FUJITSU Software
Interstage Application Server

Tuning Guide

Windows/Solaris/Linux

B1WS-1084-03ENZ0(00)
April 2014

Preface

Purpose of this Document

This manual is the Interstage Application Server Tuning Guide. This manual explains how to tune Interstage.

Intended Readers

This manual is intended for the engineers who run Interstage.

It is assumed that readers of this manual have a basic knowledge of:

- C
- C++
- COBOL
- OOCOBOL
- Java
- The Internet
- Object-oriented technology
- Distributed object technology (CORBA)
- Relational databases.
- Basic knowledge of the OS used

Structure of this Manual

The structure of this manual is as follows:

Chapter 1 Resource Requirements

This chapter describes the resource requirements of Interstage Application Server.

Chapter 2 Tuning Interstage

This chapter explains how to tune Interstage.

Chapter 3 Tuning J2EE Applications

This chapter describes the tuning of J2EE applications.

Chapter 4 Tuning WorkUnits

This chapter describes the tuning of WorkUnits.

Chapter 5 System Tuning

This chapter describes system tuning.

Chapter 6 JDK/JRE Tuning

This chapter provides the basic knowledge and methods required for tuning Java applications.

Appendix A CORBA Service Environment Definition

This appendix describes the CORBA Service operating environment.

Appendix B Component Transaction Service Environment Definition

This appendix explains the Component Transaction Service environment definition file.

Appendix C Database Linkage Service Environment Definition

This appendix explains the Database Linkage Service environment definition.

Appendix D Event Service Environment Definition

This appendix describes the Event Service operating environment.

Appendix E Interstage HTTP Server Environment Definition

This appendix explains the Web Server (Interstage HTTP Server) environment definition.

Appendix F Environment Definition for Interstage Single Sign-on

This appendix explains the Interstage Single Sign-On environment definition.

Appendix G Defining the Multi Server Management Environment

This appendix explains how to tune the environment definition file for a multi server management.

Appendix H Web Server (Sun Java System Web Server) Environment Definition

This appendix explains how to tune the environment definition file for a Sun Java System Web Server.

Appendix I Setting IPC Resources

This appendix explains the setting of IPC resources.

Appendix J Portable-ORB Environment Definition

This describes the Portable-ORB operating environment set up.

Representation of Platform-specific Information

In the manuals of this product, there are parts containing content that relates to all products that run on the supported platform. In this case, an icon indicating the product platform has been added to these parts if the content varies according to the product. For this reason, refer only to the information that applies to your situation.

Windows32	Indicates that this product (32-bit) is running on Windows.
Windows64	Indicates that this product (64-bit) is running on Windows.
Windows32/64	Indicates that this product (32/64-bit) is running on Windows.
Solaris32	Indicates that this product (32-bit) is running on Solaris.
Solaris64	Indicates that this product (64-bit) is running on Solaris.
Solaris32/64	Indicates that this product (32/64-bit) is running on Solaris.
Linux32	Indicates that this product (32-bit) is running on Linux.
Linux64	Indicates that this product (64-bit) is running on Linux.
Linux32/64	Indicates that this product (32/64-bit) is running on Linux.

Conventions

Abbreviations

Read occurrences of the following Components as their corresponding Service.

Service	Component
CORBA Service	ObjectDirector
Component Transaction Service	TransactionDirector

Export Controls

This document or a portion thereof may not be exported (or re-exported) without authorization from the appropriate government authorities in accordance with the pertinent laws.

Trademarks

Trademarks of other companies are used in this documentation only to identify particular products or systems.

Product Trademarks/Registered Trademarks
Microsoft, Active Directory, ActiveX, Excel, Internet Explorer, MS-DOS, MSDN, Visual Basic, Visual C++, Visual Studio, Windows, Windows NT, Windows Server, Win32 are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.
Oracle and Java are registered trademarks of Oracle and/or its affiliates.

Other company and product names in this documentation are trademarks or registered trademarks of their respective owners.

Copyrights

April 2014 Third Edition
November 2012 First Edition

The contents of this manual may be revised without prior notice.

Contents

Chapter 1 Resource Requirements

This chapter describes the resource requirements of Interstage Application Server.

1.1 Disk Space Requirements

This section describes the disk space requirements for:

- 1.1.1 Using the Server Function of Application Server
- 1.1.2 Using the Multi Server Management Function of Application Server

1.1.1 Using the Server Function of Application Server

The free disk space required for operation using the server function of Application Server is shown in Table 1.1 Disk Space Required for Operation Using the Application Server.

Table 1.1 Disk Space Required for Operation Using the Application Server

No.	Function	Directory	Disk space (MB Units)	Remarks
1	Interstage operating environment	Installation directory for Component Transaction Service \var\td001 (Specify this directory using "TD path for system" in the Interstage operating environment definition file.)	60 or more	When the Interstage operating environment is created
2	Interstage Management Console	Windows32/64 Installation directory for Interstage Management Console \isAdmin\var\download	(*1)	log information
		Solaris32/64 Linux32/64 /var/opt/FJSVisgui/tmp/download		
3	Interstage HTTP Server	Access log ,error log , and trace log directory	Determine the required size from operation details.	Interstage HTTP Server access log , error log, and trace log
		Windows32/64 Interstage HTTP Server installation directory\var\opelog	2	Interstage HTTP Server operation log
		Solaris32/64 Linux32/64 /var/opt/FJSVihs/var/opelog		
		Windows32/64 Interstage HTTP Server installation directory\var\.ihsapi	10	Interstage HTTP Server maintenance log
		Solaris32/64 Linux32/64 /var/opt/FJSVihs/var/.ihsapi		

No.	Function	Directory	Disk space (MB Units)	Remarks
		Contents directory	Determine the required size from operation details.	Contents (e.g., HTML documents)
4	J2EE	J2EE common directory	Determine the required size from operation details.	For setting of J2EE application properties
5	IJServer WorkUnit	Windows32/64 J2EE common directory\ijserver\ IJServer name\ log directory Solaris32/64 Linux32/64 J2EE common directory/ijserver/ IJServer name/ log directory	24 or more (*2)	When an IJServer WorkUnit is used.
		Windows32/64 Interstage install directory\F3FMjs5\logs\jk2 Solaris32/64 Linux32/64 /var/opt/FJSVjs5/logs/jk2	2 or more (*3)	
		Windows32/64 J2EE common directory\ijserver\Session Registry Server(IJServer) name\apps\srs.ear\srs.war\serializedata\sessionrecovery Solaris32/64 Linux32/64 J2EE common directory/ijserver/Session Registry Server(IJServer) name/apps /srs.ear/srs.war/serializedata/sessionrecovery	(*29)	When session serialization is enabled, it must exist in the environment used to run the Session Registry Server.
6	CORBA Service	Windows32/64 Installation directory for CORBA Service	0.1 or more	When the CORBA Service is run. Depends on implementation information, Naming Service, and interface repository data size.
		Windows32/64 Installation directory for CORBA Service\var	42 (maximum size by default) (*5)	log information
			2 or more (*6)	When an internal log is collected (non-preinstalled Java library)
			4.0 or less (*7)	Naming Service runtime trace information (only when the service is used)
			6 or more	When the CORBA Service is run.

No.	Function	Directory	Disk space (MB Units)	Remarks
			4.0 or less (*7)	Naming Service user exception log information
			32.3 or less	Interface Repository log information (only when the service is running)
		Windows32/64 Installation directory for CORBA Service\etc	4.1 or more (*7)	Naming Service information
		Windows32/64 Component Transaction Service installation directory\var\IRDB (Interstage setup command is used)	10.3 or more (*9)	Interface Repository information
		Specified with system property user.dir	(*8)	Log information (preinstalled Java library)
		Solaris32/64 Linux32/64 /var/opt	24 (maximum size by default) (*5)	log information
			(*4)	Log information (for non-preinstalled version Java library)
			6 or more	When the CORBA Service is run.
			4.0 or less (*7)	Naming Service runtime trace information (only when the service is used)
			4.0 or less (*7)	Naming Service user exception log information
			32.3 or less	Interface Repository log information (only when the service operates)
		Solaris32/64 Linux32/64 /etc/opt	4.1 or more (*7)	Naming Service information
		Solaris32/64 Linux32/64 Component Transaction Service installation directory/var/IRDB (Interstage Setup Command is used)	10.3 or more (*9)	Interface Repository information
		Solaris32/64 Linux32/64 /tmp	1.0 or more (The capacity of the disk depends on the size of the IDL.) When the C/C++ compiler is operated, a disk volume is required for separate work.	IDL compiler operation

No.	Function	Directory	Disk space (MB Units)	Remarks
		Solaris32/64 var directory specified by environment variable OD_HTTPGW_HOME or OD_HOME	2 or more (*10)	When an HTTP-IIOP gateway internal log is collected
7	Component Transaction Service	Windows32/64 Component Transaction Service installation directory\var Solaris32/64 /var/opt/FSUNtd	25 or more	Log trace file
		Solaris32/64 /opt/FSUNtd/etc/isreg (Interstage operating environment directory) Linux32/64 /opt/FJSVtd/etc/isreg (Interstage operating environment directory)	15.0 or more	Operating environment
		Solaris32/64 Linux32/64 /var	Determine the required size from operation details (*11)	core file to be collected when an ABEND occurs
8	Load Balance (*26) (*27)	Windows32/64 Installation directory for CORBA Service\etc	Number of load balance groups x ((Number of Object references for one load balance group x Length of Object reference) + 0.0005) (*12)	data file
		Solaris32/64 Linux32/64 /etc/opt		
9	Windows32/64 Solaris32 Linux32/64 Database Linkage Service	Solaris32 Linux32/64 -	(*13)	system log file
		Windows32/64 Log file directory	Number of transactions x 0.008 + 0.001	When the Database Linkage Service is used
		Trace log file directory	OTS_TRACE_SIZE of the operating environment x 0.001	
		Resource management trace log file directory	RESOURCE_ TRACE_SIZE of the operating environment x 0.001	
		Recovery trace log file directory	RECOVERY_ TRACE_SIZE of the operating environment x 0.001	

No.	Function	Directory	Disk space (MB Units)	Remarks
		Monitoring process trace log file directory	OBSERVE_TRACE_SIZE of the operating environment x 0.001	
		Windows32/64 Resource definition file directory	Number of resource definition files registered x 0.001	
		Solaris32 opt/FSUNots/var (directory storing dump files from the otsgetdump command)	5.0 or more	
10	Event Service	Windows32/64 Installation directory for Event Service\etc	0.1 or more	Channel information
		Solaris32/64 Linux32/64 /etc/opt	1.0 or more	
		Windows32/64 Installation directory for Event Service\var Solaris32/64 Linux32/64 /var/opt	61 MB + value specified in -s -logsize option of the +essetcnf command x 2 KB or more	Log information
			Windows32/64 Solaris32/64 trace_size of traceconfig file x Number of trace file generations	Trace information
			Linux32/64 Trace is collected by the process (trace_buffer of traceconfig file = process) trace_size of traceconfig file x Number of event channel processes (*14) x Number of trace file generations Trace is collected by the Event Service (trace_buffer of traceconfig file = system) trace_size of traceconfig file x	

No.	Function	Directory	Disk space (MB Units)	Remarks
			Number of trace file generations	
		A directory specified in the save destination (newly created) directory by the Interstage Management Console, or a directory specified by "trandir", "sysdir", or "userdir" of the Event Service unit definition file	Windows32/64 38 x number of units created by Event Service or more (*15) Solaris32/64 Linux32/64 Determine the required size from operation details (*15)	Required if the Event Service is operated in persistent mode
11	Portable-ORB	Windows32/64 Installation directory for Portable-ORB (*16)	When log information collection is specified with the *porbeditenv* command: The set log file size x 2 x number of applications or applets which operate	Log information
		Solaris32/64 Linux32/64 /var/opt (*16)		
12	Interstage JMS	Windows32/64 Installation directory for JMS Service\etc	0.01 + (Number of durable subscribers x 0.002)	Definition information
		Solaris32/64 Linux32/64 /etc/opt		
		Windows32/64 Installation directory for JMS Service\var	10 or more	Console file
		Solaris32/64 Linux32/64 /var/opt		
13	Performance monitoring tool	Solaris32/64 Linux32/64 Directory which stores the performance log files	1.0 or more (*19)	Performance log files
		Solaris32/64 /var	6.4 x performance monitoring tool shared memory size (*20) x 6	Core file to be collected when an ABEND occurs
14	WorkUnit	Solaris32/64 Linux32/64 Specified with "TD path for system," a definition item of the Interstage operating environment definition	Definition size of one WorkUnit x Number of WorkUnit definitions (*21)	When registering WorkUnit definition
			One WorkUnit definition size x	When operating WorkUnit

No.	Function	Directory	Disk space (MB Units)	Remarks
			Number of starting WorkUnits (*21)	
15	Interstage JServlet(OperationManagement)	Windows32/64 Interstage JServlet(OperationManagement) installation directory\log Solaris32/64 Linux32/64 /var/opt/FJSVjs2su/log	12	Log information
16	Portal Function	-	Number of users x 1.5 MB + 40 MB	Estimate the Interface Repository database size required for operation.
17	Framework Function	Directory specified in the JVM system property java.io.tmpdir	Windows32/64 File size uploaded from the client (Web browser)	When operating web application using file upload function(*22)
			Solaris32/64 Linux32/64 Determine the required size from operation details.	
		Directory specified in log function of Framework	Directory storing the file specified by the framework log function	When operating using log function
18	Interstage Single Sign-on Business server function	Windows32/64 Installation directory for Interstage single sign-on \ssoatzag\log (Output directory of access log file)	Consider the operation details to determine the required size (*23)	Log information such as the business server access log.
		Solaris32/64 Linux32/64 /var/opt/FJSVssoaz/log (Output directory of access log file)		
		Windows32/64 Installation directory for Interstage single sign-on\ssocm \etc	2	
		Solaris32/64 Linux32/64 /var/opt/FJSVssocm/etc		
		Windows32/64 Installation directory for Interstage single sign-on	Consider the operation details to determine the required size.	Access control information
		Solaris32/64 Linux32/64		

No.	Function	Directory	Disk space (MB Units)	Remarks
		/etc/opt		
19	Interstage Single Sign-on Authentication server function	Windows32/64 Installation directory for Interstage single sign-on \ssoatcag\log (Output directory of access log file)	Consider the operation details to determine the required size (*23)	Log information such as the authentication server access log.
		Solaris32/64 Linux32/64 /var/opt/FJSVssoac/log (Output directory of access log file)		
		Windows32/64 Installation directory for Interstage single sign-on\ssocm\etc	2 (*35)	
		Solaris32/64 Linux32/64 /var/opt/FJSVssocm/etc		
20	Interstage Single Sign-on Repository server function	Windows32/64 Installation directory for Interstage single sign-on \ssoatcsv\log (Output directory of access log file and session management log file)	Consider the operation details to determine the required size (*23)(*24)	Log information such as the repository server access log.
		Solaris32/64 Linux32/64 /var/opt/FJSVssosv/log (Output directory of access log file and session management log file)		
		Windows32/64 Installation directory for Interstage single sign-on\ssocm\etc	2	
		Solaris32/64 Linux32/64 /var/opt/FJSVssocm/etc		
21	Interstage Directory Service (*32)(*33)	Windows32/64 Installation directory for Interstage Directory Service\var	20 x number of repositories created + 20	Interstage Directory Service log information
		Solaris32/64 Linux32/64 /var/opt		
		Windows32/64	0.5 x number of repositories created	Interstage Directory Service environment definition

No.	Function	Directory	Disk space (MB Units)	Remarks
		Installation directory for Interstage Directory Service\etc		
		Solaris32/64 Linux32/64 /etc/opt		
		Interstage Directory Service access log storage directory	Depends on the Interstage Management Console access log setting value "Size" x "number of access log files to maintain"	Interstage Directory Service access log
		Interstage Directory Service database storage directory (*28)	0.1 x n x s/500 + 200 (*25) "n" indicates the number of entries; "s" is the size when one entry is indicated by LDIF (bytes)	Interstage Directory Service database storage directory
22	Interstage Directory Service SDK (*32)(*33)	Windows32/64 Installation directory for Interstage Directory Service \var	Windows32/64 The number of processes x 8	Interstage Directory Service SDK log information
		Solaris32/64 Linux32/64 /var/opt/FJSVirepc	Solaris32/64 Linux32/64 Number of simultaneous connections x 8 (*31)	
23	MessageQueueDirector	Solaris32	Determine the required size from operation details.	For details, refer to "File Volume Estimate" in the MessageQueueDirector Handbook.
		Solaris32 /opt/FJSVmqd/gui/var/ trace (Trace file storage directory)	5	Trace information
24	Interstage JMX Service	Windows32/64 Interstage JMX Service install directory	14 or more (*30)	
		Solaris32/64 Linux32/64 /var/opt	14 or more (*30)	
		Solaris32/64 Linux32/64 /etc/opt	0.1 or more	
25	Configuration Manager	Configuration Manager repository Windows32/64	Determine the required size from operation details.	

No.	Function	Directory	Disk space (MB Units)	Remarks
		Installation directory for Interstage JMX Service\var\repository Solaris32/64 /var/opt/FJSVisas/ repository		

- *1 If the log file is downloaded in the following Interstage Management Console window, ensure that there is sufficient disk space to download the file.

Function	Window (Standalone)
Interstage HTTP Server	[System] > [Services] > [Web Server] > [Web Server name] > [Refer to Log] tab
	[System] > [Services] > [Web Server] > [Web Server name] > [Virtual Host] > [Virtual Host name] > [Refer to Log] tab
Web Server Connector	[System] > [Services] > [Web Server] > [Web Server name] > [Web Server Connector] > [View Log] tab
IJServer WorkUnit	[System] > [WorkUnit] > [WorkUnit name] > [View Log] tab

For details of the log file size, refer to the disk space in the log information for each function, and investigate the required size according to the operation contents.

If the log file is large and is not downloaded successfully due to insufficient disk space, use FTP to download it.

- *2 Add the following per IJServer WorkUnit.

  - process multiple *

  - 4 (amount of default disk used for container log and container information log) *

  - 6 (generation backup) above

  When application timeouts occur frequently and the application outputs many messages in a short time (and outputs debugging information), the amount of the disk used as a container information log of J2EE common directory/ijserver/(IJServer name)/(log) increases. When such a situation occurs, ensure that sufficient disk capacity is available.

- *3 By default, each Web server has 2Mbytes. When the application outputs a lot of messages in a short time and outputs debugging information, the amount of disk space used increases. When such a situation occurs, ensure sufficient disk capacity is available.

- *4 The maximum size of the log file can be set in log_file_size, access_log_size, error_log_size, process_log_size, and info_log_size of the CORBA Service config file. A disk space size of "log_file_size" x 2 is required because 1 back up file will be left. Delete log files when they are no longer necessary.

- *5 When the CORBA Service log collection function is used, the following maximum disk volume is used. (Each parameter is defined in the config file.)

  access_log_size x 2 + error_log_size x 2 + process_log_size x 2 + info_log_size x 2

  In Windows, the following disk volume is used in addition to the above:

  error_log_size x 2 + process_log_size x 2 + info_log_size x 2

  For details of the above parameters, refer to "config" in the "CORBA Service Environment Definition" appendix.

- *6 The required disk volume is the value (bytes) of (max_processes(*) + 2) x log_file_size(*) x 2. (*: CORBA-service-install-folder\etc\config file parameters) The log files to be collected are "appNNNN.log" and "appNNNN.old" (NNNN indicates alphanumeric characters) in addition to "log" and "log.old" for each server application. Delete log files when they are no longer required.

  When a Naming Service and interface repository are operated on the local host, a 4-megabyte area and a 32-megabyte area are required respectively.

- *7The required disk space for building the Naming Service on the server machine of the CORBA Service is described in Table 1.2 Required Disk Space for Building the Naming Service below.

Table 1.2 Required Disk Space for Building the Naming Service

Use		Capacity
Naming Service information	Object repository (fixed)	16 KB
	Management file (fixed)	2056 KB
	Data file (variable)	2048 (KB) x No. of contexts + (Object Reference length x No. of objects x 2)
Executive tracing information (maximum)		4096 KB
User exception log information (maximum)		4096 KB

- *8 The maximum size of the log file can be set in log_file_size of the CORBA Service config file. Log information is collected with the name JVxxxxxxxxxx.log/JVxxxxxxxxxx.old (xxxxxxxxxx is a unique numeric character string) for each application. Delete log files when they are no longer necessary.

- *9 The required disk space when the Interface Repository is used is described. Specify the Interface Repository database size according to Table 1.3 Required Disk Space When the Interface Repository is Used. Estimate the Interface Repository database size using the figures below and ensure that the required amount of disk space is available.

  Note that the Interface Repository database is extended automatically from the initial value (10,240 kilobytes).

Table 1.3 Required Disk Space When the Interface Repository is Used

Use	Capacity
Physical file	Administration area 220 KB (fixed)
	User definition area 10240 KB (not fixed)
Executive tracing file	33000 KB (maximum)

The user definition area (disk capacity required for object) can be calculated using the formulae listed in Table 1.4 User Definition Area Calculator.

Table 1.4 User Definition Area Calculator

IDL-specified definition	Formula (byte)
Module declaration	1708+((a-1)/32+1) x 176
Interface declaration	1712+((a-1)/32+1) x 176+((b-1)/32+1) x 176+512 x b
Operation declaration	2304+((e-1)/32+1) x 176+((f-1)/32+1) x 176+((g-1)/32+1) x 176
Attribute declaration	2224
Constant declaration	2160+c
Exception declaration	1712+((d-1)/32+1) x 176+836 x d
Type declaration	2220
String declaration	1716
Wide string declaration	1716
Enumeration declaration	1824+((j-1)/32+1) x 176+64 x j
Sequence declaration	2228
Structure declaration	1712+((h-1)/32+1) x 176+836 x h
Union declaration	2436+((i-1)/32+1) x 176+972 x i
Fixed decimal point type declaration	1716
Array declaration	2228

Table 1.5 User Definition Area Calculator Explanations

Symbol	Variable	Explanation
a	Number of types included	Number of type declarations included
b	Number of type declarations	Total number of type declarations except character-string type, sequence type, and array type
c	Number of inherited interfaces	Number of interfaces inherited by interface declaration
d	Constant value length	Value length of constant declaration
e	Number of exception structure members	Number of structure members in exception declaration
f	Number of parameters	Number of parameters in operation declaration
g	Number of contexts	Number of contexts in operation declaration
h	Number of exceptions	Number of exceptions in operation declaration
i	Number of character-string type declarations	Total number of character-string type declarations
j	Number of sequence type declarations	Total number of sequence type declarations

- *10 The upper limit of the log file size can be set by "max_log_file_size" of the HTTP tunneling "gwconfig file". The disk volume should be "value specified by max_log_file_size x 2" because one backup file is required. When an Interstage HTTP Server is used as the Web server, a log file is created for each Interstage HTTP Server communication process. Delete the log file when it is no longer required.

- *11 Required disk space is calculated as follows:

Solaris32 Linux32

```
Shared memory size of related CORBA Service (*1) x 3
+ Shared memory size of Component Transaction Service (*2)
+ The number of WorkUnits x 0.26
+ The number of parameters of the IDL definition included in a WorkUnit (*3) x 0.00005
+ Basic size (*4)
```

Solaris64 Linux64

```
Shared memory size of related CORBA Service (*1) x 3
+ The number of WorkUnits x 0.26 + Basic size (*4)
```

  - (*1) Shared memory size of related CORBA Service

    This is calculated by the following formulas from each parameter in the config file (/opt/FSUNod/etc/config) of the CORBA Service.

    ```
    limit_of_max_IIOP_resp_con x 0.016
    + limit_of_max_IIOP_resp_requests x 0.016
    + max_impl_rep_entries x 0.006
    + 0.01
    ```

  - (*2) Shared memory size of Component Transaction Service

    The number of clients x 0.1 + 100

    The number of clients depends on the system scale that was specified using the *isgendef* command to generate the Interstage system definition. Estimate the number of clients in accordance with the system scales shown below.

    - small: 50

    - moderate: 100

    - large: 500

    - super: 1000

    - (*3) The number of WorkUnits

      Add when there are many WorkUnits and parameters in an IDL definition.

      When a structure object is in the parameter of each operation, add the number of members of a structure object for every parameter.

    - (*4) Basic size

      According to the value specified as the System Scale: statement of the Component Transaction Service environment definition file (/var/opt/FSUNtd/etc/sysdef), the basic size is as follows:

      Solaris32 Linux32

        - small: 250
        - moderate: 330
        - large: 840
        - super: 1400

      Solaris64 Linux64

        - small: 270
        - moderate: 350
        - large: 860
        - super: 1420

- *12 The initial value allocated on first start is 8256 KB. When this is exceeded, it is increased in 1024KB units.

- *13 Estimate as follows because the Database Linkage Service system log file differs depending on the system scale specified by the *isgendef* command.

    - small: 1 MB or more
    - moderate: 2 MB or more
    - large: 8 MB or more
    - super: 16 MB or more

- *14 Number of Event Channel processes =

  Number of static Event Channel groups + Number of dynamic Event Channel processes

  (Number of dynamic Event Channel processes: The value specified for "Event maximum Process" of the Interstage operating environment definition, set when Interstage is initialized using the *isinit* command. If the Notification Service is used, make this value the "Number of dynamic Event Channel processes x 2".)

- *15 When the environment is set using the Interstage Management Console, the following capacity is required in the "Storage Directory" of "Store (Create a New Unit)".

    - File capacity for the event data
    - File capacity for the system
    - File capacity for the transaction: ((The transaction multiplicity x 4) + 256 + (The maximum message size in one transaction x 2)) x 16 (Kbytes)

  When the environment is set using the unit definition file, the following capacity is required in the directory specified with the unit definition file.

    - Directory specified in "sysdir": Size specified by "syssize"
    - Directory specified in "userdir": Size specified by "usersize"
    - Directory specified in "trandir": ((tranmax x 4) + 256 + (tranunitmax x 2)) x 16 (Kbytes)

- *16 When operating as an applet, this is the directory specified as 'log storage directory' from the *porbeditenv* command in the local disk of the client machine where the applet runs.

- *19 Required amount (MB) = Common memory size specified by the *ispmakeenv* command (MB) x (Measurement time/interval) x Measurement period (days)

- *20 Solaris32/64 Shared memory size to be specified by the *ispmakeenv* command -m option

- *21 One WorkUnit definition size =

  1000 + (500 x "Number of [Application Program] section definitions") +

  (500 x "Number of [Resource Manager] section definitions") +

  (500 x "Number of [Nonresident Application Process] section definitions") +

  (500 x "Number of [Multiresident Application Process] section definitions") +

  User optional specified string data length.

- *22 When the size of the file up-loaded from a Web browser exceeds the memory size for the file transfer specified by the Web application, the up-loaded file is stored in this directory.

- *23 If the default settings are enabled, there is no upper limit to the amount of disk that can be used. To prevent disk shortages from occurring, either regularly delete log files that are no longer needed or change the log collection method.

- *24 Add 52 MB if the repository server conducting session management is to operate in a cluster system.

- *25 The amount of disk per repository. For the replication environment, use the following calculation (master repository only)

  0.2 x n x s / 500 + 200

  (n is number of entries and s is a size describing one entry with LDIF (byte unit))

  The calculation formula is a standard. Set sufficient disk capacity.

  Message irep30023 is displayed when the disk area specified at the database storage destination is insufficient, and the repository is canceled. Refer to "Messages Beginning with 'irep'" in the Messages manual to determine the required action for message irep30023.

- *26 This is not valid for Windows (64 bit).

- *27 This is not valid for Linux (64 bit).

- *28 This is required if the standard database is used as the repository database.

- *29 When session recovery is used to enable session serialization, the session serialization file is created in the directory specified in the Session Registry Server environment definition file. The following amount of disk space is required for each deployed Web application:

  - Windows32/64

    (0.005 + (0.005 + data capacity for maintaining the session) x number of sessions) x 2 (MB)

  - Solaris32/64

    (0.001 + (0.002 + data capacity for maintaining the session) x number of sessions) x 2 (MB)

  - Linux32/64

    (0.008 + (0.008 + data capacity for maintaining the session) x number of sessions) x 2 (MB)

  "Data capacity for maintaining the session" is the object and key total set for the session attribute (Attribute) in the Web application.

  The values shown above may vary depending on the file system that is used.

  Interstage Application Server Enterprise Edition can be used to run Session Registry Server.

- *30 If the upper limit for the login log file size is changed during customization of Interstage JMX Service, the following disk space is required:

  - Login log

    The upper limit for the login log file size x 2 (Mbytes)

  If the upper limit is not changed, [1] is set for the maximum login log file size.

- *31 To estimate the number of simultaneous connections, specify the required total for each of the following:
    - The number of application processes needed to access the repository using a thread concurrency application
    - The application process concurrency needed to access the repository using a process concurrency application
    - The number of simultaneous client connections made using Interstage Single Sign-on to a Web server built in to the Interstage Single Sign-on repository server function
- *32 This is not valid for Standard-J Edition on Windows (64 bit).
- *33 This is not valid for Standard-J Edition on Linux (64 bit).
- *35 To use Integrated Windows Authentication, add 2MB.

1.1.2 Using the Multi Server Management Function of Application Server

Windows32/64 Solaris32 Linux32/64

1.1.2.1 Using the Admin Server

Sufficient free disk space is necessary to use the following required services for Admin Server functions. For details refer to "1.1.1 Using the Server Function of Application Server".

- Interstage Management Console
- Interstage HTTP Server
- Interstage Directory Service
- Interstage JMX Service
- Configuration Manager

The free disk space required for operation using the Admin Server function of Application Server is shown in Table 1.6 Disk Space Required for Operation Using the Admin Server function

Table 1.6 Disk Space Required for Operation Using the Admin Server function

No.	Function	Directory	Disk Space (MB Units)	Remarks
1	Deploy Applications	Windows32/64 Installation directory for Interstage JMX Service\var\ssv_ijs	Application size x 2	-
		Solaris32 Linux32/64 /var/opt/FJSVisjmx/ssv_ijs		
2	View Log	Windows32/64 Installation directory for Interstage Management Console \isAdmin\var\download	(*1)	log information
		Solaris32 Linux32/64 /var/opt/FJSVisgui/tmp/download		

- *1 If the log file is downloaded in the following Interstage Management Console window, ensure that there is sufficient disk space to download the file.

Function	Window
Interstage HTTP Server	[Integrated Control] > [Interstage] > [Interstage Application Server] > [Server Group name] > [Server name] > [System] > [Services] > [Web Server] > [FJapache] > [Refer to Log] tab

Function	Window
	[Integrated Control] > [Interstage] > [Interstage Application Server] > [Server Group name] > [Server name] > [System] > [Services] > [Web Server] > [FJapache] > [Virtual Host] > [Virtual Host name] > [Refer to Log] tab
Web Server Connector	[Integrated Control] > [Interstage] > [Interstage Application Server] > [Server Group name] > [Server name] > [System] > [Services] > [Web Server] > [FJapache] > [Web Server Connector] > [View Log] tab
IJServer WorkUnit	[Integrated Control] > [Interstage] > [Interstage Application Server] > [Server Group name] > [Server name] > [System] > [WorkUnit] > [WorkUnit name] > [View Log] tab

For details of the log file size, refer to the disk space in the log information for each function, and investigate the required size according to the operation contents.

If the log file is large and is not downloaded successfully due to insufficient disk space, use FTP to download it.

1.1.2.2 Operating as a Managed Server

Sufficient free disk space is necessary for operating as a Managed Server. For details refer to "1.1.1 Using the Server Function of Application Server".

The free disk space required for operation using the Admin Server function of Application Server is shown in Table 1.7 Disk Space Required for Operation Using the Admin Server function.

Table 1.7 Disk Space Required for Operation Using the Admin Server function

No.	Function	Directory	Disk Space (MB Units)	Remarks
1	Added a Site	Windows32/64 Installation directory for Interstage JMX Service\etc Solaris32 Linux32/64 /etc/opt/FJSVisjmx/	1	-

1.1.2.3 Running the Server Functions as the Combined Server

In the Combined Server, the Admin Server function and the Interstage Server function (the Managed Server) are run on the same machine. Refer to the above description before listing the services to be used in the Admin Server function and the Interstage Server function. For details of the disk space that is required for each service, refer to "1.1.1 Using the Server Function of Application Server".

Note

To use the same service in the Admin Server function and the Interstage Server function, there is no need to count the system resources (for example, the Admin Server and Interstage Directory Service) twice.

1.2 Memory Requirements

This section describes the memory requirements for:

- 1.2.1 Using the Server Function of Application Server
- 1.2.2 Using the Multi Server Management Function of Application Server
- 1.2.3 Using the Client Function of Application Server

1.2.1 Using the Server Function of Application Server

The memory capacity required for operating this software using the Application Server is shown in Table 1.8 Memory Requirements (Using the Application Server).

Table 1.8 Memory Requirements (Using the Application Server)

No.	Function	Memory Required(MB)	Remarks
1	Interstage Management Console	121 or more	
2	Interstage HTTP Server	Windows32 22.7 + (0.05 x m) + (0.06 x n) or more	The HTML file is accessed by more than one client simultaneously: m: Maximum number of requests processed simultaneously, specified in the ThreadsPerChild directive of the environment definition file (httpd.conf). n: Number of clients that access the HTML file concurrently
		Windows64 25.2 + (0.04 x m) + (0.12 x n) or more	
		Solaris32 17.0 + (3.5 x n) or more	
		Solaris64 25.0 + (4.5 x n) or more	
		Linux32/64 8.0 + (3.0 x n) or more	
3	IJServer WorkUnit	Windows32/64 59.6 or more (*1) 56.5 or more (*15)	When the IJServer WorkUnit is operated in "Web Applications and EJB Applications run in same Java VM" mode, the sample of EjbBmp(Web,Session,BMP).
		Solaris32/64 107.4 or more (*1)	
		Linux32 64.6 or more (*1)	
		Linux64 107.0 or more (*1)	
		Windows32/64 62.1 or more (*1) 56.6 or more (*15)	When the IJServer WorkUnit is operated in "Web Applications and EJB Applications run in same Java VM" mode, the sample of EjbCmp11(Web,Session,CMP1.1).
		Solaris32/64 108.7 or more (*1)	
		Linux32 65.8 or more (*1)	
		Linux64 108.6 or more (*1)	
		Windows32/64 65.8 or more (*1) 61.2 or more (*15)	When the IJServer WorkUnit is operated in "Web Applications and EJB Applications run in same Java VM" mode, the sample of EjbCmp20(Web,Session,CMP2.0).
		Solaris32/64	

No.	Function	Memory Required(MB)	Remarks
		114.7 or more (*1)	
		Linux32 69.1 or more (*1)	
		Linux64 119.3 or more (*1)	
		Windows32/64 71.6 or more (*1) 66.2 or more (*15)	When the IJServer WorkUnit is operated in "Web Applications and EJB Applications run in same Java VM" mode, the sample of EjbMessageDriven(Web,Session,MDB).
		Solaris32/64 119.2 or more (*1)	
		Linux32 71.6 or more (*1)	
		Linux64 125.6 or more (*1)	
		Windows32/64 51.6 or more (*1) 53.1 or more (*15)	When the IJServer WorkUnit is operated in "Web Applications Only" mode, the sample of HelloServlet(Web).
		Solaris32/64 97.7 or more (*1)	
		Linux32 66.5 or more (*1)	
		Linux64 89.1 or more (*1)	
		Windows32/64 54.3 or more (*2) 55.8 or more (*15)	When the IJServer WorkUnit is operated in "EJB Applications Only" mode, the sample of EjbBmp(Web,Session,BMP).
		Solaris32/64 98.2 or more (*2)	
		Linux32 62.5 or more (*2)	
		Linux64 114.0 or more (*2)	
		Windows32/64 55.8 or more (*2) 55.9 or more (*15)	When the IJServer WorkUnit is operated in "EJB Applications Only" mode, the sample of EjbCmp11(Web,Session,CMP1.1)
		Solaris32/64	

No.	Function	Memory Required(MB)	Remarks
		100.9 or more (*2)	
		Linux32 58.5 or more (*2)	
		Linux64 113.5 or more (*2)	
		Windows32/64 57.9 or more (*2) 58.6 or more (*15)	When the IJServer WorkUnit is operated in "EJB Applications Only" mode, the sample of EjbCmp20(Web,Session,CMP2.0)
		Solaris32/64 107.9 or more (*2)	
		Linux32 61.0 or more (*2)	
		Linux64 115.5 or more (*2)	
		Windows32/64 59.8 or more (*2) 60.2 or more (*15)	When the IJServer WorkUnit is operated in "EJB Applications Only" mode, the sample of EjbMessageDriven(Web,Session,MDB)
		Solaris32/64 109.9 or more (*2)	
		Linux32 56.5 or more (*2)	
		Linux64 122.5 or more (*2)	
4	CORBA Service	Windows32/64 16 or more (*3)	
		Solaris32/64 Linux32/64 8 or more	
		8 or more	Naming Service is operated
		45.6 or more (*4)	Interface Repository is operated
		Solaris32/64 2.4	When the COBOL Web subroutine is used
5	Event Service	Windows32/64 16 or more (*5)	
		Solaris32/64 Linux32/64 8 or more (*5)	

No.	Function	Memory Required(MB)	Remarks
6	Notification Service	Number of units x 100 + Total of shmmax of the Event Service unit definition file (*5)	When the persistent channel is operated.
7	Portable-ORB	1.5 or more	
		Windows64 3.0 or more	
8	Component Transaction Service	Windows32/64 48 or more (*6)	The Service is started
		Solaris32/64 Linux32/64 50.0 or more (*7)	
		Solaris32/64 Linux32/64 4.0 or more (*8)	Service start (when the process concurrency is assumed to be 1 per WorkUnit)
9	Windows32/64 Solaris32 Linux32/64 Database Linkage Service	Windows32/64 18.0 + 10.0 x n + 0.008 x m	Service start (on the machine where the Database Linkage Service is run) n: Total number of (concurrency for each Resource Management Program + 1) m: Maximum number of transactions
		Solaris32 Linux32/64 6.0 + 0.004 x (Maximum number of transactions)	Service start (on the machine that starts both the resource management program and OTS system) When the maximum number of transactions is 512
		Windows32/64 8.9 + 10.0 x n + 0.008 x m	Service start (on the machine that starts only the resource management program) n: Total number of (concurrency for each Resource Management Program + 1) m: Maximum number of transactions
		Solaris32 Linux32/64 4.0 + 0.004 x (Maximum number of transactions)	Service start (on the machine that starts only the resource management program)
10	Load Balance (*14) (*17)	2.0	
11	Solaris32/64 Session Information Management Function	7.0 or more	
12	Portal Function	Number of simultaneous users x 0.1	
13	Framework Function	Windows32/64 Memory used by Application Server + 32MB	
		Solaris32/64 Linux32/64 2.9 (Reference value) (*9)	When the sample "model" is operated.
14	Interstage Single Sign-on Business server function	10 or more (*10)	

No.	Function	Memory Required(MB)	Remarks
15	Interstage Single Sign-on Authentication server function	10 or more (*11)	
16	Interstage Single Sign-on Repository server function	10 or more (*12)	
17	Interstage Directory Service	Windows32/64 340.0 or more (*13)	When Interstage Directory Service is run in stand-alone, database sharing or slave mode (*18)
		Solaris32/64 400.0 or more (*13)	
		Linux32/64 217.0 or more (*13)	
		50.0 or more	When Interstage Directory Service is run as the master mode (*18) (Value that must be added when the above Interstage Directory Service is operated in stand-alone mode)
		2.0 or more	When the entry management command is used
		Windows32/64 22.0 or more	When the entry management tool is used
		Solaris32/64 Linux32/64 60.0 or more	
		m x n x 3	When the entry searching m: Size of the LDIF file used in registration of an entry. n: The number of search results notified by the server to a client search request.
18	Interstage JMX Service	Windows32/64 80.0 or more 81.3 or more (*15) Solaris32/64 200.0 or more Linux32/64 130.0 or more	
19	Solaris32/64 Linux32/64 Session Registry Server (*14)	Solaris32/64 Example : 254 Linux32/64 Example : 120 (*16)	-

- *1 Estimate the details using the formulae below.

  Web Server Connector:

- Windows32/64

  2 + 0.2 x k

- Solaris32/64

  1.9 x k + 30

- Linux32/64

  1 x k + 30

  (k: simultaneous access count to the Servlet Service)

IJServer WorkUnit (per process):

- For Web Applications and EJB Applications run in the same Java VM (MB)

  - Windows32/64

    48 + (1.4 x k) + (0.7 x w) + (P1 + P2 + P3 + .. + Pn)

  - Solaris32/64

    121 + (2.1 x k) + (0.7 x w) + (P1 + P2 + P3 + .. + Pn)

  - Linux32/64

    28 + (1.5 x k) + (0.7 x w) + (P1 + P2 + P3 + .. + Pn)

- For Web Applications Only (MB)

  - Windows32/64

    47 + (1.3 x k) + (0.7 x w) + (P1 + P2 + P3 + .. + Pn)

  - Solaris32/64

    84 + (2.5 x k) + (0.7 x w) + (P1 + P2 + P3 + .. + Pn)

  - Linux32/64

    27 + (1.3 x k) + (0.7 x w) + (P1 + P2 + P3 + .. + Pn)

    (k: simultaneous access count to the Servlet container)

    (w: number of Web applications)

    (Pn: execution size of each Servlet or JSP. In the above table, it is calculated as 1 MB.)

  - Solaris32/64 Linux32/64

    If the Session Registry Client is used (this is not valid for Linux (64 bit)), add the following values:

    (0.002 + data capacity for maintaining the session) x assumed number of sessions(MB)

    where 'data capacity for maintaining the session' is the object and key total set for the session attribute (Attribute) in the Web application.

The Servlet Container:

The Servlet container operates on JavaVM, so the actual memory usage (including the heap area) depends on the following factors:

- Class name for which new instance (process) is created
- Number of new instances(processes)
- Lifecycle of an instance
- GC operation status
- Various definitions of the IJServer WorkUnit
- JavaVM to be used

Fujitsu therefore recommends accurately estimating Servlet container memory usage (heap area) by actually measuring it as follows:

    - Operate the Servlet Service under full operation peak time conditions. If the heap area used by Java VM is exhausted, an OutOfMemoryError is output to the log. Increase the maximum value of the heap area to an appropriate value. Use the determined maximum value of the heap area for full operation.

- *2 Estimate the required amount of memory for operating the EJB Service using the following description as reference.

  When operating an EJB application, the memory amount for JavaVM (initial value and maximum value) and the memory amount required in one process differ depending on the following factors.

  - Class type to be renewed
  - Number of instances to be renewed
  - Life cycle of instance
  - GC operation status
  - Definitions of EJB applications

  As the memory amount cannot be calculated easily for any of the above, estimate it by actually measuring it using the following method.

  1. Initial value of memory amount used by JavaVM (the value specified with the -Xms option of the java command)

     Operate the EJB application in the same conditions as during ordinary time (instead of peak time) of actual operation. If the memory amount (maximum value) used by JavaVM is running short, an IJServer21033 or EJB1033 message is output. Set the memory amount (maximum value) to an appropriate value by trial and error. Use the memory amount obtained in this way as the memory amount (initial value) in actual operation. The memory amount default (initial value) is 2MB.

  2. Maximum value of memory amount used by JavaVM (the value specified in the -Xmx option of the java command)

     Operate the EJB application in the same conditions as peak time of actual operation. If the memory amount (maximum value) used by JavaVM is running short, an IJServer21033 or EJB1033 message is output. Set the memory amount (maximum value) to an appropriate value by trial and error. Use the memory amount obtained in this way directly as the memory amount (maximum value) in actual operation. The memory amount default (maximum value) is 64MB.

  3. Total memory amount required in one process

     When estimating the memory amount used by JavaVM in Items (1) and (2), also estimate the total memory amount required in one process by actual measurement.

- *3 Windows32/64 16 megabytes plus the added value (refer to the table below) is required depending on the CORBA Service operating environment (config file) settings.

Operating mode	Required value (added value)
Module declaration	3902+a(2b+2)
When the CORBA Service is run	100KB + max_IIOP_resp_con x 16KB + max_IIOP_resp_requests x 16KB + max_impl_rep_entries x 6 KB (or more)
When the trace function is used	(When the CORBA Service is run) + 20KB + max_processes x trace_size_per_process (or more)
When the snapshot function is used	(When the CORBA Service is run) + 10KB + snap_size (or more)

  When a CORBA application of the client package is run, 1.5 megabytes of memory is required for each process.

- *4 The Interface Repository deploys the object saved in the database on memory. The required memory amount when the Interface Repository is used is described below.

  - Fixed available area: 45.6 MB
  - Variable available area

  In the Interface Repository, memory is used for each object. The available memory for each object can be calculated using the formulae listed in Table 1.9 Available Memory Calculator for Interface Repository Objects.

Table 1.9 Available Memory Calculator for Interface Repository Objects

IDL-specified definition	Formula (unit byte)
Module declaration	3902+a(2b+2)
Interface declaration	3902+a(2b+2)+abc
Operation declaration	3934+a(3b+2+f)+abg+(12+a+ab)h
Attribute declaration	3910+a(3b+2)
Constant declaration	7704+a(3b+3)+d
Exception declaration	3836+a(2b+e+1)+(78+a+ab)e
String declaration	3882+a(b+1)
Enumeration declaration	3918+a(2b+k+2)
Sequence declaration	3882+a(2b+1)
Structure declaration	3766+a(2b+i+1)+i(78+a+ab)
Union declaration	3840+a(3b+j+1)+(3880+2a+ab)j
Fixed decimal point declaration	3882+a(b+1)
Array declaration	3886+a(2b+1)

Table 1.10 Interface Repository Object Memory Calculator Explanations

Symbol	Variable	Explanation
a	Identifier length	Length of the object's identifier
b	Number of hierarchies	Hierarchy within which the object exists
c	Number of inheritances	Number of interfaces inherited by the interface declaration
d	Constant value length	Value length of the constant declaration
e	Number of exception structure members	Number of structure members in the exception declaration
f	Number of contexts	Number of contexts in the operation declaration
g	Number of exceptions	Number of exceptions in the operation declaration
h	Number of parameters	Number of parameters in the operation declaration
i	Number of struct members	Number of members in the structure declaration
j	Number of union members	Number of members in the shared declaration
k	Number of enum members	Number of members in the enumerated type declaration

- *5 Add the required memory estimated using the following formula according to the Event Service application mode:
  - When you increase the number of consumers/suppliers of statically generated event channels using the essetcnf command

    (a+b) x c (KB)
  - When you increase the number of consumers/suppliers of dynamically generated event channels using the essetcnf command

    (a+b) x d (KB)
  - When you increase the number of consumers/suppliers of statically generated event channels using the essetcnf and essetcnfchnl commands

    (a+b) x (c-e) + (f+g) x e (KB)

    - The any type is used for the event data format stored in the event channel

      Size of the message x Number of stored messages

    - The StructuredEvent type is used for the event data format stored in the event channel

      (Size of the message + (Number of QoS property items x 4KB)) x Number of stored messages

  Notes

  a: Additional number of consumers specified using the -coninit option of the essetcnf command

  b: Additional number of suppliers specified using the -supinit option of the essetcnf command

  c: Number of event channel groups

  d: Maximum number of event channels specified using the -dchmax option of the essetcnf command

  e: Number of event channel groups set using the essetcnfchnl command

  f: Additional number of consumers specified using the -coninit option of the essetcnfchnl command

  g: Additional number of suppliers specified using the -supinit option of the essetcnfchnl command

- *6 Windows32/64 This value does not include the CORBA Service memory size.

  Add the CORBA Service memory size to this value.

- *7 When the user authentication function is used, add 0.9 megabyte.

  When access control is used, add 0.6 megabyte.

- *8 Estimate the details using the following formula:

  4 MB x Total number of processes under the WorkUnit.

- *9 Estimate the required memory capacity and include it in the memory requirements to run the Servlet Service when running a Web application created using the framework. Apply the memory usage of the Web application as a value of Pn (execution size of each servlet or JSP) in the formulas in (*1).

  This value is 2.9MB for sample "model" of the framework. The memory requirements for running the Servlet Service can be estimated using the methods described in (*1).

  When an EJB application created using the framework is run, the memory requirements include the memory usage required to run the EJB Service, and can be estimated using the methods described in (*2).

- *10 Add the value estimated using the following formula according to the application:

  (2,400 + (Number of roles + Number of role sets + (Number of role sets x Number of roles)) x 2,048 or greater) x Number of PATH configurations + number of business server x (2,000,000 + Cache size x Cache count)

    - Number of roles: Total number of roles of the tuning business server. This total is set in the PATH configuration of the protection resources defined in the SSO repository

    - Number of role sets: Total number of role sets of the tuning business server. This total is set in the PATH configuration of the protection resources defined in the SSO repository

    - Number of PATH configurations: Total number of PATH configurations of the tuning business server. This total is set in the PATH configuration of the protection resources defined in the SSO repository.

    - For details on the cache size, refer to "Tuning for Setting up the Business Server" in the "Environment Definition for Interstage Single Sign-on" appendix.

    - For details on the cache count, refer to "Tuning for Setting up the Business Server" in the "Environment Definition for Interstage Single Sign-on" appendix.

- *11If session management is not to be performed, add the value estimated using the following formula according to the application:

  ((Number of Site definitions x 1024) + (Number of path definitions x 1024)) x 2

    - Number of Site definitions: Total number of Site definitions in the SSO repository

    - Number of path definitions: Total number of path definitions defined for all the Site definitions in the SSO repository

If you are using Integrated Windows Authentication, add 256MB.

- *12 Add the value estimated using the following formula according to the application:

  ((Number of roles + Number of role sets + Number of role sets x Number of roles) x 2048 bytes or more) x 2

  - Number of roles: Total number of roles defined in the SSO repository
  - Number of role sets: Total number of role sets defined in the SSO repository

  If session management is to be performed, add the value calculated with the formula below to the calculation above:

  23,500,000 + ((Number of users using the Single Sign-on system at the same time x (2,560 + Y)) x 2)

  [Y: Extended user information]

  The value calculated using the formula above changes according to the amount of extended user information that is notified.

  Size of extended user information that is notified x 2

  To use Active Directory as the directory service for registering user information without using Single Sign-on extended schema, add the value calculated above to the value calculated with the formula below:

  Total number of attributes used for the Active Directory role/role set x 524 x 2

- *13 Calculate the memory space using the following formulas when several repositories are created and used.

  - Windows32/64

    Memory space: number of repositories x 340.0 (megabytes)

  - Solaris32/64

    Memory space: number of repositories x 150.0 (megabytes)

  - Linux32/64

    Memory space: number of repositories x 217.0 (megabytes)

- *14 This is not valid for Linux (64 bit).

- *15 For Windows Server(TM) 2003 x64 Editions.

- *16 Estimate the details according to the following formula:

  - Solaris32/64

    85.7 + (2.5 x k) + (0.01 x a) + ((0.002 + d) x s) x 2 (MB)

  - Linux32/64

    28.7 + (1.3 x k) + (0.01 x a) + ((0.002 + d) x s) x 2 (MB)

    k: Number of simultaneous Session Registry Server processes

    a: Number of Web applications (deployed in IJServer)

    d: Data capacity for maintaining the session =

    The object and key total set for the session attribute (Attribute) in the Web application.

    s: Number of sessions

  Example: The number of simultaneous IJServer processes is [64], there is [1] application, the data capacity for maintaining the session is [2]KB, and the number of sessions is [1000]

  - Solaris32/64

    85.7 + (2.5 x 64) + (0.01 x 1) + ((0.002 + 0.002) x 1000) x 2

    = 85.7 + 160 + 0.01 + 8

    = 254

  - Linux32/64

    28.7 + (1.3 x 64) + (0.01 x 1) + ((0.002 + 0.002) x 1000) x 2

= 28.7 + 83.2 + 0.01 + 8

=120

The Session Registry Server runs on the Java VM. For this reason, the actual memory used (including the Heap) depends on the load and GC operating conditions.

When estimating how much memory will be used, it is recommended that the memory is calculated so that the estimate is correct.

- The Session Registry Server runs according to the same conditions as the actual application peak. If there is insufficient memory for using the Java VM, a message is output in the event log. For this reason, increase the maximum Heap and set an optimum value. The maximum value for the requested Heap is used for the actual application.

Interstage Application Server Enterprise Edition can be used to run the Session Registry Server.

Note

If the memory size required to operate this software is not reserved, an error may occur.

- *17 This is not valid for Windows (64 bit).
- *18 "master" and "slave" in the table refer to the master and slave servers that run in the replication mode that uses the standard database for the repository database.

  The memory requirement for running the master and slave server mode in the replication mode that uses the RDB for the repository database is the same as the one used for the standalone.

1.2.2 Using the Multi Server Management Function of Application Server

Windows32/64 Solaris32 Linux32/64

1.2.2.1 Using the Admin Server

Sufficient memory capacity is necessary to use the following required Admin Server services. For details refer to "1.2.1 Using the Server Function of Application Server".

- Interstage Management Console
- Interstage HTTP Server
- Interstage Directory Service
- Interstage JMX Service

1.2.2.2 Operating as a Managed Server

Sufficient memory capacity is necessary to use required services for operating as a Managed Server. For details refer to "1.2.1 Using the Server Function of Application Server".

1.2.2.3 Running the Server Functions as the Combined Server

In the Combined Server, the Admin Server function and the Interstage Server function (the Managed Server) are run on the same machine. Refer to the above description before listing the services to be used in the Admin Server function and the Interstage Server function. For details of the memory capacity that is required for each service, refer to "1.2.1 Using the Server Function of Application Server".

1.2.3 Using the Client Function of Application Server

This is not valid for Linux (64 bit).

This section lists the memory size required when this software is run.

No.	Function	Memory Required(MB)	Remarks
1	Interstage Directory Service	22.0 or more	When using the Entry Administration Tool of Interstage Directory Service

1.3 Swap Space

Solaris32/64

This section describes the swap space requirements for this software.

1.3.1 Swap Space for Interstage Application Server Enterprise Edition Server Package

The swap space requirements for normal operation of this software are shown in Table 1.11 Swap Space for Interstage Application Server Enterprise Edition Server Package. If the swap space provided by the system is not sufficient, expand it.

