************************
#*   start.sh                          *
#***************************************
if [ "$CLP_EVENT" = "START" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "NORMAL1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "NORMAL2"
   else
     echo "ON_OTHER1"
   fi

   date +"%Y/%m/%d  %T"
   echo "Sybase ASE start"

#
# Correct the path of the ASE start file.
# Correct the Sybase installation path and the DB-Library path
# to appropriate values.
#
   run_script=/opt/sybase/ASE-12_5/install/RUN_SERVER
   export LD_LIBRARY_PATH=/opt/sybase/OCS-12_5/lib
   export SYBASE=/opt/sybase

#
# Correct the account for Sybase and the installation path of Sybase.
#
   su - sybase -c "export SYBASE=/opt/sybase; $run_script"  >
/dev/null 2>&1 &
   sleep 10

else
   echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "FAILOVER2"
   else
     echo "ON_OTHER2"
   fi

   date +"%Y/%m/%d  %T"
   echo "Sybase ASE start"

#
# Correct the path of the ASE start file.
# Correct the Sybase installation path and the DB-Library path
```

```
# to appropriate values.
#
   run_script=/opt/sybase/ASE-12_5/install/RUN_SERVER
   export LD_LIBRARY_PATH=/opt/sybase/OCS-12_5/lib
   export SYBASE=/opt/sybase

#
# Correct the account for Sybase and the installation path of Sybase.
#
   su - sybase -c "export SYBASE=/opt/sybase; $run_script" > /dev/null
2>&1 &
   sleep 10

else
   echo "ERROR_DISK from FAILOVER"
fi
else
echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

stop.sh

```
#! /bin/sh
#***************************************
#*    stop.sh                          *
#***************************************
if [ "$CLP_EVENT" = "START" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "NORMAL1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "NORMAL2"
   else
     echo "ON_OTHER1"
   fi

   date +"%Y/%m/%d  %T"
   echo "Sybase ASE stop"
   #
   # Correct the path of the ISQL command.
   #
   ISQL=/opt/sybase/OCS-12_5/bin/isql

   #
   # Correct the path of the shutdown script.
   #
   shutdown=/home/sybase/shutdown
   sleep 10

   #
   # Correct the ASE server name and the user name/password to appropriate
values.

   # Correct the account for Sybase and the installation path of Sybase.
   #
   su - sybase -c "export SYBASE=/opt/sybase; $ISQL -S SERVER -U sa
-P -i $shutdown"
   sleep 10

else
   echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "FAILOVER2"
   else
     echo "ON_OTHER2"
   fi

   date +"%Y/%m/%d  %T"
   echo "Sybase ASE stop"
   #
   # Correct the path of the ISQL command
   #
   ISQL=/opt/sybase/OCS-12_5/bin/isql

   #
   # Correct the path of the shutdown script.
```

```
   #
   shutdown=/home/sybase/shutdown
   sleep 10

   #
   # Correct the ASE server name and the user name/password to appropriate
values.

   # Correct the account for Sybase and the installation path of Sybase.
   #
   su - sybase -c "export SYBASE=/opt/sybase; $ISQL -S SERVER -U sa
-P -i $shutdown"
   sleep 10

else
   echo "ERROR_DISK from FAILOVER"
fi
else
echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

Start file and shutdown file

When ASE is started, a start file created during the installation can be used (in the example above, ***RUN_SERVER***). You can connect to the database server and execute the shutdown file when ASE is stopped. You can assign any name for the path and file, but the name needs to be same as the full path specified by “start.sh/stop.sh.”

Write the following in each file. A template is not available.

shutdown

shutdown with nowait

go

For ASE monitoring

Two script templates, start.sh and stop.sh, are provided for connecting/disconnecting a database and for monitoring ASE.

Modify according to the system requirements and use them. Modifications to the scripts need to be made are underlined boldface italics.

start.sh

```
#!/bin/sh
#***************************************
#*   start.sh                          *
#***************************************

#
# Correct the Sybase installation path and the DB-Library path
# to appropriate values.
#
export SYBASE=/opt/sybase
export LD_LIBRARY_PATH=/opt/sybase/OCS-12_5/lib

if [ "$CLP_EVENT" = "START" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "NORMAL1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
      echo "NORMAL2"
   else
      echo "ON_OTHER1"
   fi
   #
   # Correct the database server name and the database name to appropriate
values.
   #
   clp_sybmon sybwatch -d XXXX -s SERVER -u user

else
   echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
      echo "FAILOVER2"
   else
      echo "ON_OTHER2"
   fi
   #
   # Correct the database server name and the database name to appropriate
values.
   #
   clp_sybmon sybwatch -d XXXX -s SERVER -u user

else
   echo "ERROR_DISK from FAILOVER"
fi
else
```

```
echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

stop.sh

```
#! /bin/sh
#***************************************
#*    stop.sh                          *
#***************************************

#
# Correct the DB-Library path to an appropriate value.
#
export LD_LIBRARY_PATH=/opt/sybase/OCS-12_5/lib

if [ "$CLP_EVENT" = "START" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "NORMAL1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "NORMAL2"
   else
     echo "ON_OTHER1"
   fi
   #
   # correct the watch id
   #
   clp_sybmon sybwatch --stop

else
   echo "ERROR_DISK from START"
fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
if [ "$CLP_DISK" = "SUCCESS" ]
then
   echo "FAILOVER1"
   if [ "$CLP_SERVER" = "HOME" ]
   then
     echo "FAILOVER2"
   else
     echo "ON_OTHER2"
   fi
   #
   # correct the watch id
   #
   clp_sybmon sybwatch --stop

else
   echo "ERROR_DISK from FAILOVER"
fi
else
echo "NO_CLP"
fi
echo "EXIT"
exit 0
```