Table 1.11 Swap Space for Interstage Application Server Enterprise Edition Server Package

No.	Function	Swap capacity (unit: MB)	Remarks
1	Interstage HTTP Server	24 + (6 x n)	The HTML file is accessed by more than one client simultaneously: n: Number of clients that access the HTML file concurrently
2	CORBA Service	10.0 or more	The CORBA Service communication base service is used.
		10.0 or more	The Naming Service is used.
		50.0 or more	The Interface Repository is used.
3	Load Balance function	4.0 or more	
4	Event Service	10.0 or more	
5	Portable-ORB	2.0 or more	
6	Component Transaction Service	48.0 or more	The Component Transaction Service is started (SMALL system scale).
		55.0 or more	The Component Transaction Service is started (MODERATE system scale).
		65.0 or more	The Component Transaction Service is started (LARGE system scale).
		65.0 or more	The Component Transaction Service is started (SUPER system scale).
		2.1 or more	In the Component Transaction Service, the CORBA Service linkage function and Web service linkage function are used. (The process multiplicity of the WorkUnit is 1.)
		3.0 or more	In the Component Transaction Service, the user authentication function is used.
7	IJServer WorkUnit	233.7 or more	When the IJServer WorkUnit is operated in "Web Applications and EJB Applications run in same Java VM" mode, the sample of EjbBmp(Web,Session,BMP).
		234.1 or more	When the IJServer WorkUnit is operated in "Web Applications and EJB Applications run in same Java VM" mode, the sample of EjbCmp11(Web,Session,CMP1.1).
		234.9 or more	When the IJServer WorkUnit is operated in "Web Applications and EJB Applications run in same Java VM" mode, the sample of EjbCmp20(Web,Session,CMP2.0).
		243.7 or more	When the IJServer WorkUnit is operated in "Web Applications and EJB Applications run in same Java VM" mode, the sample of EjbMessageDriven(Web,Session,MDB).
		179.7 or more	When the IJServer WorkUnit is operated in "Web Applications Only" mode, the sample of HelloServlet(Web).
		230.0 or more	When the IJServer WorkUnit is operated in "EJB Applications Only" mode, the sample of EjbBmp(Web,Session,BMP).

No.	Function	Swap capacity (unit: MB)	Remarks
		234.5 or more	When the IJServer WorkUnit is operated in "EJB Applications Only" mode, the sample of EjbCmp11(Web,Session,CMP1.1).
		231.9 or more	When the IJServer WorkUnit is operated in "EJB Applications Only" mode, the sample of EjbCmp20(Web,Session,CMP2.0).
		241.4 or more	When the IJServer WorkUnit is operated in "EJB Applications Only" mode, the sample of EjbMessageDriven(Web,Session,MDB).
8	Previous version of the EJB Service	45.7 or more	The previous version of the EJB Service is used. (BMP)
		45.2 or more	The previous version of the EJB Service is used. (CMP)
9	Interstage Single Sign-on	1.0 or more	Business server function
		1.0 or more	Authentication server function
		1.0 or more	Repository server function
10	Interstage Directory Service	190.0 or more	Calculate the swap space using the following formulas when several repositories are created and used. Swap space: number of repositories x 190.0 (MB)
11	Interstage JMX Service	200.0 or more	
12	Session Registry Server (*1)	252.9 or more	When the Session Registry Server is operated by tuning "Xms256m-Xmx256m" of heap area size, swap space is set to the value given in 'Swap capacity'. Note: The swap space changes with tuning and the environment of heap area size.

- *1 This is not valid for Linux (64 bit).

1.3.2 Using the Multi Server Management Function of Application Server

Windows32/64 Solaris32 Linux32/64

1.3.2.1 Using the Admin Server

Sufficient swap space is necessary to use the following required Admin Server services. For details refer to "1.3.1 Swap Space for Interstage Application Server Enterprise Edition Server Package".

- Interstage Management Console
- Interstage HTTP Server
- Interstage Directory Service
- Interstage JMX Service

1.3.2.2 Operating as a Managed Server

Sufficient swap space is necessary to use required services for operating as a Managed Server. For details refer to "1.3.1 Swap Space for Interstage Application Server Enterprise Edition Server Package".

1.3.2.3 Running the Server Functions as the Combined Server

In the Combined Server, the Admin Server function and the Interstage Server function (the Managed Server) are run on the same machine. Refer to the above description before listing the services to be used in the Admin Server function and the Interstage Server function. For details of the swap space that is required for each service, refer to "1.3.1 Swap Space for Interstage Application Server Enterprise Edition Server Package".

Chapter 2 Tuning Interstage

This chapter explains how to tune Interstage.

By merely specifying the system scale, you can make Interstage set up a model scenario in which the system can operate, and register a definition for each service. In some cases, however, the system will need to be specified in more detail.

After tuning Interstage, you will need to use the isregistdef command to register the definition for each service before executing the system. The changes made in the tuning operation will be put into effect by Interstage's initialization function, and reflected when Interstage is started.

Tuning Interstage is carried out by modifying the following files:

- The Component Transaction Service system environment definition file
- The CORBA Service environment definition file
- The Database Linkage Service system environment definition file

  Note: The Database Linkage Service is only used for the Enterprise Edition.
- The Interstage operating environment definition file

2.1 Hypothetical System Configuration

Linkages using transaction applications will be set up as model scenarios. The following types of linkage use transaction applications:

- Local transaction linkage
- Global transaction linkage
- Linkage using session control

Windows32/64 Solaris32/64

- Linkage with an existing system (global server)

  Note: Linkage can only be performed with the Enterprise Edition.

The transaction applications will be designed to the following conditions.

- It is assumed that all the transaction applications will link to other servers (including linkages within the local system).
- The number of transaction application object processes is one tenth of the maximum number of clients that can connect.
- Transaction application objects can only connect to one server machine.
- There is only one of each resource per server machine, and the multilevel concurrency of resource managers per server machine is 5.
- For database linkage services, the multilevel concurrency is 5, and for recovery programs it is 2.

If the system is going to run CORBA applications, perform load balancing, or use server machine state monitoring, then Interstage will need to be tuned.

Load balancing is not valid for Linux (64 bit).

2.2 Definition File Setting Values

Each definition file will set the system scale specified in the Interstage system definition System Scale statement.

Specify the system scale when the Interstage system definition is generated using the isgendef command (for details, refer to the Reference Manual (Command Edition)).

There are four types of system scale (as shown below):

- small
- moderate
- large

- super

Note that the following system scales will have been set in the setup environment immediately after the installation.

Windows32/64

- large

Solaris32/64 Linux32/64

- small

The values set in each definition file are shown below.

Note

- "FJapache" in the Interstage operating environment definition is valid only in Interstage Application Server Enterprise Edition.
- Environment definitions for the Component Transaction Service are valid only in Interstage Application Server Enterprise Edition.

Values Set for the Different Scale Systems

The following table lists the values that are set for each system scale.

Table 2.1 Values Set for the Different Scale Systems

Definition	Statement	Value			
		small	moderate	large	super
Interstage operating environment definition	Corba Host Name	No value			
	Corba Port Number	No value			
	IR path for DB file	Windows32/64 TD_HOME\var\IRDB (*1) Solaris32/64 Linux32/64 TD_HOME/var/IRDB (*2)			
	IR USE	No value (*3)			
	IR Host Name	No value Solaris32/64 Linux32/64 (*3)			
	IR Port Number	8002			
	NS USE	No value (*3)			
	NS Host Name	No value.			
	NS Port Number	8002			
	NS JP	no			
	Windows32 Solaris32 Linux32 LBO USE (*7)	no			
	TD path for system	Windows32/64 TD_HOME\var (*1) Solaris32/64 /var/opt/FJSVisas/system/default/FSUNextp Linux32/64 /var/opt/FJSVisas/system/default/FJSVextp			

Definition	Statement	Value			
		small	moderate	large	super
	Windows32/64 Solaris32 Linux32/64 OTS Multiple degree	5			
	Windows32/64 Solaris32 Linux32/64 OTS Recovery	2			
	Windows32/64 Solaris32 Linux32/64 OTS path for system log	No value. (*4)			
	Windows32/64 Solaris32 Linux32/64 OTS maximum Transaction	Windows32/64 5 Solaris32 Linux32/64 50	Windows32/64 10 Solaris32 Linux32/64 100	Windows32/64 50 Solaris32 Linux32/64 500	Windows32/64 100 Solaris32 Linux32/64 1000
	Windows32/64 Solaris32 Linux32/64 OTS Setup mode	sys			
	Windows32/64 Solaris32 Linux32/64 OTS JTS's RMP Multiple degree of Process	5			
	Windows32/64 Solaris32 Linux32/64 OTS JTS's RMP Multiple degree of Thread	16			
	Windows32/64 Solaris32 Linux32/64 OTS Participate	4			
	Windows32/64 Solaris32 Linux32/64 OTS Host	No value			
	Windows32/64 Solaris32 Linux32/64 OTS Port	No value			
	Event Service	no			
	Event maximum Process	2			
	Event maximum Connection	Windows32/64 5 Solaris32/64 Linux32/64 50	Windows32/64 10 Solaris32/64 Linux32/64 100	Windows32/64 50 Solaris32/64 Linux32/64 500	Windows32/64 100 Solaris32/64 Linux32/64 1000
	Event Auto Disconnect	no			
	Event SSL	no			
	SSL USE	no			

Definition	Statement	Value			
		small	moderate	large	super
	SSL Port Number	4433			
	IS Monitor Mode	mode2			
	Fjapache	no			
CORBA Service environment definition (*5)	max_IIOP_resp_con	Windows32/64 33 Solaris32/64 Linux32/64 80	Windows32/64 40 Solaris32/64 Linux32/64 135	Windows32/64 100 Solaris32/64 Linux32/64 575	Windows32/64 175 (*6) Solaris32/64 Linux32/64 1024
	max_IIOP_resp_requests	Windows32/64 772 Solaris32/64 Linux32/64 1920	Windows32/64 896 Solaris32/64 Linux32/64 2944	Windows32/64 1920 Solaris32/64 Linux32/64 10112	Windows32/64 3968 Solaris32/64 Linux32/64 20352
	max_processes	Windows32/64 29 Solaris32/64 Linux32/64 31	Windows32/64 31 Solaris32/64 Linux32/64 36	Windows32/64 51 Solaris32/64 Linux32/64 76	Windows32/64 76 Solaris32/64 Linux32/64 126
	max_exec_instance	448	448	Windows32/64 448 Solaris32/64 Linux32/64 1046	Windows32/64 448 Solaris32/64 Linux32/64 2046
Component Transaction Service Environment Definition	[SYSTEM ENVIRONMENT] System Scale	small	moderate	large	super
Database Linkage Service Environment Definition	No statement.	No value.			

- *1 TD_HOME : Installation folder \td of Interstage.
- *2 TD_HOME : Installation directory of Component Transaction Service.
- *3 This must be specified without fail if the system configuration is TYPE3.
- *4 This must be specified without fail if the system configuration is TYPE2.
- *5 When the isregistdef command is initially submitted, the value shown in Table 2.1 Values Set for the Different Scale Systems is added to the value in the CORBA service definition file. When the isregistdef command is submitted the second time or later, the value added when the previous command was submitted is subtracted from the current setting, and the new value specified in SystemScale is added.
- *6 The maximum value that can be set is 1023.
- *7 This is not valid for Windows (64 bit).

2.3 How to Tune Interstage

Broadly speaking, Interstage needs to be tuned when new applications have been added and when Interstage functions are to be used.

2.3.1 Tuning Interstage According to the Type of Application Added

This section explains the service definition statements and values to be added when a client application or a server application has been added. The values to be added depend on whether the new application is a CORBA application or a transaction application.

We shall now explain how to perform the tuning required when the application added is one of the following:

- a client application
- a server application

  Note: A server application must be tuned in the Enterprise Edition and Standard-J Edition.
- a client/server application

  Note: A client/server application must be tuned in the Enterprise Edition and Standard-J Edition.

If the New Addition is a Client Application

CORBA Application

The following table lists the CORBA application values.

Table 2.2 CORBA Application

Definition	Statement	Value to Add
CORBA service environment definition	max_processes (*1)	Total number of processes
	max_IIOP_resp_con (*1 and 2)	

- *1 Solaris32/64 Linux32/64

  If max_processes, max_IIOP_resp_con has been altered, the system parameters must be set.
- *2 SSL connections and non-SSL connections must be counted separately. For this reason, when using SSL Linkage the total is 'Number of processes x 2'.

EJB Client Application

The following table lists the EJB client application values.

Table 2.3 EJB Client Application

Definition	Statement	Value to Add
CORBA service environment definition	max_processes (*1)	Number of processes of added client applications

- *1 Solaris32/64 Linux32/64

  If max_processes has been altered, you will need to set the system parameters.

If the New Addition is a Server Application

CORBA Application

The following table lists the CORBA application values.

Table 2.4 CORBA Server Application

Definition	Statement	Value to Add
CORBA Service environment definition	max_processes (*1)	Total number of processes
	max_exec_instance	Total number of threads for request execution

- *1 Solaris32/64 Linux32/64

  If max_processes has been altered, you will need to set the system parameters.

EJB Application

The following table lists the EJB application values.

Table 2.5 EJB Application

Definition	Statement	Value to Add
CORBA service environment definition	max_processes (*1)	Number of processes of added EJB applications
	max_exec_instance	Windows32/64 [When the EJB application is that the thread can operate] Number of processes of added EJB applications * 16 [When the EJB application is that the thread cannot operate] Number of processes of added EJB applications Solaris32/64 Linux32/64 Number of processes of added EJB applications * 16

- *1 Solaris32/64 Linux32/64

  If max_processes has been altered, you will need to set the system parameters.

Transaction Application Windows32 Solaris32 Linux32

This eventuality does not need to be considered.

If the New Addition is a Client/Server Application

Application to work even if a CORBA client, is taken when other objects are called from the server application and it gets an object reference and session control function, XA connection, and so on are used is shown.

CORBA Application

The following table lists the CORBA client/server application values

Table 2.6 CORBA Application

Definition	Statement	Value to Add
CORBA Service environment definition	max_processes (*1)	Total number of processes
	max_IIOP_resp_con (*1) (*2)	
	max_exec_instance	Total number of threads for request execution

- *1 Solaris32/64 Linux32/64

  If max_processes, max_IIOP_resp_con has been altered, the system parameters must be set.

- *2 SSL connections and non-SSL connections must be counted separately. For this reason, when using SSL Linkage the total is 'Number of processes x 2'.

Transaction Application Windows32 Solaris32 Linux32

The following table lists the transaction application values.

Table 2.7 Transaction Application

Definition	Statement	Value to Add
CORBA Service environment definition	max_processes (*1)	Total number of processes
	max_IIOP_resp_con (*1) (*2)	

- *1 Solaris32 Linux32

  If max_processes, max_IIOP_resp_con has been altered, the system parameters must be set.

- *2 SSL connections and non-SSL connections must be counted separately. For this reason, when using SSL Linkage the total is 'Number of processes x 2'.

2.3.2 Tuning Interstage in Order to Use Interstage Functions

This section explains how to perform the tuning required to use any of the following Interstage functions:

Refer to the following table and perform tuning for the relevant services that are shown, depending on the products used.

Table 2.8 Tuning for Interstage Products

	Interstage Application Server Enterprise Edition	Interstage Application Server Standard-J Edition
Windows32/64 Solaris32 Linux32/64 Database Linkage Services	Must be tuned.	Must be tuned.
Windows32 Solaris32 Linux32 Load balancing (*3)	Must be tuned.	Tuning is not necessary. (*1)
Event Service (*2)	Must be tuned.	Must be tuned.
Windows32 Solaris32 Linux32 Server Machine Condition Monitoring	Must be tuned.	Tuning is not necessary. (*1)

- *1 The service is not available in the relevant product.
- *2 Tuning for the event service is required when Interstage JMS is used.
- *3 This is not valid for Linux (64 bit).

2.3.2.1 Database Linkage Services Windows32/64 Solaris32 Linux32/64

Tuning the Multilevel Concurrency of Database Linkage Services

To change the multilevel concurrency of database linkage services, set or adjust the values given in the following table.

Table 2.9 Values for Multilevel Concurrency of Database Linkage Services

Definition	Statement	Value to Add or Set
Interstage operating environment definition	OTS Multiple degree (*1)	Multilevel concurrency of the database linkage services
CORBA Service environment definition	max_IIOP_resp_requests (*2) (*3)	

Definition	Statement	Value to Add or Set
	max_exec_instance (*2)	

- *1 Set the value.
- *2 Adjust the value.
- *3 If the multilevel concurrency of database linkage services is greater than *max_IIOP_resp_requests*, set the former.

Tuning the Multilevel Concurrency of Recovery Programs

To change the multilevel concurrency of recovery programs, set or adjust the values given in the following table.

Table 2.10 Values for Multilevel Concurrency of Recovery Programs

Definition	Statement	Value to Add or Set
Interstage operating environment definition	OTS Recovery (*1)	Multilevel concurrency of recovery programs
CORBA service environment definition	max_IIOP_resp_requests (*2)	
	max_exec_instance (*2)	

- *1 Set the value.
- *2 Adjust the value.

Resource Manager Tuning

If you are going to start more than one resource manager, or change the multilevel concurrency of resource managers, adjust the values given in the following table.

Table 2.11 Values for Resource Manager Tuning

Definition	Statement	Value to Add
CORBA service environment definition	max_processes (*1)	Total (multilevel concurrency of resource managers + 1)
	max_IIOP_resp_con (*1)	
	max_exec_instance	

- *1 Solaris32 Linux32/64

  If max_IIOP_resp_con, max_processes has been altered, you will need to set the system parameters.

2.3.2.2 Load Balancing Windows32 Solaris32 Linux32

This is not valid for Linux (64 bit).

If you are going to use the load balancing function, adjust the values given in the following table.

Table 2.12 Values for the Load Balancing Function

Definition	Statement	Value to Add
CORBA service environment definition	max_exec_instance	Value specified in -m option of *odsetlbo* command
	max_processes (*1)	Constant1
	max_IIOP_resp_con (*1)	

- *1 Solaris32 Linux32

  If max_processes, max_IIOP_resp_con has been altered, you will need to set the system parameters.

2.3.2.3 Event Service

If you are going to use the Event Service function, set or adjust the values given in the following table.

Table 2.13 Values for the Event Service Function

Definition	Statement	Value to Add or Set
CORBA service environment definition	max_exec_instance	(*2)
	max_IIOP_local_init_con	One of the following maximum values: - max_IIOP_local_init_con - Maximum value for the number of processes for consumers/suppliers to be started + 3 (*3)
	max_IIOP_local_init_requests	One of the following maximum values: - max_IIOP_local_init_requests - Maximum value for the number of processes for consumers/suppliers to be started + 3 (*3) * Number of requests that 'mix' model consumers/suppliers can connect to (send) at the same time using one connection - Maximum value for the number of processes for consumers/suppliers to be started + 3 (*3) * Number of requests that 'push' model consumers/'pull' model suppliers can connect to (receive) at the same time using one connection
	max_IIOP_resp_con (*1)	Total number of consumers and suppliers that are connected to all event channel, + 1 (*4)
	max_IIOP_resp_requests	One of the following maximum values: - The value added to max_IIOP_resp_con * (Number of requests that 'mix' model consumers/suppliers can connect to (send) at the same time using one connection + 1) - The value added to max_IIOP_resp_con * (Number of requests that 'push' model consumers/'pull' model suppliers can connect to (receive) at the same time using one connection + 1)
	max_processes (*1)	Number of processes of the event channels to be activated plus the number of consumers and suppliers plus 2 (*4)
	max_impl_rep_entries	Total number of static generation processes to be created and the total number of static generation event channels. (*5)
	period_receive_timeout	Timeout limit before connections are returned in case of a malfunction (*6)

- *1 Solaris32/64 Linux32/64

  If max_IIOP_resp_con, max_processes has been altered, you will need to set the system parameters.

- *2 The sum value is different in the event channel system and in the consumer and supplier systems. Add the following value depending on the system.

  Windows32/64

  - Event channel (In the case of static activation of the event channel)

    Total number of "number of connections of its event channel groups (specified by the *-m* option of the *esmkchnl* command) (*1)".

    *1: If "number of connections * 2" is less than 256, then it will assume the value of 256.

    - Event channel (If using event factory)

      "Number of processes (the value specified for "Event maximum Process" of the Interstage operating environment definition, set when Interstage is initialized using the *isinit* command)" * "Number of connections (the value specified for "Event maximum Connection" of the Interstage operating environment definition, set when Interstage is initialized using the *isinit* commandt)(*2)" + 17.

      *2: If "number of connections * 2" is less than 256, then it will assume the value of 256.

    - Consumers and suppliers

      "Number of server applications (number of Push model consumers plus number of Pull model suppliers)" * "Initial thread concurrency (the thr_conc_maximum value specified by the *-ax* option of the *OD_impl_inst* command).

  Solaris32/64 Linux32/64

    - Event channel (In the case of static activation of the event channel)

      Total number of "number of connections of its event channel groups (specified by the *-m* option of the *esmkchnl* command) (*3)".

      *3: If "number of connections + 16" is less than 256, then it will assume the value of 256.

    - Event channel (If using event factory)

      "Number of processes (the value specified for "Event maximum Process" of the Interstage operating environment definition, set when Interstage is initialized using the *isinit* command)" * "Number of connections (the value specified for "Event maximum Connection" of the Interstage operating environment definition, set when Interstage is initialized using the *isinit* command)(*4)" + 17.

      *4: If "number of connections + 16" is less than 256, then it will assume the value of 256.

    - Consumers and suppliers

      "Number of server applications (number of Push model consumers plus number of Pull model suppliers)" * "Initial thread concurrency (the thr_conc_init value specified by the *-ax* option of the *OD_impl_inst* command).

- *3 The further 3 is added to run the EventChannel.

- *4 if you execute the Event Service operation commands while communicating with the event channel, add 1.

- *5 The number of processes for the static generation event channels is the number of static generation event channel groups created by the esmkchnl command or Interstage Management Console.

  The number of static generation event channel processes (when using Event Factory) is the maximum number of event channel processes specified in "Event maximum Process" in the Interstage operating environment definition on execution of *isinit*.

- *6 Set a longer time than the event data queuing time specified in the Event Service (specify it using "-wtime" in the *essetcnf* or *essetcnfchnl* command).

  period_receive_timeout x 5 > event data waiting time

  If timeout occurs due to the period_receive_timeout before the event data waiting time has elapsed, the following may occur:

  Loss of event data

  Sending response ends with failure (Error 0d10605)

  The event data waiting time must be shorter than the value of period_receive_timeout times 5 (Fujitsu recommends specifying the event data waiting time so that the difference between the two values is at least 20 seconds).

  If the event data waiting time is set to 0, the event data waiting time becomes infinite, which results in a timeout due to period_receive_timeout. For this reason, refrain from specifying 0 for the event data waiting time.

2.3.2.4 Server Machine Condition Monitoring Windows32 Solaris32 Linux32

If you are going to use the server machine condition monitoring function, adjust the values given in the following tables.

Tuning the Server Performing the Monitoring

Table 2.14 Values for the Server Machine Condition Monitoring Function

Definition	Statement	Value to Add
CORBA Service environment definition	max_processes (*1)	Constant1
	max_IIOP_resp_con (*1)	
	max_exec_instance	Constant4

- *1 Solaris32 Linux32

  If max_processes, max_IIOP_resp_con has been altered, you will need to set the system parameters.

Tuning the Server to be Monitored

Table 2.15 Values for the Server to be Monitored

Definition	Statement	Value to Add
CORBA Service environment definition	max_processes (*1)	Constant1
	max_IIOP_resp_con (*1)	
	max_exec_instance	

- *1 Solaris32 Linux32

  If max_processes, max_IIOP_resp_con has been altered, you will need to set the system parameters.

2.4 Environment Variables

The following table lists the environment variables used by Interstage.

Table 2.16 Environment Variables Used by Interstage

Name of Environment Variable	Purpose
OD_HOME	Specifies the install path for the CORBA Service. Example: OD_HOME=/opt/FSUNod
OD_CODE_SET	Specifies the client code system when code conversion is performed. Example: OD_CODE_SET=UNICODE
OTS_HOME	Specifies the install path for the Database Linkage Service. Setting of OTS_HOME is not required to operate OTS. Example: OTS_HOME=/opt/FSUNots
OTS_SCROLL_SIZE	Specifies the number of rows in the transaction list when it is displayed on one screen using the *otstranlist* or *otspendlist* command. Example: OTS_SCROLL_SIZE=30 Default: 20
OTS_INVOKE_MODE	Specifies the machine on which the OTS system or resource management program runs when an OTS system or resource management program on an older version/level of Interstage is linked with a distributed transaction. The only value that can be specified is 2. Example: OTS_INVOKE_MODE=2
ES_HOME	Specifies the install path for the Event Service. Example: ES_HOME=/opt/FJSVes
PORB_HOME	Specifies the install path for the Portable ORB. (Also used to search for the Portable ORB operating environment file). Example: PORB_HOME=/opt/FJSVporb

2.5 Operating in an IPv6 Environment

The section explains the method used to operate using Interstage in an IPv6 environment.

Note

- Interstage only supports IPv4/IPv6 dual stack. Use Interstage in IPv4/IPv6 dual stack. Interstage is not supported where IPv4 has been disabled in the operating system.
- Operations that use global addresses and unique local addresses are possible. For link-local addresses, whether or not there is support varies according to the function. For details, refer to the explanation for each function below.
- To perform an IPv6 operation by specifying a host name, it must be possible to resolve the name of that host name in the IPv6 address. For example, execute the ping command to check that it is possible to use IPv6 communication for the target host name correctly in advance.
- Operations in environments where temporary address support (privacy extension) is enabled are not supported. For details on temporary address support (privacy extension), refer to each operating system manual.
- When the IPv6 address is output to a log/message, the format may sometimes be different from that of the value (input value) specified in the configuration file.
- For functions that can run in IPv6 environments, it is also possible to set IPv6 communication using the Interstage Management Console. For details on how to set IPv6 communication, refer to the Interstage Management Console Help.

Services

Function name		Standard-J Edition	Enterprise Edition
JDK/JRE		Yes	Yes
Java EE 6		Yes	Yes
J2EE	EJB	Yes	Yes
	JTS/JTA	Yes	Yes
Multilanguage Service	CORBA Service	No	Yes
	Event Service	No	Yes
	Windows32/64 Solaris32 Linux32/64 Database Linkage Service	No	Yes
	Solaris32 Component Transaction Service	No	Yes
Interstage HTTP Server		Yes	Yes
Interstage Single Sign-on		Yes	Yes
Interstage Directory Service		Yes	Yes
The Certificate/Key Management Environment Using the SMEE Command		Yes	Yes

JDK/JRE

IPv6 communication can be performed using JDK/JRE.

Note

Windows32/64

To use the java.nio package, a Windows Vista(R)/Microsoft(R) Windows Server(R) 2008 or later operating system must be used.

Java EE 6

Java EE 6 can be used in an IPv6 environment.

Note

- Windows32/64

  A Windows Vista(R)/Microsoft(R) Windows Server(R) 2008 or later operating system must be used.

- link-local addresses cannot be used.

J2EE

IPv6 communication is possible when the functions shown below are used. However, IPv6 communication cannot be performed when SSL linkage has been used.

- EJB

- JTS/JTA

Note

- The CORBA service environment settings must be configured. In the CORBA service operating environment file (config), set "v4-dual" or "v6" for "IP-version", and then restart Interstage. The default is "v4-dual".

- Linux32/64

  link-local addresses cannot be used.

- IPv4-mapped addresses cannot be used as input values.

Multilanguage Service

IPv6 communication is possible when the functions shown below are used. However, IPv6 communication cannot be performed when SSL linkage has been used.

- CORBA Service

- Event Service

- Database Linkage Service Windows32/64 Solaris32 Linux32/64

- Component Transaction Service (expected for server machine status monitoring mechanism, and load balancing for IPCOM linkage) Solaris32

Note

- The CORBA service environment settings must be configured. In the CORBA service operating environment file (config), set "v4-dual" or "v6" for "IP-version", and then restart Interstage. The default is "v4-dual".

- Solaris32

  To use the Component Transaction Service, the Component Transaction Service environment settings must be configured. In the Component Transaction Service environment definition file, set "v6" for "IP-version", and then restart Interstage. The default is "v4".

- Linux32/64

  link-local addresses cannot be used.

- IPv4-mapped addresses cannot be used as input values.

Interstage HTTP Server

HTTP/HTTPS communication can be used in an IPv6 environment.

Note

- Linux32/64

  link-local addresses cannot be used.

- IPv4-mapped addresses cannot be used as input values.

Interstage Single Sign-on

Interstage Single Sign-on can be used in an IPv6 environment.

Note

- link-local addresses cannot be used.
- IPv4-mapped addresses cannot be used as input values.
- IPv6 addresses cannot be specified directly. Specify the host name or FQDN.

Interstage Directory Service

Interstage Directory Service can be used in an IPv6 environment.

Note

- link-local addresses cannot be used.
- IPv4-mapped addresses cannot be used as input values.
- IPv6 addresses cannot be specified directly. Specify the host name.

The Certificate/Key Management Environment Using the SMEE Command

IPv6 communication is possible when the *cmgetcrl* command (get CRL) is executed.

Note

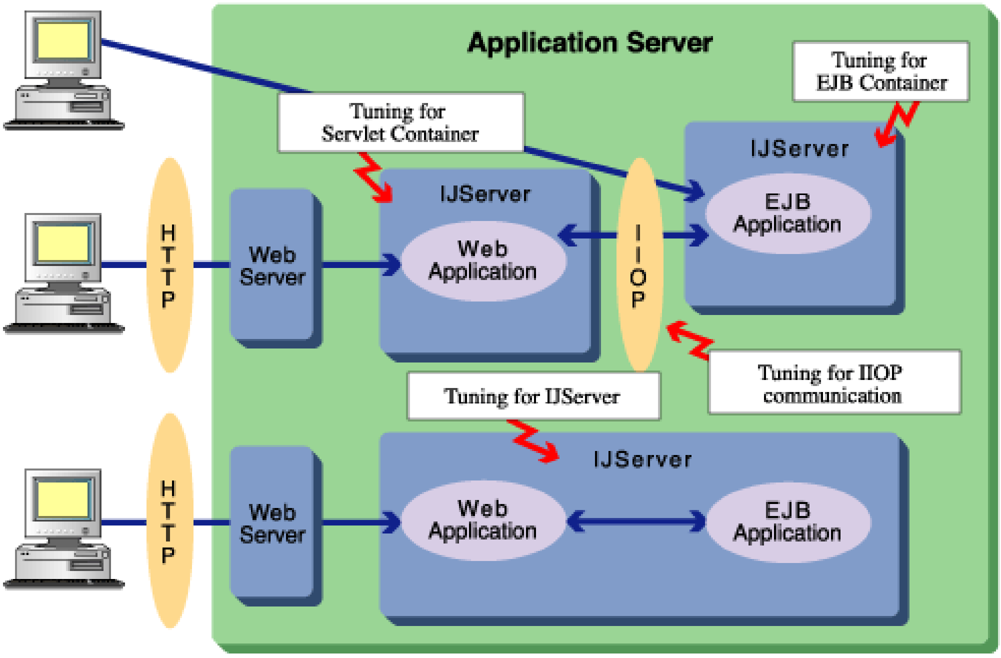
- link-local addresses cannot be used.
- IPv4-mapped addresses cannot be used as input values.

Chapter 3 Tuning J2EE Applications

With Interstage, service definitions are registered based on a model case where you can enable system operation only by specifying a system scale. To run a J2EE application, you must tune each of the components that make up J2EE in addition to the above definitions.

This section describes tuning-related settings using the following pattern of a J2EE application as an example. This section also explains the 3.1 J2EE Monitor Logging Function, which is useful for tuning J2EE applications.

Figure 3.1 Tuning a J2EE Application



- 3.2 Tuning of IJServer
- 3.3 Tuning Servlet Containers
- 3.4 Tuning of EJB Containers
- Tuning of IIOP communication (*1)
- 3.5 Tuning the CORBA Service
- 3.6 Tuning Directory Service as an LDAP Server

*1 Refer to the "CORBA Service Environment Definition" appendix for more information on how to tune IIOP communication.

.

3.1 J2EE Monitor Logging Function

The J2EE monitor logging function logs performance information for IJServer. This function can be used to periodically collect performance information for Java VMs, JDBC data sources and so on, and to output the results to log files. Log files are output in CSV format, so performance information can be analyzed easily by importing log files into a program such as Microsoft® Excel, which is also useful for accumulating statistical information.

When the *isj2eemonitor* command is executed, it sends a logging request to the Interstage JMX Service.

If the Interstage JMX Service has received a request to commence logging, it periodically sends a log output request to IJServer at specified intervals, and outputs performance information to CSV format files as part of the IJServer process. The Interstage JMX Service stops logging processing if it receives a request to stop logging.

Target for J2EE Monitor Logging

J2EE monitor logging can be used with IJServer of V9.0 or later.

This product can also be used with IJServers running in V8.0 compatibility mode or IJServers that have been created using V8.0 or earlier, but the J2EE monitor logging function cannot be used with these kinds of IJServer.

An error will be output if the J2EE monitor logging command is executed on IJServers running in V8.0 compatibility mode or IJServers that have been created using V8.0 or earlier.

3.1.1 Operation Procedure for J2EE Monitor Logging

The operation procedure for J2EE monitor logging is explained below.

- 3.1.1.1 Starting J2EE Monitor Logging
- 3.1.1.2 Stopping J2EE Monitor Logging
- 3.1.1.3 Flow of Monitoring Operations

3.1.1.1 Starting J2EE Monitor Logging

Start J2EE monitor logging by executing the *isj2eemonitor* command as follows:

```
isj2eemonitor -start -n IJServer name
```

Monitoring can be performed by starting J2EE monitor logging either before or after the IJServer whose performance is to be monitored has been started.

It is also possible to make additional specifications (for items such as the log collection interval or the information to be collected) when the command is executed. Refer to the Reference Manual (Command Edition) for more information.

If logging commences before the IJServer starts, the first data collected will be aggregate information for the interval between the time when IJServer started and the next time when the log collection interval has elapsed. When IJServer is stopped, log output also stops at that point.

3.1.1.2 Stopping J2EE Monitor Logging

Stop J2EE monitor logging by executing the *isj2eemonitor* command as follows:

```
isj2eemonitor -stop -n IJServer name
```

If the following service has stopped, J2EE monitor logging will stop.

Windows32/64

"Interstage Operation Tool" Service

Solaris32/64 Linux32/64

Interstage JMX Service

3.1.1.3 Flow of Monitoring Operations

Collecting Logs Only at Particular Times

To collect performance information only at particular times (in order to investigate a problem, for example), analyze the performance information by starting the J2EE monitor logging function when the user wants to collect the logs after IJServer has started, as follows:

1. Start IJServer using either the Interstage Management Console or the *isstartwu* command.
2. Start J2EE monitor logging using the *isj2eemonitor* command.
3. Use a tool such as Microsoft® Excel to analyze the performance information that has been output.
4. Stop J2EE monitor logging using the *isj2eemonitor* command.

5. Repeat steps 2 to 4.

6. Stop IJServer using either the Interstage Management Console or the *isstopwu* command.

Collecting Logs Continuously

To check the validity of performance tuning by collecting logs continuously, analyze performance information by first starting the J2EE monitor logging function and then starting IJServer, as follows:

1. Start J2EE monitor logging using the *isj2eemonitor* command.

2. Start IJServer using either the Interstage Management Console or the *isstartwu* command.

3. Use a tool such as Microsoft ® Excel to analyze the performance information that has been output.

4. Stop IJServer using either the Interstage Management Console or the *isstopwu* command.

5. Stop J2EE monitor logging using the *isj2eemonitor* command.

6. Repeat steps 1 to 5.

3.1.2 Log Files for J2EE Monitor Logging

The log files for J2EE monitor logging are explained below.

3.1.2.1 Log File Names

The log files that are used to output monitor information are output for each process and logging target. The file names for log files are assigned according to the following naming rule.

```
monitor-[ logging target name].log
```

The following table shows a list of file names.

File	File name
Java VM information log files	monitor-JavaVM.log
Database information log files	monitor-DataSource.log
Transaction information log files	monitor-Transaction.log
Servlet container information log files	monitor-ServletContainer.log
EJB container information log files	monitor-EJBContainer.log

3.1.2.2 Output Conditions for Log Files

Log files are created and performance information is output only when the specified IJServer is running.

Also, the type of log file that can be output depends on the type of IJServer being monitored. The following table shows the log files that can be output for each IJServer type.

	Running Web applications and EJB applications on the same Java VM	Running Web applications and EJB applications on different Java VMs	Running only Web applications	Running only EJB applications
Java VM information log files	Yes	Yes	Yes	Yes
Database information log files	Yes (*1)	Yes (*1)	Yes (*1)	Yes (*1)
Transaction information log files	Yes	Yes	Yes	Yes
Servlet container information log files	Yes	Yes (*2)	Yes	-
EJB container information log files	Yes	Yes (*3)	-	Yes

*1 Database information log files are output only when a data source is being used.

*2 Servlet container information log files are output only for Java VMs that are running Web applications.

*3 EJB container information log files are output only for Java VMs that are running EJB applications.

3.1.2.3 File Names after Log Files have been Rolled Over

Log files that have been output roll over at fixed intervals. After log files are rolled over, they are backed up by appending the date and time when they were rolled over, as shown below. A string indicating the date and time is inserted between the ".log" extension and the part of the log file name that indicates the name of the logging target (e.g., "JavaVM"). The name of the logging target and the date/time string are also separated by a hyphen ("-").

```
monitor-[ logging target name]-YYYY_MM_DD-hh_mm_ss.log
```

The following table explains the date/time information.

YYYY	Indicates years using a four-digit number (0000~9999).
MM	Indicates months using a two-digit number (01~12).
DD	Indicates days using a two-digit number (01~31).
hh	Indicates hours using a two-digit number (00~23).
mm	Indicates minutes using a two-digit number (00~59).
ss	Indicates seconds using a two-digit number (00~59).

Example

monitor-JavaVM-2006_06_24-01_00_00.log

3.1.2.4 The Lifecycle of Log Files

When an IJServer process outputs a log, new files are created if the log files explained in '3.1.2.6 Output Directory' do not exist in the log output directory. If a log file exists then the log message will be appended to this file. Log files are also rolled over when the following conditions are met.

Rollover Conditions

- The IJServer process is running at the rollover time

- When the log is collected, the previous log file is still in the log output directory, and the update time for this file is earlier than the previous rollover time

The rollover start time can be specified when J2EE monitor logging is started. Refer to the Reference Manual (Command Edition) for more information.

Rollovers are performed as follows:

1. If the number of files that have been backed up is greater than or equal to the specified number of log file generations, the file with the oldest update date and time is deleted. Files are deleted until the number of backed up files is equal to one less than the specified number of generations.
2. Existing log files are backed up by changing their names as explained in '3.1.2.3 File Names after Log Files have been Rolled Over'.
3. New files are created with the name explained in '3.1.2.6 Output Directory'.

3.1.2.5 File Access Rights Solaris32/64 Linux32/64

The owner of the files that are output will be the IJServer startup user, and the file permissions will be "644".

3.1.2.6 Output Directory

The default output directory for log files is as follows:

Windows32/64

```
[J2EE Common Directory]\ijserver\[ IJService name]\log\[process serial number]
```

Solaris32/64 Linux32/64

```
[J2EE Common Directory]/ijserver/[IJService name]/log/[process serial number]
```

The output directory can be changed either with the IJServer definitions for the *isj2eeadmin* command or by selecting [Work Unit] > "*work unit name*" > [Environment Settings] tab > [Detailed Settings] > [Work Unit Settings] > [Log Output Directory] from the Interstage Management Console.

3.1.3 Analyzing and Responding To Performance Information

This section explains how to analyze the performance information that is collected in log files, and how to respond to this information.

Output Information

Monitor information log files are output in CSV format (with items separated by commas) as follows:

```
D1,D2,D3,D4,D5,...
```

The Content of Performance Information Items

The monitor information that can be looked up using the Interstage Management Console displays aggregate data that has been collected since IJServer started. The data collected using the J2EE monitor logging function, on the other hand, collects aggregate data (such as maximum values during the data collection period) within the data collection interval, which is useful for analyzing performance bottlenecks and so on.

For the Interstage Management Console, data for the entire IJServer (except for some data, such as Java VM data) can be aggregated and looked up as monitor information, but with the J2EE monitor logging function, data can be collected for each IJServer process, as shown below, which makes more detailed analysis possible.

	Interstage Management Console	J2EE monitor logging
Upper limit values	The upper limit values that can be used for the entire IJServer	The upper limit values that can be used by each process
Maximum values	Maximum values for the entire IJServer	Maximum values for each process
Minimum values	Minimum values for the entire IJServer	Minimum values for each process
Average values	Average values for the entire IJServer	Average values for each process
Current values	Current values for the entire IJServer	Current values for each process

Date/time information in performance information is output using "DD/MM/YYYY hh:mm:ss:SSS" format.

DD	Indicates days using a two-digit number (01~31).
MM	Indicates months using a two-digit number (01~12).
YYYY	Indicates years using a four-digit number (0000~9999).
Hh	Indicates hours using a two-digit number (00~23).
Mm	Indicates minutes using a two-digit number (00~59).
Ss	Indicates seconds using a two-digit number (00~59).
SSS	Indicates milliseconds using a three-digit number (000~999).

Example

24/06/2006 01:00:00:200

The "millisecond" units in performance information are output using "h:mm:ss:SSS" format.

h	Displays the hour as a variable-digit number (0 or more). Displayed as "0" if the value is 0.
mm	Indicates minutes using a two-digit number (00~59).
ss	Indicates seconds using a two-digit number (00~59).
SSS	Indicates milliseconds using a three-digit number (000~999).

Example

0:00:00:200

This section explains the items that are output as performance information. The D1, D2, ..., Dn in the item numbers of each table correspond to the D1, D2, ..., Dn that are output in CSV format.

1) Java VM Information

Item number	Item name	Unit	Content
D1	The date/time when data collection started	-	The date/time when measurements of performance information for this record started
D2	Date/time when data collection ended	-	The date/time when measurements of performance information for this record ended
D3	Process serial number	-	The serial number for the process started by IJServer
D4	Process ID	-	The process ID for the IJServer being measured

Item number	Item name	Unit	Content
D5	Container type	-	The container type for the Java VM. One of the following will be output: - 1VM (a process that operates both servlets and EJBs) - Web (a process that operates only servlets) - EJB (a process that operates only EJBs)
D6	Operating time for the Java VM	-	The total time that has elapsed since the IJServer started
D7	Current heap usage	KB	The heap usage for the Java VM when the log was taken
D8	Minimum heap usage	KB	The minimum heap usage for the Java VM during the measurement period (*1)
D9	Maximum heap usage	KB	The maximum heap usage for the Java VM during the measurement period (*1)
D10	Upper limit value for the heap	KB	The upper limit value for the heap size (a value that is the same as or less than the value specified with the -Xmx option)
D11	Current usage of the "Perm" area	KB	The "Perm" area usage for the Java VM when the log was taken
D12	Minimum usage of the "Perm" area	KB	The minimum usage for the "Perm" area for the Java VM during the measurement period (*1)
D13	Maximum usage of the "Permission" area	KB	The maximum usage for the "Perm" area for the Java VM during the measurement period (*1)
D14	Upper limit value for the "Perm" area	KB	The upper limit value for the size of the "Perm" area a value that is almost the same as the value specified with the -XX:MaxPermSize option)
D15	Number of times garbage collection occurred	Times	The number of times that garbage collection was performed during the measurement period (*2)
D16	Total time for garbage collection processing	Millisecond s	The total value for the time spent on garbage collection processing during the measurement period (*2)

*1 The values that are output for the minimum and maximum values of the Java VM heap and the "Permanent" area are the maximum and minimum values of samples taken at three second intervals, in the same way as the Monitor Display window in the Interstage Management Console.

*2 Garbage collection uses Full GC information.

2) Data Source Information

Item number	Item name	Unit	Content
D1	The date/time when data collection started	-	The date/time when measurements of performance information for this record started
D2	Date/time when data collection ended	-	The date/time when measurements of performance information for this record ended
D3	Process serial number	-	The serial number for the process started by IJServer
D4	Process ID	-	The process ID for the IJServer being measured
D5	Data source name	-	Data source name
D6	Current number of physical connections	Number	Number of physical connections that were established when the log was taken (*1)

Item number	Item name	Unit	Content
D7	Minimum number of physical connections	Number	The minimum number of simultaneous connections to the database during the measurement period (*1)
D8	Maximum number of physical connections	Number	The maximum number of simultaneous connections to the database during the measurement period (*1)
D9	Upper limit for the number of physical connections	Number	The upper limit for the number of physical connections to the database that can be established (*1)
D10	Current number of connections used	Number	The number of connections being used when the log was taken (*1)
D11	Minimum number of connections used	Number	The minimum number of pooled connections used simultaneously during the measurement period (*1)
D12	Maximum number of connections used	Number	The maximum number of pooled connections used simultaneously during the measurement period (*1)
D13	Cumulative number of times connection wait occurred	Times	Number of times connection wait occurred during the measurement period (connection wait timeouts are not included) (*2)
D14	Average connection waiting time	Milliseconds	Average waiting time when connection wait occurred during the measurement period (connection wait timeouts are not included) (*2)
D15	Minimum connection waiting time	Milliseconds	The minimum waiting time when connection wait occurred during the measurement period (connection wait timeouts are not included) (*2)
D16	Maximum connection waiting time	Milliseconds	The maximum waiting time when connection wait occurred during the measurement period (connection wait timeouts are not included) (*2)
D17	Current number of threads waiting for connection	Number	The number of threads waiting for a connection when the log was taken (*2)
D18	Minimum number of threads waiting for connection	Number	The minimum number of threads waiting for a connection during the measurement period (*2)
D19	Maximum number of threads waiting for connection	Number	The maximum number of threads waiting for a connection during the measurement period (*2)
D20	Number of connection wait timeouts	Times	The number of times connection wait timeouts occurred during the measurement period (*2)
D21	Number of physical connections established	Times	The number of times that physical connections to the database were established during the measurement period (*2)
D22	Maximum establishment time for physical connections	Milliseconds	The maximum time taken for a pooled connection to establish a physical connection to the database during the measurement period (*2)
D23	Number of times connections were closed due to idle timeouts	Times	The number of times that physical connections were closed due to idle timeouts during the measurement period (*2)
D24	Number of times connections were closed because of exceptions	Times	The number of times that physical connections were closed because an exception occurred when connections were established or closed during the measurement period (*2)

Item number	Item name	Unit	Content
D25	Number of times connections were acquired	Times	The number of times that applications acquired connections by executing the getConnection method during the measurement period (*3)
D26	Number of times connections were closed	Times	The number of times that applications released connections by executing the close method during the measurement period (*3)
D27	Number of communication wait timeouts	Times	Number of times communication wait timeouts occurred during the measurement period (*3)
D28	Average usage time per connection	Millisecon ds	The average time that connections were used by applications during the measurement period (*3)
D29	Minimum connection usage time	Millisecon ds	The minimum time that connections were used by applications during the measurement period (*3)
D30	Maximum connection usage time	Millisecon ds	The maximum time that connections were used by applications during the measurement period
D31	Number of connection usage timeouts	Times	The number of times that connection usage timeouts occurred during the measurement period (*3)

*1 This value is output if Interstage is pooling JDBC connections or if the database is Oracle 10g or later and the JDBC driver is pooling JDBC connections. Otherwise, a hyphen ("-") is output.

*2 This value is output only if Interstage is pooling JDBC connections. A hyphen ("-") is output if the JDBC driver is pooling JDBC connections.

*3 Connection information used in container control processing that runs before and after the application is also displayed.

Note: No performance information is output for unused data sources.

3) Transaction Information

Item number	Item name	Unit	Content
D1	The date/time when data collection started	-	The date/time when measurements of performance information for this record started
D2	Date/time when data collection ended	-	The date/time when measurements of performance information for this record ended
D3	Process serial number	-	The serial number for the process started by IJServer
D4	Process ID	-	The process ID for the IJServer being measured
D5	Total number of transactions executed	Number	The number of J2EE transactions executed by applications during the measurement period (*1)
D6	Number of committed transactions	Number	The number of J2EE transactions that applications committed during the measurement period (*1)
D7	Number of rolled back transactions	Number	The number of J2EE transactions that applications rolled back during the measurement period (*1)
D8	Average transaction time	Millisecond s	The average time for J2EE transactions executed by applications during the measurement period (*1)
D9	Minimum transaction time	Millisecond s	The minimum time for J2EE transactions executed by applications during the measurement period (*1)
D10	Maximum transaction time	Millisecond s	The maximum time for J2EE transactions executed by applications during the measurement period (*1)

Item number	Item name	Unit	Content
D11	Current number of transactions executing	Number	The number of transactions currently executing when the log was taken (*1)
D12	Minimum number of transactions executing simultaneously	Number	The minimum number of J2EE transactions that were executed simultaneously during the measurement period (*1)
D13	Maximum number of transactions executing simultaneously	Number	The maximum number of J2EE transactions that were executed simultaneously during the measurement period (*1)

*1 Transaction information controlled by the container is also displayed.

4) Servlet Container Information

Item number	Item name	Unit	Content
D1	The date/time when data collection started	-	The date/time when measurements of performance information for this record started
D2	Date/time when data collection ended	-	The date/time when measurements of performance information for this record ended
D3	Process serial number	-	The serial number for the process started by IJServer
D4	Process ID	-	The process ID for the IJServer being measured
D5	Current number of threads	Number	Number of threads being used when the log was taken
D6	Minimum number of threads used simultaneously	Number	Minimum number of threads used simultaneously during the measurement period (*1)
D7	Maximum number of threads used simultaneously	Number	Maximum number of threads used simultaneously during the measurement period (*2)
D8	Current total number of pooled threads	Number	Number of threads being pooled when the log was taken (*3)
D9	Minimum total number of pooled threads	Number	The minimum number of threads being pooled during the measurement period (*3)
D10	Maximum total number of pooled threads	Number	The maximum number of threads being pooled during the measurement period (*3)

*1 One thread is used to receive requests from clients, so a value of 1 will be output for the minimum number of simultaneously used threads even if there are no requests from clients.

*2 The ServerSocket for the Servlet container that receives requests from clients comes out of standby mode every few seconds, and then goes back to waiting for requests from clients. As a result, a value of 2 will be output for the maximum number of simultaneously used threads even when there are no requests from clients.

*3 If there are no requests from clients for a certain period of time, excess processing threads will be destroyed according to the value that has been set for the maximum number of standby threads for the Servlet container. At this point, the thread that receives requests from clients will not be destroyed, so the minimum value will be one more than the value for the maximum number of standby threads.

5) EJB Container Information

Item number	Item name	Unit	Content
D1	The date/time when data collection started	-	The date/time when measurements of performance information for this record started
D2	Date/time when data collection ended	-	The date/time when measurements of performance information for this record ended
D3	Process serial number	-	The serial number for the process started by IJServer
D4	Process ID	-	The process ID for the IJServer being measured
D5	Maximum number of message-driven beans that can be simultaneously processed	Number	The maximum number of threads that can be pooled by message-driven beans
D6	Current size of the thread pool for message-driven beans	Number	The number of threads that had been pooled by message-driven beans when the log was taken
D7	Minimum size of the thread pool for message-driven beans	Number	The minimum number of threads that have been pooled by message-driven beans
D8	Maximum size of the thread pool for message-driven beans	Number	The maximum number of threads that have been pooled by message-driven beans
D9	Current number of idle message-driven bean threads	Number	The number of unused threads that had been pooled by message-driven beans when the log was taken
D10	Minimum number of idle message-driven bean threads	Number	Minimum number of unused threads that had been pooled by message-driven beans
D11	Maximum number of idle message-driven bean threads	Number	Maximum number of unused threads that had been pooled by message-driven beans
D12	Number of idle timeouts for message-driven beans	Number	The number of threads that had been pooled by message-driven beans but then destroyed because of idle timeouts

How to Evaluate and Respond to Performance Information

1) Java VM Information

Item number	Evaluation method	Response/Action
1	Heap usage is tending to increase even though garbage collection is being performed in between.	There may be a memory leak. Review the server applications by checking for things such as objects that are still being looked up even though they are no longer needed. Alternatively, if a very long time has been set for session timeouts for Web applications, review whether this setting is appropriate.

Item number	Evaluation method	Response/Action
2	The minimum heap usage stays close to the upper limit for the heap size.	The heap may run out of space if operations continue. Review IJServer settings by increasing the upper limit that has been specified for the heap size, for example.
3	The minimum "Perm" area usage stays close to the upper limit for the size of the "Perm" area.	The "Perm" area may run out of space if operations continue. Review IJServer settings by increasing the upper limit that has been specified for the size of the "Perm" area, for example.
4	Garbage collection occurred too frequently.	Application processing performance may drop as a result of garbage collection being performed too frequently. If the minimum usage for the Java VM heap is close to the upper limit, garbage collection may occur often because there is not enough heap space. Review IJServer settings by increasing the upper limit for the heap size, for example.
5	The total processing time for garbage collection sometimes takes a long time.	The CPU for the server running IJServer may be working at high load continuously. Either reduce the load on the CPU (by stopping unnecessary applications, for example) or review whether the CPU performance for the server running IJServer is appropriate. Also, if the size of the heap is increased, more objects will be released with each garbage collection, and so the processing time will tend to take longer each time garbage collection is performed. Check the heap usage status, and consider reducing the upper limit for the heap size if there is enough spare room to do so.

Note: Refer to "3.2.2 Size of Java VM Heap Area" and "3.2.3 Number of Garbage Collections" in this chapter, and "Tuning Methods" in the "JDK/JRE Tuning" chapter for information about tuning Java VMs.

2) Data Source Information

Item number	Evaluation method	Response/Action
1	The maximum number of physical connections is close to the upper limit.	Connection wait may occur if operations continue. Review whether the specification for the maximum number of connections is appropriate.
2	The maximum number of connections used is much smaller than the minimum number of physical connections	Because fewer connections are being used than are being pooled, connections may be being pooled unnecessarily. Review the values that have been specified for the number of pre-opened connections and idle connection timeouts. The number of constantly connected physical connections can be reduced by reducing the number of pre-opened connections. Unnecessary connections can also be released by reducing the idle connection timeout value.
3	The cumulative number of times connection wait occurred is too high	Application processing performance may be dropping because connection wait occurred too frequently. Review whether the specification for the maximum number of connections is appropriate.
4	The average connection waiting time is long.	Application processing performance may be dropping because connection wait occurred too frequently. Review whether the specifications for the maximum number of connections and the connection waiting time are appropriate.
5	The minimum connection waiting time is short and	Application processing performance may be inconsistent because the connection waiting time varies with the timeframe.

Item number	Evaluation method	Response/Action
	the maximum connection waiting time is long.	Consider counter measures such as reducing the connection waiting time settings and returning an error if the waiting time becomes too long.
6	The maximum number of threads waiting for connections is increasing.	The number of connections simultaneously used by the application is increasing, so application performance may drop if operations continue. Review whether the specification for the maximum number of connections is appropriate.
7	The number of connection wait timeouts is increasing.	The number of applications producing errors due to connection wait timeouts is increasing. Review whether the specifications for the maximum number of connections and the connection waiting time are appropriate.
8	Many physical connections are established. Many connections are closed due to idle timeouts.	Because physical connections are being released due to idle timeouts, the number of attempts to establish physical connections is increasing, which may cause the processing performance of the application to drop. Review whether the idle timeout value is appropriate. Constant physical connections can also be established by specifying the number of pre-opened connections.
9	Many physical connections are being established. Many connections are closed because exceptions are occurring.	Because physical connections are being released due to idle timeouts, the number of attempts to establish physical connections is increasing, which may cause the processing performance of the application to drop. Eliminate the cause of the exception by reviewing the application.
10	There is a large difference between the number of connections acquired and the number of connections closed.	Some connections may not be being closed. Review the application.
11	The average usage time for connections is long.	Database access processing may be causing the processing performance of the application to drop. Review the application and the database settings.
12	There is a large difference between the minimum connection usage time and the maximum connection usage time.	Application processing performance may have become inconsistent for some reason to do with database access processing. Review the application and the database settings.
13	Many communication wait timeouts are occurring.	The application processing performance may have dropped because of database access processing. Check the database processing time and whether processing is hanging.

Note: For details on how to tune JDBC data sources, refer to "3.2.5 JDBC Connection" and "3.2.6 Statement Cache Function".

3) Transaction Information

Item number	Evaluation method	Response/Action
1	The total number of transactions executed and the maximum number of transactions executing simultaneously are increasing, and the average	As a result of the increased number of transactions executing simultaneously, the processing performance of the application may have dropped because of exclusive control for the database or for some other reason. Either stabilize processing performance by reducing the IJServer specifications for the number of transactions that can be processed simultaneously or review the processing performance of the server running IJServer.

Item number	Evaluation method	Response/Action
	transaction time is getting longer.	
2	The total number of transactions executed does not change but there are a large number of transactions executing simultaneously and the maximum transaction time is long.	As a result of transaction processing being concentrated during certain time periods, application processing may have been delayed because of exclusive control for the database or for some other reason. Either stabilize processing performance by reducing the IJServer specifications for the number of transactions that can be processed simultaneously or review the processing performance of the server running IJServer.
3	There is a large difference between the minimum connection usage time and the maximum connection usage time.	Application processing performance may have become inconsistent for some reason relating to database access processing. Review the application and the database settings. Otherwise, the CPU for the server running IJServer may be experiencing high load temporarily. Either reduce the load on the CPU (by stopping unnecessary applications, for example) or review whether the CPU performance for the server running IJServer is appropriate.
4	More transactions are committed than rolled back.	Application processing performance may have dropped because the load associated with commit processing is generally higher than for rollback processing. Review the application to see if unnecessary commit processing is being executed.

Note: Refer to "3.2.4 Transaction Isolation Level" for more information about tuning JDBC data sources.

4) Servlet Container Information

Item number	Evaluation method	Response/Action
1	The number of active processes is close to the total number for a long time.	[If the CPU is running well within capacity] - The number of simultaneous processes for the Servlet container may be too small. - Increase the number of simultaneous processes for the Servlet container. However, if there is no spare capacity in the heap usage for the Java VM, an OutOfMemoryError may occur. In this case, take appropriate measures (such as increasing the maximum value for Java VM heap usage) at the same time. - There may be a conflict between threads due to a factor within the process. Increase the number of concurrent processes for the Work Unit.
		[If the CPU does not have spare capacity] - The processing capacity of the server may have been exceeded. Consider adding more servers or replacing the server with one with better performance.

Note: For details on how to tune Servlet containers, refer to "3.3 Tuning Servlet Containers".

5) EJB Container Information

Item number	Evaluation method	Response/Action
1	The maximum number of message-driven beans processed simultaneously	Message-driven beans may have to wait to acquire threads if operations continue.

Item number	Evaluation method	Response/Action
	is close to the maximum size of the thread pool for message-driven beans.	Review whether the specification for the number of message-driven beans that can be processed simultaneously is appropriate.
2	There is a large difference between the minimum size of the thread pool for message-driven beans and the maximum size of the thread pool for message-driven beans, and idle timeouts for message-driven beans occur often.	Application processing performance may have dropped because of thread generation processing for message-driven beans. Review whether the specification for the minimum number of message-driven beans that can be processed simultaneously is appropriate.
3	The maximum number of idle threads for message-driven beans is large.	Memory resources may be being consumed unnecessarily because the specification for the minimum number of message-driven beans that can be processed simultaneously is large. Review whether the specification for the minimum number of message-driven beans that can be processed simultaneously is appropriate.

Note: For details on how to tune EJB Containers, refer to "3.4 Tuning of EJB Containers".

3.2 Tuning of IJServer

The points to be considered when tuning IJServer are listed below.

- 3.2.1 Process Concurrency
- 3.2.2 Size of Java VM Heap Area
- 3.2.3 Number of Garbage Collections
- 3.2.4 Transaction Isolation Level
- 3.2.5 JDBC Connection
- 3.2.6 Statement Cache Function
- 3.2.7 Monitoring Information
- 3.2.8 Notes on Linkage with IPCOM

Note that the tuning information described here is valid for both Servlet and EJB containers.

3.2.1 Process Concurrency

Two or more processes can activate one IJServer. Doing so can distribute the load.

IJServer process concurrency can be specified using the Interstage Management Console WorkUnit settings, or the *isj2eeadmin* command.

The process concurrency of the IJServer can be defined in the WorkUnit setting on the Interstage Management Console.

For details about the *isj2eeadmin* command, refer to "isj2eeadmin" in the Reference Manual (Command Edition).

3.2.2 Size of Java VM Heap Area

The Java VM option can be specified in the WorkUnit setting on the Interstage Management Console, or *isj2eadmin* command. Doing so can change the parameters of the Java VM in which the IJServer runs.

The size of the Java VM heap area can be changed using parameters. An example of the maximum size of the heap area under JDK 6 is shown below.

The default maximum size of the heap area varies depending on the Java VM. Refer to the JDK document for details. If a java.lang.OutOfMemoryError occurs frequently, increase the maximum size of the Java VM heap area using this definition item.

Sample

Setting the maximum size of Java VM heap area to 1024 megabytes

```
-Xmx1024m
```

Interstage provides a proactive monitoring function that posts an alarm message for heap area problems.

If work is continued after an alarm message is output, a problem such as a memory shortage or response deterioration may occur. To solve such a problem, tune the IJServer, based on the resource shortage information provided by the alarm message.

The cause of the JavaVM error is a space shortage in the heap area or Perm area. To avoid the problem, increase the current upper limit by 20% and restart the operation. If an alarm message is still produced, increase the upper limit by another 20%. Repeat this operation until no alarm message is posted. To construct a system with stable operation, repeat the tuning until the alarm message does not appear.

3.2.3 Number of Garbage Collections

In an IJServer, automatic garbage collection using the Java RMI function runs at 1-minute intervals (by default).

To set the interval for automatic garbage collection using the RMI function, in the Interstage Management Console, click 'WorkUnit', and click the [Environment Settings] tab. For the Java VM options of [WorkUnit Settings], specify "-Dsun.rmi.dgc.client.gcInterval= occurrence interval" and "-Dsun.rmi.dgc.server.gcInterval= occurrence interval" and execute tuning. Specify the value for the garbage collection interval in microseconds.

If this is not specified, '-Dsun.rmi.dgc.client.gcInterval=3600000' and '-Dsun.rmi.dgc.server.gcInterval=3600000' are set automatically when IJServer starts up.

The *isj2eeadmin* command can also be used to make the settings.

If the number of garbage collections is not reduced by tuning the interval for automatic garbage collection using the RMI function, it is possible that the heap size for the Java VM is insufficient. Tuning the Java VM heap area size may cause a reduction in heap size. For details, refer to "3.2.2 Size of Java VM Heap Area".

3.2.4 Transaction Isolation Level

The Transaction Isolation Level (Isolation Level) is the exclusive consistency level for the database. When an EJB application accesses the database, the appropriate Isolation Level must be selected to improve multiprocessing of the EJB application. Select one of the following Isolation Levels, as required. For details of the Isolation Level, refer to the relevant database manual.

- Transaction-read-committed
- Transaction-read-uncommitted
- Transaction-repeatable-read
- Transaction-serializable

The Isolation Level is valid from when the UserTransaction.begin() method is issued to when the UserTransaction.commit() or UserTransaction.rollback() method is issued.

3.2.4.1 Setting Method

Use the *isj2eeadmin* command or Interstage Management Console to set the Isolation Level.

[When Oracle is used as DBMS]

An error "ORA-8177: Access to this transaction cannot be serialized." Notifies the user that a serialized transaction cannot be guaranteed. For example, when multiple users update the same table at the same time, even though Transaction-serializable is set for the transaction isolation level.

When an error "ORA-8177" occurs with Transaction-serializable set for the transaction isolation level, "Abnormal end" should not be forced on the application system. Instead,"Retry" should be used after the transaction's rollback.

If the transaction isolation level is Transaction-read-committed (Oracle's default), the error "ORA-8177" does not occur. If the Transaction-serializable setting is not specifically required, set Transaction-read-committed. Using this setting enhances the concurrent execution capability and prevents the "ORA-9177" error from occurring.

3.2.5 JDBC Connection

The following JDBC connections are described in this section:

- Connection pooling types
- Connection pooling tuning parameters

When a JDBC data source from the Interstage JNDI service provider is used, the JDBC connection is pooled and re-used.

Connection Pooling Types

The connection pooling types are as follows:

- Interstage connection pooling
- JDBC driver connection pooling

The characteristics of each are as follows:

Interstage Connection Pooling

Detailed connection pooling settings can be made in the Interstage Management Console to control Interstage connection pooling. The pooled information can be referenced using the Interstage Management Console Monitor function. Connections are also pooled by Interstage if Oracle's distributed transactions are used.

JDBC Driver Connection Pooling

To control connection pooling in the JDBC driver, use the JDBC driver functions to make the connection pooling settings. For details, refer to the JDBC driver manual.

The monitor information for JDBC data sources that can be looked up using the Interstage Management Console in order to perform pooling control on the JDBC driver side is either both "connection pool information" and "information about connections established from applications" for Oracle databases or only "information about connections established from applications" for non-Oracle databases.

Refer to "JDBC (Database) Connections" in the J2EE User's Guide for an overview of the connection pooling function.

The following table shows how connection pooling works for each database.

DB	Connection pooling
Symfoware	If [Use Interstage connection pooling] has been selected for the data source type, connection pooling is performed by Interstage. If [Use Symfoware connection pooling] has been selected, connection pooling is performed by the JDBC driver.
Oracle	If [Use Interstage connection pooling] has been selected for the data source type, connection pooling is performed by Interstage. If [Use Oracle connection pooling] has been selected, connection pooling is performed by the JDBC driver.
SQL Server	Connection pooling is performed by Interstage regardless of the data source type.
Generic definition	If the data source class implements the java.sql.ConnectionPoolDataSource interface, connection pooling is performed by Interstage. Otherwise, whether the JDBC driver will perform connection pooling depends on the implementation of the JDBC driver.

Note

The platform determines whether the database is supported.

For details, refer to "Database" in the "Supported Software" chapter of the Product Notes.

Connection Pooling Tuning Parameters

The tuning parameters are set in the IJServer environment database connection settings. The parameters that can be set depend on the database type and data source type, and are shown in the following table.

	Oracle			Symfoware		SQL Server	Generic definition	
	Interstage (*1)	JDBC (*2)	XA (*3)	Interstage (*1)	JDBC (*2)	Interstage (*1)	Interstage (*1)	JDBC (*2)
Transaction Isolation Level	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Pre-existing Connections	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Maximum Connections	Yes	Yes	Yes	Yes	No	Yes	Yes	No
Connection Timeout	Yes	Yes	Yes	Yes	No	Yes	Yes	No
Idle Timeout	Yes	Yes	Yes	Yes	No	Yes	Yes	No
Connection Monitoring Time	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Statement Cache Size	No	Yes	No	Yes	No	No	No	No
Statement Automatic Close	No	No	No	Yes	No	No	No	No
Communication Wait Time	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Reconnect on Failure?	Yes	No	No	Yes	No	Yes	No	No
Interval	Yes	No	No	Yes	No	Yes	No	No
Retry Count	Yes	No	No	Yes	No	Yes	No	No

Yes: Valid

No: Invalid

*1 If Interstage connection pooling is used

*2 If JDBC connection pooling is used

*3 If distributed transactions are used

*4 If the data source class implements the java.sql.ConnectionPoolDataSource interface

*5 If the data source class implements the java.sql.ConnectionPoolDataSource interface (Whether connection pooling is performed by JDBC depends on the implementation of the JDBC driver.)

The following table lists the parameters that can be set using the Interstage Management Console or *isj2eeadmin* Command. The parameters are set in the IJServer environment settings for each database.

Table 3.1 Parameters Set Using Interstage Management Console

Parameter	Explanation	Value
Pre-existing Connections (*1) (*5)	The connections required for operation can be obtained in advance during startup. Doing so can obtain the same processing speed from the very beginning of connection as that obtained for the second and subsequent connections. If the database type is Oracle and the data source type is [Use Oracle connection pooling and if InitialLimit has been specified in the connection cache properties for the implicit connection cache and there is at least one pre-opened connection, either the pre-opened connection count or the value of InitialLimit will take effect, whichever is larger. (*4)	Max value: 2147483647 Min value: 0 Default settings value: 0
Maximum Connections	The memory space can be suppressed by suppressing the maximum number of connections.	Max value: 2147483647

Parameter	Explanation	Value
	If a connection request is received while J2EE applications use all of the connections requests that can be set up, the container waits for a connection to be returned to the pool within the period specified for Connection Timeout. If a connection is returned to the pool, the container allocates it. If no connection is returned, the container returns an SQL Exception. Note the following points if the database type is Oracle and the data source type is "Use Oracle connection pooling]: - This item will be the MaxLimit settings in the connection cache properties for the implicit connection cache. - Even if MaxLimit is specified in the cache properties when JDBC resource definitions are made, the value specified will not take effect. Specify this value using this item.	Min value: 1 Default settings value: 64
Connection Timeout	The waiting time specified until a connection is returned to the pool when a connection request is received and the maximum number of connections in the J2EE application is in use. If the time is exceeded and a connection is not returned, an SQLException is returned. Timeout monitoring will not be performed if 0 is specified. An SQLException will be returned as soon as a connection wait state occurs. Note the following points if the database type is Oracle and the data source type is "Use Oracle connection pooling]: - This item will be the ConnectionWaitTimeout settings in the connection cache properties for the implicit connection cache. - Even if ConnectionWaitTimeout is specified in the cache properties when JDBC resource definitions are made, the value specified will not take effect. Specify this value using this item.	Max value: 2147483647 Min value: 0 Default settings value: 5 (Unit: Seconds)
Idle Timeout	Abandoning unused connections according to timeout can free memory that has been used wastefully. Note that pre-opened connections are excluded from the idle timeout. Note the following points if the database type is Oracle and the data source type is [Use Oracle connection pooling]: - This item will be the InactivityTimeout settings in the connection cache properties for the implicit connection cache. - Even if InactivityTimeout is specified in the cache properties when JDBC resource definitions are made, the value specified will not take effect. Specify this value using this item. - The time interval for checking timeouts is the value set for PropertyCheckInterval in the connection cache properties of the implicit connection cache. Define PropertyCheckInterval in the data source environment settings. Refer to the Oracle manuals for more information about properties. - If Oracle connection pooling is used, connection destruction that occurs when there is an idle timeout is performed using the Oracle JDBC driver. Connections that use pre-connect are also targets of idle timeout.	Max value: 2147483647 Min value: 0 Default settings value: 600(Unit: Seconds)
Connection Monitoring Time(*2)	Monitors the usage time for connections that are acquired by applications before transactions start until they close.	Maximum value: 2147483647 Minimum value: 0

Parameter	Explanation	Value
	If a connection does not close even though the specified time has elapsed, a warning message will be output to either the container log or the system log. If 0 is specified, connection usage time will not be monitored.	Default value: 60 (Units: minutes)
Close timed out connections	When connection usage monitoring time is used (when a value of 1 or more is specified for the connection usage monitoring time), turn this parameter ON if connections that have timed out are to be released automatically. If this parameter is turned OFF, connections that have timed out will not be released automatically.	- ON - OFF (default)
Statement Cache Size	Specifies the size of the statement cache. Specify the size of the area for caching statements that applications execute (without closing these statements) If 0 is specified, statements will not be cached. Refer to "3.2.6 Statement Cache Function" for more information about the statement cache function.	Maximum value: 32000 Minimum value: 0 Default value: 10
Statement Automatic Close	Specifies whether the JDBC driver should automatically close statements when the statement cache function is used (when a value of 1 or more is specified for the size of the statement cache).	- Enabled - Disabled (default)
Communication Wait Time (*2)	Monitors the time from when the following SQL statements start executing until they finish executing: - java.sql.Statement - execute(String) - execute(String, int) - execute(String, int[]) - execute(String, String[]) - executeBatch() - executeQuery(String) - executeUpdate(String) - executeUpdate(String, int) - executeUpdate(String, int[]) - executeUpdate(String, String[]) - java.sql.PreparedStatement - execute() - executeQuery() - executeUpdate() A warning message will be output to the container log or the system log if an SQL statement has not returned after a certain amount of time has elapsed. If 0 is specified, communication waiting time will not be monitored.	Maximum value: 2147483647 Minimum value: 0 Default value: 400 (Units: seconds)
Output SQL statements in the log	Turn this parameter ON to output SQL statements in the warning messages that are output to logs when communication waiting times are used (if a value of 1 or more is specified for the communication waiting time). If this parameter is turned OFF, SQL statements will not be output.	- ON - OFF (Default)

Parameter	Explanation	Value
Reconnect on Failure? (*2)	Specify whether to use the automatic reconnection function for the JDBC connection (*3). To use the automatic reconnection function, check whether the pooled JDBC connection can be used. If the connection cannot be used, reconnect automatically to DBMS.	- Enabled - Disabled (default)
Interval (*2)	Specify the time interval until reconnection is performed if the automatic reconnection function for the JDBC connection (*3) fails to use the pooled JDBC connection or connect to DBMS. The specified value is only valid when [Yes] is selected for "Reconnect on Failure".	Max value: 2147483647 Min value: 1 Default value: 10 (Unit: second)
Retry Count (*2)	Specify the reconnection retry count to be used if the automatic reconnection function for the JDBC connection (*3) fails to use the pooled JDBC connection or connect to DBMS. The specified value is only valid when [Yes] is selected for "Reconnect on Failure".	Max value: 2147483647 Min value: 0 Default value: 10 (Unit: count)

*1 At startup of the IJServer to which a CMP2.0 EJB application has been deployed, it checks the maximum value for the DBMS identifier length. For this reason, only one connection to the DBMS is set up even if pre-opened connection optimization is not used. If Interstage pools connections, the connections are released after completion of start processing.

*2 Set up timeout values for connection usage monitoring times and communication waiting times by referring to the following calculation formula.

Maximum application processing time > Connection usage monitoring time > Communication waiting time

*3 With Symfoware, the functions of the Connection Manager can be used to perform the same kind of operations when the database server fails or the communication line fails. Refer to the Connection Manager User's Guide for more information about the Connection Manager.

*4 For Oracle, the following table shows the timing when pre-opened connections are acquired.

Oracle data source type and settings		Timing when pre-opened connections are acquired
Connection pooling performed by Interstage	A value of 1 or more set for the number of pre-opened connections in the database connection settings	When IJServer starts
	If this setting is not made	Connections are not established in advance
Connection pooling performed by Oracle	A value of 1 or more set for the number of pre-opened connections in the database connection settings However, if "InitialLimit" has also been set in the properties for the implicit connection cache, the number of pre-opened connections established will be the larger of these two values.	When IJServer starts
	The number of pre-opened connections is not set in the database connection settings (the default value is 0) and a value of 1 or more is set for "InitialLimit" in the properties for the implicit connection cache	The first time the getConnection method is invoked
	In all other cases	Connections are not established in advance

*5 If a value of 1 or more was set for the number of pre-opened connections, then when IJServer starts, there will be a connection to DBMS - therefore the target DBMS must already be running.

3.2.6 Statement Cache Function

The statement cache function can be used in the following cases:

- If the database type is Oracle and the data source type is [Use Oracle connection pooling]

  In this case, the statement cache function for Oracle 10g or later can be used.

  Note that this function cannot be used in a distributed transaction environment because distributed transactions cannot be used by [Use Oracle connection pooling] data sources

- If the database type is Symfoware and the data source type is [Use Interstage connection pooling]

Caching statements has the benefits of reducing overheads when SQL statements are analyzed or created and reducing the number of communications with the database.

The following statements are cached:

- PreparedStatement objects acquired using the Connection.prepareStatement(String) method
- PreparedStatement objects acquired using the Connection.prepareStatement(String, int, int) method
- PreparedStatement objects acquired using the Connection.prepareStatement(String, int) method
- PreparedStatement objects acquired using the Connection.prepareStatement(String, int[]) method
- PreparedStatement objects acquired using the Connection.prepareStatement(String, int, int, int) method
- PreparedStatement objects acquired using the Connection.prepareStatement(String, String[]) method
- CallableStatement objects acquired using the Connection.prepareCall(String) method
- CallableStatement objects acquired using the Connection.prepareCall(String, int, int) method
- CallableStatement objects acquired using the Connection.prepareCall(String, int, int, int) method

Tuning Method

Size of the Statement Cache

The size of the statement cache is set for the data source (to be used) by using either the Interstage Management Console or the *isj2eeadmin* command.

With the Interstage Management Console, set [IJServer] > [Settings] > [DB Connection Settings] > [Statement Cache Size]. Statements will not be cached if 0 is set for [Statement Cache Size].

Refer to the Reference Manual (Command Edition) for more information about the *isj2eeadmin* command.

If the size of the statement cache has been set using the connection options in the JDBC resource definition window and the cache size has also been specified in the database connection settings at the same time, the value for the cache size that has been specified in the database connection settings will take effect.

Settings Guidelines

Statements are cached separately for each connection, and connections are shared between all applications within the same IJServer process being executed.

As the number of statements issued by applications becomes larger than the size of the statement cache, more and more statements will be deleted from the cache. If a statement requested by an application has been deleted from the cache, overheads resulting from communicating with the database and analyzing the SQL statement will accrue in order to recreate the statement.

To reduce the number of times that statements are deleted from the cache, it is recommended that the size of the statement cache be equal or close to the total number of statements issued by the IJServer process. However, statement caching consumes machine resources. The machine specifications should be taken into account when setting the size of the statement cache.

For CMP applications that issue statements on the application server (container) side, add the total of the following values to the cache size rather than the number of statements issued by applications.

- For CMP 1.1 entity beans
    - Statement The number of statements calculated for each bean

      Four (insertion, deletion, updating and searching (primary keys))

      The number of methods with single item searches or multiple item searches
- For CMP2.0 entity beans
    - Statement The number of statements calculated for each bean

      Four (insertion, deletion, updating and searching (primary keys))

      The number of EJB QL queries defined in the deployment descriptor
    - The number of statements calculated for each relationship

      The number of relationships between CMP 2.0 entity beans that are one-way and 1-to-1

      (The number of relationships between CMP 2.0 entity beans that are two-way and 1-to-1) x 2

      (The number of relationships between CMP 2.0 entity beans that are one-way and 1-to-many or many-to-many) x 3

      (The number of relationships between CMP 2.0 entity beans that are two-way and 1-to-many or many-to-many) x 4

OPEN_CURSORS Settings (If Oracle is Used)

If the statement cache function is being used, statements will not be destroyed when connections close, and the number of statements simultaneously issued by a single connection will increase. As a result, the number of statements may exceed the "upper limit for the number of statements that can be simultaneously issued by a single connection (transaction) (hereafter referred to as "OPEN_CURSORS", with a default value of 50)". An SQLException will occur if the number of statements exceeds OPEN_CURSORS.

In this case, make settings so that OPEN_CURSORS is larger than or equal to the size of the statement cache. Refer to the Oracle manuals for information on how to set OPEN_CURSORS.

Deletion Triggers for Cached Statements

The triggers for deleting statements that have been cached are shown below. Also, if the number of SQL statements executed reaches the value set for the [Statement cache size], the treatment of any SQL statements that are then executed depends on the specification of the JDBC driver. (Statements that have been cached may be deleted, depending on the specification for the JDBC driver.) Refer to the JDBC driver manual for information about JDBC driver specifications.

- The time monitoring function for JDBC pooled connections

  Statements are cached separately for each connection instance (connection), and so if a connection object is released because it has remained idle in the connection cache for longer than is possible, the statement objects cached with that connection will also be deleted.
- Physical release of connections

  Statements are cached separately for each connection instance (connection), and so if connections become disabled (because the database has crashed, for example) when the connections have been pooled, cached statements will also be deleted when these connections are released.

3.2.7 Monitoring Information

The Interstage Management Console displays the operation information for the currently operating IJServer. The output information can be used to detect performance bottlenecks and check the effects of tuning.

The following information is output. Refer to the Interstage Management Console Help for details of the output information.

- JavaVM information
- Servlet container information
- Data source information
- Transaction information
- Queue information

- Message Driven Bean thread information

Note that if IJServer of V9.0 or later is being used, performance information can be output to log files at fixed intervals. Refer to "3.1 J2EE Monitor Logging Function" for more information.

3.2.8 Notes on Linkage with IPCOM

When IPCOM is used to separate IJServer and the Web server to execute load balancing for the operating system, a connection (thread) is required for fault monitoring.

In this case, when setting the number of simultaneous processing events for IJServer, take into account the number for monitoring as well as the actual number of simultaneous processing events. The settings are as follows:

- Number of simultaneous processing events = number of simultaneous processing events + 1 (for monitoring)

When setting the permanent number of IPCOM connections, set the actual number of simultaneous processing events (do not add the number for monitoring). Refer to the IPCOM manual for details of the permanent number of connections.

3.3 Tuning Servlet Containers

The point to be considered when tuning the Servlet container is the number of simultaneous processing tasks.

3.3.1 Number of Simultaneous Processing Tasks

Increasing the number of simultaneous processing tasks or the process concurrency can increase the execution concurrency of Web applications.

Increasing the number of simultaneous processing tasks can increase the execution concurrency per process, but may not always be effective as it also increases loads and resources. We recommend normally operating the system below the default values set up in the Interstage Management Console or using the *isj2eeadmin* command.

Tune the number of simultaneous processing tasks according to the number of JDBC connections and the number of EJBs (that are called from applications) that can be processed simultaneously.

Specify the number of simultaneous processing tasks in the Interstage Management Console "Servlet Container Settings" or using the *isj2eeadmin* command.

The following settings can be made for the number of simultaneous processing tasks:

- The initial value (increments)
- The maximum number of processing threads that are on standby (idle)
- The maximum value

If there are more standby processing threads (processing threads that have gone on standby and are not being used after their processing has completed) than the maximum number of standby processing threads, excess threads will be released at one-minute monitoring intervals.

This means that even if the load increases temporarily, server resources are released and saved if the load decreases.

3.3.2 Number of Connections

The following items relating to the number of connections can be set up.

- The maximum number of connections to the Servlet container for the Web server connector

  Specify this item using the Interstage Management Console.

  - If a Web server and a Work Unit are running on the same machine

    The Web server connector settings can be changed in the environment settings for the Work Unit

  - If Web servers and Work Units are not running on the same machine

    The environment settings can be changed for the Web server connector directly

Settings can also be made using the *isj2eeadmin* command.

- The maximum number of Servlet container connections

  Make this specification using the Servlet container settings in the Interstage Management Console.

  These settings can also be made using the *isj2eeadmin* command.

The number of client requests that the Servlet container can accept can be increased by increasing the values for these items.

If it is expected that there may be periods of temporary high load during operations, where more requests are received than the number of simultaneous processes for the Servlet container, the response times for the whole server can be prevented from deteriorating by reducing the number of simultaneous processes for the Servlet container and setting a large value for the number of connections (the maximum number of connections) to the Servlet container.

Set the number of connections according to the different requirements for the system being operated, as shown below.

- If requests that cannot be processed immediately because of high load on the Servlet container need to be processed normally, even if it takes a long time

Item	Setting value
Maximum number of connections for the Servlet container	Set an enough large value to process requests.
Restrictions on the number of connections to the Servlet container for the Web server connector	None (no restrictions)

- If requests that cannot be processed immediately due to high load on the Servlet container need to be notified to the client as an error without having the client wait for a long time

Item	Setting value
Maximum number of connections for the Servlet container	Set an enough large value to process requests
Restrictions on the number of connections to the Servlet container for the Web server connector	Set a number of connections that can be processed by the Servlet container

Notes

- If the number of connections to the Servlet container exceeds the maximum number of simultaneous Servlet container processes, the excess connections will not be kept alive, and so more sockets will be consumed than if the maximum number of simultaneous Servlet container processes was not exceeded.
- The value that has been set for the maximum number of connections is set in the backlog property for the socket used by the Servlet container.

  The range of valid values that can be set for the sockets backlog property depends on the operating system, and so not all values set for the maximum number of connections will necessarily take effect.

  Refer to the operating system documentation for information about the range of valid values for the backlog property of sockets.

  For Windows®, values larger than 200 are not valid.

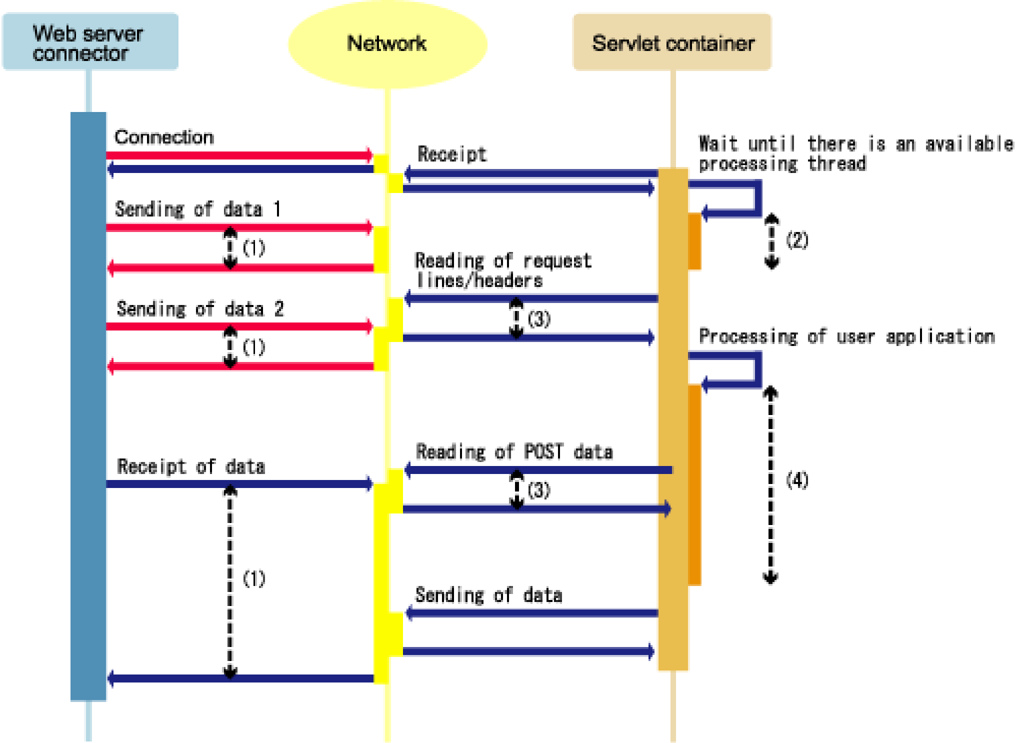
3.3.3 Timeouts

Timeout Monitoring Items

The following items relating to timeouts are monitored.

Monitoring item	Meaning
(1)	The time to wait when the Web server connector sends and receives data packets between Servlet containers.
(2)	The time to wait for request processing to start after a connection to the servlet container is established. (This value cannot be specified.)
(3)	Time to wait when the Servlet container receives data packets from the Web server connector.
(4)	Processing times for user applications

The following figure shows the monitoring timing for each of these items.



Timeout Setting Items

The following table shows the setting items relating to timeouts.

Setting item	Meaning
(a)	Transmission/reception timeout for the Web server connector Specify this timeout using the following items in the Interstage Management Console: - If Web servers and Work Units are running on the same machine [Work Unit] > [Web Server Connector Settings] > [Send/Receive Timeout] - If Web servers and Work Units are not running on the same machine [Web Server Connectors] > [Settings] > [Send/Receive Timeout]
(b)	Timeout for the Servlet container Specify this timeout using [Servlet Container Settings] > [Timeout] from the Interstage Management Console.
(c)	Maximum application processing time for Work Units Specify this time using [WorkUnit Settings] > [Maximum application processing Time] in the Interstage Management Console.

These settings can also be made using the *isj2eeadmin* command.

Note

- If the following problems occur frequently, it may be possible to avoid the problem by increasing the values of (a) and (c). If the problem still occurs even though the values of (a) and (c) have been increased, the processing capacity of the server may have been exceeded. Consider either installing additional servers or replacing the server with a server that has better performance.

  - IJServer32113 is output to the container log

  - IJServer12035 and IJServer12044 are output to the Web server connector log

- If the following problems occur frequently, it may be possible to avoid the problem by increasing the value of (b). If the problem still occurs, there may be a problem with the network or the client application. Resolve the problem that is causing these messages to be output.
  - A java.net.SocketTimeoutException is thrown to the Servlet when request data is read
  - IJServer12026, IJServer12027, IJServer12034 and IJServer12036 are output to the Web server connector log

How to Set Up Timeouts

Set up timeout values so that the following relationships hold

- Setting item (a) > Monitoring item (2) + Monitoring item (3) + Monitoring item (4)
- Setting item (b) > Monitoring item (3)
- Setting item (c) > Monitoring item (4)

Notes

- IJServer will be stopped forcibly in the following cases:
  - Setting item (b) > Setting item (c); and
  - [WorkUnit Settings] > [Forcefully end application on timeout?] has been set to "Forcefully stop all the running processes."; and
  - The "Sending of data 2" in the figure for timeout monitoring items does not complete before the time set for setting item (c) elapses.
- If the following settings are made, a Web server connector timeout will be notified to the Web browser.

  Setting item (a) < Monitoring item (2) + Monitoring item (3) + Monitoring item (4)

  However, the Servlet container cannot detect that the Web server connector has timed out, so no error will be detected on the Servlet container side if the application continues to execute and processing completes within the time set for monitoring item (3).
- The time for monitoring item (4) includes the time taken to read POST data.

3.4 Tuning of EJB Containers

The points to be considered when tuning the EJB container are as follows:

3.4.1 Number of Simultaneous Processing Tasks

Notes

The simultaneous processing count is equal to the number of executions that are run simultaneously for each process. The number of client requests that IJServer can process at the same time, depends on the process concurrency and simultaneous processing count. If the process concurrency is set to 2, and the simultaneous processing count is set to 16, a total of 32 tasks (2 x 16 = 32) can be processed concurrently. If requests from clients exceed the number of requests being able to be processed, they are placed in a queue.

- By changing the simultaneous processing count, the execution concurrency for each process can be raised. However, this may not be effective because the resources (CPU use rate and memory volume) used will increase as the simultaneous processing count increases.
- If there is a resource shortage, the response can be stabilized by putting the latest requests into the queue.

Task distribution to processes

- When the process concurrency is set to 2 or more, tasks from clients are distributed over optimum processes with a few EJB objects. If the number of EJB objects of one process is the same as that of another, the tasks are distributed over specific processes.
- When multiple EJB applications are deployed, the tasks are distributed over the processes of the deployed EJB applications with a small total number of EJB objects.
- If all threads of a process with a small total number of EJB objects are used, the tasks are distributed to the process with the next small total number of EJB objects.

The timing for creating and deleting EJB objects according to the EJB application type is shown in the table below.

The EJB container determines the number of EJB objects created in each process and distributes the request to the appropriate process.

EJB application type	Timing for creating EJB objects	Timing for deleting EJB objects
STATEFUL Session Bean	When the 'create' method is executed	When the 'remove' method is executed When Idle Timeout occurs When IJServer is stopped
STATELESS Session Bean	One EJB object is created when IJServer starts up	When IJServer stops
Entity Bean	Execution Create method Execution finder method	When the 'remove' method is executed When the Entity Bean EJB object times out When IJServer is stopped
Message-driven Bean	- (EJB objects are not created because Message-driven Bean does not have the Remote interface).	-

Table 3.2 Recommended Values

Definition item	Default value	Explanation
Minimum value	16	Number of threads that the EJB container can process simultaneously from the very beginning after the IJServer is started. When the number of threads requested by the client exceeds the minimum value, the available number of threads is automatically increased for operation.
Maximum value	64	Maximum number of threads that can be processed simultaneously. If the number of threads requested by the client exceeds the maximum value, the excess request is queued.

3.4.2 Session Bean

To use resources effectively, make the following settings for the Session Bean.

3.4.2.1 Selecting STATEFUL or STATELESS

Using a STATELESS Session Bean suppresses the number of times memory resources and objects are generated and accordingly improves the processing performance.

The differences between STATEFUL and STATELESS are listed below.

Table 3.3 STATEFUL and STATELESS Beans

	STATEFUL	STATELESS
Interactive status	Because access is made to the same object from "create" to "remove," the interactive status with the client can be maintained.	Because the interactive status with the client is not maintained, the client must retain information.

	STATEFUL	STATELESS
Transaction	The synchronization function of the Session Bean can be used for synchronization with transactions.	A transaction must be completed within one method.
Performance	An EJB object is generated for each client and therefore more memory space is used than STATELESS.	Memory usage can be suppressed because the same instance and EJB object are used for multiple clients. The object generation frequency is also suppressed.

3.4.2.2 Idle-time Monitoring Function for STATEFUL Session Bean

If an object created with the create method ends without executing the remove method, the remaining object is automatically removed. This function can thus prevent unnecessary consumption of memory.

The default value is 30 minutes.

3.4.2.3 Maximum Number of Times a Session Bean Can Execute the Create Method

The number of times the create method can be executed can be changed according to the high load execution environment.

If the value calculated from the following expression exceeds the default value of 1,024, change the maximum number of times the Session Bean can execute the create method.

- (Number of client application processes) x (average number of execution threads per process)

  * Use the Interstage Management Console to set the number of times the create method can be executed.

Note

If an object created with the create method is not deleted with the remove method, the object remains and consumes memory space until a timeout is detected by the idle-time monitoring function. Set an adequate value for the maximum number of times the create method can be executed.

Table 3.4 Settings

Parameter	Setting
Initial setting value	1024
Minimum value	1
Maximum value	2147483647

3.4.2.4 Creating STATELESS Session Bean Startup Instances

STATELESS Session Bean instances can be created to start up when IJServer starts up so that the instance creation time is eliminated when the access is made and processing performance is improved.

In the Interstage Management Console, click [System] > [WorkUnit] > 'WorkUnit Name' > 'EJB Module' > 'EJB Application Name' > [Application Environment Definition] > [Interstage Additional Settings]. Make the settings here. The default is 'No'.

Parameter	Setting
Minimum value(default)	0
Maximum value	2147483647

Note

- If the initial start instance number is increased, the heap amount will also increase.
- If [1] or more is set for the initial start instance number, the operations shown below and access are not possible from the 'setSessionContext' or 'ejbCreate' methods.
  - The javax.ejb.TimerService method
  - The javax.ejb.Timer method
  - Access to other EJB applications

    - Access to the resource manager (such as the database)

- Instances cannot be deleted once they are created. For this reason, make sure that the initial start instance number does not exceed the maximum number of simultaneous instances. Calculate the maximum number of simultaneous instances according to the following formula:
    - The IJServer type is [Web and EJB Applications run in the same Java VM]Number of simultaneous instances set in the Servlet container settings + Maximum number of simultaneous Message-driven Bean instances
    - The IJServer type is [Web and EJB Applications run in separate Java VMs] or [EJB Applications Only]Maximum number of simultaneous IIOP call processes set in the EJB container settings + Maximum number of simultaneous Message-driven Bean instances

3.4.3 Entity Bean

3.4.3.1 Notes on Calling an Entity Bean

An Entity Bean frequently executes a method to obtain record information. If an Entity Bean is called from outside of processes, IIOP communication occurs frequently and performance is deteriorated.

Fujitsu recommends calling an Entity Bean from an application within the same IJServer.

3.4.3.2 Number of Instances

Instances are cached inside the transaction. If there is no instance in the pool, an instance's record data (from the same transaction), is saved to the database and the instance is reused to store another record's data. If the instance is frequently reused, there will be an increase in the number of times the database is accessed. This may affect performance so the number of instances must be set with the performance considered. The number of instances is related to the number of database records to be retrieved and the number of clients to be connected at the same time. An effective value can be obtained by multiplying the number of records normally retrieved by 1.25.

The expression for this calculation is shown below:

- Number of instances = a x b x 1.25 (the rate of safety)

  a: Number of records that can be retrieved at a time

  b: Number of clients that access simultaneously per process

Sample

When ten clients retrieve 100 records simultaneously

Number of instances = 10 x 100 x 1.25 = 1250

Note

Note that increasing the number of instances consumes more memory space.

3.4.3.3 Instance Management Mode

Database processing can be tuned using Entity Bean instance management modes.

The following table lists the instance management modes and their optimum usage for processing.

Table 3.5 Instance Management Modes

ReadWrite	Effective for executing a retrieval or updating an online database
ReadOnly	Effective for retrieving (referencing) major data that is not updated
Sequential	Effective for batch processing of mass data

3.4.3.4 Map CMP DataStream

If the data handled in the CMP Entity Bean exceeds the JDBC size limit, make the following settings in the Interstage Management Console.

Click [System] > [WorkUnit] > 'WorkUnit Name' > 'EJB Module' > 'EJB Application Name' > [Application Environment Definition], and set [Map CMP DataStream] of the CMF mapping definition to 'Yes'. The default is 'No'.

3.4.3.5 Speed up CMP2.0 Item Search

This is an option for loading all record data at once when multiple 'finder' methods are executed in CMP2.0 Entity Bean.

Data can be processed quickly, even if it is all loaded from DBMS.

For details about the CMP2.0 Entity Bean multiple search, refer to "Entity Bean Optimization", "Increasing the Speed of CMP2.0 Multi-Item Searches" in the "Basic Functions of the EJB Service" chapter of the J2EE User's Guide.

3.4.3.6 Update CMP1.1 Records in Batch

Make the multiple record batch update select option settings in the Interstage Management Console.

In the Interstage Management Console, click [System] > [WorkUnit] > 'WorkUnit Name' > 'EJB Module' > 'EJB Application Name' > [Application Environment Definition] > [Interstage Additional Settings]. Make the settings here.

To use this option, select 'Yes' in [Batch update of multiple records] of the above window. The default is 'Yes'.

The CMP1.1 Entity Bean for which the settings were made is batch updated using the following API when the database is updated.

'addBatch' method of the 'java.sql.PreparedStatement' class

Note

- The database and JDBC driver must support the JDBC2.0 batch update function.
- If the database and JDBC driver do not support the JDBC2.0 batch update function, normal database update processing is performed.

  If [Instance Management Mode] is 'Read-Only', the data itself is not updated. For this reason, the [Batch update of multiple records] settings are disabled.
- [Batch update of multiple records] is only enabled if distributed transactions are not used. If distributed transactions are used, normal database update processing is performed.
- If CMP1.1 Entity Bean stream forwarding is used, it might cause the data update to fail. In this case, change [Batch update of multiple records] to 'No'.

3.4.3.7 Judge CMP1.1 Byte Array Update

If byte arrays are used in CMP1.1 Entity Bean, a method to judge whether the byte array data has been updated can be set.

In the Interstage Management Console, click [System] > [Environment Settings] > [Detailed Settings] > [EJB Service Detailed Settings] > [Judge CMP1.1 byte array update]. Make the settings here. The *isj2eeadmin* command can also be used to make the settings.

3.4.4 Message-Driven Bean

If the number of instances of a message-driven bean is defined, multiple messages can be processed simultaneously.

Usually, define the number of instances to the extent that messages will not be stacked in the queue. Because the optimum number depends on the number of clients and the processing time of the message-driven bean, carry out a trial run according to the environment, make adjustments, and then define the appropriate number of instances.

3.4.4.1 Message-driven Bean Thread Pool

Pooled threads are used to process received Point-To-Point or Publish/Subscriber messaging model Message-driven Bean messages.

These pooled threads are called 'simultaneous Message-driven Bean processes'.

Note

In the following cases, the receipt of messages in thread concurrency is not supported. For this reason, the Message-driven Bean thread pool is not used.

- Messaging Model: 'Publish/Subscriber' and

- Transaction Type: 'Container' and
- Transaction Attribute: 'Required'

Thread Pool Units

Pooled threads are created for one IJServer process, and shared to process all Message-driven Bean messages deployed to the IJServer.

Creating Threads

Threads are created for the minimum number of simultaneous Message-driven Bean processes when the IJServer process starts and stored in the pool.

If there is a search for the pool before the message is deployed and there are found to be no threads in the pool, and the number of threads currently used is less than the maximum number of simultaneous Message-driven Bean processes, the threads are created and received messages are processed.

After a message is received, the created thread is not deleted but returned to the pool.

Note

If there are no threads in the pool and the number of threads currently used exceeds the maximum number of simultaneous Message-driven Bean processes, there is a wait until the currently used threads are returned to the pool before these threads are used to process the received messages.

Deleting Threads

Pooled threads are destroyed when there is an Idle Timeout. If a thread is not used for a specified time following its return to the pool, the thread is destroyed.

Initial start threads are continuously pooled, however, so they are not destroyed. Threads for IJServer processes that have stopped are also destroyed.

Tuning Thread Pools

The thread pool can be tuned as shown below using the Interstage Management Console or the *isj2eeadmin* command.

Simultaneous Message-driven Bean Processes

Specify the maximum and minimum number of simultaneous Message-driven Bean processes (threads) for the process used to run EJB, and the Timeout Time until unused pooled threads are released.

In the Interstage Management Console, click [WorkUnit] > 'WorkUnit Name' > [Environment Settings] > [Detailed Settings] > [EJB Container Settings]. Make the settings here.

The *isj2eeadmin* command can also be used to make the settings.

Option	Description	Description of the value
Minimum	Specify the number of threads created when the IJServer process starts. The created threads are not pooled, nor are they targets of Idle Timeout.	Specify a number from [0] to [2147483647].The default is [0].
Maximum	Only the minimum number of threads is created when the IJServer process starts. This increases the number of threads if necessary.	Specify a number from [1] to [2147483647].The default is [64].
Idle Timeout	Specify the Idle Timeout. If a thread is not used for a specified time following its return to the pool, the thread is destroyed. If the minimum is set for the number of simultaneous Message-driven Bean processes, however, the thread is not destroyed. If [0] is specified, timeout monitoring is not used	Specify a number from [0] to [2147483647].The default is [600] (Seconds).

Note the following points when tuning the maximum number of simultaneous instances.

- If processing is intermittent, set the minimum value for the number of simultaneous threads. This reduces the CPU activity required to create and destroy threads and improves performance. If Message-driven Bean processing is infrequent, set a low minimum value to reduce the memory used.
- If Interstage Management Console monitor information is referenced it increases the Idle Timeout Count intermittently, and threads are created and destroyed frequently. There is a possibility that a low value has been set for the Idle Timeout, therefore investigate the possibility of increasing the Idle Timeout value. If the value set for Idle Timeout is increased, however, it will cause the threads that are created to remain in the pool for longer. This will also increase the memory used. Consider the Java heap and system resources before making the settings.
- If a large number of processing requests are to be received, CPU activity can be reduced by setting a low value for the maximum number of simultaneous threads. If the CPU activity for the new maximum value is still low, a higher value can then be set so that more Message-driven Bean requests will be processed simultaneously. The intention is to allow the greatest number of simultaneous requests to be processed without significantly impairing the system performance
- If the Message-driven Bean initial start thread total exceeds the maximum number of simultaneous Message-driven Bean instances, there is a possibility that the number of messages distributed to the IJServer process exceeds the maximum number of simultaneous Message-driven Bean instances. In this case, there is a wait until the currently used threads are returned to the pool. If you do not want to wait until the currently used threads are returned to the pool, specify the total Message-driven Bean initial start instance number for the maximum number of simultaneous Message-driven Bean instances using the Interstage Management Console or isj2eeadmin command.

3.4.5 JNDI

3.4.5.1 Holding and Using an Object

By holding and using an object (obtained by the lookup method) in an EJB application, the execution count of the lookup method can be reduced.

3.4.5.2 Deployment Descriptor Definition

When object information is defined in the deployment descriptor file, the object is obtained from each Naming Service (based on the defined information) during IJServer start-up and held in memory. The processing performance is therefore improved.

When object information is not defined in the deployment descriptor file, check whether or not the object is being located in each Naming Service when the lookup method is executed in the application.

It is better to define object information in the deployment descriptor file. If, however, the contents of the deployment descriptor file cannot be edited because a developed EAR file was used, the following options can be used to reduce the number of accesses to the Naming Service.

If these options are used, the information collected from each Naming Service can be held in memory during the lookup method execution, so the processing of subsequent lookup methods for the same object is improved. When the following object is acquired, operate IJServer with this option set.

- Other IJServer's EJB application Home object
- Data Source
- JMS Connection Factory
- JMS Destination

Table 3.6 Options to Reduce Naming Service Accesses

Item	Set contents
Definition file directory	Windows32/64 Interstage installation directory\ejb\etc Solaris32/64 Linux32/64 /opt/FJSVejb/etc
Definition file	FJEJBconfig.properties

Item	Set contents
Specified key name	"LookupCache" (predetermined)
Optional value	1: Returns information held in memory during subsequent lookup method execution for the same object. Other than 1: (Default) Accesses each Naming Service every time the lookup method is executed.

3.5 Tuning the CORBA Service

Generating Interstage System Definition Files

To change the number of client connections that can be made to Interstage, set a value for the scale-value option of the *isgendef* command appropriately for the scale of the system.

Tuning CORBA Service Environment Definition

There is no need for customization for normal operation.

If resources become insufficient due to system expansion or to the addition of IJServers, increase the values of the CORBA service environment definition in the scale statement of the computer where Interstage is installed. For details, refer to the information on increasing default settings of the CORBA Service environment definition in "Definition File Setting Values" in the "Chapter 2 Tuning Interstage" chapter.

When using an EJB Service

To create and delete IJServers or to deploy and undeploy EJB applications, use CORBA service resources as EJB services. Increase values as follows:

Statement	Addition value
max_processes	1
max_IIOP_resp_con	1

When an IJServer is Added

To add an IJServer or change the process concurrency level of an IJServer, add the values shown below:

Statement	Addition value
max_processes	Process concurrency level of the IJServer that is added (*1)
max_exec_instance	Number of processes in IJServer (the EJB container) added x 64 (the maximum value for number of threads specified when an IJServer is created) (*2)
max_impl_rep_entries	3 (*2)

- *1 Add this for all IJServer types.
- *2 This is only applicable when the IJServer type is one of the following:
  - Web applications and EJB applications running in separate Java VMs
  - EJB applications only

When a Client Application is Added

With J2EE application clients, or when invoking EJB applications from applets, increase values as follows:

Statement	Addition value
Max_IIOP_resp_con	Number of processes in the client application added

When the Application that Invokes the EJB Application is Deployed in an IJServer

When invoking EJB applications from separate computers and processes, increase values as follows:

Statement	Addition value
Max_IIOP_resp_con	Number of IJServer processes on invoking side

For details on the value to set, see the following:

- IJServer with Web application and EJB application running in the same JavaVM

	Condition	Addition value
1	An EJB application deployed in a separate IJServer is invoked from a Web application or EJB application	IJServer process concurrency
2	An EJB application deployed in a separate IJServer is not invoked from a Web application or EJB application	None

- Web applications and EJB applications running in separate Java VMs

	Condition	Addition value
1	An EJB application deployed in the same IJServer is invoked from a Web application	Servlet container process concurrency (*1)
2	A Web application deployed in the same IJServer is invoked from an EJB application	EJB container process concurrency (*2) (*3)
3	An EJB application deployed in a separate IJServer is invoked from a Web application	Servlet container process concurrency (*1)
4	A Web application deployed in a separate IJServer is invoked from a EJB application	EJB container process concurrency (*2)

*1 f conditions 1 and 3 are met, increase by 1.

*2 If conditions 2 and 4 are met, increase by 1.

*3 For EJB applications whose 'Local Invocation' setting in the application environment definition is set to "On", there is no need to increase.

- EJB applications only

	Condition	Addition value
1	A Web application deployed in the same IJServer is invoked from an EJB application	EJB container process concurrency (*1) (*2)
2	A Web application deployed in a separate IJServer is invoked from an EJB application	EJB container process concurrency (*1)

*1 If conditions 1 and 2 are met, increase by 1.

*2 For EJB applications whose 'Local Invocation' setting in the application environment definition is set to "On", there is no need to increase.

- Web applications only

	Condition	Addition value
1	An EJB application deployed in a separate IJServer is invoked from a Web application	Servlet container process concurrency

Note

When an EJB application is used, the no-communication monitoring function of the CORBA service cannot be used.

3.6 Tuning Directory Service as an LDAP Server

For details about tuning Interstage Directory Service, refer to the Directory Service Operator's Guide.

Note

In Windows(64 bit) and Linux(64 bit), Interstage Directory Service can only be used with Interstage Application Server Enterprise Edition.

Chapter 4 Tuning WorkUnits

A WorkUnit has various functions that can be tuned to provide optimum performance. This chapter explains the tuning of WorkUnits.

4.1 Tuning CORBA WorkUnits

CORBA WorkUnits can be used with the following products:

- Interstage Application Server Enterprise Edition
- Interstage Application Server Standard-J Edition

Refer to the following manuals for information on tuning WorkUnits:

- Designing WorkUnits in "Designing the OLTP Server" of the OLTP Server User's Guide
- CORBA WorkUnit in "Designing the OLTP Server" of the OLTP Server User's Guide

4.2 Tuning IJServer WorkUnits

IJServer WorkUnits can be used with the following products:

- Interstage Application Server Enterprise Edition
- Interstage Application Server Standard-J Edition

Refer to Tuning of IJServer in "Tuning J2EE Applications" for information on tuning WorkUnits.

Chapter 5 System Tuning

This chapter describes system tuning.

5.1 System Resources for Operating Interstage Server Function (for Solaris)

This section describes the system resources that are required for operating each Interstage service.

Refer to the following table and perform tuning of the relevant services, depending on the products used.

Table 5.1 Tuning for Interstage Products

Operating Service	Interstage Application Server Enterprise Edition	Interstage Application Server Standard-J Edition
Setting the System Environment for a CORBA Service	Must be tuned.	Must be tuned.
Setting the System Environment for a Component Transaction Service	Must be tuned.	Must be tuned.
System Resources of the Event Service System (*1)	Must be tuned.	Must be tuned.
System Resources of the IJServer or EJB Service	Must be tuned.	Must be tuned.
System Resources of the Interstage Single Sign-on	Must be tuned	Must be tuned
System Resources of Interstage Directory Service	Must be tuned	Must be tuned
System Resources of the Web server connector	Must be tuned	Must be tuned

*1 Tuning for the event service is required when Interstage JMS is used.