Appendix B. Glossary

Cluster partition	A partition on a mirror disk. Used for managing mirror disks. (Related term: Disk heartbeat partition)
Interconnect	A dedicated communication path for server-to-server communication in a cluster. (Related terms: Private LAN, Public LAN)
Virtual IP address	IP address used to configure a remote cluster.
Management client	Any machine that uses the WebManager to access and manage a cluster system.
Startup attribute	A failover group attribute that determines whether a failover group should be started up automatically or manually when a cluster is started.
Shared disk	A disk that multiple servers can access.
Shared disk type cluster	A cluster system that uses one or more shared disks.
Switchable partition	A disk partition connected to multiple computers and is switchable among computers. (Related terms: Disk heartbeat partition)
Cluster system	Multiple computers are connected via a LAN (or other network) and behave as if it were a single system.
Cluster shutdown	To shut down an entire cluster system (all servers that configure a cluster system).
Active server	A server that is running for an application set. (Related term: Standby server)
Secondary server	A destination server where a failover group fails over to during normal operations. (Related term: Primary server)
Standby server	A server that is not an active server. (Related term: Active server)
Disk heartbeat partition	A partition used for heartbeat communication in a shared disk type cluster.
Data partition	A local disk that can be used as a shared disk for switchable partition. Data partition for mirror disks. (Related term: Cluster partition)
Network partition	All heartbeat is lost and the network between servers is partitioned. (Related terms: Interconnect, Heartbeat)

Node	A server that is part of a cluster in a cluster system. In networking terminology, it refers to devices, including computers and routers, that can transmit, receive, or process signals.
Heartbeat	Signals that servers in a cluster send to each other to detect a failure in a cluster. (Related terms: Interconnect, Network partition)
Public LAN	A communication channel between clients and servers. (Related terms: Interconnect, Private LAN)
Failover	The process of a standby server taking over the group of resources that the active server previously was handling due to error detection.
Failback	A process of returning an application back to an active server after an application fails over to another server.
Failover group	A group of cluster resources and attributes required to execute an application.
Moving failover group	Moving an application from an active server to a standby server by a user.
Failover policy	A priority list of servers that a group can fail over to.
Private LAN	LAN in which only servers configured in a clustered system are connected. (Related terms: Interconnect, Public LAN)
Primary (server)	A server that is the main server for a failover group. (Related term: Secondary server)
Floating IP address	Clients can transparently switch one server from another when a failover occurs. Any unassigned IP address that has the same network address that a cluster server belongs to can be used as a floating address.
Master server	The server displayed on top of the **Master Server** in **Cluster Properties** in the Builder.
Mirror connect	LAN used for data mirroring in a data mirror type cluster. Mirror connect can be used with primary interconnect.
Mirror disk type cluster	A cluster system that does not use a shared disk. Local disks of the servers are mirrored.

Appendix C. Index

A

adding a group, 63
adding a group resource (disk resource), 64
adding a group resource (floating IP address), 63
adding the exec resource (exec 1), 61, 65
adding the exec resource (exec 2), 61, 71
alert messages, 31, 48, 49

B

before using the Database Agent, 17

C

clp_db2mon, 24
clp_mysql50mon, 37
clp_ora10mon, 28
clp_psql81mon, 32
clp_sybmon, 41
command reference, 21
confirmation test for startup, 61, 67
creating a failover group, 63

D

Database Agent, 13, 14
DB2 V9, 16, 24
details on script templates, 81

E

ExpressCluster, 17

F

failing over a group, 70, 78
for ASE monitoring, 112
for DB2, 81
for DB2 monitoring, 84
for MySQL, 100
for MySQL monitoring, 105
for Oracle10g, 87
for Oracle10g monitoring, 91
for PostgreSQL, 94
for PostgreSQL monitoring, 97
for Sybase Adaptive Server Enterprise(ASE), 108

H

hardware, 17

I

installing script templates on Linux machine, 80
installing the script templates on Windows machine, 79

L

license registration, 13, 17, 18
list of monitoring commands, 22
log collection when an error has occurred, 48

M

mechanism of monitoring, 15
memory space, 17
messages being output by clp_db2mon, 47, 49
messages being output by clp_mysql50mon, 55
messages being output by clp_ora10mon, 51
messages being output by clp_psql81mon, 53
messages being output by clp_sybmon, 57
messages being output when an error is detected in database monitoring, 50, 54, 56, 59
messages being output when error is detected in database monitoring, 52
messages due to setting error, 49, 51, 53, 56, 58
messages due to system error, 51, 53, 55, 57, 59
messages indicating normal operation, 49, 51, 53, 55, 57
monitor information with monitoring commands, 48
monitoring chart, 22
monitoring command, 61, 77
moving a group, 69, 78
my.cnf, 104
MySQL5.0, 37

O

Oracle10g R2, 28
OS, 17

P

pid monitor resource, 74
PostgreSQL 8.1, 32

R

registering the license, 19
registering the license interactively, 18

S

setting up, 61, 62
setting up script templates, 79
shutdown file, 111
shutdown.sql, 90
specifying a license file, 19
start file, 111
start.sh, 81, 84, 87, 91, 94, 97, 100, 105, 108, 112
starting up a group, 69, 77
startup.sql, 90

stop.sh, 83, 86, 89, 93, 96, 99, 102, 107, 110, 114
stopping a group, 69, 77
suspending and resuming monitoring, 23
Sybase 12.5.2, 41
system requirements, 17

T

target monitoring application, 16, 61, 67

U

uninstalling the script templates, 80
uninstalling the script templates for Linux, 80
uninstalling the script templates for Windows, 80
Using the Database Agent, 17

V

verifying the behavior of the target monitoring application, 69
verifying the settings, 61, 77

W

WebManager, 48
writing scripts, 23