5.1.1 Setting the System Environment for a CORBA Service

When running a system using a CORBA Service, the system resources will need to be increased according to the number of clients/servers to be connected, the number of objects, and so forth. This section describes the system resources (system parameters, processes, threads and file descriptors) required by the CORBA Service and the client and server applications.

5.1.1.1 System Parameters

The example below shows typical additions to the kernel parameters for shared memory, semaphores, and message queues used by an ordinary CORBA Service.

If any application other than the CORBA Service is also going to use shared memory, semaphores and message queues, add the resource quantities for the CORBA Service to those required by the other application.

5.1.1.1.1 Changing the System Parameters

For details on how to change system parameters, refer to the "Setting IPC Resources" appendix.

5.1.1.2 CORBA Service

The system resources required by the CORBA Service are shown in 5.2.1.2 CORBA Service.

Note

In the subsequent descriptions, "type" has the following meaning.

Setup Value

Change the values according to conditions applying to 'Required amount'.

Additional Value

Add the 'Required amount' to the already set value.

5.1.1.2.1 Shared Memory

Table 5.2 System Resources Required by the CORBA Service

Parameter	Resource control	Type	Required amount (value to add)	Remarks (description)
Shmmax	project.max-shm-memory	Additional value	max_IIOP_resp_con x 0.4KB + limit_of_max_IIOP_resp_con (*1) x 0.5KB + max_IIOP_resp_con_extend_number (*2) x 0.1KB + max_IIOP_resp_requests x 8KB + limit_of_max_IIOP_resp_requests (*1) x 3KB + max_IIOP_resp_requests_extend_number (*2) x 0.1KB + limit_of_number_of_common_buffer(*4) x 5KB + number_of_common_buffer_extend_number (*2) x 0.1KB + max_processes x 0.6KB + max_exec_instance x 0.2KB + max_impl_rep_entries x 12KB + (max_IIOP_resp_con + limit_of_max_IIOP_resp_con (*1) x 2) x max_impl_rep_entries x 0.004KB + max_bind_instances x 0.1K + 3200KB or more	Maximum shared memory segment size
			The above value + limit_of_max_IIOP_resp_con x 5KB or more	When the SSL coordinated function is used
			The above value + max_processes x trace_size_per_process + trace_size_of_daemon (*3) + 20KB or more	When the trace function is used
			The above value + snap_size + (max_impl_rep_entries + max_processes) x 0.1KB or more	When the snapshot function is used
			The above value + (Buffer Size + 0.2KB) x Buffer Number x The number of WorkUnit (*5)	When CORBA WorkUnit is used
Shmmni	project.max-shm-ids	Additional value	max_IIOP_resp_con_extend_number (*2) + max_IIOP_resp_requests_extend_number (*2) + number_of_common_buffer_extend_number (*2) + The number of WorkUnit which specified Buffer Size and Buffer Number (WorkUnit definition) + 13	Number of shared memory IDs (*6)

*1 The default value of limit_of_[parameter name] is:

- [parameter name] x 1.3 (rounding up of the fractional part)

This is the default value even if 0 is specified.

If "MANUAL" is specified in the 'AutoConfigurationMode' configuration option of the isconfig.xml file, the value for making non-automatic expansion settings is:

- [parameter name]

*2 The default value of [parameter name]_extend_number is

- (limit_of_[parameter name] - [parameter name]) / [parameter name] (rounding up of the fractional part)

This is the default value even if 0 is specified.

If "MANUAL" is specified in the 'AutoConfigurationMode' configuration option of the isconfig.xml file, the value for making non-automatic expansion settings is.

- 0

*3 The default is:

- trace_size_per_process x 32

This is the default value even if 0 is specified.

*4 The default is:

- max_IIOP_resp_requests x 0.2

This is the default value even if 0 is specified.

*5 When the WorkUnit which specified Buffer Size and Buffer Number (CORBA WorkUnit definition) starts, this value is used. The maximum value of '(Buffer Size + 0.2KB) x Buffer Number' corresponds to the CORBA WorkUnit definition which specified 'Buffer Size' and 'Buffer Number'.

Solaris32

*6 When the multi system is used, the total of the extension system numbers should be added. The multi system function can be used in the Enterprise Edition.

5.1.1.2.2 Semaphores

Table 5.3 Semaphores

Parameter	Resource control	Type	Required amount (value to add)	Remarks (description)
Semmni	project.max-sem-ids	Additional value	Whichever is higher: 512 or max_IIOP_resp_con_extend_number (*1) x 5 + max_IIOP_resp_requests_extend_number (*1) + max_impl_rep_entries + The number of WorkUnit which specified Buffer Size and Buffer Number (WorkUnit definition) x 2 + 100 or more	Total number of semaphore identifiers in system
Semmsl	process.max-sem-nsems	Setup value	Whichever is higher: max_IIOP_resp_con or max_processes or more	Maximum number of semaphores for each semaphore identifier
Semopm	process.max-sem-ops	Setup value	50 or more	Maximum number of operators for each semaphore call

*1 The default value of [parameter name]_extend_number is

- (limit_of_[parameter name] - [parameter name]) / [parameter name] (rounding up of the fractional part)

This is the default value even if 0 is specified.

If "MANUAL" is specified in the 'AutoConfigurationMode' configuration option of the isconfig.xml file, the value for making non-automatic expansion settings is.

- 0

5.1.1.2.3 Message Queues

Table 5.4 Message Queues

Parameter	Resource control	Type	Required amount (value to add)	Remarks (description)
msgmnb	process. max-msg-qbytes	Setup value	32768 or more	Maximum number of messages that can be held in one message queue
msgmni	project.max-msg-ids	Additional value	512 or more	Maximum number of message queue IDs

5.1.1.2.4 Interface Repository

The system resources required to use an Interface Repository are as shown in the following table.

Shared Memory

Table 5.5 Shared Memory

Parameter	Resource control	Type	Required amount (value to add)	Remarks (description)
shmmax (*1)	project.max-shm-memory	Additional value	"logging memory size" + 16KB x 3 (*2)	Total size of shared memory allocated to project when collecting log data (the EJB Service is not used)
			"logging memory size" + 16KB x 4 (*2)	Total size of shared memory allocated to project when collecting log data (the EJB Service is used)
shmmni	project.max-shm-ids	Additional value	3	Number of shared memory IDs (the EJB Service is not used)
			4	Number of shared memory IDs (the EJB Service is not used)

*1 Before configuring a value for shmmax in Solaris 10 or later, refer to the Solaris documentation and the "Setting IPC Resources" appendix.

*2 Specify "logging memory size" in the CORBA Service irconfig file. Refer to "irconfig" in the "CORBA Service Environment Definition" appendix for details.

5.1.1.2.5 Naming Service

The system resources required to create multiple naming contexts in the Naming Service are shown in the following table.

Table 5.6 Naming Service

Parameter	Required amount (value to add)	Remarks (description)
(*1)	Number of naming contexts + 16 or more	Number of files that can be opened for each process

*1 The relevant parameter does not exist.

Using the ulimit command for the Bourne shell and the limit command for the C shell, set the values only as far as required for opening the file with the naming service process. For details about commands, refer to the OS documents.

5.1.1.3 Number of Threads and Processes Used by Applications

When applications are to be executed in the CORBA Service, system parameters will need to be changed if large numbers of processes and threads are going to be generated by the applications.

The targets for the number of (multiple) threads started when applications are started are shown in the following table.

Table 5.7 Targets for the Number of Threads

Component	Number of Threads
CORBA Service	25 + number of connections to client application
Server application	(6 + number of thread primary multiples) for each process
Client application	8 (maximum) per process

The system parameters that require modification are listed in the following table.

Table 5.8 System Parameters

Parameter	Description
max_nprocs	Total number of processes started in the system

Table 5.9 System Parameters

Parameter	Description
kernel.threads-max	Total number of processes started in the system

Table 5.10 Parameters other than System Parameters

Parameter	Description
(*1)	The maximum number of processes available to a single user
(*2)	The maximum stack size

*1 There is no corresponding parameter. Use the 'ulimit' (for bash or Bourne shell types) or 'limit' (for C shell types) command to make the settings. Set a value that is equal to or greater than the total number of processes and threads created by the user.

*2 The corresponding parameter does not exist. Use the 'ulimit' (for bash or Bourne shell types) or 'limit' (for C shell types) command to make the settings.

This value is multiplied by the number of threads, and the resulting value used in defining the size of the process stack area. A thread cannot be created if the memory that can be used in one process is exceeded. For this reason, the number of threads that can be created for one process is limited. Request processing concurrency for CORBA server applications and EJB applications is calculated as 'thread concurrency x process concurrency'. If it is not possible to increase the thread concurrency using the memory size that can be used per process, investigate the possibility of increasing the process concurrency. For details of thread concurrency/process concurrency for CORBA server applications, refer to descriptions of "proc_conc_max", "thr_conc_init" and "thr_conc_maximum" in "OD_impl_inst" in the "CORBA Service Operation Commands" chapter of the Reference Manual (Command Edition). For details of EJB application thread concurrency, refer to "Number of Simultaneous Processing Tasks" in "Tuning of EJB Containers" in the "Tuning J2EE Applications" chapter.

5.1.1.4 Number of File Descriptors

If multiple applications are going to be executed in the CORBA Service (because multiple terminals are connected, for instance), and the number of file descriptors used exceeds the default value, also set the parameter of the config file.

Table 5.11 File Descriptors

Parameter	Description
rlim_fd_cur (System parameter)	Set when the number of file descriptors in use exceeds the default value.

5.1.2 Setting the System Environment for a Component Transaction Service

When the Component Transaction Service is in operation, the system resources must be expanded depending on the functions used. The system resources (system parameters) required by the Component Transaction Service and the system resources (system parameters) required by each function are described below.

For details on how to change system parameters, refer to the "Setting IPC Resources" appendix.

Note

The following values are not included in the values of the CORBA Service. Refer to "Setting the System Environment for a CORBA Service" and add the values as required.

5.1.2.1 System Parameters

This section describes tuning of the system parameters of shared memory as well as semaphores and message queues that the Component Transaction Service uses.

When various functions are going to be used in addition to the basic function of the Component Transaction Service, add the amount of resources that are used by each function to the resources of the Component Transaction Service basic function.

Note

In the subsequent descriptions, "type" has the following meaning.

- Setting value

  Change the values according to conditions applying to 'Required amount'.

- Additional value

  Add the 'Required amount' to the already set value.

5.1.2.1.1 Basic Function of the Component Transaction Service

The system resources that are required to use the basic function of the Component Transaction Service are shown in the following tables.

Shared Memory

Table 5.12 Shared Memory

Parameter	Resource control	Type	Required amount	Description
-	project.max-shm-memory	Additional value	[Interstage Application Server Enterprise Edition] - Operation without using wrapper WorkUnit For system scale "small": 36,096,468 or more For system scale "moderate": 48,500,404 or more For system scale "large": 61,728,660 or more For system scale "super": 83,718,036 or more - Operation using wrapper WorkUnit Solaris32 For system scale "small": 36,222,932 or more	Maximum size of shared memory segment

Parameter	Resource control	Type	Required amount	Description
			For system scale "moderate": 48,744,628 or more For system scale "large": 62,914,964 or more For system scale "super": 86,081,940 or more [Interstage Application Server Standard-J Edition] For system scale "small": 35,971,540 or more For system scale "moderate": 48,257,716 or more For system scale "large": 60,543,892 or more For system scale "super": 81,355,668 or more	
shmmax	-	Setup value	For system scale "small": 12,498,508 or more For system scale "moderate": 23,736,108 or more For system scale "large": 34,973,708 or more For system scale "super": 54,736,908 or more	Maximum size of shared memory segment
shmmni	project.max-shm-ids	Additional value	Add 22	Number of shared memory IDs (*1)

Solaris32

*1 When the multi system is used, the total of the extension system numbers should be added.

The multi system function can be used in the Enterprise Edition.

Semaphores

Table 5.13 Semaphores

Parameter	Resource control	Type	Required amount	Description
semmni	project.max-sem-ids	Additional value	Add 29	Number of semaphore IDs (*1)
semmsl	process.max-sem-nsems	Setup value	12 or more	Maximum number of semaphores for each semaphore ID (*1)
semopm	process.max-sem-ops	Setup value	3 or more	Maximum number of operations for each semaphore call

Solaris32

*1 When the multi system is used, the total of the extension system numbers should be added.

The multi system function can be used in the Enterprise Edition.

5.1.2.1.2 Message Queues

Table 5.14 Message Queues

Parameter	Resource control	Type	Required amount	Description
msgmnb	process.max-msg-qbytes	Setup value	4572 + (528 x number of simultaneously executed commands) (*1)	Maximum number of bytes in the queue
msgmni	project.max-msg-ids	Additional value	Add 11 (*2)	Number of message queue IDs
msgtql	process.max-msg-messages	Additional value	15 + number of simultaneously executed commands (*1) (*2)	Number of system message headers

*1 Number of simultaneously executed commands refers to the number of commands when the following commands are executed at the same time.

For Enterprise Edition:

- isstartwu, isstopwu, tdstartwu, tdstopwu, tdinhibitobj, tdpermitobj, tdmodifyprocnum, tdmodifywu

For Standard-J Edition:

- isstartwu, isstopwu

Solaris32

When start/stop of the WorkUnit, object closure/closure release and acquisition of the wrapper WorkUnit object information are executed using the Systemwalker OperationMGR and Interstage operation API, the number of simultaneous operations is counted as the number of simultaneously executed commands.

Object closure and closure cancellation can be used in the Enterprise Edition. Object information allocation of the wrapper work unit can be used in Enterprise Edition.

*2 When the multi system is used, the total of the extension system numbers should be added.

The multi system function can be used in the Enterprise Edition.

5.1.2.2 Session Information Management Function

This section describes the system resources that are additionally required when the session information management function is used.

5.1.2.2.1 Shared Memory

Table 5.15 Shared Memory

Parameter	Resource control	Type	Required amount	Description
shmmni	project.max-shm-ids	Additional value	Add 1	Number of shared memory IDs (*1)

Solaris32

*1 When the multi system is used, the total of the extension system numbers should be added.

The multi system function can be used in the Enterprise Edition.

5.1.2.2.2 Semaphores

Table 5.16 Semaphores

Parameter	Resource control	Type	Required amount	Description
semmni	project.max-sem-ids	Additional value	Add 1	Number of semaphore IDs (*1)

Solaris32

*1 When the multi system is used, the total of the extension system numbers should be added.

The multi system function can be used in the Enterprise Edition.

5.1.2.3 Performance Monitoring Tool

For details on the Performance Monitoring Tool, refer to "5.4 Environment Setup for Performance Monitoring Tool".

5.1.3 System Resources of the Event Service System

To run a system using an Event Service, you will need to increase the system resources according to the number of channels, the number of consumers/suppliers to be connected, and so forth. This section describes how to set the system resources (system parameters) required by the Event Service.

Note

The following values are not included in the values of the CORBA Service. Refer to "Setting the System Environment for a CORBA Service" and add the values as required.

5.1.3.1 System Parameters

The example below shows typical additions to the kernel parameters for shared memory, semaphores, and message queues used by an ordinary Event Service.

5.1.3.1.1 Changing the System Parameters

For details on how to change system parameters, refer to the "Setting IPC Resources" appendix.

Note

- In the subsequent descriptions, "type" has the following meaning.
    - Setting value

      Change the values according to conditions applying to 'Required amount'.
    - Additional value

      Add the 'Required amount' to the already set value.
- When both volatile and persistent channel operations are used, use the 'Required amount' for persistent channel operation.

5.1.3.1.2 Shared Memory

Table 5.17 Shared Memory

Parameter	Resource control	Type	Required amount	Remarks (description)
-	project.max-shm-memory	Additional value	The following values are added: - (1040 bytes x maximum number of created event channel (*1) + 600 kilobytes) x 4 - Number of bytes specified in the trace_size parameter of the traceconfig file	Maximum segment size of shared memory (volatile channel operation)
			The following values are added: - (1040 bytes x maximum number of created event channel (*1) + 184 bytes x number of	Maximum segment size of shared memory (persistent channel operation)

Parameter	Resource control	Type	Required amount	Remarks (description)
			simultaneously executable global transactions (*2) + 600 kilobytes) x (Unit number x 100) - (17 x 1024 x 1024 + 576 x Number of concurrent transactions + 88 x (Number of system data storage areas + Number of event data storage areas) + Size of shared memory to be used by unit x 1024 x 1024 bytes) x (Unit number x 100) (*3) - Number of bytes specified in the trace_size parameter of the traceconfig file	
shmmax	-	Setup value	Whichever is higher: - (1040 bytes x maximum number of created event channel (*1) + 600 kilobytes) x 4 Number of bytes specified in the trace_size parameter of the traceconfig file	Maximum segment size of shared memory (volatile channel operation)
			Whichever is higher: - (1040 bytes x maximum number of created event channel (*1) + 184 bytes x number of simultaneously executable global transactions (*2) + 600 kilobytes) x (Unit number x 100) - (17 x 1024 x 1024 + 576 x Number of concurrent transactions + 88 x (Number of system data storage areas + Number of event data storage areas) + Size of shared memory to be used by unit x 1024 x 1024 bytes) x (Unit number x 100) (see *3) Number of bytes specified in the trace_size parameter of the traceconfig file	Maximum segment size of shared memory (persistent channel operation)
shmmni	project.max-shm-ids	Additional value	4	Number of identifiers of shared memory (volatile channel operation)
			Unit number x 100 or more	Number of identifiers of shared memory (persistent channel operation)

*1. Maximum number of event channels created = maximum number of static generation event channels created + maximum number of dynamic generation event channels created.

*2. Number of global transactions that can be executed simultaneously = the setting value of the -gtrnmax option using the configuration information management command (essetcnf) of Event Service. Solaris32

*3. Number of concurrent transactions = Setting value of the unit definition (tranmax) using the unit generation command (esmkunit) of Event Service

Number of system data storage areas = Setting value of the unit definition (sysqnum) using the unit generation command (esmkunit) of Event Service

Number of event data storage areas = Setting value of the unit definition (userqnum) using the unit generation command (esmkunit) of Event Service

Size of shared memory to be used by unit = Setting value of the unit definition (shmmax) using the unit generation command (esmkunit) of Event Service (When the size of shared memory to be used by unit is smaller than 42, it is 42.)

Calculate each unit when two or more units are used.

5.1.3.1.3 Semaphores

Table 5.18 Semaphores

Parameter	Resource control	Type	Required amount	Remarks (description)
semmni	project.max-sem-ids	Additional value	3	Total number of identifiers of semaphores in system

5.1.3.1.4 Message Queues

Table 5.19 Message Queues

Parameter	Resource control	Type	Required amount	Description
msgmnb	process.max-msg-qbytes	Setting value	4,096 or more	Maximum number of bytes in the queue memory (persistent channel operation)
msgmni	project.max-msg-ids	Additional value	2	Number of message queue IDs memory (volatile channel operation)
			Unit number x 3 + 2	Number of message queue IDs memory (persistent channel operation)
msgtql	process.max-msg-messages	Setting value	The total set for each unit is as follows: 6 + tranmax value of unit definition file x 4 x 1.3	Number of headers of system message memory (persistent channel operation)

5.1.4 System Resources of the IJServer or EJB Service

In the IJServer or EJB Service, the system resources should be expanded when the following functions are used. The system resources (system parameter) that the IJServer or EJB Service requires are described below.

5.1.4.1 System Parameters

When the IJServer or EJB Service is used, tune the following system parameters.

5.1.4.1.1 Changing the System Parameters

For details on how to change system parameters, refer to the "Setting IPC Resources" appendix.

5.1.4.1.2 Message Queues

Table 5.20 Message Queues

Parameter	Resource control	Type	Required amount	Description
msgmnb	process.max-msg-qbytes	Setting value	4,096 or more	Maximum value of message that a single message queue can hold.
msgmni	project.max-msg-ids	Additional value	2 or more	Maximum value of message queue ID

Parameter	Resource control	Type	Required amount	Description
msgtql	process.max-msg-messages	Setting value	1024 or more	Maximum number of messages that can be created

5.1.5 System Resources of the Interstage HTTP Server

For system operation using the Interstage HTTP Server, it is necessary to expand the system resources. This section explains the system resources (file descriptors).

5.1.5.1 Number of File Descriptors

The number of file descriptors that are required to run Interstage HTTP Server will depend on the features that are used in the Web server and on the content that is defined in the environment definition file (httpd.conf).

Refer to the following table before calculating the number of required file descriptors. If this value exceeds the system default, set this value as the system parameter.

Table 5.21 File Descriptors

Parameter	Type	Required amount
rlim_fd_cur	Setup value	130 or more
		[When the following features are used] Add the following values to the above: - Basic Authentication function: 1 - Online Collation function: 1 - SSL communication: 21 - Proxy function: 1 - CGI function (*1) : 5
		[When the following directives are added to the environment definition file (httpd.conf)] Add the following values to the above: - CustomLog(specify ihsrlog command execution statement): 2 x directive number - CustomLog(not specify ihsrlog command execution statement): 1 x directive number - ErrorLog: 1 x directive number - Listen: 1 x directive number

*1 Also add the number of file descriptors that are required in the CGI program that is to be run.

5.1.6 System Resources of the Interstage Single Sign-on

For system operation using Interstage Single Sign-on, it is necessary to expand the system resources. This section explains the system resources (system parameters).

5.1.6.1 System Parameters

This section explains the tuning of the system parameters used by Interstage Single Sign-on.

5.1.6.1.1 Changing the System Parameters

For details on how to change system parameters, refer to the "Setting IPC Resources" appendix.

Note

In the subsequent descriptions, "type" has the following meaning.

- Setting value

  Change the values according to conditions applying to 'Required amount'.

- Additional value

  Add the 'Required amount' to the already set value.

5.1.6.1.2 Tuning System Parameters when using the Interstage Single Sign-on Repository Server Function

Shared Memory

Table 5.22 Shared Memory

Parameter	Resource control	Type	Required amount	Remarks (description)
shmmax (*1)	project.max-shm-memory	Additional value	26000000 + (164 x number of site configurations (*2)) + (number of roles (*3) + number of role sets (*4) + number of role sets (*4) x number of roles (*3)) x 1024 x 3 (*5)	Maximum size of shared memory
shmmni	project.max-shm-ids	Additional value	13	Number of shared memory segments in the system

*1 Before configuring a value for shmmax, refer to the Solaris documentation and the "Setting IPC Resources" appendix.

*2 Total number of site configurations registered as protected resources in the SSO repository.

*3 The total number of roles defined in the SSO repository.

*4 The total number of role sets defined in the SSO repository.

*5 To use Active Directory as the directory service that is used for registering user information without using Single Sign-on extended schema, add the value that is estimated using the following formula for the application.

Total number of attributes used for the Active Directory role/role set x 524 x 3

Semaphores

Table 5.23 Semaphores

Parameter	Resource control	Type	Required amount	Remarks (description)
shmmni	project.max-shm-ids	Additional value	9	Maximum number of semaphore identifiers

"Interstage Directory Service" is used in the repository server. Refer to "System Resources of the Interstage Directory Service" for details about the tuning required.

5.1.6.1.3 Tuning System Parameters when using the Interstage Single Sign-on Authentication Server Function

Shared Memory

Table 5.24 Shared Memory

Parameter	Resource control	Type	Required amount	Remarks (description)
shmmax (*1)	project.max-shm-memory	Additional value	27000000 + number of protected paths (*2) x 2048	Maximum size of shared memory

Parameter	Resource control	Type	Required amount	Remarks (description)
shmmni	project.max-shm-ids	Additional value	11	Number of shared memory segments in the system

*1 Before configuring a value for shmmax, refer to the Solaris documentation and the "Setting IPC Resources" appendix.

*2 Total number of protected paths for each site configurations registered as protected resources in the SSO repository.

Semaphores

Table 5.25 Semaphores

Parameter	Resource control	Type	Required amount	Remarks (description)
shmmni	project.max-shm-ids	Additional value	10	Maximum number of semaphore identifiers

5.1.6.1.4 Tuning System Parameters when using the Interstage Single Sign-on Business Server Function

Shared Memory

Table 5.26 Shared Memory

Parameter	Resource control	Type	Required amount	Remarks (description)
shmmax (*1)	project.max-shm-memory	Additional value	(13000000 + (1 + cache size (*1)) x 1024 x cache count (*2) + (number of roles (*3) + number of extended user information (*4) + 1) x largest number of path configurations (*5) x 1200 x 2) x number of business servers	Maximum size of shared memory
shmmni	project.max-shm-ids	Additional value	10 x number of business servers	Number of shared memory segments in the system

*1 Before configuring a value for shmmax, refer to the Solaris documentation and the "Setting IPC Resources" appendix.

*2 When session management is used, the size (Kbytes) of the authentication information about the user cached in the business server is set. In applications in which session management is not used, this is calculated as 0. For details about the cache size, refer to "Tuning for Setting up the Business Server" in the "Environment Definition for Interstage Single Sign-on" appendix.

*3 When session management is used, a value greater than the maximum number of simultaneous accesses is set. In applications in which session management is not used, this is calculated as 0. For details about the cache count, refer to "Tuning for Setting up the Business Server" in the "Environment Definition for Interstage Single Sign-on" appendix.

*4 Total number of roles defined in the SSO repository.

*5 Number of items of user information that is notified to the business application

When session management is used, the number of attribute names set in the Interstage Management Console of the repository server, shown below, is set. In applications in which session management is not used, this is calculated as 0.

[System] > [Security] > [Single Sign-on] > [Authentication infrastructure] > [Repository Server] > [Settings] > [Repository server detailed settings [Show]] > [Information notified to the Business System] > [Extended user information]

*6 This is the largest number of path configurations across all site configurations that are registered as protection resources in the SSO repository.

Semaphores

Table 5.27 Semaphores

Parameter	Resource control	Type	Required amount	Remarks (description)
Shmmni	project.max-shm-ids	Additional value	7 x number of business servers	Maximum number of semaphore identifiers

5.1.7 System Resources of the Interstage Directory Service

For system operation using Interstage Directory Service, it is necessary to expand the system resources. This section explains the system resources (system parameters).

5.1.7.1 System Parameters (System Resources Required for Interstage Directory Service)

This section explains the tuning of the system parameters used by Interstage Directory Service.

5.1.7.1.1 Changing the System Parameters

For details on how to change system parameters, refer to the "Setting IPC Resources" appendix.

In subsequent descriptions, "type" has the following meaning:

- Setting value

  Change the values according to conditions applying to 'Required amount'.

- Additional value

  Add the 'Required amount' to the value already set.

Shared Memory

Parameter	Resource control	Type	Required amount	Remarks (description)
shmmax (*1)	project.max-shm-memory	Additional value	((number of repositories x 1,572,864) + (5 x (number of repositories x 1,843,200)) (*2)) x (4 x number of repositories) or more	Maximum size for shared memory
Shmmni	project.max-shm-ids	Additional value	4 x (number of repositories)	Number of shared memory segments in the system

*1 Before configuring a value for shmmax in Solaris, refer to the Solaris documentation and the "Setting IPC Resources" appendix.

*2 If you are using a standard database as the repository database for replication applications, add an extra 1,048,576.

Tune the system resources required to run the RDB, if an RDB is used as the repository database. Refer to the following prior to changing system parameters:

- 5.1.7.2 System Parameters (System Resources Required for Symfoware Server)
- 5.1.7.3 System Parameters (System Resources Required for Oracle Database)

5.1.7.2 System Parameters (System Resources Required for Symfoware Server)

Change the system parameter in the machine used to install Symfoware Server. These system parameters must be set to the values in the table below in the Symfoware Server system operating environment file or in the RDB configuration parameter file. When the value of the parameter of the table below is changed, refer to the Symfoware Server manual and re-calculate the values for the required system parameters.

Parameter	Required value (bytes)
Memory used by the local connection (COMMUNICATION_BUFFER)	32K
Number of local connections (MAX_CONNECT_SYS) (*1)	256
Daemon concurrency (RDBCNTNUM)	712
Shared memory (RDBEXTMEM)	13,208K

*1 Add the number of connections used by other applications to the maximum number of allowed connections from the repository to the RDB that is required to use the Interstage Directory Service.

If the result exceeds the value set for the number of local connections (256), re-calculate the values for the required system parameters.

For details on the maximum number of repository connections, refer to "Specifying the Maximum Number of Connections" in the "Creating a Load Distribution Environment" chapter of the Directory Service Operator's Guide.

Installing Symfoware Server

Shared Memory

Parameter	Resource control	Type	Required amount	Remarks (description)
Shmmax	project.max-shm-memory	Setting value	13,524,992 or more	Maximum segment size for shared memory
Shmmni	project.max-shm-ids	Additional value	10	Number of shared memory segments in the system

Semaphores

Parameter	Resource control	Type	Required amount	Remarks (description)
Semmni	project.max-sem-ids	Additional value	300	Number of semaphore identifiers
Semmsl	process.max-sem-nsems	Setting value	48 or more	Maximum number of semaphores for each semaphore ID

Message Queues

Parameter	Resource control	Type	Required amount	Description
msgmnb	process.max-msg-qbytes	Setup value	4,096 or more	Maximum number of bytes in the queue
Msgmni	project.max-msg-ids	Additional value	2	Number of message queue IDs
Msgtql	process.max-msg-messages	Setup value	64 or more	Number of system message headers

5.1.7.3 System Parameters (System Resources Required for Oracle Database)

Change the system parameters in the machine used to install Oracle Database. Configure system resources required to run the replication, if one is being used. For details on these settings, refer to the Oracle database manual.

Installing Oracle Database

Shared Memory

Parameter	Type	Required amount	Remarks (description)
Shmmax	Setting value	4,294,967,295 or more	Maximum segment size for shared memory
Shmmni	Setting value	100	Number of shared memory segments in the system

Semaphores

Parameter	Type	Required amount	Remarks (description)
Semmni	Setting value	100 or more	Number of semaphore identifiers
Semmsl	Setting value	256 or more	Maximum number of semaphores for each semaphore ID

5.1.8 Setting the System Environment for the Interstage Setup Commands

When Java EE, the multilanguage service, or the J2EE compatibility feature are used, in addition to the system resources that are added with each feature it will also be necessary to extend these system resources with the Interstage Setup commands.

5.1.8.1 System Parameters

This section describes tuning of the system parameters of shared memory as well as semaphores and message queues that the Setup commands uses.

Shared Memory

Parameter	Resource control	Type	Required amount	Description
-	project.max-shm-memory	Additional value	2,015,552 or more	Maximum size of shared memory segment
shmmax	-	Setup value	1,106,440 or more	Maximum size of shared memory segment
shmmni	project.max-shm-ids	Additional value	Add 22	Number of shared memory IDs (*1)

Solaris32

*1 When the multi system is used, add the total of the extension system numbers.

The multi system function can be used in the Enterprise Edition.

Semaphores

Parameter	Resource control	Type	Required amount	Description
semmni	project.max-sem-ids	Additional value	Add 2	Number of semaphore IDs (*1)
semmsl	process.max-sem-nsems	Setup value	12 or more	Maximum number of semaphores for each semaphore ID (*1)
semopm	process.max-sem-ops	Setup value	3 or more	Maximum number of operations for each semaphore call

Solaris32

*1 When the multi system is used, add the total of the extension system numbers.

The multi system function can be used in the Enterprise Edition.

5.1.8.1.1 Message Queues

Table 5.28 Message Queues

Parameter	Resource control	Type	Required amount	Description
msgmnb	process.max-msg-qbytes	Setup value	4572 + (528 x number of simultaneously executed commands) (*1)	Maximum number of bytes in the queue
msgmni	project.max-msg-ids	Additional value	Add 12	Number of message queue IDs (*2)
msgtql	process.max-msg-messages	Setup value	15 + number of simultaneously executed commands (*1)	Number of system message headers (*2)

*1 Number of simultaneously executed commands refers to the number of commands when the Interstage Setup commands are executed at the same time.

Solaris32

When start/stop of the WorkUnit, object closure/closure release and acquisition of the wrapper WorkUnit object information are executed using the Systemwalker Operation Manager and Interstage operation API, the number of simultaneous operations is counted as the number of simultaneously executed commands.

*2 When the multi system is used, add the total of the extension system numbers.

The multi system function can be used in the Enterprise Edition.

5.1.9 System Resources required by a Web Server Connector

For system operation using Web server connector, it is necessary to expand the system resources. This section explains the system resources (system parameters).

5.1.9.1 System Parameters

When the Web server connector is used, tune the following system parameters.

When Web server connector fault monitoring is used, add the resource amount used for Web server connector fault monitoring to the resource amount used for the Web server connector.

5.1.9.1.1 Changing the System Parameters

For details on how to change system parameters, refer to the "Setting IPC Resources" appendix.

Note

In the subsequent descriptions, "type" has the following meaning.

- Setting value

  Change the values according to conditions applying to 'Required amount'.

- Additional value

  Add the 'Required amount' to the already set value.

5.1.9.2 Web Server Connector

The system resources required for using Web server connector fault monitoring are shown in the table below.

5.1.9.2.1 Semaphores

Table 5.29 Semaphores

Parameter	Resource control	Type	Required amount	Remarks (description)
semmni	project.max-sem-ids	Additional value	Total required across all Web servers (*1) Each web server requires: ((IJServer WorkUnit total + IJServer WorkUnit process concurrency total + 2) x 2) + 4	Number of semaphore identifiers
Semmsl	process.max-sem-nsems	Setting value	1 or more	Maximum number of semaphores for each semaphore ID
Semopm	process.max-sem-ops	Setting value	2 or more	Maximum number of operations for each semaphore call

*1 The following is a tuning example:

- There are two Web servers (web001 and web002)
- There are three IJServer WorkUnits (WU001, WU002, and WU003)
- WorkUnit process concurrency is set to 3 on each IJServer
- web001 connector destinations are WU001 and WU002; web002 connector destination is WU003

semmni

Required number of semaphores in each web server

= ((IJServer WorkUnit total + WorkUnit process concurrency total for each IJServer + 2) x 2) + 4

Required number of semaphores in web001

= ((2 + 6 + 2) x 2) + 4

= 24

Required number of semaphores in web002

= ((1 + 3 + 2) x 2) + 4

= 16

semmni = Total required across all Web servers

= Required web001 number + Required web002 number

= 24 + 16

= 40

5.1.9.3 Web Server Connector Fault Monitoring

Additional system resources used with Web server connector fault monitoring are shown in the table below.

5.1.9.3.1 Shared Memory

Table 5.30 Shared Memory

Parameter	Resource control	Type	Required amount	Remarks (description)
shmmax (*1)	project.max-shm-memory	Additional value	6,720,012 x (Number of Web servers + 1)	Maximum size of shared memory
Shmmni	project.max-shm-ids	Additional value	Number of Web servers + 1	Number of identifiers of shared memory

*1 Before configuring a value for shmmax in Solaris, refer to the Solaris documentation and the "Setting IPC Resources" appendix.

5.1.9.3.2 Semaphores

Table 5.31 Semaphores

Parameter	Resource control	Type	Required amount	Remarks (description)
Semmni	project.max-sem-ids	Additional value	3	Number of semaphore identifiers

5.2 System Resources for Operating Interstage Server Function (for Linux)

This section describes the system resources that are required for operating each Interstage service.

Refer to the following table and perform tuning of the relevant services, depending on the products used.

Table 5.32 Tuning for Interstage Products

Operating service	Interstage Application Server Enterprise Edition	Interstage Application Server Standard-J Edition
Setting the System Environment for a CORBA Service	Must be tuned.	Must be tuned.
Setting the System Environment for a Component Transaction Service	Must be tuned.	Must be tuned.
System Resources of the Event Service System (*1)	Must be tuned.	Must be tuned.
System Resources of the IJServer or EJB Service	Must be tuned.	Must be tuned.
System Resources of the Interstage HTTP Server	Must be tuned.	Must be tuned.
System Resources of the Interstage Single Sign-on	Must be tuned	Must be tuned (*2)
System Resources of the Interstage Directory Service	Must be tuned	Must be tuned (*3)
System Resources of the Interstage Management Console	Must be tuned.	Must be tuned.
System Resources of the Web server connector	Must be tuned	Must be tuned

*1 Tuning for the event service is required when Interstage JMS is used.

*2 In Linux (64 bit), only the system resources of the authentication server and the business server need to be tuned.

*3 This is not valid for Standard-J Edition on Linux (64 bit).

5.2.1 Setting the System Environment for a CORBA Service

When running a system using a CORBA Service, the system resources will need to be increased according to the number of clients/servers to be connected, the number of objects, and so forth. This section describes the system resources (system parameters, processes, threads and file descriptors) required by the CORBA Service and the client and server applications.

5.2.1.1 System Parameters

The example below shows typical additions to the kernel parameters for shared memory, semaphores, and message queues used by an ordinary CORBA Service.

If any application other than the CORBA Service is also going to use shared memory, semaphores and message queues, add the resource quantities for the CORBA Service to those required by the other application.

5.2.1.1.1 Changing the System Parameters

Modify the parameter value by editing /etc/sysctl.conf. After completing the modification, either execute "sysctl -p /etc/sysctl.conf" or reboot the system.

For the modification procedure, refer to the OS documentation.

5.2.1.2 CORBA Service

The system resources required by the CORBA Service are shown in the table below.

Note

In the subsequent descriptions, "type" has the following meaning.

Setup Value

Change the values according to conditions applying to 'Required amount'.

Additional Value

Add the 'Required amount' to the already set value.

5.2.1.2.1 Shared Memory

Table 5.33 System Resources Required by the CORBA Service

Parameter (*1)	Type	Required amount	Remarks (description)
kernel.shmmax	Setup value	The maximum is specified using the following values. - max_IIOP_resp_con x 16KB + (max_IIOP_resp_con_extend_number (*1) + 1) x 0.2KB + max_IIOP_resp_requests x16KB + (max_IIOP_resp_requests_extend_number (*1) + 1) x 0.2KB + max_impl_rep_entries x 6KB + max_bind_instances x 0.1K + 100KB or more [For trace_use=yes] The above value + max_processes x trace_size_per_process + trace_size_of_daemon (*2) + 20KB or more [For snap_use=yes] The above value + snap_size + 10KB or more - number_of_common_buffer (*3) x 4KB or more - (Buffer Size + 0.2KB) x Buffer Number or more - (*4)	Maximum shared memory segment size

*1 The default value of [parameter name]_extend_number is:

- (limit_of_[parameter name] - [parameter name]) / [parameter name] (rounding up of the fractional part)

This is the default value even if 0 is specified. limit_of_[parameter name] is calculated automatically if '0' is specified. For details of the calculation formula, refer to "config" in the "CORBA Service Environment Definition" appendix.

If "MANUAL" is specified in the 'AutoConfigurationMode' configuration option of the isconfig.xml file, the value for making non-automatic expansion settings is:

- 0

*2 The default is:

- trace_size_per_process x 32

This is the default value even if '0' is specified.

*3 The default is:

- max_IIOP_resp_requests x 0.2

This is the default value even if '0' is specified.

*4 When the WorkUnit which specified Buffer Size and Buffer Number (CORBA WorkUnit definition) starts, this value is used. The maximum value of '(Buffer Size + 0.2KB) x Buffer Number' corresponds to the CORBA WorkUnit definition which specifies Buffer Size and Buffer Number.

5.2.1.2.2 Semaphores

For the semaphore setting value, specify each parameter in the following format.

- In the case of Red Hat

  kernel.sem = para1 para2 para3 para4

Table 5.34 Semaphores

Parameter	Type	Required amount (value to add)	Remarks (description)
para1	Setup value	Whichever is higher: max_IIOP_resp_con or max_processes	Maximum number of semaphores for each semaphore identifier
para2	Additional value	limit_of_max_IIOP_resp_con (*1) x 4 + max_IIOP_resp_con_extend_number (*2) + max_IIOP_resp_requests_extend_number (*2) + max_impl_rep_entries + max_processes x 4 + max_exec_instance + [Number of WorkUnit which specified Buffer Size and Buffer Number (WorkUnit definition)] x 2 + 14 or more	Total number of semaphores in system
		The above value + 1 or more	When the trace function is used
		The above value + 1 or more	When the snapshot function is used
		The above value + limit_of_max_IIOP_resp_con (*1) or more	When the SSL coordinated function is used
para3	Setup value	50 or more	Maximum number of operators for each semaphore call
para4	Additional value	The maximum value (that calculated using the following formula or 512) max_IIOP_resp_con_extend_number (*2) x 5 + max_IIOP_resp_requests_extend_number (*2) + max_impl_rep_entries + [Number of WorkUnit which specified Buffer Size and Buffer Number (WorkUnit definition)] x 2 + 100 or more	Total number of semaphore identifiers in system

*1 The default value of limit_of_[parameter name] is:

- [parameter name] x 1.3 (rounding up of the fractional part)

This is the default value even if 0 is specified.

If "MANUAL" is specified in the 'AutoConfigurationMode' configuration option of the isconfig.xml file, the value for making non-automatic expansion settings is shown:

- [parameter name]

*2 The default value of [parameter name]_extend_number is:

- (limit_of_[parameter name] - [parameter name]) / [parameter name] (rounding up of the fractional part)

This is the default value even if 0 is specified.

If "MANUAL" is specified in the 'AutoConfigurationMode' configuration option of the isconfig.xml file, the value for making non-automatic expansion settings is:

- 0

5.2.1.2.3 Message Queues

Table 5.35 Message Queues

Parameter	Type	Required amount (value to add)	Remarks (description)
kernel.msgmax	Setup value	16384 or more	Maximum message size
kernel.msgmnb	Setup value	32768 or more	Maximum number of messages that can be held in one message queue
kernel.msgmni	Additional value	512 or more	Maximum number of message queue IDs

5.2.1.2.4 Interface Repository

The system resources required to use an Interface Repository are as shown in the table below.

Shared Memory

Table 5.36 Shared Memory

Parameter (*1)	Type	Required amount (value to add)	Remarks (description)
Kernel.shmmax	Setup value	"logging memory size" + 16KB (*1)	Maximum shared memory segment size (When collecting log data)

*1 Specify "logging memory size" in the CORBA Service irconfig file. Refer to "irconfig" in the "CORBA Service Environment Definition" appendix for details.

5.2.1.2.5 Naming Service

The system resources required to create multiple naming contexts in the Naming Service are shown in the table below.

Table 5.37 Naming Service

Parameter	Required amount (value to add)	Remarks (description)
(*1)	Number of naming contexts + 16 or more	Number of files that can be opened for each process

*1 The relevant parameter does not exist.

Using the ulimit command for the Bourne shell and the limit command for the C shell, set the values only as far as required for opening the file with the naming service process. For details about commands, refer to the OS documents.

5.2.1.3 Number of Threads and Processes Used by Applications

When applications are to be executed in the CORBA Service, system parameters will need to be changed if large numbers of processes and threads are going to be generated by the applications.

The targets for the number of (multiple) threads started when applications are started are shown in the following table.

Table 5.38 Targets for the Number of Threads

Component	Number of Threads (*1)
CORBA Service	25 + number of connections to client application
Server application	(6 + number of thread primary multiples) for each process
Client application	8 (maximum) per process

The system parameters that require modification are shown in the following table.

Table 5.39 System Parameters

Parameter	Description
kernel.threads-max	Total number of processes started in the system

Table 5.40 Parameters other than System Parameters

Parameter	Description
(*1)	The maximum number of processes available to a single user
(*2)	The maximum stack size

*1 There is no corresponding parameter. Use the 'ulimit' (for bash or Bourne shell types) or 'limit' (for C shell types) command to make the settings. Set a value that is equal to or greater than the total number of processes and threads created by the user.

*2 The corresponding parameter does not exist. Use the 'ulimit' (for bash or Bourne shell types) or 'limit' (for C shell types) command to make the settings.

This value is multiplied by the number of threads, and the resulting value used in defining the size of the process stack area. A thread cannot be created if the memory that can be used in one process is exceeded. For this reason, the number of threads that can be created for one process is limited. Request processing concurrency for CORBA server applications and EJB applications is calculated as 'thread concurrency x process concurrency'. If it is not possible to increase the thread concurrency using the memory size that can be used per process, investigate the possibility of increasing the process concurrency. For details of thread concurrency/process concurrency for CORBA server applications, refer to descriptions of "proc_conc_max", "thr_conc_init" and "thr_conc_maximum" in "OD_impl_inst" in the "CORBA Service Operation Commands" chapter of the Reference Manual (Command Edition). For details of EJB application thread concurrency, refer to "Number of Simultaneous Processing Tasks" in "Tuning of EJB Containers" in the "Tuning J2EE Applications" chapter.

5.2.1.4 Number of File Descriptors

If multiple applications are going to be executed in the CORBA Service (because multiple terminals are connected, for instance), and the number of file descriptors used exceeds the default value, also set the parameter of the config file.

Table 5.41 File Descriptors

Parameter	Description
rlim_fd_cur (System parameter)	Set when the number of file descriptors in use exceeds the default value.
fs.file-max (System parameter)	Set when the number of file descriptors in use exceeds the default value.

5.2.2 Setting the System Environment for a Component Transaction Service

When the Component Transaction Service is in operation, the system resources must be expanded depending on the functions used. The system resources (system parameters) required by the Component Transaction Service and the system resources (system parameters) required by each function are described below.

Refer to the OS document for details of system parameter settings.

Note

The following values are not included in the values of the CORBA Service. Refer to "Setting the System Environment for a CORBA Service" and add the values as required.

5.2.2.1 System Parameters

This section describes tuning of the system parameters of shared memory as well as semaphores and message queues that the Component Transaction Service uses.

When various functions are going to be used in addition to the basic function of the Component Transaction Service, add the amount of resources that are used by each function to the resources of the Component Transaction Service basic function.

Note

In the subsequent descriptions, "type" has the following meaning.

- Setting value

  Change the values according to conditions applying to 'Required amount'.

- Additional value

  Add the 'Required amount' to the already set value.

5.2.2.1.1 Basic Function of the Component Transaction Service

The system resources that are required to use the basic function of the Component Transaction Service are shown in the following tables.

5.2.2.1.2 Shared Memory

Table 5.42 Shared Memory

Parameter	Type	Required amount	Description
kernel.shmmax	Setup value	10,629,296 or more	Maximum size of shared memory segment
kernel.shmmni	Additional value	Add 22	Number of shared memory IDs

5.2.2.1.3 Semaphores

For the semaphore setting value, specify each parameter in the following format.

- In the case of Red Hat

  kernel.sem = para1 para2 para3 para4

Table 5.43 Semaphores

Parameter	Type	Required amount	Description
para1	Setup value	12 or more	Maximum number of semaphores for each semaphore ID
para2	Additional value	Add 21	Number of semaphores of the entire system
para3	Setup value	3 or more	Maximum number of operations for each semaphore call
para4	Additional value	Add 29	Number of semaphore IDs

5.2.2.1.4 Message Queues

Table 5.44 Message Queues

Parameter	Type	Required amount	Description
kernel.msgmax	Setup value	528 or more	Maximum message size
kernel.msgmnb	Setup value	4572 + (528 x number of simultaneously executed commands)	Maximum number of bytes in the queue

Parameter	Type	Required amount	Description
kernel.msgmax	Setup value	528 or more	Maximum message size
		(*1)	
kernel.msgmni	Additional value	Add 11	Number of message queue IDs

*1 Number of simultaneously executed commands refers to the number of commands when the following commands are executed at the same time:

For Enterprise Edition:

isstartwu, isstopwu, tdstartwu, tdstopwu, tdinhibitobj, tdpermitobj, tdmodifyprocnum, tdmodifywu

For Standard-J Edition:

isstartwu, isstopwu

Linux32

When start/stop of the WorkUnit and object closure/closure release are executed using the Systemwalker OperationMGR and Interstage operation API, the number of simultaneous operations is counted as the number of simultaneously executed commands.

Object closure and closure cancellation can be used in the Enterprise Edition.

5.2.2.2 Session Information Management Function

This section describes the system resources that are additionally required when the session information management function is used.

5.2.2.2.1 Shared Memory

Table 5.45 Shared Memory

Parameter	Type	Required amount	Description
kernel.shmmni	Additional value	Add 1	Number of shared memory IDs

5.2.2.2.2 Semaphores

For the semaphore setting value, specify each parameter in the following format.

- In the case of Red Hat

  kernel.sem = para1 para2 para3 para4

  Table 5.46 Semaphores

Parameter	Type	Required amount	Description
para2	Additional value	Add 1	Number of semaphores of the entire system
para4	Additional value	Add 1	Number of semaphore IDs

5.2.2.3 Performance Monitoring Tool

In the Web server connector, system resources should be expanded when the following function is used.

- Web server connector fault monitoring function

The system resources (system parameters) required by the Web server connector are described below.

5.2.2.4 System Parameters

When the Web server connector fault monitoring function is used, tune the following system parameters.

5.2.2.4.1 Shared Memory

Table 5.47 Shared Memory

Parameter	Type	Required amount	Remarks (description)
Shmmax	Setup value	Value of 6,720,012 or more	Maximum segment size of shared memory
Shmmin	Setup value	Number of Web servers + 1	Minimum segment size of shared memory
Shmseg	Additional value	1	Number of shared memory segments that can be attached with a single process
Shmmni	Additional value	1	Number of identifiers of shared memory

5.2.2.4.2 Semaphores

Table 5.48 Semaphores

Parameter	Type	Required amount (value to add)	Remarks (description)
Semmni	Additional value	2 or more	Number of semaphore identifiers
Semmns	Additional value	3	Number of semaphores in the system
semmnu	Additional value	2 or more	Number of groups for cancellation of semaphore operation in the system
semume	Additional value	3	Number of groups for cancellation of semaphore operation per process

5.2.3 System Resources of the Event Service System

To run a system using an Event Service, you will need to increase the system resources according to the number of channels, the number of consumers/suppliers to be connected, and so forth. This section describes how to set the system resources (system parameters) required by the Event Service.

Note

The following values are not included in the values of the CORBA Service. Refer to "Setting the System Environment for a CORBA Service" and add the values as required.

5.2.3.1 System Parameters

The example below shows typical additions to the kernel parameters for shared memory, semaphores, and message queues used by an ordinary Event Service.

5.2.3.1.1 Changing the System Parameters

Modify the parameter value by editing /etc/sysctl.conf. After completing the modification, either execute "sysctl -p /etc/sysctl.conf" or reboot the system.

For the modification procedure, refer to the OS documentation.

5.2.3.1.2 Shared Memory

Table 5.49 Shared Memory

Parameter	Type	Required amount	Remarks (description)
kernel.shmmax	Setup value	2064 bytes x maximum number of event channel creations (*1) + 600 kilobytes + trace_size (*2) + 3328 bytes	Maximum segment size of shared memory (volatile channel operation)
		The maximum is specified using the following values: - 2064 bytes x maximum number of event channel creations (*1) + 184 bytes	Maximum segment size of shared memory (in persistent channel operation)

Parameter	Type	Required amount	Remarks (description)
		x number of simultaneously executable global transactions (*3) + 600 kilobytes + trace_size (*2) + 3328 bytes - 134217728 or more - Size of shared memory to be used by unit (This is the maximum size when two or more units exist.) (*4)	
kernel.shmmni	Additional value	- Trace is collected by the process (trace_buffer of traceconfig file = process) (*2) 4 + Number of EventChannel processes (*5) - Trace is collected by the Event Service (trace_buffer of traceconfig file = system) 4	Number of identifiers of shared memory (volatile channel operation)
		- Trace is collected by the process (trace_buffer of traceconfig file = process) (*2) Value of 100 or more (to be added in units of "unit") + Number of EventChannel processes (*5) - Trace is collected by the Event Service (trace_buffer of traceconfig file = system) Value of 100 or more (to be added in units of "unit")	Number of identifiers of shared memory (in persistent channel operation)

*1 Maximum number of event channels created = maximum number of static generation event channels created + maximum number of dynamic generation event channels created.

*2 trace_buffer and trace_size are specified in the Event Service operating environment file (traceconfig).

*3 Number of global transactions that can be executed simultaneously = -gtrnmax (use the configuration information management command (essetcnf) of the Event Service to set -gtrnmax)

*4 The size of shared memory is a set value for the item 'shmmax' and it is specified within the unit definition file for the command (esmkunit).

*5 Number of Event Channel processes =

Number of static Event Channel groups + Number of dynamic Event Channel processes

(Number of dynamic Event Channel processes: This value is set in the -p option of the Event Service setup command (essetup). If the Notification Service is used, set this value to "Number of dynamic Event Channel processes x 2".)

5.2.3.1.3 Semaphores

- For the semaphore setting value, specify each parameter in the following format.

  kernel.sem = para1 para2 para3 para4

Table 5.50 Semaphores

Parameter	Type	Required amount	Description
para1	Setup value	29	Maximum number of semaphores for each semaphore identifier
para2	Additional value	6 or more	Total number of semaphores in system (volatile channel operation)
		Unit number x 29 + 13 or more	Total number of semaphores in system (persistent channel operation)
para3	Setup value	29	Maximum number of operators for each semaphore call

Parameter	Type	Required amount	Description
para4	Additional value	Unit number x 256	Maximum number of semaphores for each semaphore identifier

5.2.3.1.4 Message Queues

Table 5.51 Message Queues

Parameter	Type	Required amount	Description
kernel.msgmax	Setup value	2,048 or more	Maximum message size (persistent channel operation)
kernel.msgmnb	Setup value	4,096 or more	Maximum number of bytes in the queue (persistent channel operation)
kernel.msgmni	Additional value	Unit number x 9	Number of message queue Ids (persistent channel operation)

5.2.4 System Resources of the IJServer or EJB Service

In the IJServer or EJB Service, the system resources should be expanded when the following functions are used. The system resources (system parameter) that the IJServer or EJB Service requires are described below.

5.2.4.1 System Parameters

When the IJServer or EJB Service is used, tune the following system parameters.

5.2.4.1.1 Changing the System Parameters

Modify the parameter value by editing /etc/sysctl.conf. Upon completion of modification, either execute "sysctl -p /etc/sysctl.conf" or reboot the system.

For the modification procedure, refer to the OS documentation.

5.2.4.1.2 Message Queues

Table 5.52 Message Queues

Parameter	Type	Required amount	Description
kernel.msgmax	Setup value	4096 or more	Maximum message size
kernel.msgmnb	Setup value	4096 or more	Maximum bytes in the queue
kernel.msgmni	Additional value	2 or more	Maximum value of message queue ID

5.2.5 System Resources of the Interstage HTTP Server

For system operation using the Interstage HTTP Server, it is necessary to expand the system resources. This section explains the system resources (system parameters).

5.2.5.1 System Parameters

This section details how to tune the system parameters used by the Interstage HTTP Server.

5.2.5.1.1 Changing the System Parameters

Edit /etc/sysctl.conf to change the parameter values. After making changes, execute "sysctl -p /etc/sysctl.conf" or reboot the system. To change the parameter values, set and add values according to the parameter type.

For details on changing system parameters, refer to the Linux documents.

Note

In the subsequent descriptions, "type" has the following meaning:

- Setting value

  Change the values according to conditions applying to 'Required amount'.

- Additional value

  Add the 'Required amount' to the already set value.

Semaphores

- For the semaphore setting value, specify each parameter in the following format:

  kernel.sem = para1 para2 para3 para4

Table 5.53 Semaphores

Parameter	Type	Required amount	Remarks (description)
para1	Setup value	1 or more	Maximum number of semaphores for each semaphore identifier
para2	Additional value	Number of Web servers x 2	Total number of semaphores in system
para3	Setup value	1 or more	Maximum number of operators for each semaphore call
para4	Additional value	Number of Web servers x 2	Maximum number of semaphores for each semaphore identifier

5.2.5.2 Number of File Descriptors

The number of file descriptors that are required to run Interstage HTTP Server will depend on the features that are used in the Web server and on the content that is defined in the environment definition file (httpd.conf).

Refer to the following table before calculating the number of required file descriptors.

If this value exceeds the system default for the resource limit, set this value in the /etc/security/limits.conf file.

If the number of file descriptors is insufficient, add this value as the system parameter setting. When multiple Web servers have been created, add the required number for each Web server.

Table 5.54 File Descriptors

Parameter	Type	Required amount
[RESOURCE_LIMIT] nofile	Setup value	20 or more
		[When the following features are used] Add the following values to the above: - Basic Authentication function: 1 - Online Collation function: 1 - SSL communication: 21 - Proxy function: 1 - CGI function (*1): 5
		[When the following directives are added to the environment definition file (httpd.conf)] Add the following values to the above: - CustomLog(specify ihsrlog command execution statement): 2 x directive number - CustomLog(not specify ihsrlog command execution statement): 1 x directive number - ErrorLog: 1 x directive number

Parameter	Type	Required amount
		- Listen: 1 x directive number
[System Parameter] fs.file-max	Additional value	Same as the above values

*1 Also add the number of file descriptors that are required in the CGI program that is to be run.

5.2.6 System Resources of the Interstage Single Sign-on

For system operation using Interstage Single Sign-on, it is necessary to expand the system resources. This section explains the system resources (system parameters).

5.2.6.1 System Parameters

This section explains the tuning of the system parameters used by Interstage Single Sign-on.

The values shown in the table below are not included in the values required for the Interstage HTTP Server. For details, refer to "5.2.5 System Resources of the Interstage HTTP Server" and add the required values.

5.2.6.1.1 Changing the System Parameters

Edit /etc/sysctl.conf to change the parameter values. After making changes, execute "sysctl -p /etc/sysctl.conf" or reboot the system. To change the parameter values, set and add values according to the parameter type.

For details on changing system parameters, refer to the Linux documents.

Note

In the subsequent descriptions, "type" has the following meaning.

- Setting value

  Change the values according to conditions applying to 'Required amount'.

- Additional value

  Add the 'Required amount' to the already set value.

5.2.6.1.2 Tuning System Parameters when using the Interstage Single Sign-on Repository Server Function

Shared Memory

Table 5.55 Shared Memory

Parameter	Type	Required amount	Remarks (description)
kernel.shmmax	Setting value	12800000 + (164 x number of site configurations (*1)) + (number of roles (*2) + number of role sets (*3) + number of role sets (*3) x number of roles (*2)) x 1024 or more (*4)	Maximum segment size for shared memory
kernel.shmmni	Additional value	13	Number of shared memory segments in the system

*1 Total number of site configurations registered as protected resources in the SSO repository

*2 The total number of roles defined in the SSO repository

*3 The total number of role sets defined in the SSO repository

*4 To use Active Directory as the directory service that is used for registering user information without using Single Sign-on extended schema, add the estimated value that is calculated using the following formula for the application.

Total number of attributes used for the Active Directory role/role set x 524

Semaphores

- For the semaphore setting value, specify each parameter in the following format.

  kernel.sem = para1 para2 para3 para4

Table 5.56 Semaphores

Parameter	Type	Required amount (value to add)	Remarks (description)
para2	Additional value	9	Total number of semaphores in system
para4	Additional value	9	Maximum number of semaphores for each semaphore identifier

"Interstage Directory Service" is used in the repository server. Refer to "5.2.7 System Resources of the Interstage Directory Service" for details of the tuning required.

5.2.6.1.3 Tuning System Parameters when using the Interstage Single Sign-on Authentication Server Function

Shared Memory

Table 5.57 Shared Memory

Parameter	Type	Required amount	Remarks (description)
kernel.shmmax	Setting value	14000000 + Total no. of path configurations (*1) x 2048 or more	Maximum segment size for shared memory
kernel.shmmni	Additional value	11	Number of shared memory segments in the system

*1 Total number of path configurations in each site configurations registered as protected resources in the SSO repository

Semaphores

- For the semaphore setting value, specify each parameter in the following format.

  kernel.sem = para1 para2 para3 para4

Table 5.58 Semaphores

Parameter	Type	Required amount (value to add)	Remarks (description)
para2	Additional value	10	Total number of semaphores in system
para4	Additional value	10	Maximum number of semaphores for each semaphore identifier

5.2.6.1.4 Tuning System Parameters when using the Interstage Single Sign-on Business Server Function

Shared Memory

Table 5.59 Shared Memory

Parameter	Type	Required amount	Remarks (description)
kernel.shmmax	Setting value	13000000 + (1 + cache size (*1)) x 1024 x cache count (*2) + (number of roles (*3) + number of extended user information (*4) + 1) x largest number of path configurations (*5) x 1200 or more	Maximum segment size for shared memory

Parameter	Type	Required amount	Remarks (description)
kernel.shmmni	Additional value	10 x number of business server	Number of shared memory segments in the system

*1 When session management is used, the size (Kbytes) of the authentication information about the user cached in the business server is set. In applications in which session management is not used, this is calculated as 0. For details about the cache size, refer to "Tuning for Setting up the Business Server" in the "Environment Definition for Interstage Single Sign-on" appendix.

*2 When session management is used, a value greater than the maximum number of simultaneous accesses is set. In applications in which session management is not used, this is calculated as 0. For details about the cache count, refer to "Tuning for Setting up the Business Server" in the "Environment Definition for Interstage Single Sign-on" appendix.

*3 Total number of roles defined in the SSO repository.

*4 Number of items of user information that is notified to the business application

When session management is used, the number of attribute names set in the Interstage Management Console of the repository server is set, as shown below. In applications in which session management is not used, this is calculated as 0.

[System] > [Security] > [Single Sign-on] > [Authentication infrastructure] > [Repository Server] > [Settings] > [Repository server detailed settings [Show]] > [Information notified to the Business System] > [Extended user information]

*5 This is the largest number of path configurations across all site configurations that are registered as protection resources in the SSO repository

Semaphores

- For the semaphore setting value, specify each parameter in the following format.

  kernel.sem = para1 para2 para3 para4

Table 5.60 Semaphores

Parameter	Type	Required amount (value to add)	Remarks (description)
para2	Additional value	7 x number of business server	Total number of semaphores in system
para4	Additional value	7 x number of business server	Maximum number of semaphores for each semaphore identifier

5.2.7 System Resources of the Interstage Directory Service

For system operation using Interstage Directory Service, it is necessary to expand the system resources. This section explains the system resources (system parameters).

5.2.7.1 System Parameters (System Resources Required for Interstage Directory Service)

This section explains the tuning of the system parameters used by Interstage Directory Service.

5.2.7.1.1 Changing the System Parameters

Edit /etc/sysctl.conf to change the parameter values. After making changes, execute "sysctl -p /etc/sysctl.conf" or reboot the system.

For details on changing system parameters, refer to the OS documents.

In subsequent descriptions, "type" has the following meaning:

- Setting value

  Change the values according to conditions applying to 'Required amount'.

- Additional value

  Add the 'Required amount' to the value already set.

Shared Memory

Parameter	Type	Required amount	Remarks (description)
kernel.shmmax	Setting value	(5 x (number of repositories x 1,843,200)) or more	Maximum segment size for shared memory
kernel.shmmni	Additional value	4 x (number of repositories)	Number of shared memory segments in the system

Tune system resources required for running the RDB, if an RDB is used as the repository database. Refer to the following items before changing system parameters:

- 5.2.7.2 System Parameters (System Resources Required for Symfoware Server)
- 5.2.7.3 System Parameters (System Resources Required for Oracle Database)

5.2.7.2 System Parameters (System Resources Required for Symfoware Server)

Change the system parameters in the machine used to install Symfoware Server. These system parameter s must be set to the values in the table below in the Symfoware Server system operating environment file or in the RDB configuration parameter file. When the value of the parameter of the table below is changed, refer to the Symfoware Server manual and re-calculate the values for the required system parameters.

Parameter	Required value (bytes)
Memory used in the local connection (COMMUNICATION_BUFFER)	32K
Number of local connections (MAX_CONNECT_SYS) (*1)	256
Daemon concurrency (RDBCNTNUM)	712
Shared memory (RDBEXTMEM)	13,208K

*1 Add the number of connections used by other applications to the maximum number of connections from the repository to the RDB that is required to use the Interstage Directory Service before making the calculation.

If the result exceeds the value set for the number of local connections (256), re-calculate the value for the required system parameters.

For details on the maximum number of repository connections, refer to "Specifying the Maximum Number of Connections" in the "Creating a Load Distribution Environment" chapter of the Directory Service Operator's Guide.

Installing Symfoware Server

Shared Memory

Parameter	Type	Required amount	Remarks (description)
kernel.shmmax	Setting value	13,524,992 or more	Maximum segment size for shared memory
kernel.shmmni	Additional value	10	Number of shared memory segments in the system

Semaphores

- For the semaphore setting value, specify each parameter in the following format.

  kernel.sem = para1 para2 para3 para4

Parameter	Type	Required amount	Remarks (description)
para1	Setting value	48 or more	Maximum number of semaphores for each semaphore ID
para2	Additional value	1,112	Number of semaphores in the system
para3		The value that is already set	Number of groups for cancellation of semaphore operation in the system

Parameter	Type	Required amount	Remarks (description)
para4	Additional value	300	Number of semaphore identifiers

Message Queues

Parameter	Type	Required amount	Description
kernel.msgmax	Setup value	128 or more	Maximum message size
kernel.msgmnb	Setup value	4,096 or more	Maximum number of bytes in the queue
kernel.msgmni	Additional value	2	Number of message queue IDs

5.2.7.3 System Parameters (System Resources Required for Oracle Database)

Change the system parameters in the machine used to install Oracle Database. Configure system resources required to run the replication server, if one is being used. For details on settings, refer to the Oracle database manual.

Installing Oracle Database

Shared Memory

Parameter	Type	Required amount	Remarks (description)
kernel.shmall	Setting value	2,097,152 or more	Maximum size for shared memory over entire system
kernel.shmmax	Setting value	At least half the physical memory size (in bytes)	Maximum segment size for shared memory
kernel.shmmni	Setting value	4,096 or more	Number of shared memory segments in the system

Semaphores

- For the semaphore setting value, specify each parameter in the following format.

  kernel.sem = para1 para2 para3 para4

Parameter	Type	Required amount	Remarks (description)
para1	Setting value	250 or more	Maximum number of semaphores for each semaphore ID
para2	Setting value	32,000 or more	Number of semaphores in the system
para3	Setting value	100 or more	Maximum number of operators for each semaphore call
para4	Setting value	128 or more	Number of semaphore identifiers

File Systems

Parameter	Type	Required amount	Description
fs.file-max	Setup value	65,536 or more	Maximum number of file handles

Network

Parameter	Type	Required amount	Description
net.ipv4.ip_local_port_range	Setup value	Minimum: 1,024 Maximum: 65,000	Port number range
net.core.rmem_default	Setup value	1,048,576 or more	Default window size (receive)

Parameter	Type	Required amount	Description
net.core.rmem_max	Setup value	1,048,576 or more	Maximum window size (receive)
net.core.wmem_default	Setup value	262,144 or more	Default window size (transmit)
net.core.wmem_max	Setup value	262,144 or more	Maximum window size (transmit)

5.2.8 System Resources of the Interstage Management Console

For system operation using the Interstage Management Console, it is necessary to expand the system resources. This section explains the system resources (system parameters).

5.2.8.1 System Parameters

This section details how to tune the system parameters used by the Interstage Management Console.

5.2.8.1.1 Changing the System Parameters

Edit /etc/sysctl.conf to change the parameter values. After making changes, execute "sysctl -p /etc/sysctl.conf" or reboot the system. To change the parameter values, set and add values according to the parameter type.

For details on changing system parameters, refer to the Linux documents.

Note

In the subsequent descriptions, "type" has the following meaning:

- Setting value

  Change the values according to the conditions applying to 'Required amount'.

- Additional value

  Add the 'Required amount' to the already set value.

Semaphores

- To configure values for semaphores, specify each parameter in the following format:

  kernel.sem = para1 para2 para3 para4

Table 5.61 Semaphores

Parameter	Type	Required Value	Remarks (description)
para1	Setup value	1 or more	Maximum number of semaphores for each semaphore identifier
para2	Additional value	1	Total number of semaphores in the system
para3	Setup value	1 or more	Maximum number of operators for each semaphore call
para4	Additional value	1	Maximum number of semaphores for each semaphore identifier

5.2.9 Setting the System Environment for the Interstage Setup Commands

When Java EE, the multilanguage service, or the J2EE compatibility feature are used, in addition to the system resources that are added with each feature it will also be necessary to extend these system resources with the Interstage Setup commands.

5.2.9.1 System Parameters

This section describes tuning of the system parameters of shared memory as well as semaphores and message queues that the Setup commands Setup commands Service uses.

5.2.9.1.1 Shared Memory

Table 5.62 Shared Memory

Parameter	Type	Required amount	Description
kernel.shmmax	Setup value	(For 32 bit) 12,503,496 or more (For 64 bit) 13,208,308 or more	Maximum size of shared memory segment
kernel.shmmni	Additional value	Add 19	Number of shared memory IDs

5.2.9.1.2 Semaphores

For the semaphore setting value, specify each parameter in the following format.

- In the case of Red Hat

  kernel.sem = para1 para2 para3 para4

Table 5.63 Semaphores

Parameter	Type	Required amount	Description
para1	Setup value	12 or more	Maximum number of semaphores for each semaphore ID
para2	Additional value	Add 2	Number of semaphores of the entire system
para3	Setup value	3 or more	Maximum number of operations for each semaphore call
para4	Additional value	Add 2	Number of semaphore IDs

5.2.9.1.3 Message Queues

Table 5.64 Message Queues

Parameter	Type	Required amount	Description
kernel.msgmax	Setup value	528 or more	Maximum message size
kernel.msgmnb	Setup value	4572 + (528 x number of simultaneously executed commands) (*1)	Maximum number of bytes in the queue
kernel.msgmni	Additional value	Add 12	Number of message queue IDs

*1 Number of simultaneously executed commands refers to the number of commands when the Setup commands are executed at the same time.

Linux32

When start/stop of the WorkUnit and object closure/closure release are executed using the Systemwalker Operation Manager and Interstage operation API, the number of simultaneous operations is counted as the number of simultaneously executed commands.

5.2.10 System Resources Required by a Web Server Connector

For system operation using Web server connector, it is necessary to expand the system resources. This section explains the system resources (system parameters).

5.2.10.1 System Parameters

When the Web server connector is used, tune the following system parameters.

When Web server connector fault monitoring is used, add the resource amount used for Web server connector fault monitoring to the resource amount used for the Web server connector.

5.2.10.1.1 Changing the System Parameters

Modify the parameter value by editing /etc/sysctl.conf. After completing the modification, either execute "sysctl -p /etc/sysctl.conf" or reboot the system.

For the modification procedure, refer to the OS documentation.

Note

In the subsequent descriptions, "type" has the following meaning.

- Setting value

  Change the values according to conditions applying to 'Required amount'.

- Additional value

  Add the 'Required amount' to the already set value.

5.2.10.2 Web Server Connector

The system resources required for using Web server connector fault monitoring are shown in the table below.

5.2.10.2.1 Semaphores

- For the semaphore setting value, specify each parameter in the following format.

  kernel.sem = para1 para2 para3 para4

Table 5.65 Semaphores

Parameter	Type	Required amount	Description
para1	Setting value	1 or more	Maximum number of semaphores for each semaphore ID
para2	Additional value	Total required across all Web servers (*1) Each web server requires: ((IJServer WorkUnit total + IJServer WorkUnit process concurrency total + 2) x 2) + 4	Number of semaphores over the entire system
para3	Setting value	2 or more	Maximum number of operations for each semaphore call
para4	Additional value	Total required across all Web servers (*1) Each web server requires: ((IJServer WorkUnit total + IJServer WorkUnit process concurrency total + 2) x 2) + 4	Number of semaphore IDs

*1 The following is a tuning example:

- There are two Web servers (web001 and web002)
- There are three IJServer WorkUnits (WU001, WU002, and WU003)
- WorkUnit process concurrency is set to 3 on each IJServer

- web001 connector destinations are WU001 and WU002; web002 connector destination is WU003
    - para2

      Required number of semaphores in each web server

      = ((IJSe0.rver WorkUnit total + WorkUnit process concurrency total for each IJServer + 2) x 2) + 4

      Required number of semaphores in web001

      = ((2 + 6 + 2) x 2) + 4

      = 24

      Required number of semaphores in web002

      = ((1 + 3 + 2) x 2) + 4

      = 16

      para2 = Total required across all Web servers

      = Required web001 number + Required web002 number

      = 24 + 16

      = 40

    - para4

      Calculate this using the method used for para2.

5.2.10.3 Web Server Connector Fault Monitoring

Additional system resources used for Web server connector fault monitoring are shown in the table below.

5.2.10.3.1 Shared Memory

Table 5.66 Shared Memory

Parameter	Type	Required amount	Remarks (description)
kernel.shmmax	Setting value	Value of 6,720,012 or more	Maximum segment size of shared memory
kernel.shmmni	Additional value	Number of Web servers + 1	Number of identifiers of shared memory

5.2.10.3.2 Semaphores

- For the semaphore setting value, specify each parameter in the following format.

  kernel.sem = para1 para2 para3 para4

Table 5.67 Semaphores

Parameter	Type	Required amount	Description
para1	Setting value	2 or more	Maximum number of semaphores for each semaphore ID
para2	Additional value	3	Number of semaphores of the entire system
para3	Setting value	2 or more	Maximum number of operations for each semaphore call
para4	Additional value	3	Number of semaphore IDs

5.3 System Resources for Operating the Multi Server Function

Solaris32 Linux32/64

This section describes the system resource required by the Multi Server function.

5.3.1 System Resources Required by the Admin Function

When using the Admin Server function, the system resources are required for each server function service shown below:

- Interstage Directory Service

5.3.2 System Resources Required by Managed Server

When using the Managed Server function, the system resources are required for each server function service. For details of each service, refer to "5.1 System Resources for Operating Interstage Server Function (for Solaris)", and "5.2 System Resources for Operating Interstage Server Function (for Linux)".

5.3.3 System Resources Required by Combined Function

In the Combined Server, the Admin Server function and the Interstage Server function (the Managed Server) are run on the same machine. For this reason, in operations where the server is run as the Combined Server, the system resources needed to use the Admin Server function and the Managed Server function, will be required. For details of the system resources that are required for each service, refer to "5.1 System Resources for Operating Interstage Server Function (for Solaris)", and "5.2 System Resources for Operating Interstage Server Function (for Linux)".

Note

To use the same service in the Admin Server function and the Interstage Server function, there is no need to count the system resources twice.

5.4 Environment Setup for Performance Monitoring Tool

Windows32/64

This section describes how to estimate the shared memory used by the performance monitoring tool as the performance monitoring environment.

Solaris32/64 Linux32/64

This section describes how to estimate the resources used by the performance monitoring tool as the performance monitoring environment.

5.4.1 Estimating Shared Memory

Solaris32/64

Use the following reference estimation formula for determining the amount of shared memory. Round off the value obtained to the nearest megabyte.

Compare with the estimated value specified by shmsys:shminfo_shmmax in the /etc/system file and set the larger value in shmsys:shminfo_shmmax. After changing the settings, reboot the system. Refer to the Solaris documents for details on the changing the settings.

Linux32/64

Use the following reference estimation formula for determining the amount of shared memory. Round off the value obtained to the nearest megabyte.

5.4.1.1 Memory Estimation (Bytes)

1. For all objects that are to be monitored by the performance monitoring tool, find the amount of common memory required for each object.

   - Determine the total process concurrency defined for each application.

   - Find the average number of operations registered in each application. (Find the number of operations from the IDL definition file. Discard decimals in the averages.)

- Depending on the average number of operations, find the amount of common memory required for each object in the following way:

  If the average number of operations is 3 or less

  Common memory required for each object = (Total process concurrency x 1536) + 400

  If the average number of operations is greater than 3

  Common memory required for each object = (Average number of operations x Total process concurrency x 546) + 400

2. Find the amount of shared memory required in the following way:

   Amount of shared memory required = Total amount of shared memory required for WorkUnits + 261188

3. Find the amount of shared memory in the following way:

   Amount of shared memory = Amount of shared memory required / 1048576

Round up any decimals in this calculation.

5.4.2 Estimating System Configuration Information

Solaris32/64 Linux32/64

This section describes how to estimate system configuration information. Set values that are equal to or greater than the estimated values. The following table shows the system configuration values.

Solaris32/64

Table 5.68 System Configuration Values

System Configuration Information	Estimation
Semsys: Seminfo_semmnu (*)	semsys:seminfo_semmnu >= Existing value + Semaphore quantity Semaphore quantity Share memory capacity specified when performance monitoring tool started (MB) x 10 + 2 Maximum is 52
semsys: seminfo_semmni	Add 1
semsys: seminfo_ semmns	Add the same value as semsys:seminfo_semmnu. The parameter semsys:seminfo_semmnu makes comparison based on the value estimated by (*).
semsys: seminfo_ semmsl	Compared with semsys:seminfo_semmnu and set the larger value.
shmsys: shminfo_shmmax	Refer to "5.4.1 Estimating Shared Memory".
shmsys: shminfo_shmmni	Add 1
shmsys: shminfo_shmmin	Set 1
shmsys: shminfo_shmseg	Add 1

5.4.2.1 Semaphores

- Linux32/64

  kernel.sem = para1 para2 para3 para4

Table 5.69 System Configuration Values

System Configuration Information	Estimation
para1	Compare with the following estimate values and set the larger one. para1 >= Existing value + Semaphore quantity(*)

System Configuration Information	Estimation
	Semaphore quantity = Shared memory capacity specified when performance monitoring tool started (MB) x 10 + 2 but Maximum is 52
para2	The same value as para1 is added.
para3	Compare with para1 and set the larger one. Para1 is compared based on the value which (*) estimated.
para4	Add 1

5.4.2.2 Shared Memory

Linux32/64

System Configuration Information	Estimation
kernel.shmmax	Refer to "5.4.1 Estimating Shared Memory".
kernel.shmmni	Add 1

5.5 Tuning TCP/IP Parameters

Tune the TCP/IP parameters as follows.

Windows32/64

Using the registry editor, add the following registry information and then restart the system.

- Registry key

  HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters

  - Name : TcpTimedWaitDelay
  - Type : REG_DWORD
  - Recommended value : 1E(30 seconds)

- Registry key

  HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters

  - Name : MaxUserPort
  - Type : REG_DWORD
  - Recommended value : 65534(decimal)

If a large number of clients connect in a short period of time, there may not be enough TCP/IP sockets and "Internal Server Error" may be output in the Web browser.

If the above information is added, the number of available sockets is increased, and sockets that have already been used open quickly.

Note

- Create registry information if it does not exist.
- To turn the TCP/IP parameters, change the values of the above two registry keys.
- If the registry edit operation is incorrect, the system may become unstable. For this reason, only change the specified keys when performing this operation. Do not change any other keys. Additionally, back up the registry before performing this operation.
- TCP/IP parameter tuning affects all TCPs. Consult the system administrator before tuning TCP/IP parameters.

Solaris32/64

Use the ndd command to change tcp_time_wait_interval to 60 seconds. The OS default value is 60 seconds, so it does not need to be set if the value was not changed.

To make the setting valid permanently, it must be added to the RC procedure (/etc/rc2.d).

An example of RC procedure is shown below.

```
#!/bin/sh
ndd -set /dev/tcp tcp_time_wait_interval 60000
```

If a large number of clients connect in a short period of time, there may not be enough TCP/IP sockets and "Internal Server Error" may be output to the Web browser.

If the above information is added, the number of available sockets is increased, and sockets that have already been used will open quickly.

Note

- TCP/IP parameter tuning affects all TCPs. Consult the system administrator before tuning these parameters.

5.6 Other Customization Items

Solaris32/64 Linux32/64

The other required customization items are described below.

5.6.1 Customizing the IPC Key Value of the System V IPC Resource

Interstage uses the System V IPC resources (message queue, semaphore and shared memory) that the OS provides for communication between the processes constituting Interstage. These IPC resources can be uniquely identified within the system by the value (IPC key value) specified during resource creation.

The IPC key value must be unique in a system, but can be duplicated with the products and application programs using other IPC resource because arbitrary value can be used.

If duplication of the IPC key value occurs, Interstage issues the following message to notify duplication of the IPC key value.

Example of message that is output from the Component Transaction Service when duplication of IPC key values is detected.

TD: ERROR: td11038: Could not get IPC resource due to duplicated key(key=%x path=%s)

In such a case, various functions of the Interstage service that uses the IPC resource corresponding to the IPC key value cannot be used.

The IPC key value that Interstage uses can be customized in order to prevent occurrence of this problem. This measure can be used when the message notifying duplication of IPC key value is output.

5.6.1.1 Overview

The IPC key value consists of 4 bytes (32 bits). Duplication with the IPC key values used in other products can be prevented by defining an arbitrary value in the lower 12 bits (3 digits in hexadecimal notation). The remaining upper 20 bits are determined by Interstage.

5.6.1.2 Method of Defining the IPC Key Value

Create a new IPC key value definition file as shown below to specify the lower 12 bits of the IPC key value using a 3-digit hexadecimal number.

Solaris32/64

For services, the specification in the common definition file is effective.

Common definition file

```
/var/opt/FJSVisas/system/system name/FJSVisas/etc/ipc_key
```

Linux32/64

Common definition file

```
/var/opt/FJSVisas/system/default/FJSVisas/etc/ipc_key
```

Note

- If the content of the definition file is other than a 3-digit hexadecimal number, the same operation as when no IPC key value is specified is performed.

Solaris32

- "System name" is the system name of the multi system function. If the multi system function is not used, "default" is set.

Note

- When defining the IPC key value, stop Interstage using the all forced stop mode beforehand. Do not modify this definition while Interstage is operating.

Solaris32/64

- Define the IPC key value so that it is not duplicated among the systems

Chapter 6 JDK/JRE Tuning

- In C and C++, areas allocated to memory must be explicitly released when they are no longer required. If this release process is omitted or performed incorrectly, memory leaks may occur and/or programs may freeze.
- Java reduces the memory management workload for programmers by introducing a process called garbage collection (GC). Garbage collection needs to be managed, however, so that performance is not degraded by Java applications pausing each time garbage collection occurs. There are also memory leaks peculiar to Java.
- In addition to these factors, there is a tendency to create large numbers of threads because they are easy to manage in Java. This can generate a large stack and cause memory shortages.

For these reasons, JDK/JRE tuning is required to adjust the stack size or the frequency of garbage collection and its processing time. This chapter provides the basic knowledge and methods required for tuning Java applications.

6.1 Basic Knowledge

This section provides the knowledge required to tune JDK/JRE.

6.1.1 JDK Documentation

For JDK 6 and JDK 7 documentation, refer to the following URLs:

- JDK 6: http://download.oracle.com/javase/6/docs/
- JDK 7: http://download.oracle.com/javase/7/docs/

These URLs are correct as of 04 October, 2011.

The documents provided at the above URLs are also stored in the following files on the Manual package:

- JDK 6 \ApplicationServer\javadocs\jdk6-jdkdocs.zip
- JDK 7 \ApplicationServer\javadocs\jdk7-jdkdocs.zip

Note

The documentation for Java VisualVM is not included on the Manual package.

For information on Java VisualVM, refer to the following URL:

- JDK6

  http://docs.oracle.com/javase/6/docs/technotes/guides/visualvm/index.html

- JDK7

  http://docs.oracle.com/javase/7/docs/technotes/guides/visualvm/index.html

Note that Java VisualVM is a tool added as of JDK 6.

Note

The documentation for jcmd is not included in the Manual package.

For details on jcmd, refer to the following URL:

- Windows32/64

  http://docs.oracle.com/javase/7/docs/technotes/tools/windows/jcmd.html

- Solaris32/64 Linux32/64

  http://docs.oracle.com/javase/7/docs/technotes/tools/solaris/jcmd.html

Note that jcmd is a new tool, added in JDK 7.

The java command and Java VM options also include tuning-related options. These options are introduced in this chapter as required.

For details on these options, refer to the following:

- Java command options

  "java" in "Tool Docs" in the JDK documentation

- Java HotSpot VM Options

  http://www.oracle.com/technetwork/java/javase/tech/vmoptions-jsp-140102.html (as of 06 April, 2011)

Options related to Java VM tuning not specifically explained in this manual are not supported in FJVM.

Java Platform Migration Guide

For information on compatibility and other problems when migrating Java programs developed in JDK 1.3 to JDK 5.0, refer to the following URL:

- http://www.oracle.com/technetwork/java/javase/community/jm-white-paper-r6a-149981.pdf

For details on migrating from JDK5.0 to JDK6, refer to the following URL:

- http://www.oracle.com/technetwork/java/javase/adoptionguide-137484.html

For details on migrating from JDK6 to JDK7, refer to the following URL:

- http://docs.oracle.com/javase/7/docs/webnotes/adoptionGuide/index.html

6.1.2 Java VM

Java VM (Included in Product)

JDK/JRE contains a Java virtual machine (Java VM) that functions as an engine for interpreting and executing Java bytecode.

The table below lists the different types of Java VM and their features.

Table 6.1 Java VM Types and Features

Java VM	Features
Java HotSpot Client VM	A Java VM for client environments. It is designed to shorten application startup times and reduce memory consumption. Fujitsu's **Java HotSpot Client VM** is based on Oracle Corporation's Java VM, "Java HotSpot Client VM", but includes additional features that employ original Fujitsu technology, such as enhancement of the troubleshooting functionality. Accordingly, it is inherently functionally compatible with the Java HotSpot Client VM on which it is based.
Java HotSpot Server VM (FJVM)	A Java VM for server environments. It is designed for improved stability and throughput rather than shorter application startup times. Fujitsu's **Java HotSpot Server VM** is based on Oracle Corporation's Java VM, "Java HotSpot Server VM", but includes additional features that employ original Fujitsu technology, such as performance improvement and enhancement of the troubleshooting functionality. Accordingly, it is inherently functionally compatible with the Java HotSpot Server VM on which it is based. Since this is the default Fujitsu version Java VM in Interstage Application Server, this Java VM is called FJVM.

Interstage Application Server is equipped with the Java HotSpot Client VM and the FJVM (Java HotSpot Server VM).

The FJVM is the default Java VM. It can also be selected for use by specifying the java command option "-server" or "-fjvm". Specify the option "-client" to use the Java HotSpot Client VM.

Windows64 Solaris64 Linux64

Only the FJVM is supported. The Java HotSpot Client VM is not available.

Note

In Fujitsu version JDK/JRE 6 and 7, the JDK/JRE 6 and 7 ergonomics function, Java VM automatic selection function (the function for selecting the Java VM that is used automatically according to the machine CPU and the physical memory) is disabled.

FJVM Details

Refer to 6.1.3 FJVM for a detailed description of the FJVM.

Java VM-related Information

For detailed information about the Java VM, refer to the following JDK documents:

- JDK 6:

  [Java SE 6 Overview] > [Java Virtual Machines] and

  http://download.oracle.com/javase/6/docs/technotes/guides/vm/index.html

- JDK 7:

  [Description of Java Conceptual Diagram] > [Java Virtual Machine] and

  http://docs.oracle.com/javase/7/docs/technotes/guides/vm/index.html

Information can also be obtained from the following sources:

- Java Language and Virtual Machine Specifications

  http://docs.oracle.com/javase/specs/

- Java HotSpot VM Options
- http://www.oracle.com/technetwork/java/javase/tech/vmoptions-jsp-140102.html
- Java SE 6 HotSpot Virtual Machine Garbage Collection Tuning
- http://www.oracle.com/technetwork/java/javase/gc-tuning-6-140523.html

Document categories and URLs mentioned were confirmed as correct as of 06 April, 2011.

6.1.3 FJVM

Information regarding Java VM in this section primarily refers to the "Fujitsu Java HotSpot Server VM (FJVM)", which is the default Java VM. Except where a specific disclaimer is given, that information is also equally applicable to the "Fujitsu Java HotSpot Client VM" (Java HotSpot Client VM).

The Fujitsu technology-based enhanced and unique functions provided in the FJVM are listed below.

With the exception of the FJVM-specific functions, this list also applies to Java HotSpot Client VM.

- 6.2.3 GC with the New Generation Area Control Processing Parallelization Function (Parallel GC)
- 6.2.4 Parallel GC with Concurrent Mark Sweep (Parallel GC with CMS)
- 6.2.5 Object Reference Compression Function (*1)
- 6.2.6 Outputting a Garbage Collection Log
- 6.3.1 Compiler Error Automatic Recovery Function (*1)
- 6.3.2 Long Time Compilation Detection Function (*1)
- 6.3.3 Dynamic Compile Status Log Output Function (*1)
- 6.5.4 Class Instance Information Output Function
- 6.5.5 Stack Trace Output Function that runs when java.lang.System.gc() is Executed
- 6.5.6 Java VM Termination Status Output Function
- 6.5.7 Time Information Format Specification Function in the Log Output
- 6.5.8 FJVM Log
- 6.6.1.6 Enhanced Message Output Function for Insufficient Memory Size Events
- 6.6.3.1 Stack Overflow Message Output Function

1. *1FJVM-specific function only provided in the FJVM.

6.1.4 Virtual Memory and the Virtual Address Space

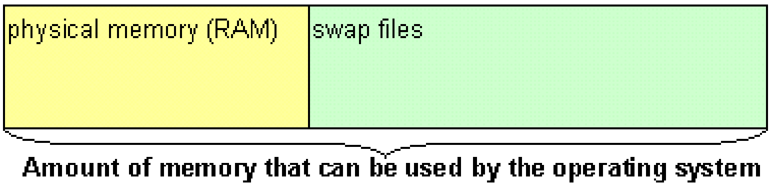
A general understanding of how the operating system manages memory is required when developing and running Java applications. This section explains a memory management technique known as the virtual address space that is commonly used by operating systems.

The actual architecture of the virtual address space varies between operating systems, but the information contained in this section is common to all operating systems.

Virtual Memory

The operating system can utilize many memory areas by using both physical memory (RAM) and swap files. This technology is referred to as virtual memory. The capacity of a system's virtual memory is the total amount of RAM plus the size of the swap file(s).

Figure 6.1 Operating System Memory Capacity



Note, however, that access to the hard disk is slower than access to RAM, so memory swapping can have a major impact on performance.

Virtual Address Space

When a program runs on an operating system, the operating system generates a process that becomes the unit by which the program is executed and managed. Likewise, a Java process is generated when a Java application is run.

A virtual address space is allocated to a process generated on the operating system. Virtual address spaces are created independently for each process, and one process cannot access the virtual address space of another. Regardless of the amount of physical memory (RAM) installed in a system, the size of the virtual address space is always constant. For example, in an operating system with a 32-bit architecture, the size of the virtual address space is always 4 GB (232 bytes), regardless of the size of the physical memory (RAM).

This means that even if a large amount of **virtual memory** is available, the maximum amount of memory that can be used by a single process can be no more than the **maximum virtual address space**. If a process needs more memory than the size of the **virtual address space** (actually the size of the user space referred to below), a memory shortage occurs.

However, even if the memory needed by the process is less than the size of the virtual address space (actually the size of the user space referred to below), a memory shortage will still occur if the operating system does not have enough virtual memory to meet the memory needs of the process.

Likewise, if a system has a large amount of virtual memory, and a large number of processes are running and consuming large amounts of virtual memory, a memory shortage can occur even if the virtual address space has not been used up.

User Space

The area of a virtual address space that a process can actually use is referred to as the user space. The user space contains a copy of the body of the program (for example, java.exe when a Java application is run under Windows(R)), and also contains segments such as the stack and the heap. The user space is used by running programs, and by the operating system programs used to run those programs. The user space of a Java process not only contains the segments mentioned above, but also a segment for storing Java objects (the Java heap). This means that the maximum size of the Java heap targeted for Java application tuning must be smaller than the user space.

When tuning Java applications, it is necessary to consider the amount of virtual memory and the processes that are being used. Care is needed since the maximum available **user space** varies depending on the type of operating system and the application execution mode.

The actual control processing for allocating and releasing segments is carried out by the operating system. For this reason, how each segment is managed, its behavioral specs and its size will depend on the specs of the OS that executes JDK/JRE. Even in situations where it appears from outside the process that there is **free user space** available, in some cases the operating system, due to its control processing, determines that there is no available space, so that the memory space for the segment cannot be allocated and the process ends in an error.

There may also be situations where, due to the operating system control processing, there is intervening space between segments that cannot be used from the application. For this reason, the amount of virtual memory that can be used is normally less than the upper limit on the user space. So applications should not be designed with the assumption that the maximum user space size will necessarily be available as the amount of memory used by a process.

Execution Mode

Changing the execution mode changes the amount of basic memory required to run a program.

Specifically, if you change the program execution mode from 32-bit mode to 64-bit mode before running the program, the memory size unit required for handling pointers changes from 4 bytes to 8 bytes. (Depending on the operating system, the memory size unit required when handling integers may also change.) Consequently, even where programs have been created from the same source, running applications using a 64-bit mode program on a 64-bit operating system will require more memory than running applications using a 32-bit mode program on a 32-bit or 64-bit operating system.

C Data Type Model in 64-bit Mode

Solaris64 Linux64

LP64 model: A long type/pointer is changed from 32 bits (4 bytes) to 64 bits (8 bytes).

Windows64

P64 model: A pointer is changed from 32 bits (4 bytes) to 64 bits (8 bytes).

For Java applications, just as for C/C++ applications, changing the execution mode changes the amount of basic memory required to run a program.

Particularly since the data content that makes up objects includes large amounts of information that handles pointers, the size of the memory needed per object in 64-bit mode is larger than in 32-bit mode.

This means that if you are running a Java application using 64-bit mode JDK/JRE on a 64-bit operating system, you will need a Java heap size that is 1.5 to 2 times larger than the size you would set to run a Java application using 32-bit mode JDK/JRE on a 32-bit or 64-bit operating system.

6.1.5 Stack

This section explains the stack.

Stack

When a program generates a thread, the operating system automatically allocates an area of memory called a stack to that thread until the thread terminates. The stack is used as a work area for storing temporary data such as local variables used by methods and functions that run on the thread.

Many methods and functions are called in a nested arrangement on a thread, and the operating system manages the temporary work areas used by those methods and functions by loading them into the stack in individual sections called frames. (Each time a method or function is called, a work area for use by that method or function is added to the stack in a frame, and data within a frame is discarded when the called method or function returns control.)

Therefore, if a method is called deeply in a nested structure (such as during an infinite recursion) or if a method that uses extremely large local variables is called, the stack area will be used up. This can prevent frames from accumulating in the stack and cause a stack overflow.

- For Java applications, a java.lang.StackOverflowError is normally thrown when a stack overflow occurs, although not if a native module is running within the Java process. Stack overflows that occur when a native module is running within a Java process using FJVM may be detected by the 6.6.3.1 Stack Overflow Message Output Function

Stack Area Management

The actual management of the **stack** area is performed by the operating system. For this reason, how the **stack** area is managed, its behavioral specification, and its size, will depend on the specification for the operating system that is used to execute JDK/JRE.

Part of the stack area is used as a Java VM control area so that Java VM can detect any stack overflows that occur while Java applications are running. Consequently, the size of the stack area that can be used from Java applications is slightly smaller than the area actually allocated as the stack.

Using Stack Area During Class File Execution

Once Java VM has been started up as the Java execution environment, Java VM reads the class files that make up the programs to be executed and then executes the class files in one of the following two ways:

- Byte code execution using an interpreter
- Execution of byte codes translated into machine instructions using dynamic compilation

Even where the same Java method in a class file is executed, the size of the stack used at execution will differ if the Java VM uses a different execution method.

For more information on class file execution methods, refer to "6.3 Dynamic Compilation".

Thread Generation Errors Caused by Insufficient User Space Memory

The stack is allocated from the user space of a process.

If a stack cannot be allocated due to insufficient memory in the user space, an error will occur during the thread creation process.

In the case of a Java process, if a large area is reserved for the Java heap, less area is available for allocation as the stack, and fewer threads can be created within the Java process.

If an error occurs during the creation of a thread within a Java process, it may be due to a memory shortage in the user space caused by a large Java heap size.

Disappearance of Java Processes Windows32/64

If a stack overflow occurs in a Java process in Windows(R), the status of the system and/or programs may prevent control from passing from the operating system to the FJVM or Dr. Watson, and the Java process may extinguish without leaving a trace.

Refer to 6.5.9.1 Crash Dumps for a description of Dr. Watson.

6.1.6 Java Heap and Garbage Collection

This section briefly explains the Java heap and garbage collection (GC) used in JDK/JRE.

Java Heap and Garbage Collection

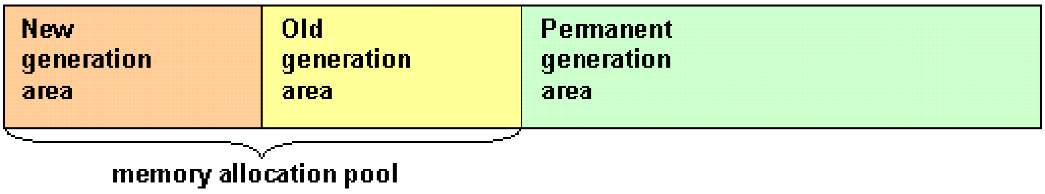
The Java heap is an area for storing Java objects that exist within Java processes.

The Java heap is broadly divided into a New generation area, an Old generation area and a Permanent generation area. These areas are controlled and managed by the Java VM. The New and Old generation areas are collectively managed and controlled in a form referred to as a memory allocation pool.

When a Java application is executed, the Java VM stores Java objects in each area of the Java heap. If the Java heap runs out of space, a java.lang.OutOfMemoryError will be thrown.

Java objects that are no longer needed are released by a GC process, which increases the free space within the Java heap. A **GC** process that releases unnecessary Java objects in the **New generation area** is referred to as a **NewGC** process (or **MinorGC** process). (These names may be abbreviated to simply NewGC and MinorGC respectively.) And, a **GC** process that releases unnecessary Java objects not just in the New generation area, but in the **entire Java heap**, including the **Old and Permanent generation areas**, is referred to as a **FullGC** process (or simply FullGC).

Figure 6.2 Java Heap Structure



- New generation area and Old generation area (memory allocation pool)

  New and Old generation areas are used for managing Java objects such as instances and arrays.

  The New generation area manages Java objects that are short-lived. Objects requested by a Java application are normally created in the New generation area. Java objects that have existed in the New generation area for a certain time are transferred to the Old generation area by a GC process. Java objects in the Old generation area that are no longer needed are then released by a FullGC process. Generation areas are divided into New and Old so that GC processes can be performed on different generations of objects.

The default and maximum sizes of the entire memory allocation pool can be specified using the "-Xms" and "-Xmx" options respectively.

- Permanent generation area

  The Permanent generation area is used for managing static objects such as Java classes, methods and constants that are continually referenced.

  Objects in the Permanent generation area that are no longer needed are released by a FullGC process.

  The default and maximum sizes of the Permanent generation area can be specified using the "-XX:PermSize" and "-XX:MaxPermSize" options respectively.

Note

The Java heap is allocated within the user space of a Java process.

In order to use a Java heap until its specified maximum value is reached, reserve the "memory allocation pool" and the "permanent generation area" at each of their respective maximum values, and as continuous domains. This is done when Java VM starts (so that it can be used from another process within the same process).

If the maximum size of the Java heap is set to a large value, the memory area available to other processes, such as the stack, will be reduced.

In other words, the maximum amount of memory available for use as the **Java heap** is equal to the maximum amount of available memory in the user space minus the amount used by the Java application itself, by Java VM and by the native modules (including the operating system).

If the Java heap size specified cannot be used when the Java VM starts up, then it terminates the Java process after outputting the message below:

```
Error occurred during initialization of VM
Could not reserve enough space for object heap
```

If this message is output, perform tuning to reduce the Java heap.

If "insufficient user space" or "insufficient virtual memory" occurs when the Java VM starts up or the Java application is running in Solaris/Linux, the Java VM outputs the following message and terminates the Java process.

When the Java VM starts up:

```
func-name: mmap failed: errono=nn, func-inf
Error occurred during initialization of VM
mmap failure
```

When the Java application is running:

```
Control name: mmap failed: errno=error information, control information....
(The java.lang.OutOfMemoryError message might be output after the above message.)
```

- *func-name* : Java VM internal control name
- *nn* : Java VM error information
- *func-inf* : Java VM internal control information

If this message is output, then take the following action:

- In the case of "insufficient user space", perform tuning to reduce the Java heap.

- In the case of "insufficient virtual memory", either terminate other unnecessary processes so that there is sufficient virtual memory, or perform tuning to increase the virtual memory by extending the physical memory (RAM) or swap file.

Note

To use the virtual memory resources of an operating system efficiently, the Java VM assigns the default size to each area of the Java heap when it starts up, and gradually increases these areas until the maximum sizes are reached.

In concrete terms, when a Java process starts, the memory allocation pool and the Permanent generation area are set to their default sizes. If the areas are insufficient after a FullGC process is performed, they are gradually increased until they reach their maximum values. (When Parallel GC, with the function that guarantees minimum usage of the memory allocation pool is disabled, has been used, the size of the memory allocation pool may be smaller than the default value.)

As each area is increased, the physical memory resources of the operating system may be swapped to the disk. This swap process can increase the time required by the FullGC process that expands each area. If the slowdown caused by swapping during a FullGC process becomes a problem, set the default and maximum sizes of the Java heap areas to the same value.

For information on Parallel GC, refer to "6.2.3 GC with the New Generation Area Control Processing Parallelization Function (Parallel GC)".

6.1.7 Tuning Options That Can Be Specified for FJVM

The following figure shows the options for Java VM tuning that can be specified for FJVM, including Java heap tuning.

Refer to this manual for information on how to use the options.

Options related to Java VM tuning that are not specifically explained in this manual are not supported in the FJVM.

Java VM tuning options that can be specified for FJVM

[Options for Java heap tuning]

```
-Xms
-Xmx
-XX:NewSize
-XX:MaxNewSize
-XX:NewRatio
-XX:SurvivorRatio
-XX:TargetSurvivorRatio
-XX:PermSize
-XX:MaxPermSize
```

[Options for stack size tuning]

```
-Xss
-XX:CompilerThreadStackSize
```

[Options to select the Garbage Collection processing used]

```
-XX:+UseSerialGC
-XX:+UseParallelGC
-XX:UseFJcmsGC
```

[Options for tuning Garbage Collection processing]

```
For Parallel GC:
-XX:ParallelGCThreads
-XX:+UseAdaptiveSizePolicyMinHeapSizeLimit
-XX:-UseAdaptiveSizePolicyMinHeapSizeLimit
```

```
-XX:+AutomaticallyJavaHeapSizeSetting
-XX:GCTimeLimit
-XX:GCHeapFreeLimit
-XX:+UseGCOverheadLimit

For Parallel GC with CMS:
-XX:ParallelGCThreads
-XX:ConcGCThreads

Common:
-XX:-UseCompressedOops (64-bit mode version of JDK/JRE )
```

[Debug options used for tuning, such as log output]

```
Garbage Collection log output:
-verbose:gc
-XX:+UseFJverbose
-XX:+ClassUnloadingInfo
-Xloggc

Dynamic compile log output:
-XX:+PrintCompilationCPUTime
-XX:+FJPrintCompilation

All log output:
-XX:FJverboseTime

Other:
-XX:-OmitStackTraceInFastThrow
-XX:+PrintClassHistogram
-XX:+PrintJavaStackAtSystemGC
-XX:+VMTerminatedMessage
-Xcheck:jni
-XX:+PrintCompilerRecoveryMessage
-XX:CompileTimeout
```

6.1.8 Java Native Interface (JNI)

Do not use native programs (that use C or C++) via a Java Native Interface (JNI) in order to ensure that problems such as application errors are prevented. At the design stage, investigate thoroughly whether the functions you are trying to implement can be written using Java. If the use of JNI is unavoidable, keep it to the absolute minimum and carry out thorough checking and debugging.

The following prerequisite skills are mandatory for the use of JNI:

- Experience in multithreading programming using C/C++
- The ability to debug problems yourself when they arise

Note

Do not manage resources with the expectation of finalization processing. The most frequent problem relating to JNI is postprocessing leakage of the memory reserved by native programs. For example, do not use programming such as that shown below.

```
------- Java ---------------
class A {
  native long nativeAlloc();
  native void nativeFree(long a);
  long address;
  A() {
    address = nativeAlloc();
  }
  public void finalize() {
    nativeFree(address);
  }
}
------- Java ---------------
------- C ------------------
JNIEXPORT jlong JNICALL Java_A_nativeAlloc(JNIEnv *env, jobject o)
{
  return (jlong)malloc(10);
}
JNIEXPORT void JNICALL Java_A_nativeFree(JNIEnv *env, jobject o, jlong p)
{
  free((void*)p);
}
------- C -------------------
```

Note

Always carry out error processing. When a JNI function call is issued by a native program, Java level errors can occur in a variety of situations. If postprocessing for such errors is not carried out by the native program, an exception is thrown and subsequent JNI function calls fail, preventing the application from running correctly.

Note that "JNI function" refers to the functions listed in the following JNI specifications:

http://docs.oracle.com/javase/6/docs/technotes/guides/jni/spec/functions.html

When using these, you must carry out an error check using ExceptionOccurred at the time.

Solaris32/64 Linux32/64

If JNI is used on Solaris or Linux, be absolutely sure not to overwrite the signal handler in the associated native program or native library.

Also, if you are using Signal Chaining that is supported by JDK/JRE 1.4.0 or later, familiarity with programming and signal operations in the operating system or Java VM itself, such signal operations in multithreading environments, is an essential prerequisite.

The use of this function is not recommended for system designs where stable operation is a requirement.

6.2 Garbage Collection (GC)

This section explains Garbage Collection (GC).

6.2.1 Garbage Collection Processing Supported in the FJVM

Due to differences in the GC control method for the Java heap New generation area, the following three types of garbage collection (GC) processing are supported in FJVM:

- 6.2.2 Standard GC (Serial GC)
- 6.2.3 GC with the New Generation Area Control Processing Parallelization Function (Parallel GC)
- 6.2.4 Parallel GC with Concurrent Mark Sweep (Parallel GC with CMS)

Note that the types of GC supported vary depending on the JDK/JRE version, execution mode and Java VM type. The GC that runs as the default option also differs depending on the type of Java VM.

The table below shows the default and supported GC.

Table 6.2 Supported GC Types

JDK/JRE version	JDK/JRE 6,7		
JDK/JRE execution mode	32-bit mode		64-bit mode
Java VM type	Client VM	FJVM	FJVM (*1)
Serial GC	S/D	S	S

JDK/JRE version	JDK/JRE 6,7		
JDK/JRE execution mode	32-bit mode		64-bit mode
Java VM type	Client VM	FJVM	FJVM (*1)
Parallel GC	S	S/D	S/D
Parallel GC with CMS	IASEE	IASEE	IASEE

S: Supported GC, S/D: Supported and default GC, IASEE: GC supported only in the Interstage Application Server Enterprise Edition

*1: Client VMs with 64-bit execution modes are not provided.

It is recommended that the default GC be used. There is normally no need to change the default GC.

Note

In JDK/JRE 6 and 7, "FJGC", which can automatically resize the New generation area size, is not provided.

If the "-XX:+UseFJGC" option was specified for the Java HotSpot Client VM, the error message shown in the following figure is output in standard error output, and the Java process fails to start.

If the "-XX:+UseFJGC" option was specified for the FJVM, the warning message shown in the following figure is output in standard error output, and the specified option is ignored.

Messages that are output when "-XX:+UseFJGC" has been specified

- When Java HotSpot Client VM is used

```
Unrecognized VM option '+UseFJGC'
```

- When the execution mode is 32-bit mode FJVM

```
warning: -XX:+UseFJGC is not supported in Java HotSpot Server VM.
```

- When the execution mode is 64-bit mode FJVM

```
warning: -XX:+UseFJGC is not supported in Java HotSpot 64-Bit Server VM.
```

How the New Generation Area is Used

NewGC processing that is non-FJGC GC processing subdivides the New generation area into three internal spaces: "eden space", "from space" and "to space". In each space, a control method generally referred to as "GC control by generation" is used to manage and control objects generated at the request of Java applications.

Of those areas, "from space" and "to space" act in the role of work areas when Java VM conducts NewGC processing. As a result, the area size used for object generation requests from Java applications makes up only a part of the total size occupied by each of the "from space" and "to space" areas.

Consequently, it may appear from output data such as the Garbage Collection log output that there is free space available in the memory allocation pool and New generation area, but there may in fact be no free space available. (Even when there appears to be space available, the discrepancy arises because that space may already be in use as work area for NewGC processing.)

Preventing Execution of Garbage Collection Processing

Execute the Java application which uses the functionality as displayed in the following figure. This will cause the status called "Critical Section" to occur, preventing movement of any objects inside the Java heap, in accordance with the conditions of that functionality.

During the Critical Section status, objects will be prohibited from moving, preventing the execution of GC processes requiring mandatory movement of objects.

When object generation requests occur at the time when the execution of Java application GC Processing is prevented, Java VM will temporarily use space in the Old generation area, if there is no space available in the New generation area.

If the space needed by a requested object exceeds the amount of space available in the Old generation area, this exception error will occur: java.lang.OutOfMemoryError.

In the following figure, the functionality of the Java application is busy and the possibility of GCProcessing Execution being prevented is increasing. Compare this with the same function when it is not too busy, and the exception error java.lang.OutOfMemoryError is likely to occur, causing problems for the GC Processing Execution Prevention.

There is a strong tendency for the error to occur especially in cases where the Java application is executed when the Old generation area is at low capacity. This is because the maximum available space is relatively small. If the Old generation area cannot be used at all, it is in inverse proportion to the size of the Old generation area itself.

Moreover, in the FJVM, functionality is provided which supports the output of information on whether or not such an error will occur.

For more details on this, refer to "6.6.1.6 Enhanced Message Output Function for Insufficient Memory Size Events".

Preventing Execution of Garbage Collection Processing

(JNI Function when Preventing Execution of GC Processing Conditions occurs)

```
From the execution of GetPrimitiveArrayCritical() until the execution of
ReleasePrimitiveArrayCritical()
From the execution of GetStringCritical() until the execution of ReleaseStringCritical()
```

(JVMPI Function when Preventing Execution of GC Processing Conditions occurs) (*1)

```
From the execution of DisableGC() until the execution of EnableGC()
```

(JVMPI Event when Preventing Execution of GC Processing Conditions occurs) (*1)

```
JVMPI_EVENT_THREAD_START
JVMPI_EVENT_CLASS_LOAD
JVMPI_EVENT_CLASS_UNLOAD
JVMPI_EVENT_JNI_GLOBALREF_ALLOC
JVMPI_EVENT_JNI_GLOBALREF_FREE
JVMPI_EVENT_JNI_WEAK_GLOBALREF_ALLOC
JVMPI_EVENT_JNI_WEAK_GLOBALREF_FREE
JVMPI_EVENT_OBJECT_ALLOC
JVMPI_EVENT_MONITOR_CONTENDED_ENTER
JVMPI_EVENT_MONITOR_CONTENDED_ENTERED
JVMPI_EVENT_MONITOR_CONTENDED_EXIT
JVMPI_EVENT_MONITOR_WAIT
JVMPI_EVENT_MONITOR_WAITED
JVMPI_EVENT_HEAP_DUMP
JVMPI_EVENT_METHOD_ENTRY
JVMPI_EVENT_METHOD_ENTRY2
JVMPI_EVENT_METHOD_EXIT
```

*1 In JDK/JRE 6 and 7, Java Virtual Machine Profiling Interface (JVMPI) is not supported.

JVMPI and JVMTI

In JDK/JRE 6 and 7, Java Virtual Machine Profiling Interface (JVMPI) is not supported:

When using the function equivalent JVMPI with JDK/JRE 6 and 7, use a Java Virtual Machine Tool Interface (JVMTI).

Distributed GC for RMI Processing

Java RMI processing carries out processing called Distributed GC, which discards objects on the server for references that are no longer needed by the client. As one part of this processing, Full GC is implemented by running java.lang.System.gc() at time intervals set in the property below (the default time interval is one hour). If normal Garbage Collection (GC) occurs at the same time as Distributed GC due to a lack of memory, the normal GC triggered by the memory shortage is run and FullGC is not implemented by Distributed GC. (If the normal GC resulting from the memory shortage is NewGC processing, it does not become FullGC.)

```
-Dsun.rmi.dgc.server.gcInterval=time interval (ms)
-Dsun.rmi.dgc.client.gcInterval=time interval (ms)
```

Because Distributed GC is executed based on independent timer control, it is run irrespective of the execution of normal GC following a problem such as a memory shortage. For this reason, it may appear from looking at the GC processing results log that FullGC processing was implemented during a period when almost no processing for Java applications was being performed and there was little likelihood of a memory shortage.

Note also that if a short interval is set in the property, a warning (the EXTP4368 or ISJEE_OM3204 message) may be received from the Interstage Application Server's predictive monitoring function.

6.2.2 Standard GC (Serial GC)

This is GC processing configured using the "standard function only". No additional functions are added for the New generation area GC control. To contrast with the Parallel GC mentioned below, the Standard GC is also called **Serial GC**.

In Java HotSpot Client VM and FJVM if the options in the following figure are specified, **Serial GC** is executed. Note that the options in the following figure enable the GC control using Serial GC.

Options for enabling Serial GC in FJVM

```
-XX:+UseSerialGC
```

Java heap tuning options (when Serial GC is used)

```
-Xms
-Xmx
-XX:NewSize
-XX:MaxNewSize
-XX:NewRatio
-XX:SurvivorRatio
-XX:TargetSurvivorRatio
-XX:PermSize
-XX:MaxPermSize
```

6.2.3 GC with the New Generation Area Control Processing Parallelization Function (Parallel GC)

This is GC processing configured using the additional "Function for parallelization and execution of appropriate processing" for the New generation area GC control. Since it parallelizes and executes the New generation area GC control, this GC is also called **Parallel GC**.

In the JDK/JRE 6 or 7 FJVM, this GC processing is executed by default.

Option for enabling the new generation area control processing parallelization function in the JDK/JRE 6 or 7 FJVM

```
-XX:+UseParallelGC
```

Java heap tuning options (when Parallel GC is used)

```
-Xms
-Xmx
-XX:NewSize(*1)
-XX:MaxNewSize(*1)
-XX:NewRatio(*1)
-XX:PermSize
-XX:MaxPermSize
```

*1 Option used to tune the New generation area size and the balance between the New and Old generation area sizes.

Number of GC Processing Threads

When **Parallel GC** is used, a number of threads are created in the Java process, that number being dependent on the number of CPUs installed in the hardware that runs the **GC process**. For this reason, the memory area required for threads (stack area, etc.), is only the amount needed for the number of GC processing threads.

To configure the number of GC processing threads in order to, for instance, limit the amount of memory in the Java process, you can adjust the number of GC processing threads by specifying the number of GC processing threads in the options in the following figure..

Note that limiting GC processing for the number of GC processing threads may affect the GC processing performance. Carefully check the performance before using this option. Generally, there is no improvement in GC processing performance even if the number of GC processing threads created exceeds the number of CPUs.

Options that specify the number of GC processing threads used for Parallel GC

```
-XX:ParallelGCThreads=Number of threads for New generation area GC
Specify the number of GC threads to carry out GC processing for the New
generation area.
If "0" is specified, the default value is used.
The default values are as follows:
- When the number of CPUs installed in the hardware running the GC process
is 7 or less = Number of CPUs
- When the number of CPUs installed in the hardware running the GC process
is 8 or more = 8
```

Memory Allocation Pool Default Value Automatic Resizing Function

In FJVM **Parallel GC**, the JDK/JRE 6 or 7 ergonomics function memory allocation pool minimum (-Xms) and maximum (-Xmx) default value automatic resizing function is disabled. This function automatically determines the default value for the -Xms and -Xmx options, depending on the physical memory size of the machine.

In the JDK/JRE 6 or 7 FJVM, specify the option in the following figure to enable the ergonomics function memory allocation pool default value automatic resizing function.

If this option is specified, insufficient system memory resources may be an issue. Hence, do not use it to start or execute more than one Java process in the system.

Option for enabling the memory allocation pool default value automatic resizing function in the JDK/JRE 6 or 7 FJVM

```
-XX:+AutomaticallyJavaHeapSizeSetting
```

Insufficient Memory Detection Function

In FJVM **Parallel GC,** the JDK/JRE 6 or 7 ergonomics function memory shortage detection function is disabled. This function detects a memory shortage (java.lang.OutOfMemoryError) when the conditions for the values specified for the options in the following figure are met at the same time.

In the JDK/JRE 6 or 7 FJVM, specify the option in the following figure to enable the ergonomics function memory shortage detection function.

The insufficient memory event that is detected when this option is specified is determined by Java heap usage, the value specified in this option, and the statistical information obtained from garbage collection operating conditions. For this reason, note that the insufficient memory event may be detected even if the Java heap usage is sufficient.

Options for detecting memory shortage

```
-XX:GCTimeLimit= Top value for the time required for GC processing (the default is 98)
Specify the top value for the time required for GC processing as a percentage (%) of the total Java
application processing time.
If the top value that is specified is exceeded, either of the detection conditions is met.

-XX:GCHeapFreeLimit= Value for the Java heap available space after GC processing (the default is 2)
Specify the minimum value for the available Java heap space after GC processing as a percentage (%)
of the maximum memory allocation pool.
If the specified minimum value is breached, either of the detection conditions is met.
```

Option for enabling memory shortage detection in the JDK/JRE 6 or 7 FJVM

```
-XX:+UseGCOverheadLimit
```

Parallel GC processing ergonomics function and the function that guarantees minimum memory allocation pool usage

If you use **Parallel GC** in **FJVM** with the JDK/JRE 6 and 7 function parallel GC processing ergonomics function (*1), then based on the information about the Java application execution status and load/GC processing times, the size of each generation space in the pool to which memory is allocated is automatically reconciled/changed and optimized.

*1 A function that changes the size of each generation space in the pool to which memory is dynamically allocated.

At that time, pool usage might be lower than the value specified in the -Xms option (the pool's initial value), which might be the optimal configuration for GC processing.

When the Java application is executed using parallel GC, specify the following option to perform the operation for usage of the pool to which memory is allocated.

- When you do not want pool usage to be less than the value specified in the -Xms option:

```
-XX:+UseAdaptiveSizePolicyMinHeapSizeLimit
```

When the parallel GC processing ergonomics function is activated, the function that guarantees minimum pool usage using the value specified in the -Xms option is enabled.

The pool sized used when the Java application is executed varies between "value specified in the -Xms option" to "value specified in the -Xmx option".

If-Xms and -Xmx options specify the same value, the size of pools currently in use do not vary.

This status is the default status when parallel GC is used in JDK/JRE 6 and 7.

- When it is OK for pool usage to be less than the value specified in the -Xms option:

```
-XX:-UseAdaptiveSizePolicyMinHeapSizeLimit
```

When the parallel GC processing ergonomics function is activated, the function that guarantees minimum pool using the value specified in the -Xms option is disabled.

The pool size used when the Java application is executed varies between "lower limit as Java VM" to "value specified in the -Xmx option".

If-Xms and -Xmx options specify the same value, the size of pools currently in use vary.

If the size of the memory allocation pool used during Java application execution shrinks so that it is smaller than the value specified in the -Xms option, the FullGC occurrence interval may become shorter than the occurrence interval close to the time when Java processes start. However, the Parallel GC ergonomics function makes adjustments so that operation is optimized for GC processing, taking into account the information on the time taken for GC processing. (Operation is optimized even if the memory allocation pool becomes smaller and less time is taken for FullGC.)This means that even if the FullGC occurrence interval shortens due to shrinking of the memory allocation pool, it has little or no impact on performance during Java application execution.

Note that, when an application that was executed using JDK/JRE 5.0 is then executed using JDK/JRE 6/7 (both used to run parallel GC), this option is used when it is necessary to improve the compatibility of the memory allocation pool size for pools that are currently in use.

Normally, there is no need to specify this option.

Regardless of whether the function is enabled or disabled, the pool size used when the Java process starts up will be the value in the -Xms option.

6.2.4 Parallel GC with Concurrent Mark Sweep (Parallel GC with CMS)

A feature that performs parallelization for and executes the GC control for the New generation space, and the "Concurrent Mark Sweep GC (CMS-GC) feature" GC control for the Old generation space that is run in parallel with, and at the same time as, the Java application have been added to the configuration of this GC processing. Because the CMS-GC feature has been added to this Parallel GC control, this GC is also sometimes called **Parallel GC with CMS**.

Note on Parallel GC with CMS

The **Parallel GC with CMS** Garbage Collection control is only provided in JDK/JRE of Interstage Application Server Enterprise Edition.

Note on CMS-GC

CMS-GC is a "GC mechanism for collecting unnecessary objects in the Old and Permanent generation spaces" and is executed to compensate for the impact on "application response performance equalization" when the Java application is stopped by Full GC.

CMS-GC runs concurrently and in parallel with the Java application, collecting unnecessary objects in the Old and Permanent generation spaces for Full GC that stops the Java application when executed. By executing CMS-GC, the amount of free Old generation space (the space to which objects in New generation space are moved and in which large objects are generated) and free Permanent generation space (the space in which Java classes, methods and constants are stored) can be increased because CMS-GC runs in parallel with the Java application, thereby limiting the occurrences of Full GC. In this way, the impact of Java applications being stopped by Full GC can be lessened, which will in turn improve response performance equalization.

Note that, while CMS-GC is running, the start of NewGC processing/FullGC processing might lag. During this lag, the Java application will also be paused. For this reason, the Java application might pause for a longer time than the GC processing execution time that was output for NewGC processing/Full GC processing in the Garbage Collection processing results log.

Options for enabling Parallel GC with CMS in JDK/JRE 6 and 7

```
[Where the unnecessary objects to be collected by CMS-GC are in the Old generation space]
-XX:UseFJcmsGC=type[0|1|2]

[Where the unnecessary objects to be collected by CMS-GC are in the Old and Permanent generation
spaces]
-XX:UseFJcmsGC=type[0p|1p|2p]
```

Note

If the range of unnecessary objects to be collected by CMS-GC is expanded to include those in the Permanent generation space as well as those in the Old generation space, this will increase the spaces targeted for CMS-GC processing and therefore tends to lengthen the execution time until CMS-GC is completed.

As a result, depending on the application being run and the execution environment, there may not be time for collection processing by CMS-GC and this may lead to the occurrence of FullGC.

For this reason, if you expand the scope of collection to include both Old and Permanent generation spaces in an environment where collection is normally limited to the Old generation space, tuning work must be carried out again.

Where -XX:UseFJcmsGC=type0 or =type0p is specified

The following figure shows "Java heap tuning options" that can be used when -XX:UseFJcmsGC=type0 or =type0p is specified. Specify this to use Parallel GC with CMS, which can be used for fine-tuning tasks.

Java heap tuning options (when -XX:UseFJcmsGC=type0 or =type0p is specified)

```
-Xms
-Xmx
-XX:NewSize
-XX:MaxNewSize
-XX:SurvivorRatio
-XX:TargetSurvivorRatio
-XX:PermSize
-XX:MaxPermSize
```

Where -XX:UseFJcmsGC=type1 or =type1p is specified

Specify this to use Parallel GC with CMS that is configured focusing on object collection on the New generation space.

If it is a feature of the application that will be executed that "the majority of objects will be collected after very few occurrences of GC for the New generation space", then this configuration will make it easy to obtain improvement effects in response performance equalization using CMS-GC.

To perform Java heap tuning, first configure the memory allocation pool and Permanent generation space size using the various -Xms/-Xmx and -XX:PermSize/-XX:MaxPermSize options. Configure the New generation space size using the various -XX:NewSize/-XX:MaxNewSize options if necessary.

Note that a value lower than the maximum size for the memory allocation pool can be specified as the New generation space size. However, if the value for the New generation space is too large, this will make it more likely that Full GC will occur.

Java heap tuning options (when -XX:UseFJcmsGC=type1 or =type1p is specified)

```
-Xms
-Xmx
-XX:NewSize
-XX:MaxNewSize
-XX:PermSize
-XX:MaxPermSize
```

Where -XX:UseFJcmsGC=type2 or =type2p is specified

Specify this to use Parallel GC with CMS that is configured focusing on object collection on CMS-GC.

If it is a feature of the application that will be executed that "the majority of objects will be collected after several occurrences of GC (they will be collected within a relatively short time, without prolonged residence)", then this configuration will make it easy to obtain improvement effects in response performance equalization using CMS-GC.

To perform Java heap tuning, first configure the memory allocation pool and Permanent generation space size using the various -Xms/-Xmx and -XX:PermSize/-XX:MaxPermSize options. Configure the New generation space size using the various -XX:NewSize/-XX:MaxNewSize options if necessary.

Note that a value lower than the maximum size for the memory allocation pool can be specified as the New generation space size. However, if the value for the New generation space is too large, this will make it more likely that Full GC will occur.

Java heap tuning options (when -XX:UseFJcmsGC=type2 or =type2p is specified)

```
-Xms
-Xmx
-XX:NewSize
-XX:MaxNewSize
```

```
-XX:PermSize
-XX:MaxPermSize
```

Note on number of GC processing threads

When **Parallel GC with CMS** is used, the number of threads, which is created in the Java process, depends on the number of CPUs installed in the hardware that runs the GC process. For this reason, memory size such as stack area, which is allocated for the thread, will vary according on the number of GC processing threads.

To configure the number of GC processing threads that will control the amount of memory in the Java process, for example, the number of GC processing threads can be configured by specifying the number of GC processing threads using the options in the sample below.

Note that the number of GC processing threads will affect the GC processing performance. Before using this option, try a performance test to confirm the effect. Generally, there is no improvement in GC processing performance even if the number of GC processing threads that is created exceeds the number of CPUs.

Option that specifies the number of GC processing threads be used in Parallel GC with CMS

```
-XX:ParallelGCThreads
```

The option above specifies the number of GC threads that will perform GC processing of the New generation space. The minimum value is 2, and if 0 or 1 is specified then the default value will be used. The default value is the number of CPUs in the hardware that runs the GC process, (except that if 1 CPU is used, then default value is 2, and if more than 7 CPUs are used, then the default value is 8).

Number of CMS-GC processing threads

If **Parallel GC with CMS** is used, the CMS-GC processing threads shown below are created inside the Java process. For this reason, the memory area required for threads (stack area, etc.), is only the amount needed for the number of CMS-GC processing threads.

- CMS threads (always create 1)
- Dedicated threads for concurrent mark processing

Additional dedicated threads for concurrent mark processing can also be created because it performs concurrent mark processing within CMS-GC processing by running multiple threads in parallel.

Consequently, you can to adjust the number of CMS-GC processing threads by specifying the number in the options shown in the following figure.

Generally, there is no improvement in CMS-GC processing performance even if the number of CMS-GC processing threads created exceeds the number of CPUs.

Options for specifying the number of GC processing threads used in Parallel GC with CMS

```
X:ConcGCThreads=Number of CMS-GC processing threads
Specify the number of dedicated threads for concurrent mark processing.
The minimum value is "2".
If this option is omitted, or if "1" is specified, only CMS threads are generated.
If "0" specified, the values shown below are set automatically.
(Where 1/4 of the number of CPUs installed in the hardware running CMS-GC (fractions rounded up)
is set as A)
- = 0 (zero) if A is 1 (only CMS threads are generated)
- =A if A is more than 1 but less than 8
- =8 if A is 8 or more
Whether the setting is set manually or automatically, if it is larger than the value of -
XX:ParallelGCThreads, the -XX:ParallelGCThreads value is used.
```

6.2.5 Object Reference Compression Function

If Java applications are run on 64-bit mode JDK/JRE, execution mode restrictions mean that the area required for object references (pointer information) stored in the Java heap is managed in "64-bit representation/8-byte area" units, in the same way as normal C/C++ applications.

Consequently, if Java applications are run on 64-bit mode JDK/JRE, the Java heap needs to be 1.5 to 2 times larger than when they are run on 32-bit mode JDK/JRE.

However, for FJVM with JDK/JRE 6 or 7 installed running in 64-bit mode and where the Java heap size (the total size of the memory allocation pool and Permanent generation space) is less than 32 GB, the "**Object Reference Compression** function for 64-bit mode execution", which manages the Java heap by compressing the area required for object references into "32-bit representation/4-byte area" units, can be used to run Java applications with a Java heap size that is smaller than for JDK/JRE run in 64-bit mode without using that function.

Because it reduces the Java heap area required per object reference, this function allows more object references to be stored in the Java heap. To give an extreme example: if you were normally able to store 100 object references in a given heap size, this function would allow you to store 200 object references in the same Java heap. In other words, not installing or using this function is equivalent to specifying a larger value as the Java heap size compared with when this function is used. The result is that the frequency of garbage collection processing is reduced, leading to improved application execution performance.

Disabling the Object Reference Compression Function

Because this function manages object references by compressing memory area that normally requires a "64-bit representation/8-byte area" into "32-bit representation/4-byte area", the compressed information must be expanded back into the "64-bit representation/8-byte area" format when the object reference is used. Because this compression and expansion can lead to excessive CPU usage, it may in some cases result in lowered execution performance for Java applications.

In the case of FJVM equipped with JDK/JRE 6 or 7 running in 64-bit mode, the object reference compression function is enabled by default. If this results in problems with the execution performance for Java applications, you can disable the object reference compression function in FJVM by specifying the option in the following figure.

Option for disabling the object reference compression function in the FJVM running in 64-bit mode

```
-XX:-UseCompressedOops
```

Note that this option can be specified for JDK/JRE 6 or 7 running in 64-bit mode.

6.2.6 Outputting a Garbage Collection Log

Specify the option in the following figure to collect the Garbage Collection (GC) log. If you specify this option, the GC processing results log is output to standard output one line at a time each time GC occurs.

Option for outputting the GC processing results log

```
-verbose:gc
```

The following figure shows the output format and an output example.

Output format

```
 [<GC type> <Heap usage before the GC> -> <Heap usage after the GC
(heap size)>, <GC processing time>]
```

"GC" as the GC type indicates MinorGC (or NewGC) processing, while "FullGC" indicates FullGC processing.

Note that "memory allocation pool in the Java heap" is abbreviated to "heap".

If Parallel GC with CMS is used, there will also be a GC log with the following format:

GC Log Output Format

```
[GC heap usage amount(heap size), mark processing time]
```

If the log was output in this format, it indicates that initial or final mark processing was executed.

Note that "memory allocation pool in the Java heap" is abbreviated to "heap".

Example of GC Log Output

```
[GC 80229K->31691K(259776K), 0.4795163 secs]
[FullGC 57654K->4623K(259776K), 0.3844278 secs]
```

The FJVM enhances of the GC processing result log output function.

To get more detailed information on the GC processing results log, use this function. See "6.2.6.1 Enhanced Garbage Collection Logging Function" for details.

How the New Generation Area is Used

NewGC subdivides the New generation area into three internal spaces: "eden space", "from space" and "to space". In each space, a control method generally referred to as "GC control by generation" is used to manage and control objects generated at the request of Java applications.

Of those areas, "from space" and "to space" act in the role of work areas when Java VM conducts NewGC processing. As a result, the area size used for object generation requests from Java applications makes up only a part of the total size occupied by each of the "from space" and "to space" areas.

Consequently, it may appear from the Garbage Collection log output that there is free space available in the memory allocation pool and New generation area, but there may in fact be no free space available. (Even when there appears to be space available, the difference is that that space may already be in use as work area for NewGC processing.)

Increase in the log output

Log output increases when this option is specified.

Logs should be monitored in this case as they may become very large.

Class Unload Information

If some classes are unloaded, specify the following option to insert "[Unloading class <class name>]" (unloading information for the corresponding class) in the gc log during FullGC:

```
-XX:+ClassUnloadingInfo
```

Specifying the File Used to Store the GC Processing Results Log

By specifying the option in the following figure, you can switch the output destination for the GC processing results log and 6.5.4 Class Instance Information Output Function from standard output to a designated file.

If the option in the following figure is specified when a Java application is executed in Interstage Application Server, the following problems will arise:

- The file output process using the option in the following figure does not have generation management features such as log rotation. If the Java process is automatically restarted for some reason, the same file will be used as the storage destination, meaning that the pre-restart GC process results log will be overwritten by the post-restart results log and will not be available as log information.

- If Java applications run concurrently on multiple processes, those Java applications will all run with the same option definition. For this reason, the GC process results logs from multiple processes will be written to the same file, meaning that the results logs will not be available as log information.

- The output destination files for the logs when Java applications are executed in Interstage Application Server are managed under the control of Interstage Application Server. If the GC process results log is a different file specified using the option in the following figure, and an error then occurs, the GC process results logs will be separate from the other error information, which may make it difficult to analyze the error.

Accordingly, you should not specify the option in the following figure for Java applications that will be executed in Interstage Application Server.

Specify the option in the following figure as required if you are running independent Java applications that do not interact with Interstage Application Server.

Option for specifying file used to store the GC processing results log

```
-Xloggc:GC process results log storage destination file name
```

*1 When this option is specified, it is automatically assumed that the "-verbose:gc" option is also specified. Consequently, even if the "-verbose:gc" option is not specified, the GC processing results log is output. You can also use "6.2.6.1 Enhanced Garbage Collection Logging Function ".

Note also that the elapsed time (in seconds) from when the Java VM was started is automatically added as the GC processing execution start time at the beginning of the GC processing results log output by the "-Xloggc" option specification (output in the format shown below).

```
GC processing execution start time: GC processing results log
```

The format of the GC processing execution start time cannot be changed.

(If 6.2.6.1 Enhanced Garbage Collection Logging Function is used to output the GC processing results log, you can specify the GC processing execution start time format using the 6.5.7 Time Information Format Specification Function in the Log Output.)

*2 The specification for the storage destination file name can also use a format that includes an appended directory name with an absolute path or relative path.

*3 If the specified storage destination file name cannot be accessed for some reason (e.g., the directory in the storage destination file name does not exist), the GC processing results log is output to standard output as usual instead of the specified storage destination file.

6.2.6.1 Enhanced Garbage Collection Logging Function

The FJVM includes an "Enhancement of the results log output function for Garbage Collection processing", which outputs a more detailed version of the Garbage Collection (GC) processing results log output when the "-verbose:gc" option is specified.

When the "-verbose:gc" option is specified to output the GC log, the option shown in the following figure is also added so that the format of the GC log is extended, as shown from Option to extend information output to the GC log to Extended format of information output as the GC processing log (completing CMS-GC). An output example is shown from Option to extend information output to the GC log to Output example showing extended format of information output as the GC processing results log (for Parallel GC with CMS - 2).

Option to extend information output to the GC log

```
-XX:+UseFJverbose
```

Format of information output to the GC log when option to extend is used

```
$1: [$2, [$3 : $4->$5($6)], [$7 : $8->$9($10)] $11->$12($13), [$14 : $15->$16($17)], $18 secs]
```

Sample of information in the GC log when option to extend is used

```
1.495: [Full GC*, [SplitEden : 384K->0K(704K)], [Tenured : 47835K->32752K(47872K)] 48219K-
>32752K(48576K), [Perm : 4081K->4081K(16384K)], 0.6623532 secs]
```

The options in Figure Format of information output to the GC log when option to extend is used is used are described below.

$1: GC processing execution start time (time of log output)

This shows the GC processing execution start time (time of log output).

The format of the time of log output can be specified using the 6.5.7 Time Information Format Specification Function in the Log Output.

The default is "time elapsed (in seconds) from when the Java VM was started".

$2: GC Type

This field represents the GC process types logged as follows:

- GC

  Information is logged from GC processes that target the New generation area (NewGC processing or minor GC processing).

- Full GC

  Information is logged from GC processes that target the entire Java heap area (FullGC processing). This includes both the memory allocation pool (New generation area and Old generation area) and the Permanent generation area.

- Full GC*

  Information is logged from FullGC processes, when Serial GC or Parallel GC with CMS processing is used and insufficient area was secured for the NewGC processing executed immediately before this FullGC. ("*" placed after "Full GC" to differentiate this).

  If only the "-verbose:gc" option is specified, then only "Full GC" information is logged (not Full GC*).

- CMS initial-mark

  This is the results information in the initial mark processing of the CMS-GC processing that targets the Old generation space.

  In CMS-GC, the Java application is stopped for a brief period of time when detecting (initial mark processing) unnecessary objects.

  Note that since this is only a process that detects unnecessary objects, there is no change to the object amount in each generation space before and after the GC processing starts.

- CMS remark

  This is the results information in the final mark processing of the CMS-GC processing that targets the Old generation space.

  In CMS-GC, the Java application is stopped for a brief period of time when detecting (final mark processing) unnecessary objects.

  Note that since this is only a process that detects unnecessary objects, there is no change to the object amount in each generation space before and after the GC processing starts.

$3: New Generation Area Identifiers

The following New generation area identifiers are output depending on the differences in the GC processing that is used:

- DefNew: Serial GC
- PSYoungGen: Parallel GC
- ParNew: Parallel GC with CMS

$4: Object Amount (New Generation Area) before GC Processing

This is the total amount (in bytes) for objects that existed in the New generation area before GC processing is executed.

$5: Object Amount (New Generation Area) after GC Processing

This is the total amount (in bytes) for objects that exist in the New generation area after GC processing is executed.

$6: New Generation Area Size

This is the New generation area size (in bytes).

Note

If the GC processing used is Serial GC, Parallel GC or Parallel GC with CMS, the size of the "to space" area is not included in this size.

(If Serial GC, Parallel GC or Parallel GC with CMS is used, GC processing controls the New generation area by subdividing it into 3 internal areas: "eden space", "from space" and "to space".)

$7: Old Generation Area Identifiers

The following Old generation area identifiers are output depending on the differences in the GC processing that is used:

- Tenured: Serial GC
- PSOldGen: Parallel GC
- CMS: Parallel GC with CMS

$8: Object Amount (Old Generation Area) before GC Processing

This is the total amount (in bytes) for objects that existed in the Old generation area before GC processing is executed.

$9: Object Amount (Old Generation Area) after GC Processing

This is the total amount (in bytes) for objects that exist in the Old generation area after GC processing is executed.

$10: Old Generation Area Size

This is the Old generation area size (in bytes).

$11: Object Amount (Memory Allocation Pool) before GC Processing

This is the total amount (in bytes) for objects that existed in the memory allocation pool before GC processing is executed.

This is the total of $4+$8.

$12: Object Amount (Memory Allocation Pool) after GC Processing

This is the total amount (in bytes) for objects that exist in the memory allocation pool after GC processing is executed.

This is the total of $5+$9.

$13: Memory Allocation Pool Size

This is the memory allocation pool size (in bytes).

This is the total of $6+$10.

Note

If the GC processing used is Serial GC, Parallel GC or Parallel GC with CMS, the size of the "to space" area in the New generation area is not included in this size.

(If Serial GC, Parallel GC or Parallel GC with CMS is used, GC processing controls the New generation area by subdividing it into 3 internal areas: "eden space", "from space" and "to space".)

$14: Permanent Generation Area Identifiers

The following Permanent generation area identifiers are output depending on the differences in the GC processing that is used:

- Perm: Serial GC
- PSPermGen: Parallel GC
- CMS Perm: Parallel GC with CMS

$15: Object Amount (Permanent Generation Area) before GC Processing

This is the total amount (in bytes) for objects that existed in the Permanent generation area before GC processing was executed.

$16: Object Amount (Permanent Generation Area) after GC Processing

This is the total amount (in bytes) for objects that exist in the Permanent generation area after GC processing is executed.

$17: Permanent Generation Area Size

This is the Permanent generation area size (in bytes).

$18: GC Processing Execution Time

This is the time taken to execute GC processing (in seconds). GC processing is carried out by stopping the operation of Java applications.

Notes

- The information output for $2, $11, $12, $13, and $18 corresponds to the information output when only the "-verbose:gc" option is specified as the GC processing result log output function.
- Once the GC processing has been executed, the Java application will be paused.

Extended format of information output as the GC processing log (starting CMS-GC)

```
$1: CMS start
```

The elements in the sample are explained below:

$1: CMS-GC processing execution start time (time of log output)

This shows the CMS-GC processing execution start time (time of log output).

The format of the time of log output can be specified using the 6.5.7 Time Information Format Specification Function in the Log Output.

The default is "time elapsed (in seconds) from when the Java VM was started".

Extended format of information output as the GC processing log (request to complete GC with the occurrence of FullGC)

```
$1: CMS stop-req
```

The elements in the sample are explained below:

$1: Time of CMS-GC processing completion request (time of log output)

If CMS-GC processing was running when the Full GC request was issued, this indicates the time of the CMS-GC processing completion request (time of log output).

The format of the time of log output can be specified using the 6.5.7 Time Information Format Specification Function in the Log Output.

The default is "time elapsed (in seconds) from when the Java VM was started".

If there was insufficient Java heap when CMS-GC was running, or a Full GC request occurred because java.lang.System.gc() was executed, then Full GC processing will wait for the completion of CMS-GC before starting. In order to retain consistency of the data processed by CMS-GC, CMS-GC will not be forcefully terminated. Instead, processing in CMS-GC that can be terminated is made to complete and is then finished. For this reason, there might be a lag between the Full GC processing request and its start.

Note that the Java application will be paused from the point at which the CMS-GC processing completion request occurs until the execution of CMS-GC processing is complete. For this reason, the point at which the CMS-GC processing completion request occurs until the execution of Full GC processing is complete will be the actual period of time for which the Java application is paused by Full GC processing, if this information was output.

Extended format of information output as the GC processing log (completing CMS-GC)

Pattern 1:

If the CMS-GC target is "within the Old generation area"

```
$1: CMS stop($2), [CMS : $3->$4($5)], $9 secs
```

If the CMS-GC target is "within the Old and Permanent generation areas"

```
$1: CMS stop($2), [CMS : $3->$4($5)], [CMS Perm : $6->$7($8)], $9 secs
```

Pattern 2:

```
$1: CMS stop($2), $9 secs
```

The elements in the sample are explained below:

$1: CMS-GC processing execution end time (time of log output)

This shows the CMS-GC processing execution end time (time of log output).

The format of the time of log output can be specified using the 6.5.7 Time Information Format Specification Function in the Log Output.

The default is "time elapsed (in seconds) from when the Java VM was started".

$2: Exit code

This shows the exit code for the CMS-GC processing execution results.

The output format pattern for the information will vary according to the differences in the exit codes.

The exit code types and their meanings are as follows:

- 00: CMS-GC processing was terminated.

  Unnecessary objects detected in the Old generation space or in the Old and Permanent generation spaces were collected.

  Information is output in the output format shown in pattern 1.

- 10: CMS-GC processing in progress was terminated because there was a Full GC request due to a Java heap shortage.

  Unnecessary objects detected in the Old generation space or in the Old and Permanent generation spaces were collected.

  Information is output in the output format shown in pattern 1.

- 20: CMS-GC processing in progress was terminated because there was a Full GC request due to an external factor such as java.lang.System.gc().

  Unnecessary objects detected in the Old generation space or in the Old and Permanent generation spaces were collected.

  Information is output in the output format shown in pattern 1.

- 11: CMS-GC processing in progress was terminated because there was a Full GC request due to a Java heap shortage.

  Unnecessary objects detected in the Old generation space or in the Old and Permanent generation spaces were not collected.

  Information is output in the output format shown in pattern 2.

- 21: CMS-GC processing in progress was terminated because there was a Full GC request due to an external factor such as java.lang.System.gc().

  Unnecessary objects detected in the Old generation space or in the Old and Permanent generation spaces were not collected.

  Information is output in the output format shown in pattern 2.

$3: Object amount before CMS-GC Processing (Old Generation Space)

This is the total object amount (in bytes) that existed in the Old generation space before CMS-GC processing.

Normally, this equals the total object amount that existed in the Old generation space when final mark processing was executed.

If it does not equal the total object amount that existed in the Old generation space when final mark processing was executed, this shows that GC for the New generation space was executed before unnecessary object collection processing by CMS-GC, or that objects were allocated to the Old generation space by running a Java application.

$4: Object amount after CMS-GC Processing (Old Generation Space)

This is the total object amount (in bytes) that exists in the Old generation space after CMS-GC processing.

$5: Old generation space size

Old generation space size (in bytes).

$6: Object amount before CMS-GC Processing (Permanent Generation Space)

This is the total object amount (in bytes) that existed in the Permanent generation space before CMS-GC processing was executed.

Normally, this equals the total object amount that existed in the Permanent generation space at final mark execution.

If it does not equal the total object amount that existed in the Permanent generation space when final mark processing was executed, this shows that objects were allocated to the Permanent generation space by running a Java application before unnecessary object collection processing by CMS-GC.

$7: Object amount after CMS-GC Processing (Permanent Generation Space)

This is the total object amount (in bytes) that exists in the Permanent generation space after CMS-GC processing was executed.

$8: Permanent generation space size

This is the Permanent generation space size (in bytes).

$9: CMS-GC processing execution time

This is the time (in seconds) taken to execute CMS-GC processing (the elapsed time from when CMS-GC was started).

Enabled GC Processing

The output format is extended by specifying this option when the GC processing types used are as follows:

- Serial GC
- Parallel GC
- Parallel GC with CMS

Increase in log output

Log output increases when this option is specified.

Logs should be monitored in this case as they may become very large.

Log View

Example of the output of the enhanced format of information that has been output as a GC processing result log.

```
23.646: [Full GC, [PSYoungGen : 1584K->0K(5504K)], [PSOldGen : 57764K->26302K(58304K)] 59348K-
>26302K(63808K), [PSPermGen : 4655K->4655K(16384K)], 0.1353549 secs]
```

From this output information, the following is understood:

- Full GC processing was executed 23.646 seconds after the Java VM was activated
- The GC processing being used is Parallel GC
- After GC processing, the size of the New generation area is 5,504,704 KB.

- As a result of GC Processing, the volume of objects in the New generation area changed from 1,584,384 KB to 0KB. (Unnecessary objects have been deleted, and Survivor objects are moved to the Old generation area according to their relevance.)
- After GC Processing, the size of the Old generation area is 58,304 KB.
- As a result of GC Processing, the volume of objects in the Old generation area changed from 57,764 KB to 26,302 KB. (Unnecessary objects have been deleted, and Survivor objects are moved to the New generation area according to their relevance.)
- After GC Processing, the size of the memory allocation pool is 63,808 KB.
- As a result of GC Processing, the general volume of objects in the memory allocation pool changed from 59,348 KB to 26,302 KB. (Unnecessary objects have been deleted.)
- After GC Processing, the size of the Permanent generation area is 16384KB.
- As a result of GC Processing, the volume of objects in Permanent generation area has not changed.
- The time required for GC Processing was 0.1353549 seconds.

Output example showing extended format of information output as the GC processing log (for Parallel GC with CMS - 1)

```
150.207: CMS start
150.208: [CMS initial-mark, [ParNew : 1863K->1863K(14784K)], [CMS : 53791K->53791K(65536K)] 55654K-
>55654K(80320K), [CMS Perm : 4664K->4664K(16384K)], 0.0030212 secs]
150.351: [GC, [ParNew : 14782K->1598K(14784K)], [CMS : 53791K->57981K(65536K)] 68573K-
>59579K(80320K), [CMS Perm : 4664K->4664K(16384K)], 0.0328537 secs]
150.466: [CMS remark, [ParNew : 8277K->8277K(14784K)], [CMS : 57981K->57981K(65536K)] 66258K-
>66258K(80320K), [CMS Perm : 4664K->4664K(16384K)], 0.0097905 secs]
150.549: [GC, [ParNew : 14782K->1598K(14784K)], [CMS : 50163K->54371K(65536K)] 64946K-
>55969K(80320K), [CMS Perm : 4664K->4664K(16384K)], 0.0303271 secs]
150.583: CMS stop(00), [CMS : 57981K->54200K(65536K)], 0.3753996 secs
```

From this output information, it is possible to determine that:

- The GC processing used is Parallel GC with CMS (the CMS-GC target is the Old generation space).
- CMS-GC processing started 150.207 seconds and finished 150.583 seconds after the Java VM started.
- Unnecessary objects were collected by CMS-GC that has finished.
- The Old generation space size after CMS-GC processing is 65536KB.
- Because of CMS-GC processing, the object amount in the Old generation space changed from 57981KB to 54200KB.
- The time taken to execute CMS-GC processing was 0.3753996 seconds.
- GC processing of the New generation space was executed while CMS-GC was processing. Additionally, the start of the execution might have lagged. Alternatively, the start of execution may have been delayed.

Output example showing extended format of information output as the GC processing results log (for Parallel GC with CMS - 2)

```
137.803: CMS start
137.804: [CMS initial-mark, [ParNew : 206690K->206690K(314560K)], [CMS :
655731K->655731K(699072K)] 862421K->862421K(1013632K), [CMS Perm : 3892K-
>3892K(16384K)], 0.4101250 secs]
139.069: [GC, [ParNew : 279616K->34943K(314560K)], [CMS : 655731K-
>673280K(699072K)] 935347K->708223K(1013632K), [CMS Perm : 3892K-
>3892K(16384K)], 0.2177910 secs]
142.140: CMS stop-req
142.501: CMS stop(11), 4.6984060 secs
142.501: [Full GC, [ParNew : 314559K->0K(314560K)], [CMS : 673280K-
>657037K(699072K)] 987839K->657037K(1013632K), [CMS Perm : 3892K-
>3892K(16384K)], 1.8642510 secs]
```

From this output information, it is possible to determine that:

- The GC processing used is Parallel GC with CMS (the CMS-GC target is the Old generation space).
- CMS-GC processing started 137.803 seconds and finished 142.501 seconds after the Java VM started.
- There was a FullGC request 142.140 seconds after the Java VM started, therefore CMS-GC that was running finished.
- The collection of unnecessary objects by CMS-GC that has finished was not executed.
- The Java application stopped for 0.361 seconds from the point of the CMS-GC processing completion request (142.140) until the completion of execution of CMS-GC processing (142.501), plus a FullGC execution time of 1.8642510 seconds for which execution started at 142.501, therefore the Java application stopped for a total of 2.225251 seconds.

6.3 Dynamic Compilation

This section explains dynamic compilation.

To run programs written in languages such as C/C++ and COBOL, a compiler for the respective language must first be used to translate the code into machine instructions that can be run on the platform used to execute the program source code. (This is referred to as static compilation as opposed to the dynamic compilation discussed below.) Platform-dependent executable binary code must then be created.

To run programs written in Java, javac commands must first be used to convert the program source code into instruction "byte code" that can be interpreted/executed by the Java VM, and "class files", which are platform-independent executable binary code, must then be created.

Once Java VM has been started up as the Java execution environment, Java VM reads the class files that make up the programs to be executed and then executes the class files in one of the following two ways:

- Byte code execution using an interpreter

  The Java VM interpreter interprets and executes the byte codes in the class files one instruction at a time.

  Execution performance is slow compared with the execution of machine instructions.

- Execution of byte codes translated into machine instructions using dynamic compilation

  During Java application execution, the Java VM automatically translates the byte codes for Java methods within the class files into machine instructions that can be run on the platform to be used for execution, and then executes the instructions. Because this translation processing is carried out automatically during Java application execution, it is referred to as **dynamic compilation**.

  Dynamic compilation allows programs to be executed faster than byte codes executed using an interpreter.

Dynamic compile processing also obtains the information needed for optimized processing of machine instructions during translation, such as the Java method execution frequency and call correlations, from the results of the profiling for each Java method carried out concurrently with Java application execution. Consequently, retranslation is repeated until a set amount of information is obtained from the profiling and the translation results are gradually optimized for the Java application execution status, which leads to improved execution performance for the machine instruction component.

The dynamic compilation performed by Java VM is seen as an overhead from the perspective of Java application execution. For this reason, the three components (interpreter execution performance (slow), overheads due to dynamic compilation and execution performance using the machine instructions generated by dynamic compilation) need to be adjusted to provide a balance that yields improved execution performance for Java applications as a whole.

By giving priority to the compilation of Java methods with high execution frequency and leaving relatively unused Java methods to be executed by the interpreter, the Java VM achieves a balance between interpreter execution, dynamic compilation and the execution of the machine instructions resulting from dynamic compilation, thereby tuning the processes to give better execution performance for Java applications as a whole.

Within Java applications to be executed, the ranking of each Java method in terms of its execution frequency is unknown until the Java application is actually run. For this reason, the Java VM conducts profiling for the Java methods concurrently with the execution of the Java application and uses the results of that profiling to determine the Java methods to be targeted for dynamic compilation. Immediately after the Java VM starts up, until a set amount of information is obtained from the profiling, only the interpreter is used for execution. But this gradually transitions to combined operation in which interpreter execution is coupled with the execution of machine instructions yielded by dynamic compilation.

The FJVM features a number of dynamic compilation features that are original functions developed by Fujitsu, as listed below.

All these functions are specific to the FJVM.

- 6.3.1 Compiler Error Automatic Recovery Function
- 6.3.2 Long Time Compilation Detection Function
- 6.3.3 Dynamic Compile Status Log Output Function

6.3.1 Compiler Error Automatic Recovery Function

When necessary, the Java VM automatically compiles Java methods that are executed as Java applications. If an error occurs during the compilation process, compilation of the relevant Java method and operation of the Java VM itself may terminate abnormally.

The FJVM is equipped with a **compiler error automatic recovery function** that automatically performs a recovery process and enables the Java VM to continue operating if an error occurs during the compilation process.

Note

The compiler error automatic recovery function is specific to the FJVM.

The Java method that was being compiled when the recovery process was initiated by this function will no longer be targeted for compilation. That Java method is not compiled and continues to operate as a Java application in interpreter mode.

This function operates as an internal process of the FJVM, so if the recovery process takes place normally even though an error has occurred in the compiler, no external notification or report will be output. To obtain information about problems that occur within the compiler when the recovery process succeeds, specify the option shown in the following figure.

If the option shown in the following figure is specified, information about the recovery process will be sent to the standard output in the format shown in the following figure.

Option to Receive Notification Following a Recovery Process

```
-XX:+PrintCompilerRecoveryMessage
```

Information Reported following a Recovery Process

```
CompilerRecovery: Information:The compilation was canceled for method method_name
Reason for the cancellation: reason [code:c, addr:xxxxxxxx]
```

- method_name: The name of the Java method that was being compiled when the error occurred in the compilation process.
- reason: Information about the cause of the error that occurred in the compilation process. This information includes the items shown in the following table.

assert	: Internal process conflict detected in compilation process
error	: Error detected in compilation process
stack overflow	: Stack overflow detected in compilation process

- c: Error code
- xxxxxxxx: Address where the error occurred during the compilation process.

6.3.2 Long Time Compilation Detection Function

When necessary, the Java VM automatically compiles Java methods executed by a Java application. This process normally finishes in an extremely short time.

However, if a fault within the compilation process itself or in another process running in the same Java process takes exclusive control of CPU resources, the compilation process may fail to terminate even after several minutes. This can have a detrimental effect on an entire system.

To prevent this kind of problem, the FJVM monitors the time required to compile each Java method, and if compilation does not finish after the expected time, it determines that a problem has occurred within the Java process and forcibly terminates it. This function is called the "Long time compilation detection function".

Note

The long time compilation detection function is specific to the FJVM.

This function is enabled when the option shown in the following figure is used to specify a monitoring time (the maximum time required for compilation) for a compilation process. If "0" is specified, this function is not enabled.

If the compilation process does not finish after the time specified by the option shown in the following figure elapses, this function determines that a problem exists within the Java process and forcibly terminates it.

Option to Enable the Long Time Compilation Detection Function

```
-XX:CompileTimeout=<nn>
```

<nn> is the maximum time (in seconds) required by the compilation process. This value is used to determine if a problem exists.

Note that the shortest unit of time that can be used by this function is 30 seconds, so the actual monitoring time may differ from the specified time by up to 30 seconds.

The default value for <nn> is 0, which disables the function.

If this function is used to forcibly terminate a Java process, the FJVM will send the message shown in the following figure to the standard output and then terminate. A core dump (crash dump) will also be output when the Java process is forcibly terminated.

Message Output when a Java Process is Forcibly Terminated by the Long Time Compilation Detection Function

```
CompilerRecovery: Information: CompilerRecovery got the VM aborted
 because the compiler thread(nnnnnnnn) has not completed.
 (compiling method: method_name)
```

nnnnnnnn: Internal identifier of compiler thread

method_name: The name of the Java method that was being compiled when the check performed by this function detected an error in a Java process

Notes on the Long Time Compilation Detection Function

Even if something prevents sufficient CPU resources from being allocated to a compilation process within a Java process and the compilation process itself does not proceed, the Java process will still be forcibly terminated by this function after the monitoring time specified by the "-XX:CompileTimeout" option elapses.

This means that if the CPU load of the system executing the Java process is high, a forcible termination by this function may occur because sufficient CPU resources are not being allocated to the compilation process.

If a forcible termination by this function occurs, the first thing to do is to check the following conditions:

- Does the system in which the relevant Java process is running have enough CPU resources?
- Are processes other than the relevant Java process monopolizing CPU resources?
- Does specifying "-XX:CompileTimeout=0" prevent the forcible termination from occurring and allow the relevant Java process to terminate normally or to change to an idle state when there is no load?

If any of the above conditions are met, it is possible that the forcible termination was the result of insufficient CPU resources being allocated to the compilation process.

If this problem occurs when the **Long time compilation detection function** is enabled, try setting a longer monitoring time with the "-XX:CompileTimeout" option.

Monitoring Messages Output by the Long Time Compilation Detection Function

When the option shown in the following figure is specified, the monitoring message shown in the following figure will be output if compilation of the Java method does not conclude after one minute.

The same message will then continue to be output every 30 seconds.

Note that the shortest unit of time that can be used by this function is 30 seconds, so the actual monitoring time may differ from the specified time by up to 30 seconds.

Option to Enable Monitoring Message Output by the Long Time Compilation Detection Function

```
-XX:+PrintCompilerRecoveryMessage
```

Monitoring Message Output by the Long Time Compilation Detection Function

```
CompilerRecovery: Information: The compiler thread(0xnnnnnnnn) might not
 return from compiling method method_name.
```

nnnnnnnn: Internal identifier of compiler thread

method_name: The name of the Java method that was being compiled when this function detected a problem

6.3.3 Dynamic Compile Status Log Output Function

The Java VM automatically carries out compile processing as needed for Java methods executed as Java applications (dynamic compilation).

The FJVM features a Dynamic Compile Status Log Output Function that outputs the status of dynamic compilation.

The Dynamic Compile Status Log Output Function outputs the following information:

- Compiler thread CPU usage

  The elapsed time and CPU time used for compiling are output each time a Java method with 20 compiler threads (dynamically compiled threads) is compiled.

  If the CPU time is high as a proportion of the elapsed time, dynamic compilation may be impacting on Java application execution performance.

- Dynamic compilation results information

  Information is output on which Java methods were compiled and when they were compiled.

  If Java method compilation occurs continuously in a short period, dynamic compilation may be impacting on Java application execution performance.

Specify the option in the following figures to output the compiler thread CPU usage.

Specify the option in the following figures to output the compiler thread CPU usage and the dynamic compilation results information.

Specifying the options in the following figures, outputs the status log for each occurrence of dynamic compilation to standard output in the formats shown in Figure Output format for compiler thread CPU usage and Output format for dynamic compilation results information.

Output samples are shown in Sample output when -XX:+PrintCompilationCPUTime is specified and Sample output when -XX:+FJPrintCompilation is specified.

Note

The Dynamic Compile Status Log Output Function is specific to the FJVM.

Option for outputting compiler thread CPU usage

```
-XX:+PrintCompilationCPUTime
```

Option for outputting compiler thread CPU usage and dynamic compilation results information

```
-XX:+FJPrintCompilation
```

Output format for compiler thread CPU usage

```
$1: [$2: cpu=$3ms elapsed=$4ms $5]
```

The elements in Output format for compiler thread CPU usage are explained below.

$1: Time of log output

Shows the time at which the log was output.

The format of the time of log output can be specified using the 6.5.7 Time Information Format Specification Function in the Log Output.

The default is "time elapsed (in seconds) from when the Java VM was started".

$2: Compiler thread name

Shows the name of the compiler thread for which information was output in the "CompilerThreadNo." format.

$3: CPU time

Shows the CPU time (in milliseconds) used by the $2 compiler thread to compile 20 Java methods.

Note that, if the compilation target was a native method, the processing time for this method is not included in the CPU time.

$4: Elapsed time

Shows the elapsed time (in milliseconds) taken by the $2 compiler thread to compile 20 Java methods.

$5: Consecutive numbering

Shows consecutive numbering for the number of times this information has been output for each compiler thread.

The value of "consecutive numbering x 20" is the total number of Java methods compiled in that compiler thread.

Output format for dynamic compilation results information

```
$1: $2 $3 ($4 bytes) $5
```

The elements in Output format for dynamic compilation results information are explained below.

$1: Java method compile request issue time (time of log output)

Shows the time at which the Java method compile request was issued (time of log output).

The format of the time of log output can be specified using "6.5.7 Time Information Format Specification Function in the Log Output".

The default is "time elapsed (in seconds) from when the Java VM was started".

$2: Consecutive numbering

Shows consecutive numbering for the number of compile requests (the number of Java methods in which compile requests were issued).

If there is no percentage symbol (%) at the end of the consecutive number, this is a request to compile the entire java method.

If there is a percentage symbol (%) at the end of the consecutive number, this is a request to compile the java method in parts.

Separate consecutive numbering is used for numbers with and without the percentage symbol (%).

$3: Java method name

Shows the name of the Java method in which the compile request was issued.

If the request is to compile the Java method in parts (if a percentage symbol (%) in $2), information is added after the Java method name ("(@No.)") to indicate the part of the Java method (byte code) from which compilation is to begin.

$4: Java method byte count

Shows the size of the Java method to be compiled (byte code size) as a byte count.

$5: Blank space or (static)

If the compilation target was a native method, "(static)" is output.

If the compilation target is not a native method, nothing is output to this location.

Note that the native methods names are not shown when using JDK/JRE 6,they are shown only when using JDK/JRE 7.

Sample output when -XX:+PrintCompilationCPUTime is specified

```
0.586: [CompilerThread1: cpu=78.13ms elapsed=450.72ms 1]
0.822: [CompilerThread0: cpu=437.50ms elapsed=686.32ms 1]
1.312: [CompilerThread0: cpu=218.75ms elapsed=489.93ms 2]
1.637: [CompilerThread1: cpu=546.88ms elapsed=1050.52ms 2]
2.385: [CompilerThread0: cpu=296.88ms elapsed=1073.57ms 3]
3.365: [CompilerThread0: cpu=140.63ms elapsed=979.67ms 4]
3.557: [CompilerThread1: cpu=343.75ms elapsed=1919.97ms 3]
4.096: [CompilerThread1: cpu=390.63ms elapsed=539.47ms 4]
4.995: [CompilerThread1: cpu=140.63ms elapsed=898.45ms 5]
```

Sample output when -XX:+FJPrintCompilation is specified

```
0.074:   1   java.util.Properties$LineReader::readLine (383 bytes)
0.102:   2   java.io.Win32FileSystem::normalize (143 bytes)
0.107:   3   java.lang.String::hashCode (60 bytes)
0.179:   4   sun.security.provider.SHA::implCompress (494 bytes)
0.206:   5   sun.reflect.UTF8::utf8Length (81 bytes)
0.229:   6   java.util.jar.Manifest$FastInputStream::readLine (167 bytes)
0.232:   7   sun.nio.cs.UTF_8$Decoder::decodeArrayLoop (1814 bytes)
0.244:   1%  sun.text.NormalizerDataReader::read @ 38 (139 bytes)
0.261:   8   java.math.BigInteger::mulAdd (82 bytes)
  (This section omitted)
0.742: [CompilerThread1: cpu=406.25ms elapsed=677.52ms 1]
0.744:  40   java.lang.String::replace (142 bytes)
```

6.4 Tuning Methods

The following points are crucial to effective tuning:

1. There is a close relationship between memory consumption and processing speed. In general, if memory consumption is reduced, processing speed drops as well. In JDK/JRE, if more than the required amount of memory is reserved for the Java heap, garbage collection of the New generation area will become less frequent, but the time required for a FullGC will increase and adversely affect processing speed.

2. There is a limit to the memory resources that are allocated to a process. In JDK/JRE, the area required for operation of the stack, Java heap, native modules and other segments is allocated to the user space. If too much area is reserved for one segment, there will be less space available for other segments.

This information should be considered when tuning the JDK/JRE.

6.4.1 Tuning the Java Heap

This section explains how to tune the Java heap, and how changes to the Java heap size will affect a system.

Tuning Methods

It is possible to set up the size of each Java heap area by specifying the options shown in Table 6.3 Options Relating to the Java Heap.

The default initial and maximum sizes of the memory allocation pool are shown in Table 6.4 Default Sizes of the Memory Allocation Pool.

The default initial and maximum sizes of the Permanent generation area are shown in Table 6.5 Default Sizes of the Permanent Generation Area.

Additionally, default values for each option that are not defined in Table 6.3 Options Relating to the Java Heap are shown in Table 6.6 Default values for the Java heap-related options (-XX:NewSize/-XX:NewRatio).

Table 6.3 Options Relating to the Java Heap

Option	Function of option (*1)
-Xms	Specifies the initial size of the memory allocation pool. For example, if the initial size of the memory allocation pool is set to 128MB, "-Xms128m" is specified. The default value for this option is shown in "Table 6.4 Default Sizes of the Memory Allocation Pool". If the value specified is less than 1 MB or is smaller than or equal to the value in the -XX:NewSize option (including the default value), an initialization error occurs and the Java process terminates.
-Xmx	Specifies the maximum size of the memory allocation pool. For example, if the maximum size of the memory allocation pool is set to 256MB, "-Xmx256m" is specified. (The actual value used may differ slightly from the specified value because it is an adjusted value that is tuned so as to optimize control by the Java VM based on system information such as the page size.) The default value for this option is shown in "Table 6.4 Default Sizes of the Memory Allocation Pool". If the specified value (or adjusted value) is smaller than the value specified in the -Xms option, an initialization error occurs and the Java process terminates.
-XX:NewSize	Specifies the heap size of the New generation area. For example, if the heap size of the New generation area is set to 128MB, "-XX:NewSize=128m" is specified. The default value for this option is: - If type0 or type 0p Parallel GC with CMS is used, 1/8 of the initial value for the memory allocation pool. - If type1 or type 1p Parallel GC with CMS is used, 1/3 of the initial value for the memory allocation pool. - If type2 or type 2p Parallel GC with CMS is used, 1/16 of the initial value for the memory allocation pool.

Option	Function of option (*1)
	- In all other cases: Refer to "Table 6.6 Default values for the Java heap-related options (-XX:NewSize/-XX:NewRatio)". If the specified value is less than the minimum value for the Java VM, an initialization error occurs and the Java process terminates. So a value of at least 1 MB should be specified for this option.
-XX:MaxNewSize	Specifies the maximum heap size of the New generation area. For example, if the maximum heap size of the New generation area is set to 128MB, "-XX:MaxNewSize=128m" is specified. The default value for this option is: - If type0 or type 0p Parallel GC with CMS is used, 1/8 of the maximum value for the memory allocation pool. - If type1 or type 1p Parallel GC with CMS is used, 1/3 of the maximum value for the memory allocation pool. - If type2 or type 2p Parallel GC with CMS is used, 1/16 of the maximum value for the memory allocation pool. - In all other cases: No default value (the New generation space maximum heap size will not be determined by this option). If the specified value is less than the value specified in the -XX:NewSize option, the value specified in the -XX:NewSize option is used.
- XX:NewRatio (*3)	Specifies the size ratio between the New and Old generation areas. For example, if the size ratio between the New and Old generation areas is 2, "-XX:NewRatio=2" is specified. The default value for this option is shown in "Table 6.6 Default values for the Java heap-related options (-XX:NewSize/-XX:NewRatio)".
- XX:SurvivorRatio (*2) (*4)	Specifies the size ratio between the Eden and Survivor areas that make up the New generation area. For example, if the size ratio between the Eden and Survivor areas is 8, "-XX:SurvivorRatio=8" is specified. The default value for this option is 8.
- XX:TargetSurvivorRatio (*2) (*4)	Adjusts the proportion of the Survivor area occupied by objects that still exist after a garbage collection (GC) process to the specified percentage value. For example, if the proportion of the Survivor area occupied by objects that still exist after a garbage collection (GC) process needs to be adjusted to half, "-XX:TargetSurvivorRatio=50" is specified. The default value is 50.
-XX:PermSize	Specify the initial value of the Permanent generation area. For instance, specify "-XX:PermSize=32m" if 32MB is to be set for the initial value of the Permanent generation area. The default value is shown in "Table 6.5 Default Sizes of the Permanent Generation Area". If the specified value is less than 1 MB, an initialization error occurs and the Java process terminates.
-XX:MaxPermSize	Specifies the maximum size of the Permanent generation area. For example, if the maximum size of the Permanent generation area is set to 128MB, "-XX:MaxPermSize=128m" is specified. (The actual value used may differ slightly from the specified value because it is an adjusted value that is tuned so as to optimize control by the Java VM based on system information such as the page size.)

Option	Function of option (*1)
	The default value is shown in "Table 6.6 Default values for the Java heap-related options (-XX:NewSize/-XX:NewRatio)". If the specified value (or adjusted value) is less than the value specified in the -XX:PermSize option, the value specified in the -XX:PermSize option is used.

*1 In the size options, the following characters can be specified as units:

If specifying KB (kilobyte):"k" or "K"

If specifying MB (megabyte):"m" or "M"

*2 If Parallel GC is used, the value specified for this option is invalid.

*3 If Parallel GC with CMS is used, the value specified for this option is invalid.

*4 When Parallel GC with CMS is used and -XX:UseFJcmsGC=type0 or =type0p is not specified, the value specified for this option will be invalid.

Table 6.4 Default Sizes of the Memory Allocation Pool

JDK/JRE version	OS	JDK/JRE Execution mode	GC	Initial value	Maximum value
6	Windows Linux for x86 Linux for Intel64	32-bit	Serial GC	5.0MB	64MB
			Parallel GC (default)	8.0MB	
			Parallel GC with CMS	(*1)	
	Windows Server(R) x64 Editions Linux for Intel64	64-bit	Serial GC	7.75MB	84MB
			Parallel GC (default)	9.1875MB	
			Parallel GC with CMS	(*1)	
	Solaris	32-bit	Serial GC	6.125MB	64MB
			Parallel GC (default)	8.0MB	
			Parallel GC with CMS	(*1)	
		64-bit	Serial GC	7.9375MB	84MB
			Parallel GC (default)	12.0MB	
			Parallel GC with CMS	(*1)	
7	Windows Linux for x86 Linux for Intel64	32-bit	Serial GC	5.0MB	96MB
			Parallel GC (default)	7.37MB	
			Parallel GC with CMS	(*1)	
	Windows Server(R) x64 Editions Linux for Intel64	64-bit	Serial GC	6.4375MB	126MB
			Parallel GC (default)	8.3125MB	
			Parallel GC with CMS	(*1)	
	Solaris	32-bit	Serial GC	5.0MB	96MB
			Parallel GC (default)	FJVM: 8.0MB Client VM: 7.375MB	
			Parallel GC with CMS	(*1)	
		64-bit	Serial GC	6.4375MB	128MB
			Parallel GC (default)	12.0MB	
			Parallel GC with CMS	(*1)	

*1 This will be the maximum value for the memory allocation pool if it the latter is less than 64MB, otherwise it will be 64MB.

Table 6.5 Default Sizes of the Permanent Generation Area

JDK/JRE version	OS	JDK/JRE Execution mode	Java VM	Initial value	Maximum value
6 7	Windows Solaris Linux for x86 Linux for Intel64	32-bit	Java HotSpot Client VM	12MB	64MB
			FJVM (default)	16MB	
	Windows Server(R) x64 Editions Linux for Intel64	64-bit	FJVM (default)	20.75MB	84MB
	Solaris	32-bit	Java HotSpot Client VM	12MB	64MB
			FJVM (default)	16MB	
		64-bit	FJVM (default)	20.75MB or 24MB	84MB

Table 6.6 Default values for the Java heap-related options (-XX:NewSize/-XX:NewRatio)

JDK/JRE version	OS	JDK/JRE execution mode	Java VM	-XX:NewSize	-XX:NewRatio
6	Windows Linux for x86 Linux for Intel64	32-bit	Java HotSpot Client VM	1024KB	12
			FJVM (default)		8
	Windows Server(R) x64 Editions Linux for Intel64	64-bit	FJVM (default)	2624KB	2
	Solaris	32-bit	Java HotSpot Client VM	2176KB	8
			FJVM (default)		2
		64-bit	FJVM (default)	2816KB	2
7	Windows Linux for x86 Linux for Intel64	32-bit mode	Java HotSpot Client VM	1024KB	2
			FJVM (default)		
	Windows Server(R) x64 Editions Linux for Intel64	64-bit mode	FJVM (default)	1280KB	
	Solaris	32-bit	Java HotSpot Client VM	1024KB	2
			FJVM (default)		
		64-bit	FJVM (default)	1280KB	

Tuning Guidelines

The following guidelines should be observed when tuning the Java heap.

1. If a memory shortage occurs even though a FullGC has been performed, obtain the GC log and check if either the memory allocation pool or the Permanent generation area is short of space.

2. A FullGC incurs a performance cost. If a Java application becomes temporarily unresponsive and appears to have frozen even though there is no memory shortage, it may be the result of a FullGC. Obtain the GC log, and if the Java heap is larger than it needs to be, make it smaller.

3. Compared to an efficient GC on a New generation area, a FullGC incurs a performance cost. For this reason, the sizes of the New and Old generation areas must be in balance. If you use Parallel GC for GC processing, you do not normally need to consider the issue of balance because the values for the sizes of the New and Old generation areas in the Java heap are automatically adjusted and optimized.

4. If there is plenty of available virtual memory, consider increasing process concurrency by starting multiple Java processes. By increasing process concurrency, each process can use the user space more effectively. When using this method, however, users must be aware that slowdowns may occur as a result of memory-to-disk swapping.

Tuning Effects

Changes to the size of the entire Java heap can have the following effects:

- If the entire Java heap is reduced, GC operations can occur more frequently.

- If the size of the entire Java heap is increased, FullGC operations may take longer to complete.

- If you increase the size of the memory allocation pool, the amount of user space and virtual memory is reduced by the same amount, which may prevent the area needed to run native modules and stacks from being reserved and cause a memory shortage.

- If you tune the Java heap so that the New generation area is half the size of the memory allocation pool, this generally facilitates the occurrence of FullGC. However, this will not facilitate FullGC for every application since it is dependent on factors such as the generation and release of objects during Java application execution.

Notes that Apply when the Overcommit Memory Function is Enabled Linux32/64

In Linux, when the overcommit memory function is enabled, the operating system reserves virtual memory resources equivalent to the maximum amount of each Java heap area for Java processes when the Java VM starts up.

Therefore, when Java processes are started with different values specified for "-Xms" and "-Xmx", the amount of virtual memory required for the Java heap when the Java processes start up will be different depending on whether this function is enabled or disabled.

- **When the overcommit function is disabled or the function is not available:**

  Amount of virtual memory for the Java heap = Value of "-Xms" option + Initial value of Permanent generation area size

- **When the overcommit function is enabled:**

  Amount of virtual memory for the Java heap = Value of "-Xmx" option + Value of "-XX:MaxPermSize" option

Therefore, even when two systems have identical virtual memory resources, the number of Java processes that can be run concurrently on each system varies according to whether this function is enabled or disabled.

Always consider the presence of the overcommit memory function when estimating virtual memory resources in Linux.

6.4.2 Tuning the Stack

This section explains how to tune the stack of threads used in the Java application, and the impact of the tuning.

Tuning Methods

The stack size of the thread generated in Java API can be specified in the "-Xss" option in bytes (to set 512KB, specify "-Xss512k").

The stack size of the dedicated thread that automatically compiles the Java method executed in JDK/JRE (the compiler thread) can be specified in the "-XX:CompilerThreadStackSize" option.

Normally, there is no need to specify the stack size of the compiler thread.

The stack size of the compiler thread specified in the "-XX:CompilerThreadStackSize" option is specified in kilobytes (KB). For example, to set 1,024KB as the stack size, specify "-XX:CompilerThreadStackSize=1024".

Note that the actual management of the stack area is performed by the operating system. For this reason, how the stack area is managed and how it behaves will depend on the specification for the operating system that is used to execute JDK/JRE.

Table 6.7 The default stack size of the threads generated in Java API and the compiler threads are shown below.

JDK/JRE version	OS	JDK/JRE Execution mode	Thread generated in Java API (KB) (*1)	Compiler thread (KB)	
				Client VM (*1)	FJVM
6 7	Windows	32-bit	320	320	2048
	Linux for x86 Linux for Intel64		320	512	2048
	Windows Server(R) x64 Editions Linux for Intel64	64-bit	1024	-(*2)	4096
	Solaris	32-bit	512	512	2048
		64-bit	1024	-(*2)	4096

*1 The stack size in Windows JDK/JRE is the value that uses a JDK tool provided in Windows JDK/JRE, such as java.exe. If the Windows application starts the Java VM independently using JNI, the value is the same as the stack size for the main thread of the program that starts the Java VM.

*2 Java HotSpot Client VM is not bundled when the execution mode is 64-bit mode JDK/JRE.

Tuning Effects

Changing the size of the stack will have the following effects:

1. If the size of the stack is reduced, stack overflows may occur.
2. If the size of the stack is increased, there will be a commensurate reduction in the amount of user space and virtual memory. This can make it impossible to reserve the area required for the Java heap and native modules to operate, and result in memory shortages.

6.4.3 Warm-up

This section describes warm-up, which consists of the procedures for checking the impact of dynamic compilation during Java application execution, and how to deal with that impact.

6.4.3.1 Tuning

As described in "6.3 Dynamic Compilation", immediately after the Java VM starts up, Java application execution is carried out solely using interpreter execution and this gradually transitions to combined operation in which interpreter execution is coupled with the execution of machine instructions yielded by dynamic compilation. And as the execution of the Java applications progresses, the content of the machine instructions translated by dynamic compilation is gradually customized to match the Java application execution status. In other words, because the execution of Java applications involves shifts in the optimization status and overheads due to dynamic compilation as outlined below, it may take some time from when the process starts for stable execution performance for Java applications to be achieved.

- Because the class files that make up programs to be executed are loaded during execution, some overheads occur due to class file loading and content inspection immediately after a Java application starts.
- Translation of coding into machine instructions is carried out during execution, which also causes overheads.
- Because the information needed for translation to machine instructions and optimization processing is collected concurrently with Java application execution, there is minimal shift to an optimized state immediately after the Java application starts. As a result, execution performance may be slow for those machine instructions.

Note also that the time taken until stable execution performance is achieved for Java applications varies depending on the application.

(1) Checking the dynamic compile status

The presence or absence of any impact by dynamic compilation during Java application execution is determined based on whether dynamic compiling occurs frequently once transaction by the Java application have started.

More specifically, the decision is made by using the 6.3.3 Dynamic Compile Status Log Output Function to output the CPU usage by compiler threads and the dynamic compilation results information, and then determining whether trends can be observed in those results and whether there is any observable correlation with the time taken for stable execution performance to be achieved for Java applications.

- If the CPU time makes up a high proportion of the elapsed time in the CPU usage by compiler threads
- If Java method compilation occurs successively in a short interval in the dynamic compilation results information

Note

The tendency for the occurrence of dynamic compile processing to become more frequent when a Java application starts and input to the Java application begins as transactions start is a normal trend when Java applications are executed. So even where checking of the dynamic compile status reveals trends such as a high proportion of CPU time being taken up by compiler threads or very frequent occurrences of dynamic compilation, there are many instances where there is still a good balance between interpreter execution and execution using machine instructions and no particular intervention is needed.

Checking of the dynamic compile status should be carried out in situations where the time taken for stable execution performance to be achieved for Java applications is posing problems for actual system operation.

Dynamic compile processing is not the only process that impact on performance immediately after Java applications start up. You must also consider factors other than dynamic compilation that also come into play only in the period immediately after a Java application starts up, such as the initialization processing for the Java application itself and the startup wait time for related applications.

If Java applications such as Servlet and EJB are run on Interstage Application Server, this has the following implications for Java application startup:

a. Interstage Application Server startup (J2EE environment work unit startup and JavaEE environment cluster startup))

b. Job application operation startup

The frequency of dynamic compile processing increases immediately after both (a) and (b), but because the aim is to check the impact made by the job, the checking for the dynamic compile status should be applied only to (b).

To check the start of (b) in the dynamic compilation results information (the results output when the -XX:+FJPrintCompilation option is specified), check the completion time of (a) in the system log, etc., and then check dynamic compilation that occurred after that time.

If the Java application is written using JSP and JSP precompiling (conversion of the Java source to class files using the javac command) is not performed when the application is prepared, this may result in overheads arising when the javac command is run at Java application startup.

If JSP precompiling operations can be performed, use JSP precompiling and check whether this shortens the time taken until stable execution performance is achieved for the Java application.

(2) Warm-up

After a Java application starts up, an operation called warm-up is carried out so that stable execution performance is achieved by the time operation for transactions begins.

Warm-up refers to a series of operations carried out after a Java application starts up and before actual transactions begin. By simulating execution using dummy data similar to the actual transactions, warm-up runs the main Java methods in advance, loads the class files and completes dynamic compilation of the Java methods.

Warm-up provides a way to minimize the overheads of processes such as dynamic compilation, and because translation to machine instructions and optimization processing can be carried out at that point, stable execution performance can be achieved as soon as operation for the transactions begins.

The benchmark for deciding how long a period should be allowed for warm-up is the time taken until Java application execution performance is stable while still meeting the operational requirements. However, there is only a limited period between Java application startup and the start of transactions, so a decision must be made to limit the time taken for warm-up at a certain point based on the results of dynamic compilation output by the 6.3.3 Dynamic Compile Status Log Output Function.

The following figure shows an example of the dynamic compilation results information output (when the -XX:+FJPrintCompilation option is specified).

Note

Normally, Java application execution is accompanied by results such as data updates and execution results display. Care must be taken to ensure that execution results from warm-up do not affect actual operation of the Java application.

For example, take care to ensure that the warm-up dummy data is not directly stored in the database so that problems such as incorrect data being returned do not occur during operation.

If factors other than dynamic compilation, such as initialization of the Java application itself or startup wait time for related applications, result in a delay until stable execution performance is achieved for the Java application, warm-up cannot be used to remedy the problem.

Sample dynamic compilation results information output (-XX:+FJPrintCompilation option specified)

```
0.133:   1   sun.misc.ASCIICaseInsensitiveComparator::compare (143 bytes)
0.137:   2   java.lang.String::charAt (33 bytes)
0.142:   3   java.lang.String::hashCode (64 bytes)
0.204:   4   java.lang.String::equals (88 bytes)
0.210:   5   java.lang.String::indexOf (151 bytes)
0.215:   6   java.lang.AbstractStringBuilder::append (40 bytes)
0.226:   7   java.lang.String::replace (142 bytes)
(this section omitted)
0.444:  30   java.util.jar.Attributes::read (410 bytes)
0.470:  31   java.lang.String::<init> (111 bytes)
0.486: [CompilerThread0: cpu=140.63ms elapsed=367.55ms 1]
0.486:  32   java.util.jar.Attributes$Name::isValid (32 bytes)
0.487:  33   java.util.jar.Attributes$Name::isAlpha (30 bytes)
0.487:  34   sun.nio.cs.ext.MS932$Decoder::decodeSingle (19 bytes)
(this section omitted) Frequent dynamic compilation at Application Server startup
15.449: 796   net.jxta.impl.document.LiteXMLElement::addAttribute (450 bytes)
15.538: 797   java.io.ObjectOutputStream::writeHandle (21 bytes)
15.574: 798   java.util.Hashtable$Enumerator::hasNext (5 bytes)
15.600: 799   com.sun.enterprise.config.ConfigBean::addXPathToChild (27 bytes)
15.606: [CompilerThread0: cpu=250.00ms elapsed=781.02ms 24]
15.630: 800   java.util.regex.Matcher::<init> (84 bytes)
23.535: 801   java.util.Arrays::copyOf (47 bytes)
25.542: 802   java.util.Arrays::copyOf (13 bytes)
25.545: 803   net.jxta.document.MimeMediaType::findNextSeperator (39 bytes)
43.567: 804   net.jxta.impl.document.LiteXMLElement::getTagRanges (1697 bytes)
At this point, Interstage Application Server startup is complete and dynamic compilation decreases
briefly.
  The completion time for Interstage Application Server startup (J2EE environment work unit startup
and JavaEE environment cluster startup) can be checked in the Interstage Application Server log
(server information log).)
44.212: 805   org.apache.tomcat.util.buf.Ascii::toLower (14 bytes)
44.223: 806   org.apache.tomcat.util.buf.ByteChunk::equalsIgnoreCase (76 bytes)
44.224: 807   sun.nio.cs.ISO_8859_1$Decoder::decodeArrayLoop (263 bytes)
44.240:  18%  java.util.Properties$LineReader::readLine @ 21 (452 bytes)
44.258: 808   org.apache.coyote.http11.InternalInputBuffer::parseHeader (585 bytes)
44.265: 809   java.util.Properties$LineReader::readLine (452 bytes)
44.290: 810   java.lang.ThreadLocal$ThreadLocalMap::nextIndex (15 bytes)
44.332: 811   java.util.Hashtable$Enumerator::next (27 bytes)
44.387: 812   java.net.URI::quote (208 bytes)
44.412:  19%  sun.nio.cs.ext.DoubleByteEncoder::encodeArrayLoop @ 55 (608 bytes)
(this section omitted) Transactions start and dynamic compilation again becomes frequent
158.121: 1128   sun.util.calendar.ZoneInfo::getOffsetsByWall (8 bytes)
160.374: 1129
javax.management.NotificationBroadcasterSupport::sendNotification (118 bytes)
160.698: 1130   java.util.TimeZone$DisplayNames::access$000 (4 bytes)
161.405: 1131
com.sun.jmx.remote.internal.ArrayNotificationBuffer::addNotification (144 bytes)
163.572: 1132   sun.nio.cs.ext.DoubleByteDecoder::decodeLoop (28 bytes)
164.460: 1133   sun.util.calendar.CalendarDate::isDaylightTime (22 bytes)
170.743: 1134   java.util.AbstractCollection::toArray (116 bytes)
Once a short time has passed after transactions begin, occurrences of dynamic compilation decrease.
```

6.5 Tuning and Debugging Techniques

This section introduces some tuning and debugging techniques.

6.5.1 Stack Trace

The stack trace, which is output when a Java application throws an exception (java.lang.Throwable instance), shows the sequence of events (the method invocation sequence) leading up to an error. The location and cause of an error can be determined by analyzing the stack trace.

Stack Trace Output Destination

The stack trace is output to the standard error output. For normal Java applications, the stack trace is output to the console, but for a Servlet, JSP or EJB application, it is output to the container information log (info.log).

Stack Trace Output Method

The stack trace can be output by catching an exception thrown in Java with the catch clause and then executing the printStackTrace method of the exception.

Outputting a Stack Trace with the printStackTrace Method

```
try {
    SampleBMPSessionRemote bmpSessionRemote = bmpSessionHome.create();
} catch(Exception e) {

    e.printStackTrace();

}
```

If the thread does not contain a method that can process the thrown exception with try-catch, the thread will be stopped and the stack trace will be output by the Java VM.

Stack Trace Output Format

Stack Trace Output Format

```
Exception class name: Error message
  at class name.method name1(source file name:line number)  Invocation destination
  at class name.method name2(source file name:line number)
  at ...
  at ...                                               Invocation source
```

- The first line contains the class name of the thrown exception and the error message.

  Sometimes there is no error message.
- The second and subsequent lines are output from bottom to top, leading from the method invocation source to the invocation destination.

  The second line contains information about the method that threw the exception.
- When *method name* is "<init>", it signifies a constructor.
- When *method name* is "<clinit >", it signifies a static initializer.
- When (source file name:line number) is "(Native Method)", it signifies a Java native method (.so or .dll file).
- If debug information is deleted when the class is compiled, (source file name:line number) may show only a source name or "Unknown Source".

6.5.1.1 Analyzing the Stack Trace (1)

This section explains how to analyze the stack trace. It uses the sample output shown in the following figure.

The numbers shown at the beginning of each line of the following figure have been added to facilitate the explanation.

Sample Stack Trace Output

```
1:  Java.lang.NullPointerException
2:  at agency.attestation.CheckLoginInfo.doCheck(CheckLoginInfo.java:150)
3:  at agency.attestation.AttestationServlet.doGet(AttestationServlet.java:96)
4:  at agency.attestation.AttestationServlet.doPost(AttestationServlet.java:161)
5:  at javax.servlet.http.HttpServlet.service(HttpServlet.java:772)
6:  at javax.servlet.http.HttpServlet.service(HttpServlet.java:865)
              :
```

How to Read the Stack Trace

By reading upwards from line 6 in the above figure, it is evident that the exceptions occurred in the following order:

1. javax.servlet.http.HttpServlet.service() executed javax.servlet.http.HttpServlet.service() on line 865 of HttpServlet.java; then
2. javax.servlet.http.HttpServlet.service() executed agency.attestation.AttestationServlet.doPost() on line 772 of HttpServlet.java; then
3. agency.attestation.AttestationServlet.doPost() executed agency.attestation.AttestationServlet.doGet() on line 161 of AttestationServlet.java; then
4. agency.attestation.AttestationServlet.doGet() executed agency.attestation.CheckLoginInfo.doCheck() on line 96 of AttestationServlet.java; and as a result
5. An exception called "java.lang.NullPointerException" occurred on line 150 of CheckLoginInfo.java in agency.attestation.CheckLoginInfo.doCheck().

Method of Analysis

The following is sample analysis of the stack trace shown in the above figure:

1. Determine if the cause of the exception can be identified from the exception information on line 1.

   It is clear that a NullPointerException has been thrown.
2. If you are in charge of developing CheckLoginInfo.java in line 2, check if there is a problem with the implementation of line 150 of CheckLoginInfo.java.
3. If you are not in charge of developing CheckLoginInfo.java in line 2, look for a class developed by the person developing the part at the top of the stack trace and check if there are any problems with the implementation of that class. If the cause of the problem still cannot be found, ask the provider of the class used by the developed class to investigate the cause.

Another method of analysis is to check if the methods were being executed in the desired order.

6.5.1.2 Analyzing the Stack Trace (2)

This section explains how to analyze the stack trace. It uses the sample output shown in the following figure.

The numbers from the beginning of each line of the following figure have been added to facilitate the explanation.

Sample Stack Trace Output

```
1:java.util.MissingResourceException: Can't find bundle for base name
sample.SampleResource, locale ja_JP
2:  at java.util.ResourceBundle.throwMissingResourceException(Unknown Source)
3:  at java.util.ResourceBundle.getBundleImpl(Unknown Source)
4:  at java.util.ResourceBundle.getBundle(Unknown Source)
5:  at sample.SampleMessage.getMessage(SampleMessage.java:15)
6:  at sample.SampleServlet.doGet(SampleServlet.java:10)
```

```
7:  at javax.servlet.http.HttpServlet.service(HttpServlet.java:696)
8:  at javax.servlet.http.HttpServlet.service(HttpServlet.java:809)
    :
    :
```

Method of Analysis

The following is sample analysis of the stack trace shown in the above figure:

1. Check if the cause of the exception can be identified from the exception information in line 1.

   According to the Java API reference, this exception occurs when java.util.MissingResourceException does not have Java resources. In addition, the error message indicates that the Japanese version (ja_JP) of a resource file called "sample.SampleResource" is missing.

2. Check the resource file.

   a. Is the name of the resource file correct?

      An exception has been thrown as a result of java.util.ResourceBundle.getBundle() being executed within sample.SampleMessage.getMessage () in line 15 of SampleMessage.java. Check if the name of the resource file passed to java.util.ResourceBundle.getBundle() is correct.

   b. Does the resource file exist in the prescribed directory configuration?

      If the name of the resource file referred to in a) above is correct, check if one of the following resource files exists within the prescribed directory configuration (/sample/):

      - SampleResource_ja_JP.properties
      - SampleResource_ja_JP.class
      - SampleResource_ja.properties
      - SampleResource_ja.class
      - SampleResource.properties
      - SampleResource.class

6.5.1.3 Analyzing the Stack Trace (3)

The following constructors and methods have been added to JDK/JRE 1.4:

- Throwable(java.lang.String, java.lang.Throwable)
- Throwable(java.lang.Throwable)
- initCause(java.lang.Throwable)

These enable the stack trace of an exception causing a problem to be output to another stack trace.

This is explained below using the following figure as an example.

Sample Program

```
public class Test {

    public static void main(String[] args) {
       new Test();
    }

    Test() {
        try{
            parentMethod();
        } catch (Exception e) {
            e.printStackTrace();
12:        }
13:    }
14:
15:    void parentMethod() throws HiLevelException {
16:        try {
17:            childMethod();
18:        } catch (Exception e) {
19:            throw new HiLevelException("HiLevel", e);
20:        }
21:    }
22:
23:    void childMethod() throws LowLevelException {
24:        throw new LowLevelException("LowLevel");
25:    }
26:}
27:
28:class HiLevelException extends Exception {
29:    HiLevelException(String msg, Throwable cause) {
30:        super(msg, cause);
31:    }
32:}
33:
34:class LowLevelException extends Exception {
35:    LowLevelException(String msg) {
36:        super(msg);
37:    }
38:}
```

When the sample program shown in the above figure is executed, the stack trace shown in the following figure is output.

Stack Trace

```
HiLevelException: HiLevel
    at Test.parentMethod(Test.java:19)
    at Test.<init>(Test.java:9)
    at Test.main(Test.java:4)

Caused by: LowLevelException: LowLevel
    at Test.childMethod(Test.java:24)
    at Test.parentMethod(Test.java:17)
    ... 2 more
```

Following HiLevelException, the stack trace of LowLevelException, which caused the HiLevelException, is output after "Caused by:". "... 2 more" in the last line indicates that the two lines immediately preceding "Caused by:" make up the next stack trace.

The process can therefore be interpreted as shown in the following figure.

Identifying the Exception that is the Root Cause of a Problem

```
Caused by: LowLevelException: LowLevel
    at Test.childMethod(Test.java:24)
    at Test.parentMethod(Test.java:17)
    at Test.<init>(Test.java:9)
    at Test.main(Test.java:4)
```

The following facts can be concluded from the above:

- LowLevelException was the cause of the stack trace.

- Test.main (line 4 of Test.java) was the first invocation source.

  For further details, refer to the description of the printStackTrace method of java.lang.Throwable in the Java API Reference.

6.5.2 Output of the Stack Trace when an Exception Occurs

The exceptions below are raised when the Java application is executed using FJVM. From the perspective of performance, however, processing to output the stack trace when the exception occurs using optimization processing performed by the Java VM dynamic compilation might be omitted, and the stack trace might not be output when the exception occurs (output of the stack trace when the exception occurs is prevented).

- **JDK/JRE 6 and 7 FJVM**
  - java.lang.NullPointerException
  - java.lang.ArithmeticException
  - java.lang.ArrayIndexOutOfBoundsException
  - java.lang.ArrayStoreException
  - java.lang.ClassCastException

The "-XX:-OmitStackTraceInFastThrow" option is specified so that processing to output the stack trace when the exception occurs using dynamic compilation is not omitted.

If the Java application for which this option is specified is executed in circumstances in which the exception occurs frequently, the performance of the application might deteriorate.

When specifying this option, either use it after performing performance validation, or only use it when you want to identify the location in which the exception occurred in development work.

Note

The "-XX:-OmitStackTraceInFastThrow" option is specific to the FJVM.

Note

If the above exception has occurred in the Java method that was executed in the interpreter, the stack trace for the exception is output regardless of whether or not this option has been specified.

6.5.3 Thread Dumps

A thread dump contains information about each thread of a Java process (in stack trace format), and can be used to investigate operational issues such as hangups and deadlocks.

Thread dumps are output to the standard output. The following table shows where and how thread dumps are output.

Table 6.8 Timing and Locations of Thread Dump Output

Program type	Output timing	Output destination
J2EE/JavaEE application	If the fixed conditions have been filled, there is both automatic collection using the container function, and manual collection using User optional timings. Automatic collection: The application has timed out or not replied. Manual collection: Windows32/64 Collection can be done with "Thread Dump Tool". Solaris32/64 Linux32/64	The standard output is logged and then output to the file.

Program type	Output timing	Output destination
	The QUIT signal corresponding to the Java VM can be sent and collected by using kill - QUIT [Process ID]	
Java programs other than J2EE/JavaEE	Manual collection using User's optional timing can be performed. Windows32/64 If starting the Java program from Command Prompt : Collection can be done by either of the methods described below. 1) Hit [Ctrl]+[Break] 2) "Thread Dump Tool" If starting the Java program with something other than Command Prompt: Collect with "Thread Dump Tool". Solaris32/64 Linux32/64 If starting the Java program from terminal: Collection can be done by either of the methods described below. 1. press [Ctrl] + [\] (Use / on an English keyboard.) 2. kill -QUIT [Process ID] If starting the Java program with something other than terminal: Gather with kill -QUIT [ProcessID]. Note If "-Xrs" options have been specified for the Java process, the [Ctrl]+[Break] key hit which has been sent to the process in question, or the operation corresponding to the QUIT signal, will be the OS default operation. As a result, if [Ctrl]+[Break] keys are hit in relation to a Java process -Xrs" with options specified, or QUIT single has been sent, the said Java process will force quit or abnormally terminate. Please do not specify "-Xrs" in the options for a Java process for which a thread dump is a possibility. If the Java process is saved as service by Windows(R), and "-Xrs" is not specified, force quit will occur during log off. If this is inconvenient, specify "-Xrs" option.	Console (standard output)

The method for analyzing the thread dump is explained using the output example shown in the following figure.

Sample Program

```
1 :public class DeadlockSample {
2 :    static boolean flag;
3 :    static Thread1 thread1;
4 :    static Thread2 thread2;
5 :
6 :    public static void main(String[] args) {
7 :        thread1 = new Thread1();
8 :        thread2 = new Thread2();
9 :        thread1.start();
10:        thread2.start();
11:    }
12:}
13:
14:class Thread1 extends Thread {
15:    public Thread1(){
16:        super("Thread1");
17:    }
18:
19:    public void run(){
20:        synchronized(this){
21:            System.out.println("Thread1 start");
22:            while(DeadlockSample.flag==false){  // Waits for Thread2 to start
23:                yield();
24:            }
25:            DeadlockSample.thread2.method();
26:            notify();
27:        }
28:    }
29:
30:    public synchronized void method(){
31:        try{wait(1000);}catch(InterruptedException ex){}
32:        System.out.println ("Thread1.method() ends ");
33:    }
34:}
35:
36:class Thread2 extends Thread {
37:    public Thread2(){
38:        super("Thread2");
39:    }
40:
41:    public void run() {
42:        synchronized(this){
43:            DeadlockSample.flag = true;
44:            System.out.println("Thread2Start" Thread2 starts);
45:            DeadlockSample.thread1.method();
46:            notify();
47:        }
48:    }
49:
50:    public synchronized void method() {
51:        try{wait(1000);}catch(InterruptedException ex){}
52:        System.out.println("Thread2.method()ends");
53:    }
54:}
```

In the sample program in the above figure, Thread1 and Thread2 perform mutually exclusive processes.

When this sample program is run, processing takes place as follows:

1. A Thread1 lock is obtained by Thread1 (synchronized clause in line 20).

2. A Thread2 lock is obtained by Thread2 (synchronized clause in line 42).
3. Thread1 attempts to execute Thread2.method() and waits for a lock release (synchronized modifier in line 50).
4. Thread2 attempts to execute Thread1.method() and waits for a lock release (synchronized modifier in line 50).

The result of this process is that Thread1 and Thread2 enter a deadlock state as they both continue to wait for a lock that will not be released.

A thread dump collected during deadlock is shown in the following figure.

Thread Dumps

```

"DestroyJavaVM" prio=5 tid=0x002856c8 nid=0x5f4 waiting on condition [0..6fad8]

"Thread2" prio=5 tid=0x0092f4d8 nid=0x640 waiting for monitor entry [182ef000..182efd64]
    at Thread1.method(DeadlockSample.java:31)
    - waiting to lock <0x1002ffe8> (a Thread1)
    at Thread2.run(DeadlockSample.java:45)
    - locked <0x10030ca0> (a Thread2)

"Thread1" prio=5 tid=0x0092f370 nid=0x294 waiting for monitor entry [182af000..182afd64]
    at Thread2.method(DeadlockSample.java:51)
    - waiting to lock <0x10030ca0> (a Thread2)
    at Thread1.run(DeadlockSample.java:25)
    - locked <0x1002ffe8> (a Thread1)

"Signal Dispatcher" daemon prio=10 tid=0x0098eb80 nid=0x634 waiting on condition [0..0]

"Finalizer" daemon prio=9 tid=0x0092a540 nid=0x5e8 in Object.wait() [1816f000..1816fd64]
    at java.lang.Object.wait(Native Method)
    - waiting on <0x10010498> (a java.lang.ref.ReferenceQueue$Lock)
    at java.lang.ref.ReferenceQueue.remove(ReferenceQueue.java:111)
    - locked <0x10010498> (a java.lang.ref.ReferenceQueue$Lock)
    at java.lang.ref.ReferenceQueue.remove(ReferenceQueue.java:127)
    at java.lang.ref.Finalizer$FinalizerThread.run(Finalizer.java:159)

"Reference Handler" daemon prio=10 tid=0x0096da70 nid=0x5e4 in Object.wait() [1812f000..1812fd64]
    at java.lang.Object.wait(Native Method)
    - waiting on <0x10010388> (a java.lang.ref.Reference$Lock)
    at java.lang.Object.wait(Object.java:429)
    at java.lang.ref.Reference$ReferenceHandler.run(Reference.java:115)
    - locked <0x10010388> (a java.lang.ref.Reference$Lock)

"VM Thread" prio=5 tid=0x0096c950 nid=0x624 runnable

"VM Periodic Task Thread" prio=10 tid=0x0092c008 nid=0x2a0 waiting on condition
"Suspend Checker Thread" prio=10 tid=0x0098e118 nid=0x478 runnable

Found one Java-level deadlock:
=============================
"Thread2":
  waiting to lock monitor 0x00929c3c (object 0x1002ffe8, a Thread1),
  which is held by "Thread1"
"Thread1":
  waiting to lock monitor 0x00929c5c (object 0x10030ca0, a Thread2),
  which is held by "Thread2"

Java stack information for the threads listed above:
===================================================
"Thread2":
    at Thread1.method(DeadlockSample.java:31)
    - waiting to lock <0x1002ffe8> (a Thread1)
    at Thread2.run(DeadlockSample.java:45)
    - locked <0x10030ca0> (a Thread2)
"Thread1":
    at Thread2.method(DeadlockSample.java:51)
    - waiting to lock <0x10030ca0> (a Thread2)
    at Thread1.run(DeadlockSample.java:25)
    - locked <0x1002ffe8> (a Thread1)

Found 1 deadlock.
```

Method of Analysis

The information about threads in a thread dump is in stack trace format.

The stack traces of both Thread1 and Thread2 contain the strings "locked" and "waiting to lock". The string "deadlock" also appears at the bottom of the thread dump, indicating that a deadlock occurred.

In this way, the thread dump can be used to check the operation of all threads, and to check if a Java process has stopped responding, or if a deadlock has occurred. If multiple thread dumps collected at short intervals indicate that threads have stopped operating, it is likely that a hangup has occurred.

The thread that has the locked object is not output to the thread dump

Normally, this might be displayed in a thread on the thread dump as follows:

```
- waiting to lock <object ID> (a class name)
```

This happens when another thread has the lock for that object ID, and the following is displayed somewhere on the trace of that thread:

```
- locked <object ID> (a class name)
```

Depending on the timing of the thread dump display, however, "*- locked <object ID> (class name a)*" might not appear in the thread, and only "*- waiting to lock <object ID> (class name a)*" might be displayed.

An example of the program is shown below.

```
class NoLockOwner extends Thread
{
  static Object lock = new Object();

  public static void main(String[] arg)
  {
    new NoLockOwner().start();
    new NoLockOwner().start();
  }

  public void run()
  {
    while (true) {
      synchronized (lock) {
        dumb();
      }
    }
  }

  void dumb()
  {
    int n = 0;
    for (int i = 0 ; i < 1000 ; ++i)
      n += i;
    25    }
    26
    27  }
```

(0) When the thread dump is collected, the following normally occurs:

```
"Thread-1" prio=1 tid=0x10 nid=0x5 waiting for monitor entry [0x3000..0x4000]
    at NoLockOwner.run(NoLockOwner.java:14)
    - waiting to lock <0x800> (a java.lang.Object)
"Thread-0" prio=1 tid=0x20 nid=0x6 runnable [0x5000..0x6000]
    at NoLockOwner.dumb(NoLockOwner.java:23)
    at NoLockOwner.run(NoLockOwner.java:15)
    - locked <0x800> (a java.lang.Object)
```

(1) When the object is locked in the top frame, "- *waiting to lock*" might be displayed instead of "- locked".

```
"Thread-1" prio=1 tid=0x10 nid=0x5 waiting for monitor entry [0x3000..0x4000]
    at NoLockOwner.run(NoLockOwner.java:14)
    - waiting to lock <0x800> (a java.lang.Object)
"Thread-0" prio=1 tid=0x20 nid=0x6 runnable [0x5000..0x6000]
    at NoLockOwner.run(NoLockOwner.java:14)
    - waiting to lock <0x800> (a java.lang.Object)
```

In this case, check the thread status - if it is "runnable", then it is possible that the same frame is running after the lock was obtained, instead of the status being "waiting to lock".

"- *waiting to lock <0x800> (a java.lang.Object)*" is displayed for both Thread-0 and Thread-1, therefore appear to have "waiting to lock" status.

The status of Thread-0 is "runnable", however, therefore it does not have "waiting to lock" status.

(2) "- *waiting to lock*" is also displayed during processing to obtain the lock, not only in "waiting to lock" status.

```
"Thread-1" prio=1 tid=0x10 nid=0x5 waiting for monitor entry [0x3000..0x4000]
    at NoLockOwner.run(NoLockOwner.java:14)
    - waiting to lock <0x800> (a java.lang.Object)
"Thread-0" prio=1 tid=0x20 nid=0x6 waiting for monitor entry [0x5000..0x6000]
    at NoLockOwner.run(NoLockOwner.java:14)
    - waiting to lock <0x800> (a java.lang.Object)
```

"- *waiting to lock <0x800>* (a java.lang.Object)" is displayed for both Thread-0 and Thread-1, therefore appear to have "waiting to lock" status.

The thread status is also "waiting for monitor entry" in both cases.

"- *waiting to lock*" is also displayed during processing to obtain the lock, not only in "waiting to lock" status.

Accordingly, in this case, it is possible that the status of either Thread-0 or Thread-1, or both, is processing to obtain the lock.

This status does not last for a long time, however, and changes to (0) or (1) after a short time.

(3) Status immediately after the lock is released

```
"Thread-1" prio=1 tid=0x10 nid=0x5 waiting for monitor entry [0x3000..0x4000]
    at NoLockOwner.run(NoLockOwner.java:14)
    - waiting to lock <0x800> (a java.lang.Object)
"Thread-0" prio=1 tid=0x20 nid=0x6 runnable [0x5000..0x6000]
    at NoLockOwner.run(NoLockOwner.java:16)
```

Thread-0 is the status immediately after the lock is released.

This status does not last for a long time either, and changes to (0) or (1) after a short time.

In determining the application status from the thread dump, it is difficult to determine the status from one thread dump. To determine the appropriate status, multiple thread dumps must be looked at comprehensively.

Line numbers of the synchronized method displayed in the thread dump

The following type of program is possible:

```
 1    class SyncMethod extends Thread
 2    {
 3      static volatile int k;
 4
 5      public static void main(String[] arg)
 6      {
 7        new SyncMethod().start();
 8      }
 9
10      public void run()
11      {
12        while (true) {
13          dumb();
14        }
15      }
16
17      synchronized void dumb()
18      {
19        /*
20          meaningless comments
21         */
22        int i = 0;
23        for ( ; i < 10 ; ++i)
24          k += i;
25      }
26
27    }
```

When this type of program thread dump is collected, the following type of program might be output:

```
"Thread-0" prio=1 tid=0x300 nid=0x61 runnable [1000..2000]
    at SyncMethod.dumb(SyncMethod.java:23)
    - waiting to lock <0xa00> (a SyncMethod)
    at SyncMethod.run(SyncMethod.java:13)
```

In line 23, it appears that the lock has been obtained, however it appears that the source code of this line - (; i < 10 ; ++i) - has nothing to do with the obtaining of the lock.

This is because the line number for the obtaining of the lock by the synchronized method is where the code is first executed.

When multiple threads in the thread dump obtain the lock

The following type of program is possible (this program contains errors because the purpose is that you understand the detection of these errors):

```
class NoNotify extends Thread
{
  static Object o = new Object();

  public static void main(String[] arg)
  {
    new NoNotify().start();
    new NoNotify().start();
  }

  public void run()
  {
    try {
      synchronized (o) {
        o.wait();
      }
    } catch (Exception e) {}
  }

}
```

Since there is no notify thread in this program, neither thread will be triggered permanently.

When this program thread dump is collected, there are the following locations:

```
"Thread-1" prio=1 tid=0x800 nid=0x6 in Object.wait() [1000..2000]
    at java.lang.Object.wait(Native Method)
    - waiting on <0x200> (a java.lang.Object)
    at java.lang.Object.wait(Object.java:429)
    at NoNotify.run(NoNotify.java:15)
    - locked <0x200> (a java.lang.Object)
"Thread-0" prio=1 tid=0x900 nid=0x7 in Object.wait() [3000..4000]
    at java.lang.Object.wait(Native Method)
    - waiting on <0x200> (a java.lang.Object)
    at java.lang.Object.wait(Object.java:429)
    at NoNotify.run(NoNotify.java:15)
    - locked <0x200> (a java.lang.Object)
```

From this thread dump, it can be seen that no thread has obtained a lock, not that multiple threads have obtained a lock.

Thread-0 and Thread-1 both appear to have locked an object with the same object ID: "<0x200>".

If "- locked" is displayed for a thread, however, it does not mean that this is the current lock owner (the correct meaning of "- locked" is simply that the object is locked in that frame).

In the top frame, "- waiting on" is displayed. Unlike "- waiting to lock", which might be triggered when the lock is released, "- waiting on" is not triggered automatically when the lock is released.

Note: Java VM internal threads

In the thread dump, the threads shown below are the internal threads of the Java VM. For this reason, the information for those threads does not help to look into the application.

- "Attach Listener"
- "C2 CompilerThread*" (where "*" is a digit)
- "Finalizer"
- "RAS Control Thread"
- "Reference Handler"
- "Signal Dispatcher"
- "VM Periodic Task Thread"
- "VM Thread"
- "Low Memory Detector" (exists when using JDK/JRE 6)

- "Service Thread" (exists when using JDK/JRE 7)
- "GC task thread#* (ParallelGC)" (exists only when Parallel GC is used) (*1)
- "Concurrent Mark-Sweep GC Thread" (exists when Parallel GC with CMS is used)
- "Surrogate Locker Thread (Concurrent GC)" (exists when Parallel GC with CMS is used)
- "Gang worker#* (Parallel GC Threads)" (exists when Parallel GC with CMS is used) (*1)
- "Gang worker#* (Parallel CMS Threads)" (exists when Parallel GC with CMS is used) (*2)

1 There are as many GC threads as specified in the "-XX:ParallelGCThreads" option. Note that "" in the name is a digit.

2 There are as many CMS-GC threads as specified in the "-XX:ConcGCThreads" option. Note that "" in the name is a digit.

Outputting Java Heap Area Information

In JDK/JRE 6 and7, information on the Java heap area is output along with the thread dump output.

Due to differences in each type of Garbage Collection processing, the Java heap area information includes different output character strings for each of the New, Old and Permanent generation areas.

Values shown as percentages, indicate the proportion of the memory that the JavaVM had made available (committed) for use as Java heap when the information was output. It does not show the proportion of the maximum amount available. For this reason, rather than referring to the values given as percentages, use values obtained by comparing the values for memory usage shown in kilobytes (K) and the values specified in options (including default values).

- When Serial GC is used:

  "def new generation" is information for the New generation area, "tenured generation" is information for the Old generation area, and "compacting perm gen" is information for the Permanent generation area.

- When Parallel GC is used:

  "PSYoungGen" is information for the New generation area, "PSOldGen" is information for the Old generation area, and "PSPermGen" is information for the Permanent generation area.

- When Parallel GC with CMS is used:

  "par new generation" is information for the New generation area, "concurrent mark-sweep generation" is information for the Old generation area, and "concurrent-mark-sweep perm gen" is information for the Permanent generation area.

  Note that information about "object space" for Old and Permanent generation space is not output.

Sample output:

```
Heap
 PSYoungGen      total 7168K, used 5158K [0x0fd60000, 0x10470000, 0x10470000)
  eden space 7104K, 72% used [0x0fd60000,0x102658f0,0x10450000)
  from space 64K, 25% used [0x10450000,0x10454000,0x10460000)
  to   space 64K, 0% used [0x10460000,0x10460000,0x10470000)
 PSOldGen        total 4096K, used 162K [0x0c470000, 0x0c870000, 0x0fd60000)
  object space 4096K, 3% used [0x0c470000,0x0c498870,0x0c870000)
 PSPermGen       total 16384K, used 2103K [0x08470000, 0x09470000, 0x0c470000)
  object space 16384K, 12% used [0x08470000,0x0867dd40,0x09470000)
```

6.5.4 Class Instance Information Output Function

If a thread dump output procedure is implemented for a Java process with the option in the following figure specified, the thread dump output is followed by instance information for each class in the Java heap in the format shown in the following figure. In the FJVM, this functionality is provided by the Class Instance Information Output Function.

Because the class instance information output consists of the total size and number of instances for each class, it can be used to check for problems such as memory leaks in the Java heap.

Class instance information is output to standard output. The triggers and output destinations for class instance information are the same as for "6.5.3 Thread Dumps" output.

Note also that if the -Xloggc option is specified, the output destination for class instance information changes from standard output to the file specified in the -Xloggc option.

Option enabling the function for outputting class instance information after a thread dump

```
-XX:+PrintClassHistogram
```

Output format for class instance information

```
num     #instances         #bytes  class name
----------------------------------------------
 $1:            $2             $3  $4
           :
           (omitted)
           :
Total           $5             $6
```

The elements in the above figure are explained below:

- $1: Order - class instance information is sorted for output by total class instance size, with the largest listed first.
- $2: No. of class instances
- $3: Total class instance size
- $4: Class name
- $5: Total of $2 values
- $6: Total of $3 values

The Class Instance Information Output Function excludes any unnecessary instances before collecting and outputting the information. Consequently, FullGC is implemented as preprocessing. Note that excessive use of class instance information output will result in frequent FullGC procedures.

If the FullGC that should be carried out prior to class instance information output cannot be performed because garbage collection is not performed (for details, refer to "Preventing Execution of Garbage Collection Processing"), the message shown in the following figure is output to standard output and the request for class instance information output is then cancelled.

Message output when a class instance information output request is cancelled

```
The PrintClassHistogram operation was canceled because GC could not be run.
```

6.5.5 Stack Trace Output Function that runs when java.lang.System.gc() is Executed

If a Java application executes the following Java methods frequently, it can impose a load on the Java VM and cause the response performance of an application to decline.

- java.lang.System.gc()
- java.lang.Runtime.gc()

The stack trace output function that runs when java.lang.System.gc() is executed is explained below with reference to java.lang.System.gc() (shortened to System.gc()).

To enable the execution status of the System.gc() method to be checked when a Java application is executed, the FJVM provides a **stack trace output function that runs when java.lang.System.gc() is executed**. This function outputs a stack trace of the Java thread that executed the relevant method.

The **stack trace output function that runs when java.lang.System.gc() is executed** is activated by specifying the option shown in the following figure.

If the System.gc() method is executed within a Java application when this function is enabled, stack trace information relating to the Java thread that executed the method will be sent to the standard output in the format shown in the following figure.

The results output to the standard output can also be saved to a file as FJVM log information. If the garbage collection result log was specified by using the "-verbose:gc" option, then the result log after the stack trace when .System.gc() was executed is also stored to the file. The file name and storage location are the same as those used when information is output after a Java VM terminates abnormally. Refer to 6.5.8 FJVM Log for details.

Option to Enable the Stack Trace Output Function that runs when java.lang.System.gc() is Executed

```
-XX:+PrintJavaStackAtSystemGC
```

Example of Output Generated by the Stack Trace Output Function that Runs when java.lang.System.gc() is Executed

```
"main" prio=10 tid=0x087c1c00 nid=0xd2a runnable [0xb75ab000..0xb75ab214]
   java.lang.Thread.State: RUNNABLE
                        at java.lang.Runtime.gc(Native Method)
                        at java.lang.System.gc(System.java:928)
                        at SystemGC.main(SystemGC.java:8)

"main" prio=10 tid=0x087c1c00 nid=0xd2a runnable [0xb75ab000..0xb75ab214]
   java.lang.Thread.State: RUNNABLE
                        at java.lang.Runtime.gc(Native Method)
                        at SystemGC.foo(SystemGC.java:4)
                        at SystemGC.main(SystemGC.java:10)
```

The sample output in the above figure indicates that java.lang.System.gc() was executed from SystemGC.main, and that java.lang.Runtime.gc() was executed from SystemGC.foo.

Note

Information displaying the status of the thread in the stack trace information (the line before the trace information) from JDK/JRE 6 is now visible.

In the example in the centre of the figure, "java.lang.Thread.State: RUNNABLE" is displayed. However, there are also cases where in the place of "RUNNABLE", the following are displayed: "NEW", "TIMED_WAITING (sleeping)", "WAITING (on object monitor)", "TIMED_WAITING (on object monitor)", "WAITING (parking)", "TIMED_WAITING (parking)", "BLOCKED (on object monitor)", "TERMINATED", "UNKNOWN".

6.5.6 Java VM Termination Status Output Function

If a Java process terminates with an unexpected status and without outputting any special messages, the Java application may have executed one of the following processes at an unexpected location:

- java.lang.System.exit()
- java.lang.Runtime.exit()
- java.lang.Runtime.halt()

The Java VM termination status output function is explained below with reference to java.lang.System.exit() (shortened to System.exit()).

When a Java application executes System.exit() to explicitly terminate a Java process, the Java VM will not output any special messages because as far as the Java VM is concerned, everything is running as expected. Therefore, if a Java application whose internal operation is not clearly understood (because, for example, the source code is not available) terminates with an unexpected status and there is no source code to examine, there is no way of checking if System.exit() has been executed.

To prevent this problem, the FJVM **Java VM termination status output function** makes it possible to check if System.exit() was executed when a Java process terminated.

The **Java VM termination status output function** is activated by specifying the option shown in the following figure.

Option to Enable Java VM Termination Status Output Function

```
-XX:+VMTerminatedMessage
```

If a Java process is terminated by the execution of System.exit() when this function is enabled, stack trace information (and other information relating to the thread that executed System.exit()) will be sent to the standard output in the format shown in the following figure. The presence of stack trace information (and its contents) make it possible to determine whether or not System.exit() was executed.

The results output to the standard output can also be saved to a file as FJVM log information. The file name and storage location are the same as those used when information is output after a Java VM terminates abnormally. Refer to 6.5.8 FJVM Log for details.

When this function is enabled and stack trace information such as that shown in the following figure is not output when a Java process terminates (that is, when just a message beginning with "#### JavaVM terminated: ..." is output), it is likely that System.exit() was not used, and the termination was the result of the control logic of the Java application.

Similarly, when this function is enabled and no information such as that shown in the following figure is output when a Java process terminates, it is likely that the Java process terminated for a different reason, such as the exit() function of a C runtime being called from within a native module.

Example of the Java VM Termination Status Output Function

```
Thread dump at JVM_Halt(status code=1234):
"main" prio=3 tid=0x00030000 nid=0x2 runnable [0xfe5ff000..0xfe5ffd80]
   java.lang.Thread.State: RUNNABLE
                        at java.lang.Shutdown.halt0(Native Method)
                        at java.lang.Shutdown.halt(Shutdown.java:105)
        - locked <0xfa81e7d0> (a java.lang.Shutdown$Lock)
                        at java.lang.Shutdown.exit(Shutdown.java:179)
        - locked <0xf3dc8dd8> (a java.lang.Class for java.lang.Shutdown)
                        at java.lang.Runtime.exit(Runtime.java:90)
                        at java.lang.System.exit(System.java:906)
                        at JVM_Halt.main(JVM_Halt.java:5)

#### JavaVM terminated: Java HotSpot(TM) Server VM
(**.*.**_FUJITSU_MODIFIED-B** mixed mode), [pid=29500] TimeMillis=1243580170483
Time=Fri May 29 15:56:10 2012
```

In the sample output shown in the above figure, the Java process terminated as a result of System.exit() being executed by JVM_Halt.main().

Note

Information displaying the status of the thread in the stack trace information (the line before the trace information) from JDK/JRE 6 is now visible.

In the example in the centre of the figure, "java.lang.Thread.State: RUNNABLE" is displayed. However, there are also cases where in the place of "RUNNABLE", the following are displayed: "NEW", "TIMED_WAITING (sleeping)", "WAITING (on object monitor)", "TIMED_WAITING (on object monitor)", "WAITING (parking)", "TIMED_WAITING (parking)", "BLOCKED (on object monitor)", "TERMINATED", "UNKNOWN".

This information is not used in the determination of the status at the time of Java VM completion.

6.5.7 Time Information Format Specification Function in the Log Output

The option in the following figure can be used to specify the format of the "time of log output" information included in the log output by the options below.

If the option in the following figure is not specified, the time of log output is output in the "elapsed time (in seconds) from when the Java VM was started" format (the format when "-XX:FJverboseTime=type1" is specified).

- Garbage Collection log output

  -verbosegc -XX:+UseFJverbose

- Dynamic Compile Status Log Output Function

  -XX:+PrintCompilationCPUTime

  -XX:+FJPrintCompilation

Option specifying the format of the time information in log output

```
-XX:FJverboseTime=type
The following values can be specified as the type:
type1
type2
type3
```

-XX:FJverboseTime=type1 specified

The time of log output is shown as the elapsed time (in seconds) from when the Java VM was started.

For example: 0.165:

-XX:FJverboseTime=type2 specified

Shows the time of log output as "Date/time (iso8601 format)".

For example: 2010-10-01T13:37:36.881+0900:

-XX:FJverboseTime=type3 specified

Shows the time of log output as "Date/time (iso8601 format)" and shows the elapsed time (in seconds) from when the Java VM was started. The output sequence is "date/time" "elapsed time".

For example: 2010-10-01T13:37:36.881+0900: 0.164:

6.5.8 FJVM Log

In the **FJVM**, the log output function for abnormal Java VM termination has been enhanced.

The FJVM log is output when a Java process terminates abnormally.

The log can be used to investigate the reason for an abnormal termination.

FJVM Log Output Destination

The FJVM log is output as the file name shown in the following figure to the current directory of the Java process.

FJVM Log File Name

```
fjvm_pid***.log (*** represents the process ID of the Java process that
 terminated abnormally)
```

For details of the current directory when an IJServer cluster is used, refer to "IJServer Clusters" in the "Java EE Operator's Guide".

For details of the current directory when IJServer is used, refer to "Environments in which J2EE Applications are Run (IJServer)" in the "J2EE User's Guide (Backward Compatible)".

Examining the FJVM Log

The FJVM log contains a variety of information for the Java process that terminated abnormally. The following information can be used to investigate the cause of the problem:

1. Information about the location of an abnormal termination
2. Signal handler information at the time of the abnormal termination (Solaris and Linux versions only)
3. Information about the Java heap at the time of the abnormal termination

Each of these information categories is discussed in the following sections.

6.5.8.1 Information about the Location of an Abnormal Termination

1. Information about exceptions that occurred at the time of the abnormal termination (Signal codes and exception occurrence addresses)

   Information begins from "Unexpected Signal:"

2. The names of functions that terminated abnormally (symbolic names that are closest to the addresses where abnormal terminations occurred)

   Information begins from "Function name="

3. Names of libraries that included functions that terminated abnormally

   Information begins from "Library="

4. A stack trace of Java threads that were active at the time of the abnormal termination

   Information begins from "Current Java thread:"

5. A list of dynamic libraries at the time of the abnormal termination

   Information begins from "Dynamic libraries:"

6. The time of the abnormal termination

   Information begins from "Local Time ="

Examination Procedure

First, identify the function that terminated abnormally from the information provided in items 1 to 3 above, then check if the function was invoked by a Java application. Note however, that the name output as the name of the function that terminated abnormally in item 2 above is a symbolic name that is closest to the address where the abnormal termination occurred, and may differ from the name of the function that terminated abnormally.

If the function is used by a running Java application, also check if any problems occur when the relevant function is used.

If the function is not used by a running Java application, investigate the stack trace shown in item 4 above.

In cases where the first method of the stack trace information was Native Method (after the method name "(Native Method)" will appear) the probability of a problem arising which relates to the JNI process will be high. Because of this, it will be confirmed if any problems have arisen which relate to control of the JNI process that has been output by the stack trace information.

Also in cases where the library including the function that terminated abnormally is a User created library, the probability of a problem arising within the user-side created library will be high. Because of this, it will be confirmed if any problems have arisen within the library's internal process or the JNI process which calls the library.

The method used to examine the stack trace is explained in "6.5.1 Stack Trace".

Detecting a Stack Overflow

If the single code notation in the following figure appears in the information relating to the exception that occurred at the time of the abnormally terminated time exception in item 1 above, either a message stating that the stack overflow that occurred at the thread that was made by the exception, (1) or (3), or one stating that there is a possibility that it has occurred, (2) or (4), will be displayed.

In such cases, the problem may be resolved by increasing the size of the stack that corresponds to the thread where the exception occurred.

If a stack overflow is due to the size of the stack corresponding to a thread created in a Java API, refer to "6.4.2 Tuning the Stack" and adjust the size of the relevant stack.

Signal Code that Indicates a Stack Overflow

Solaris32/64 Linux32/64

- (1) SIGSEGV (Stack Overflow)
- (2) SIGSEGV (Stack Overflow ?)

Windows32/64

- (3) EXCEPTION_STACK_OVERFLOW
- (4) EXCEPTION_ACCESS_VIOLATION (Stack Overflow?)

Analyzing the Dr. Watson Log Windows32/64

When an abnormal termination is caused by a stack overflow, control may fail to pass from the operating system to the FJVM, and may pass to Dr. Watson instead. In such cases, no information will be output to the FJVM log, and the Dr. Watson log file should be examined.

If the exception number shown in the following figure has been output to the Dr. Watson log, it is possible that a stack overflow has occurred.

Refer to "6.5.9.1 Crash Dumps" for a description of Dr. Watson.

Exception Number Indicating a Stack Overflow

```
c00000fd (Stack overflow)
```

6.5.8.2 Signal Handler Information at the Time of Abnormal Termination

Solaris32/64 Linux32/64

This refers to signal handler information that corresponds to the signals shown in the following table. These signals are required to control the execution of the Java VM.

Table 6.9 Signals Needed to Control the Java VM

Java VM for Solaris	Java VM for Linux
SIGSEGV	SIGSEGV
SIGPIPE	SIGPIPE
SIGBUS	SIGBUS
SIGILL	SIGILL
SIGFPE	SIGFPE
INTERRUPT_SIGNAL (the default is SIGUSR1)	INTERRUPT_SIGNAL (the default is SIGUSR1) (*2)
ASYNC_SIGNAL (the default is SIGUSR2)	SR_SIGNUM (the default is SIGUSR2)
SIGQUIT (*1)	SIGQUIT *1)
SIGINT (*1)	SIGINT (*1)
SIGHUP (*1)	SIGHUP (*1)
SIGTERM (*1)	SIGTERM (*1)
SIGXFSZ (*3)	SIGXFSZ (*3)

*1 This is a signal operation target using the -Xrs option.

*2 This is a signal reserved in the Java VM.

*3 This is a signal used in the JDK/JRE 6 and 7 Java VM.

Signal handler information for these signals is not output.

The following information is output as signal handler information:

- Address of the registered signal handler
- Information that shows whether or not a registered signal handler was registered by the Java VM. If it was registered by a process other than the Java VM, the text "(not in VM)" will appear in the line containing the relevant signal handler information.

If the signal handlers shown in Table 6.9 Signals Needed to Control the Java VM were registered by a process other than the Java VM, the Java VM will not operate correctly. In such cases, modify the application so that it does not register the signal handler concerned.

6.5.8.3 Information about the Java Heap at the Time of the Abnormal Termination

This information can be used to determine how the Java heap was being used when an abnormal termination occurred.

If an abnormal termination was caused by the size of the Java heap, it is possible to identify which Java heap was exhausted when the abnormal termination occurred.

6.5.8.4 Sample Output and Sample Investigation

The following explanation is based on sample output produced in the Solaris version of JDK/JRE 6:

```
----------------------------------------------------------------------------
#### Java VM: Java HotSpot(TM) Server VM (**.*.**_FUJITSU_MODIFIED-B**[*****] mixed mode)
>>>> Logging process start. [pid=27758] Time=Wed May 30 19:48:01 2012
----------------------------------------------------------------------------
```

1. Information about the location of an abnormal termination

   Shows information relating to the location of an abnormal termination.

   SIGSEGV (memory access referenced an illegal segment) is occurring close to the sysThreadAvailableStackWithSlack function of libjvm.so.

   In this example, the information reveals that the error is occurring inside the Java VM.

   Because the abnormal termination is not occurring inside a Java application, the stack trace information relating to the time the abnormal termination occurred should be examined.

   In this example, the illegal access occurred as an extension of com.appli.ap.business.AL02ABB00000.toString, so check if line 489 of AL02ABB00000.java contains anything likely to cause an illegal access error.

```
Unexpected Signal : SIGSEGV [0xb] occurred at PC=0xff092068, pid=27758, nid=1
Function name=sysThreadAvailableStackWithSlack
Library=/opt/FJSVawjbk/jdk6/jre/lib/sparc/fjvm/libjvm.so

Current Java thread:
0xfb8e2850 - 0xfb8e4b7c at
com.appli.ap.business.AL02ABB00000.toString(AL02ABB00000.java:489)
0xfb8e2850 - 0xfb8e4b7c at
com.appli.ap.business.AL02ABB00000.toString(AL02ABB00000.java:520)
                        at java.lang.String.valueOf(String.java:1942)
                        at java.lang.StringBuffer.append(StringBuffer.java:365)
- locked <f6db38d8> (a java.lang.StringBuffer)
                        at
com.appli.ap.business.AL02ABB25201.doExecute(AL02ABB25201.java:774)
                        at com.appli.ap.formula.AFCC6842.doDelegate(AFCC6842.java:221)
                        at com.appli.ap.formula.ejb.session.AFSF6801.doExecuteOrdinarily
    (AFSF6801.java:381)

                        at
```

```
com.appli.ap.formula.ejb.session.FJAFSF6801_AFSF6801RemoteImpl.doExecuteOrdinarily
    (FJAFSF6801_AFSF6801RemoteImpl.java:464)
      - locked <df672838> (a com.appli.ap.formula.ejb.session.FJAFSF6801_AFSF6801RemoteImpl)
     at com.appli.ap.formula.ejb.session._FJAFSF6801_AFSF6801RemoteImpl_Tie._invoke
     (_FJAFSF6801_AFSF6801RemoteImpl_Tie.java:76)
0xfb98c930 - 0xfb98cc68
     at com.fujitsu.ObjectDirector.CORBA.ServerRequest.call_invoke(ServerRequest.java:961)
     at com.fujitsu.ObjectDirector.PortableServer.POA.MsgRecv(POA.java:2578)
     at com.fujitsu.ObjectDirector.PortableServer.POAManager.MsgRecv(POAManager.java:1061)
     at com.fujitsu.ObjectDirector.PortableServer.POAnc.MsgRecv(POAnc.java:163)

Dynamic libraries:
0x10000 /opt/FJSVawjbk/jdk6/bin/java
0xff370000 /usr/lib/libthread.so.1
0xff3fa000 /usr/lib/libdl.so.1
--------------
(Omitted)
--------------
0xbef70000 /lib/libgen.so.1
0xbd6b0000 /lib/libextpiswu.so

Local Time = Wed May 30 19:48:01 2012
Elapsed Time = 9885
```

Note

If a line containing an Error ID is output, the value of the Error ID is an internal information code output when the Java VM detects an internal process conflict itself. In the case of errors such as SIGSEGV and SIGBUS that are detected by the operating system, the same value (4F530E435050****) will always be output. Therefore, if the Error ID begins with the code "4F530E435050", it normally has no meaning. If the Error ID begins with a code other than "4F530E435050", the code is a keyword that can be used to search for or identify the problem.

```
#
# HotSpot Virtual Machine Error : SIGSEGV (0xb)
#  [ pc=0xff092068, pid=27758(0x6c6e), nid=1(0x00000001), tid=0x00034d10 ]
#
# Please report this error to FUJITSU
#
# JRE version: 6.0-b04
# Java VM: Java HotSpot(TM) Server VM (**.*.**_FUJITSU_MODIFIED-B** mixed mode solaris-
sparc )--------------
(Omitted)
--------------
```

1. Signal handler information at the time of the abnormal termination Solaris32/64 Linux32/64

   Shows information relating to signal handlers at the time of the abnormal termination.

   In this example, all of the lines containing signal handler information include the text "(in VM)", so there are no problems associated with changing the signal handler registration.

```
##>> Signal Handlers
 VM signal handler[1]=0xfe1ec0a0, VM signal handler[2]=0xfe4ff780, SIG_DFL=0x00000000,
SIG_IGN=0x00000001, INT_SIG=(16,16), ASYNC_SIG=(17,17)
  SIGSEGV :signal handler=0xfe4ff780 (in VM *)
  SIGPIPE :signal handler=0xfe1ec0a0 (in VM)
  SIGBUS :signal handler=0xfe1ec0a0 (in VM *)
  SIGILL :signal handler=0xfe1ec0a0 (in VM)
  SIGFPE :signal handler=0xfe1ec0a0 (in VM)
  SIGXFSZ :signal handler=0xfe1ec0a0 (in VM)
  INTERRUPT_SIGNAL :signal handler=0xfe4ff010 (in VM +)
  ASYNC_SIGNAL :signal handler=0xfe1ec0a0 (in VM)
```

2. Information about the Java heap area at the time of the abnormal termination

   Shows information about the Java heap area at the time of the abnormal termination.

   JDK/JRE 6 or JDK/JRE 7 FJVM:

   - When Parallel GC is used:

```

[PSYoungGen]=[New generation area],
[PSOldGen]=[Old generation area],
[PSPermGen]=[Permanent generation area]

```

   - When Serial GC is used:

```

[def new generation]=[New generation area],
[tenured generation]=[Old generation area],
[compacting perm gen]=[Permanent generation area]

```

   - When Parallel GC with CMS is used:

```

[par new generation]=[New generation space],
[concurrent mark-sweep generation]=[Old generation space],
[concurrent-mark-sweep perm gen]=[Permanent generation space]

```

     Note that information about [object space] for [Old generation space] and [Permanent generation space] will not be output.

   In this example, one can see that "New generation area" + "Old generation area" (the area whose maximum size is specified by "-Xmx") still contains empty space.

   One can also see that space still remains in "Permanent generation area".

Notes

Values expressed as a percentage represent the ratio of memory that had been made usable for (committed to) the Java heap by the FJVM when the abnormal termination occurred. These values do not represent the proportion of the maximum usable value.

Do not refer to values expressed as a percentage. Instead, when making an assessment, compare the memory usage values displayed in "K" (kilo) units with the values specified by the options (including default values).

If Parallel GC is used, the values displayed after "-Xms=" and "-Xmx=" below are the actual values used by the Java VM after the initial and maximum values were recalculated and optimized for the Java VM based on system information such as the page size and the value specified by the -Xms/-Xmx option. As a result, the values may differ from the specified values.

```

##>> Heap
 PSYoungGen      total 3584K, used 277K [0xfa400000, 0xfa800000, 0xfbc00000)
  eden space 3072K, 9% used [0xfa400000,0xfa4457b0,0xfa700000)
  from space 512K, 0% used [0xfa780000,0xfa780000,0xfa800000)
  to   space 512K, 0% used [0xfa700000,0xfa700000,0xfa780000)
 PSOldGen        total 4096K, used 0K [0xf7800000, 0xf7c00000, 0xfa400000)
  object space 4096K, 0% used [0xf7800000,0xf7800000,0xf7c00000)
 PSPermGen       total 16384K, used 2520K [0xf3800000, 0xf4800000, 0xf7800000)
  object space 16384K, 15% used [0xf3800000,0xf3a762e0,0xf4800000)
(-Xms=8192K, -Xmx=65536K, -XX:PermSize=16384K, -XX:MaxPermSize=65536K)
```

6.5.9 Crash Dumps / Core Dumps

When the Java application terminates abnormally (the process dies), it might be possible to investigate the cause by collecting the crash and core dumps provided in each operating system.

6.5.9.1 Crash Dumps

Windows32/64

This section explains how to collect the crash dump to investigate abnormalities in Windows(R).

Dr. Watson

Dr. Watson is a Microsoft Corporation software product for debugging program errors.

When a program error occurs, Dr. Watson automatically outputs debugging information to a log file named "drwtsn32.log". The location to which the log file is output can be specified after running Dr. Watson.

For a detailed description of Dr. Watson, see the Microsoft Corporation website.

Dr. Watson Settings

Dr. Watson, which is bundled with the Windows(R) operating system, is used to collect a crash dump.

Use the following example to set up Dr. Watson. These settings will enable a crash dump to be output automatically when an abnormal termination occurs.

Sample Settings for Dr Watson (Windows Server(R) 2003, Windows(R) XP)

1. Enter the command "drwtsn32 -i" from the **Command Prompt** or the **Run** dialog box. A dialog box with the message "Dr. Watson has been installed as the default application debugger" should be displayed.

2. Run "drwtsn32" (without the "-i" option) from the **Command Prompt** or the **Run** dialog box. When the **Dr. Watson for Windows** dialog box appears, check the following settings:

    - Are the **Log File Path** and **Crash Dump** set correctly?

    - Is "Complete" selected as the "Crash dump type (Y)"?

    - Is the **Dump All Thread Contexts** check box selected?

    - Is the **Append To Existing Log File** check box selected?

    - Is the **Visual Notification** check box selected?

    - Is the **Create Crash Dump File** check box selected?

Windows Server(R) 2003 x64 Edition

To configure the environment settings for the 32-bit mode version of Dr Watson, you have to run the "%SystemRoot%\SysWow64\drwtsn32" command at the MS-DOS command prompt and then configure the settings as described above.

Windows Vista(R) SP1, Windows(R) 7, Windows Server(R) 2008 or Windows Server(R) 2008 R2

The Dr Watson feature is not supplied in Windows Vista(R), Windows(R) 7, Windows Server(R) 2008 or Windows Server(R) 2008 R2.

Use Windows Error Reporting (WER) instead of Dr Watson.

Refer to the example below to set up WER.

Sample WER Settings

1. Enter the "regedit" command at the MS-DOS command prompt to launch the Registry Editor.
2. Create the "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\Windows Error Reporting\LocalDumps" key.
3. Create a REG_DWORD-type value called DumpType for the LocalDumps key and set "2".

Refer also to the information below on how to configure the settings for WER.

http://msdn.microsoft.com/en-us/library/bb787181.aspx

Notes:

The first version of Windows Server(R) 2003 occasionally failed to output a user dump, and also had problems executing Java operations.

A description of some of these problems can be obtained from the following URLs:

- http://support.microsoft.com/kb/836080/en-us
- http://support.microsoft.com/kb/837018/en-us
- http://support.microsoft.com/kb/841176/en-us

If using Windows Server(R) 2003, apply Service Pack 1(later) or the relevant Hotfix.

6.5.9.2 Core Dumps (Solaris)

Solaris32/64

This section explains the notes on collecting core dumps in Solaris.

Confirming that a core dump was not output

A possible cause of the core dump not being output is that there is a problem with the system resources. Make sure that there are write privileges for the current directory, the disk space, and the limit(1) command result.

6.5.9.3 Core Dumps (Linux)

Linux32/64

This section explains the notes on collecting core dumps in Linux.

Confirming that a core dump was not output

A possible cause of the core dump not being output is that there is a problem with the system resources. Make sure that there are write privileges for the current directory, the disk space, and the limit(1) command result.

In Linux, output of the core dump also might not be set by default when the hardware/OS is shipped or after OS Update was applied. Execute as follows so that the core dump is output.

- **Starting Interstage using the *isstart* command**

  Execute the "*ulimit -c unlimited*" command using sh (bash), and then start Interstage. If the WorkUnit and Interstage start users are different, then before starting the WorkUnit execute the "*ulimit -c unlimited*" command and then start the WorkUnit.

- **Configuring the settings in the RC procedure so that Interstage starts up automatically when the OS starts up**

  By editing the file shown below, the core will be output after the OS is restarted.

  The */etc/init.d/functions* file contains the following description:

```
# make sure it doesn't core dump anywhere; while this could mask
# problems with the daemon, it also closes some security problems
ulimit -S -c 0 >/dev/null 2>&1
```

  or

```
ulimit -S -c ${DAEMON_COREFILE_LIMIT:-0} >/dev/null 2>1
```

In the above setting, change "0" to "unlimited".

```
ulimit -S -c unlimited >/dev/null 2>&1
```

Add the description that follows <--- to */etc/rc2.d/S99startis*.

```
#!/bin/sh# Interstage Application Server
# S99starttis : Interstage Application Server start procedure

OD_HOME=/opt/FJSVod
export OD_HOME

ulimit -c unlimited   <---

/opt/FJSVod/bin/odalive > /dev/null
while [ "$?" != "0" ]
do
    sleep 1
    /opt/FJSVod/bin/odalive > /dev/null
done

/opt/FJSVtd/bin/isstart
```

6.5.10 Outputting JNI Process Error Messages

The Java Native Interface (JNI) is used to link to languages other than Java.

However, if the JNI is used incorrectly, it can cause a Java process to stop running (terminate abnormally).

The option shown in the following figure causes a message to be output when an error occurs in a JNI process. This can be used to check JNI parameters and other information.

Option to Output a Message when a JNI Process Malfunctions

```
-Xcheck:jni
```

When the "-Xcheck:jni" parameter is specified, the message shown in the following figure may be output.

Message Output when an Error Occurs in a JNI Process

```
FATAL ERROR in native method: (detailed message)
```

Some examples of the string contained in the "(detailed message)" section of the above figure are provided below, along with related notes. Use this explanation when reviewing the JNI processing component.

6.5.10.1 "FATAL ERROR in Native method" Message

Explanation of Messages

JNI received a class argument that is not a class

[Error example]

```
char buf[1];
(*env)->AllocObject(env, (jclass)buf); //A different type is specified for
 the second argument of the jclass type
```

JNI received a null class

[Error example]

```
 (*env)->AllocObject(env, NULL); //Null is specified as the second argument
 of the jclass type
```

JNI string operation received a non-string

[Error example]

```
 (*env)->GetStringUTFChars(env, NULL, 0); //Null is specified as the second
 argument of the jstring type
```

Non-array passed to JNI array operations

[Error example]

```
 (*env)->GetArrayLength(env, (jarray)(*env)->NewStringUTF(env, "abc")); //
A non-array type is specified as the second argument of the jarray type
```

Note that in the following example, no message is output with the "-Xcheck:jni" option.

```
char buf[1];
(*env)->GetArrayLength(env, (jarray)buf);
```

Static field ID passed to JNI

[Error example]

```
jclass cls = (*env)->GetObjectClass(env, obj);
jfieldID fid = (*env)->GetFieldID(env, cls, "static_data", "I");
(*env)->GetIntField(env, obj, fid); //A static field is specified
as the third argument of the jfieldID type
```

Null object passed to JNI

[Error example]

```
jclass cls = (*env)->GetObjectClass(env, obj);
jfieldID fid = (*env)->GetFieldID(env, cls, "instance_data", "I");
(*env)->GetIntField(env, NULL, fid); //Null is specified as the
second argument of the object type
```

Note that this message is output only when checking if a variable is an instance variable.

In the following example, no message is output with the "-Xcheck:jni" option.

```
 (*env)->GetObjectClass(env, NULL); //Null is specified as the second
 argument of the object type
```

Wrong field ID passed to JNI

[Error example]

```
 (*env)->GetIntField(env, obj, -1); //An numerical value is specified
as the third argument of the jfieldID type
```

Note that this message is output only when checking if a variable is an instance variable.

Non-static field ID passed to JNI

[Error example]

```
jclass cls = (*env)->GetObjectClass(env, obj);
(*env)->GetStaticIntField(env, cls, -1); //
A numerical value is specified as the second argument of the jfieldID type
```

Note that in the following example, no message is output with the "-Xcheck:jni" option.

```
jclass cls = (*env)->GetObjectClass(env, obj);
jfieldID fid = (*env)->GetStaticFieldID(env, cls, "instance_data", "I");
(*env)->GetStaticIntField(env, cls, fid);
```

Array element type mismatch in JNI

[Error example]

```
jintArray intarray = (*env)->NewIntArray(env, 2);
(*env)->GetFloatArrayElements(env, intarray, 0); //jintArray is specified as
 the second argument of the floatArray type
```

Object array expected but not received for JNI array operation

[Error example]

```
jclass cls = (*env)->GetObjectClass(env, obj);
jobjectArray objarray = (*env)->NewObjectArray(env, 1, cls, obj);
(*env)->GetIntArrayElements(env, objarray, 0); //jobjectArray is specified
as the second argument of the intArray type
```

Field type (static) mismatch in JNI get/set field operations

[Error example]

```
jclass cls = (*env)->GetObjectClass(env, obj);
jfieldID fid = (*env)->GetStaticFieldID(env, cls, "static_data", "I");
(*env)->GetStaticFloatField(env, cls, fid); //Must be GetStaticIntField,
not GetStaticFloatField
```

Field type (instance) mismatch in JNI get/set field operations

[Error example]

```
jclass cls = (*env)->GetObjectClass(env, obj);
jfieldID fid = (*env)->GetFieldID(env, cls, "instance_data", "I");
(*env)->GetFloatField(env, obj, fid); //Must be GetIntField, not GetFloatField
```

Wrong static field ID passed to JNI

[Error example]

```
jclass cls = (*env)->GetObjectClass(env, obj);

jclass cls2 = (*env)->GetObjectClass(env, (*env)->NewStringUTF(env, "abc"));
jfieldID fid = (*env)->GetStaticFieldID(env, cls, "static_data", "I");
(*env)->GetStaticObjectField(env, cls2, fid); //The second argument must be cls, not cls2
```

Using JNIEnv in the wrong thread

[Explanation]

This error occurred because JNIEnv (which is not intended for the running thread) was used.

The Java VM may allocate the area that the JNI interface pointer (JNIEnv) references for use as a thread-specific data area. This means that the JNI interface pointer is only valid for the current thread. A native method is not permitted to pass the JNI interface pointer to another thread.

JNI call made with exception pending

[Description]

This is an error that occurs because an exception of some kind occurred in a native program and the JNI function was then run without that exception being handled first.

After the JNI function is called by the native program, use ExceptionOccurred at that point to check the circumstances in which the exception occurred, and then either clear the exception or throw the exception up to a higher level method as required.

6.6 Identifying the Causes of Errors

This section explains how to identify the causes of errors when they occur.

6.6.1 When java.lang.OutOfMemoryError is Thrown

This section explains some of the likely causes of an OutOfMemoryError, and the required action.

Please refer to "6.6.1.6 Enhanced Message Output Function for Insufficient Memory Size Events" to learn more about the message information that is output when an OutOfMemoryError is thrown when using FJVM.

6.6.1.1 Memory Leak

Obtain the garbage collection log and check the Java heap consumption. If memory consumption increases with time even after repeated garbage collection operations, there might be a memory leak.

As a result of a memory leak, a Java heap shortage may occur, throwing an OutOfMemoryError.

In this case, obtain the garbage collection log and check the Java heap consumption.

The method used to obtain the garbage collection log is explained in "6.2.6 Outputting a Garbage Collection Log".

6.6.1.2 Insufficient Java Heap

An OutOfMemoryError is usually due to an insufficient Java heap.

Obtain the garbage collection log and check the Java heap consumption.

Adjust the size of the Java heap if the check reveals that there is not enough remaining free space.

The method used to obtain the garbage collection log is explained in "6.2.6 Outputting a Garbage Collection Log".

Java heap tuning is explained in "6.4.1 Tuning the Java Heap".

6.6.1.3 Insufficient User Space

If a large quantity of threads are created, and a large quantity of stack is allocated to user space, and a user space shortage subsequently occurs, the following OutOfMemoryError will be thrown, or an error message will display and the process will terminate.

```
Java.lang.OutOfMemoryError: unable to create new native thread
```

In addition, regardless of any surplus in the Java heap or OS virtual memory, if memory cannot be secured within the user space, the following OutOfMemoryError will be output and the program will terminate.

```
Java.lang.OutOfMemoryError: requested SIZE bytes NAME. Out of swap space?
```

SIZE	Size of memory which could not be secured
NAME	Control name of the Java VM in which memory could not be secured (only displayed when this information is available)

If a shortage of user space occurs, make the appropriate adjustments, such as reducing the size of the Java heap or the stack.

Java heap tuning is explained in "6.4.1 Tuning the Java Heap".

Stack size adjustment is explained in "6.4.2 Tuning the Stack".

If there is no shortage of virtual memory, another way of addressing this problem is to run multiple Java processes to increase the process concurrency. Adjust J2EE (Java EE) if J2EE(Java EE) applications are being used. J2EE(Java EE) adjustment is explained in J2EE Tuning (or Java EE Tuning).

6.6.1.4 Insufficient Virtual Memory

If a shortage of virtual memory prevents threads from being created, the following information will also be added to the OutOfMemoryError, or an error message will display and the process will terminate.

```
Java.lang.OutOfMemoryError: unable to create new native thread
```

In addition, if a shortage of OS virtual memory occurs, the following OutOfMemoryError will be output and the program will terminate.

```
Java.lang.OutOfMemoryError: requested SIZE bytes NAME. Out of swap space?
```

SIZE	Size of memory which could not be secured
NAME	Control name of the Java VM in which memory could not be secured (only displayed when this information is available)

If a shortage of virtual memory occurs, close any processes that are no longer required or increase the amount of virtual memory by adding more physical memory (RAM) or enlarging the size of the swap file.

6.6.1.5 Probable Causes (Disabled Garbage Collection Processing)

If an **OutOfMemoryError** is thrown in the critical section state due to disabled Garbage Collection (GC) processing, review the content of the application to be run as required to minimize the impact of the disabling of GC processing. (Review the use of those functions within the application processing that are related to GC processing.)

Refer to "6.2.1 Garbage Collection Processing Supported in the FJVM " for information on disabling GC processing.

It may also be possible to use Java heap tuning to lessen the occurrences of **OutOfMemoryError** due to disabled GC processing.

- Where the application does not issue a Java object generation request using JNI from a native program while in the critical section state, it may be possible to lessen the occurrences of **OutOfMemoryError** due to disabled GC processing by tuning the size of the Old generation area (increasing the overall size of the memory allocation pool).
- Where the application does issue a Java object generation request using JNI from a native program while in the critical section state, it may be possible to lessen the occurrences of **OutOfMemoryError** due to disabled GC processing by tuning the size of the New generation area.

Refer to "6.2.6 Outputting a Garbage Collection Log" for information on how to collect Garbage Collection logs.

Refer to "6.4.1 Tuning the Java Heap" for information on how to tune the Java heap.

Note

To determine whether an **OutOfMemoryError** has been thrown in the critical section state due to disabled GC processing, refer to the messages output by the Enhanced Message Output Function for Insufficient Memory Size Events.

If the EXTP4435 or ISJEE_OM1018 message is output, refer to the "Insufficient Java VM heap area detail information" that is output to the IJServer container information log (info.log) and the IJServer cluster Java VM log (jvm.log) to make the determination.

6.6.1.6 Enhanced Message Output Function for Insufficient Memory Size Events

In the **FJVM**, the message information output when an insufficient memory size event occurs is enhanced.

This means that when an insufficient memory size event occurs, the **FJVM** outputs type information for the area where the event occurred in the format shown in the following figure as well as the java.lang.OutOfMemoryError exception message information.

Type information output for the area where an insufficient memory size event occurred

```
The memory was exhausted area_name
  Java heap size / max Java heap size = heap_size / max_heap_size
  Java perm size / max Java perm size = perm_size / max_perm_size
```

Element	Description
area_name	This displays the name of the area where the insufficient memory size event occurred and the size of the object request (insufficient area information) that caused the insufficient memory event. Insufficient size information items are as follows: - on Java heap space. : requested <NNNN> bytes Where an insufficient memory size event occurred for the memory allocation pool (New generation space or Old generation space) in an object generation request for NNNN bytes. Note that this item may apply when the insufficient memory size event discovery feature is enabled in the ergonomics function in JDK/JRE 6 or 7, and an insufficient memory size event was detected using this discovery feature. - on Java heap space. : requested <NNNN> bytes (in critical section) This means the same thing as "on Java heap space. : requested <NNNN> bytes", but indicates that the execution of GC processing in the critical section state was disabled when the insufficient memory size event occurred. - on Java perm space. : requested <NNNN> bytes

Element	Description
	Where an insufficient memory size event occurred for the Permanent generation area in an object generation request for NNNN bytes. Note that this item may apply when the insufficient memory size event discovery feature is enabled in the ergonomics function in JDK/JRE 6 or 7, and an insufficient memory size event was detected using this discovery feature. - on Java perm space. : requested <NNNN> bytes (in critical section) This means the same thing as **"on Java perm space. : requested <NNNN> bytes"**, but indicates that the execution of GC processing in the critical section state was disabled when the insufficient memory size event occurred. - (None) Where an insufficient memory size event occurred for an area other than the Java heap, such as a stack or heap. Particularly if the java.lang.OutOfMemoryError exception information is output in a format where there is "insufficient user space" or "insufficient virtual memory" when a java.lang.OutOfMemoryError was thrown, it can be concluded that an insufficient memory size event occurred for an area other than the Java heap, such as a stack or heap. Alternatively, at the evaluation stage of the array generation formula during Java application execution, the result of the evaluation based on the length of the array object (the number of array elements), was that there was insufficient space to allocate the array object. (The length of the array (the number of array elements) was too great, so there will be an array definition with an array object size of 2 gigabytes or more.) Another possibility is that an insufficient memory event occurred in class loading processing. Note that this item may apply when the insufficient memory size event discovery feature was enabled in the ergonomics function in JDK/JRE 6 and 7, and an insufficient memory size event was detected by this discovery feature.
heap_size	This is the size (in bytes) of the memory allocation pool being used when the insufficient memory size event occurred.
max_heap_size	This is the maximum size (in bytes) that can be used for the memory allocation pool.
perm_size	This is the size (in bytes) for the Permanent generation area being used when the insufficient memory size event occurred.
max_perm_size	This is the maximum size (in bytes) that can be used for the Permanent generation space.

Notes

- The sizes of the areas being used when the insufficient memory size event was output (heap_size and perm_size) do not include the size of the object that caused the insufficient memory size event.

  Consequently, if the insufficient memory size event was caused by a request to generate a particularly massive object, it is important to note that there may be a large difference between the maximum size and the size being used (giving the impression that there is a large amount of free space).

- NewGC processing subdivides the New generation area into three internal spaces: "eden space", "from space" and "to space". In each space, a control method generally referred to as "GC control by generation" is used to manage and control objects generated at the request of Java applications.

  Of those areas, "from space" and "to space" act in the role of work areas when Java VM conducts NewGC processing. As a result, the area size used for object generation requests from Java applications makes up only a part of the total size occupied by each of the "from space" and "to space" areas.

  Consequently, note that it may appear from the output data that there is free space available in the memory allocation pool and New generation area, but there may in fact be no free space available. (Even when there appears to be space available, the discrepancy arises because that space may already be in use as work area for NewGC processing.)

- The heap_size value output when an insufficient memory size event occurs for the memory allocation pool is the total of the amounts of New generation area and Old generation area being used.

  Because the New and Old generation areas are managed and controlled as separate object storage areas, note that the size of the difference between max_heap_size and heap_size is not the same as the maximum size for objects that can be generated by request.

Sample message output when an insufficient memory size event has occurred

```
Exception in thread "main" java.lang.OutOfMemoryError: Java heap space
The memory was exhausted on Java heap space. : requested 4016 bytes
  Java heap size / max Java heap size = 495974032 / 536870912
  Java perm size / max Java perm size = 1678376 / 67108864
```

Given the sample output in the above figure, we can confirm that an insufficient memory size event occurred for the memory allocation pool in a request to generate a 4016-byte object.

6.6.2 When an EXTP4435 or ISJEE_OM1018 message is output

If the messages below are output when the Java application is executed in Interstage Application Server, it means an abnormality was caused by the occurrence of an insufficient memory size event, and therefore the Java heap must be tuned (for details, refer to "6.4.1 Tuning the Java Heap").

- EXTP4435 Messages
- ISJEE_OM1018 Messages

Note

For details on the content of each message, refer to the message information in "Messages".

Note that the format of the "insufficient Java VM heap area detail information" message output to the IJServer container information log (info.log) and the IJServer cluster Java VM log (jvm.log) will be as shown in the sample below.

Output format for "insufficient Java VM heap area detail information" message

```
----------------------------------------------------------------------------
OutOfMemory Log
----------------------------------------------------------------------------
pid=process_id
heap_type=heap_type_code
heap_size=heap_size
max_heap_size=max_heap_size
perm_size=perm_size
max_perm_size=max_perm_size
requested_size=requested_size
----------------------------------------------------------------------------
VM is terminated by occurred OutOfMemoryError on heap_type.
stack_trace
```

The table below describes the element in the output:

Element	Description
process_id	ID of the Java process in which the event occurred.
heap_type_code	Code of the space in which the event occurred. For details, refer to element heap_type in this table.
heap_size	Size (in bytes) of the memory allocation pool that was being used when the event occurred.
max_heap_size	Maximum size (in bytes) that can be used for the memory allocation pool.

Element	Description
perm_size	Size (in bytes) of the permanent generation space that was being used when the event occurred.
max_perm_size	Maximum size (in bytes) that can be used for the permanent generation space.
requested_size	Requested size (in bytes) for the object that caused the area insufficiency.
heap_type	Additional information about the heap, which can be one of the below: - Java heap Indicates that event has occurred for the memory allocation pool (New generation space or Old generation space) in the object generation request (the heap_type_code for this item is 1). Note that this item might apply when the event monitoring feature is enabled in the ergonomics feature in JDK/JRE 6 or 7 and the event was discovered by it. - Java heap in critical section This means the same as "Java heap", but it indicates that the GC processing in the critical section state was disabled when the event occurred. - Java Perm Indicates that the event has occurred for the Permanent generation space in the object generation request (the heap_type_code for this item is 2). Note that this item might apply when the event monitoring feature is enabled in the ergonomics feature in JDK/JRE 6 or 7, and the event was discovered by it. - Java Perm space in critical section This means the same as "Java Perm", but it indicates that the execution of GC processing in the critical section state was disabled when the event occurred. - C heap control information Indicates that the event has occurred for space other than the Java heap, such as for example, the stack or heap (the heap_type_code for this item is 0). Additionally, control information for a Java VM for which memory could not be reserved might be output in the subsequent line. - unknown space The evaluation stage of the array generation formula when the Java application was executed concluded that due to the length of the array object (number of array elements) there was insufficient space to allocate it. As a result, there will be an array definition that will have an array object size of 2 gigabytes or more. Alternatively, the event might have occurred in the class loading processing (the heap_type_code for this item is -1). Note that this item might apply when the event monitoring feature is enabled in the ergonomics feature in JDK/JRE 6 and 7, and the event was discovered by it. Note that **requested_ size** for this item is "0".
stack_trace	Report of the stack when the event occurred. If the thread in which the event occurred was the thread that executed the Java application, then a stack trace for the thread will be output. If the thread was not the thread that executed the Java application, or the stack information cannot be obtained due to insufficient resources, then a stack trace will not be output.

Notes

- The sizes of the areas being used when the insufficient memory size event was output (heap_size and perm_size) do not include the size of the object that caused the insufficient memory size event.

  Consequently, if the insufficient memory size event was caused by a request to generate a particularly massive object, it is important to note that there may be a large difference between the maximum size and the size being used (giving the impression that there is a large amount of free space).

- NewGC processing subdivides the New generation area into three internal spaces: "eden space", "from space" and "to space". In each space, a control method generally referred to as "GC control by generation" is used to manage and control objects generated at the request of Java applications.

  Of those areas, "from space" and "to space" act in the role of work areas when Java VM conducts NewGC processing. As a result, the area size used for object generation requests from Java applications makes up only a part of the total size occupied by each of the "from space" and "to space" areas.

  Consequently, note that it may appear from the output data that there is free space available in the memory allocation pool and New generation area, but there may in fact be no free space available. (Even when there appears to be space available, the discrepancy arises because that space may already be in use as work area for NewGC processing.)

- The heap_size value output when an insufficient memory size event occurs for the memory allocation pool is the total of the amounts of New generation area and Old generation area being used.

  Because the New and Old generation areas are managed and controlled as separate object storage areas, note that the size of the difference between max_heap_size and heap_size is not the same as the maximum size for objects that can be generated by request.

Output example

```
----------------------------------------------------------------------------
OutOfMemory Log
----------------------------------------------------------------------------
pid=4696
heap_type=1
heap_size=136800
max_heap_size=6291456
perm_size=2052320
max_perm_size=67108864
requested_size=40000016
----------------------------------------------------------------------------
VM is terminated by occurred OutOfMemoryError on Java heap.
"main" prio=6 tid=0x00307000 nid=0x12a8 runnable [0x0092f000]
   java.lang.Thread.State: RUNNABLE
          at test.<init>(test.java:10)
          at test.main(test.java:5)
```

Note

Information that shows the thread status in the stack trace information (the line before the trace information) is now displayed from JDK/JRE 6. In the example above, "java.lang.Thread.State: RUNNABLE" is displayed, but any of the following might also be displayed instead of the "RUNNABLE" part: "NEW", "TIMED_WAITING (sleeping)", "WAITING (on object monitor)", "TIMED_WAITING (on object monitor)", "WAITING (parking)", "TIMED_WAITING (parking)", "BLOCKED (on object monitor)", "TERMINATED" or "UNKNOWN".

6.6.3 When java.lang.StackOverflowError is Thrown

When a StackOverflowError is thrown, a stack overflow is the cause.

Adjust the size of the stack.

The method used to adjust the stack is explained in "6.4.2 Tuning the Stack".

Note that in some cases Java processes may simply terminate abnormally without throwing a **StackOverflowError.** Refer to "Stack Overflow Message Output Function" for information on how to analyze such events.

6.6.3.1 Stack Overflow Message Output Function

A Java process might terminate abnormally when it performs an invalid memory access if the size of the stack for the thread is insufficient (that is, if the action causes a stack overflow).

If an invalid memory access causes a Java process to terminate abnormally because of a stack overflow, the FJVM outputs information indicating the cause of the problem to the FJVM log. This function is called the **stack overflow message output function**.

Refer to "6.5.8 FJVM Log" for instructions on viewing the FJVM log.

If a stack overflow is due to the size of the stack corresponding to a thread created in a Java API, refer to "6.4.2 Tuning the Stack" and adjust the size of the relevant stack.

Detected Threads

The threads that are subject to stack overflow detection are threads created by a Java API.

The following threads are not targeted by this function:

- Threads created by directly using an operating system API from a native module
- Threads that execute the main method (because they are not created by a Java API)

However, in the case of the Windows(R) versions of the FJVM, even the above threads are targeted by this function if stack overflows occur directly in a native module invoked from inside the Java method executed by the relevant thread.

Notes Windows32/64

If a stack overflow occurs, control is sometimes not passed from the operating system to the FJVM process. In such cases the FJVM log will not be output.

If the control process of the operating system passes exception control directly to Dr. Watson, check the log file output by Dr. Watson. Refer to "6.5.9.1 Crash Dumps" for a description of Dr. Watson.

If there is little stack remaining when a stack overflow occurs, the Java application might close in the following cases:

- The message shown below is output in standard output, but the Java application might close without the FJVM log being output.
- The following message indicates that a stack overflow has occurred even if the FJVM log used to detect stack overflows is not output.

```
An unrecoverable stack overflow has occurred.
```

If stack overflow exception control is not passed to the FJVM process or Dr. Watson, the Java application may simply terminate.

In such cases, it is not possible to detect the occurrence of a stack overflow.

6.6.4 Abnormal Termination Following the Occurrence of SIGBUS

Solaris32/64

If a process terminates abnormally in Solaris following the occurrence of SIGBUS in the status shown below, it means that the abnormal termination occurred because of insufficient system memory resources/swap.

```
signal no   : 10(SIGBUS)
signal code :  3(BUS_OBJERR)
signal error: 12(ENOMEM)
```

In this case, the following information is output to the FJVM log output when the Java process terminates abnormally. (This information varies, depending on the JDK/JRE version).

```
siginfo:si_signo=SIGBUS: si_errno=Not enough space, si_code=3(BUS_OBJERR), si_addr=hexadecimal
```

or

```
siginfo:si_signo=10, si_errno=12, si_code=3, si_addr=hexadecimal
```

If this information is output, either terminate unnecessary processes so that there is sufficient virtual memory, or perform tuning to increase the virtual memory by extending the physical memory (RAM) or swap file.

6.6.5 When a Process is Extinguished (Terminates Abnormally)

This section explains why a process can disappear without leaving a trace or an output, and how to resolve this problem.

6.6.5.1 Likely Cause: Stack Overflow

The FJVM is equipped with a function to output a message when a stack overflow is detected. By analyzing the FJVM log, it is possible to determine if a stack overflow has occurred. Refer to "6.6.3.1 Stack Overflow Message Output Function" for the methods used to analyze the FJVM log.

If it has been confirmed that a stack overflow occurred, adjust the size of the relevant stack. Refer to "6.4.2 Tuning the Stack" for information on how to adjust the stack.

Windows32/64

Normally, when a stack overflow occurs, a java.lang.StackOverflowError is thrown. This is detected by Dr. Watson, which outputs a user dump and a Dr. Watson log.

However, if the operating system is under high load, or there is not much stack available when the stack overflow occurs, a process may disappear without leaving a trace or an output, and without control passing from the operating system to either the FJVM or Dr. Watson.

If the reason for the disappearance of a process cannot be determined, check if the problem can be avoided by increasing the size of the stack. If the problem continues even after the stack size has been increased, investigate other possible causes.

Refer to "6.5.9.1 Crash Dumps" for a description of Dr. Watson.

6.6.5.2 Likely Cause: Termination by the Long Time Compilation Detection Function

Termination may have been brought about by the **Long time compilation detection function** of the FJVM.

Refer to "6.3.2 Long Time Compilation Detection Function" for details.

If a Java application was started with the "-XX:CompileTimeout" option, check if any messages have been sent to the standard output by the FJVM.

6.6.5.3 Likely Cause: Signal Handler

Solaris32/64 Linux32/64

If a signal handler is registered with a module that is not a Java VM, the Java application may malfunction and terminate abnormally. Refer to "6.5.8.2 Signal Handler Information at the Time of Abnormal Termination" for details.

If a FJVM is used, signal handler information will be output to the FJVM log, so check that log for the relevant information.

6.6.5.4 Likely Cause: JNI Process Error

When using JNI to link to native modules developed in languages other than Java, a process may disappear if the JNI is used incorrectly.

In such cases, specify the "-Xcheck:jni" option to check if any messages are output during JNI processing. The "-Xcheck:jni" option is explained in detail in "6.5.10 Outputting JNI Process Error Messages".

Even if the JNI process is correct, a Java application process may disappear if a native module terminates abnormally or stops responding. Care must be taken when using a thread-unsafe function.

Example of a Thread-Unsafe Function

Solaris32/64

There have been cases of faults occurring when the following function is used:

- vfork

6.6.5.5 Likely Cause: Termination by Program

If a Java process terminates unexpectedly without outputting any special messages, one of the following events may have been executed at an unexpected location:

- java.lang.Runtime.exit()
- java.lang.Runtime.halt()
- java.lang.System.exit()

If the FJVM is used, handle the problem after referring to "6.5.6 Java VM Termination Status Output Function".

6.6.5.6 Likely Cause: Windows Server(R) 2003 Problem

Windows32/64

The first version of Windows Server(R) 2003 occasionally failed to output a user dump, and also had problems executing Java operations.

A description of some of these problems can be obtained from the following URLs:

- http://support.microsoft.com/kb/836080/en-us
- http://support.microsoft.com/kb/837018/en-us
- http://support.microsoft.com/kb/841176/en-us

If using Windows Server(R) 2003, apply Service Pack 1 or the relevant Hotfix.

6.6.6 Hangups (Freezing)

This section explains why programs may stop responding (hang or freeze) even when Java processes still remain. It also explains what to do when such problems occur.

6.6.6.1 Likely Cause: Deadlock

A deadlock can cause threads to stop running.

If a program stops responding, collect a thread dump to see if a deadlock has occurred.

Refer to "6.5.3 Thread Dumps" for a detailed explanation of how to collect and analyze a thread dump.

6.6.6.2 Likely Cause: Garbage Collection

When a garbage collection occurs, all of the Java application threads stop running for the duration of the garbage collection operation. This can make it appear as though a Java application has stopped responding.

Obtain the garbage collection log and check when garbage collection took place. If garbage collections are causing Java applications to appear as if they have stopped responding, adjust the Java heap to improve garbage collection operations.

The method used to obtain the garbage collection log is explained in "6.2.6 Outputting a Garbage Collection Log".

Java heap adjustment is explained in "6.4.1 Tuning the Java Heap".

6.6.6.3 Likely Cause: JNI Process Error

When using JNI to link to native modules developed in languages other than Java, a program freeze can occur if the JNI is used incorrectly.

In such cases, specify the "-Xcheck:jni" option to check if any messages are output during JNI processing. The "-Xcheck:jni" option is explained in detail in "6.5.10 Outputting JNI Process Error Messages".

Even if the JNI process is correct, a Java application can stop responding if a JNI module terminates abnormally or freezes. Care must be taken when using a thread-unsafe function, for example.

Example of a Thread-Unsafe Function

Solaris32/64

Programs have been known to stop responding when the following function is used:

- vfork

6.6.7 Slowdowns

This section explains the possible causes of Java application slowdowns, and how to resolve them.

6.6.7.1 Likely Cause: Garbage Collection

When a garbage collection occurs, all of the threads of a Java application stop running for the duration of the garbage collection operation. This can slow down the response of Java applications.

Obtain the garbage collection log and compare the timing of garbage collection with the timing of slowdowns. If garbage collections are causing slowdowns, adjust the Java heap to improve garbage collection operations.

The method used to obtain the garbage collection log is explained in "6.2.6 Outputting a Garbage Collection Log".

Java heap adjustment is explained in "6.4.1 Tuning the Java Heap".

Example of Slowdown

There have been cases in which multiple Web servers run the same software and Java applications, but only some experience slowdowns. The discrepancy was due to differences in the amount of physical memory (RAM) installed in each machine.

If a machine does not have enough physical memory (RAM) installed, the contents of memory will be swapped to disk each time garbage collection occurs, which can cause slowdowns.

6.7 Java Tools

This product provides the following tools for tuning and troubleshooting Java programs:

- The method trace function for outputting method traces
- jheap for outputting Java heap usage
- The thread dump tool for outputting thread dumps (Windows(R) only)
- Java Management Agent (JDK only)
- Tools useful for troubleshooting that are contained in JDK

Tool storage destinations

- The method trace function for outputting method traces
- jheap for outputting Java heap usage
- The thread dump tool for outputting thread dumps (Windows(R) only)

  Windows32/64

  - When JDK 6 is used: <Interstage installation folder>\jdk6\tools
  - When JRE 6 is used: <Interstage installation folder>\jre6\tools
  - When JDK 7 is used: <Interstage installation folder>\jdk7\tools
  - When JRE 7 is used: <Interstage installation folder>\jre7\tools

  Solaris32/64 Linux32/64

  - When JDK 6 is used: /$DIR/FJSVawjbk/jdk6/tools
  - When JRE 6 is used: /$DIR/FJSVawjbk/jre6/tools
  - When JDK 7 is used: /$DIR/FJSVawjbk/jdk7/tools
  - When JRE 7 is used: /$DIR/FJSVawjbk/jre7/tools

  '$DIR' is a relative directory specified during installation. The system recommended name for '$DIR' is 'opt'.

- Java Management Agent (JDK only)

  Windows32/64

    - When JDK 6 is used: <Interstage installation folder>\jdk6\tools

    - When JDK 7 is used: <Interstage installation folder>\jdk7\tools

  Solaris32/64 Linux32/64

    - When JDK 6 is used: /$DIR/FJSVawjbk/jdk6/tools

    - When JDK 7 is used: /$DIR/FJSVawjbk/jdk7/tools

  '$DIR' is a relative directory specified during installation. The system recommended name for '$DIR' is 'opt'.

- Tools useful for troubleshooting that are contained in JDK

  Windows32/64

    - When JDK 6 is used: <Interstage installation folder>\jdk6\bin

    - When JDK 7 is used: <Interstage installation folder>\jdk7\bin

  Solaris32/64 Linux32/64

    - When JDK 6 is used: /$DIR/FJSVawjbk/jdk6/bin

    - When JDK 7 is used: /$DIR/FJSVawjbk/jdk7/bin

  '$DIR' is a relative directory specified during installation. The system recommended name for '$DIR' is 'opt'.

Appendix A CORBA Service Environment Definition

This appendix describes the CORBA Service operating environment.

Each file is stored as follows:

Storage directory

Windows32/64

(Default installation path)

```
C:\Interstage\ODWIN\etc
```

Solaris32/64

(Default installation path. The directory specified as the operating environment file (fixed configuration install directory).)

```
/etc/opt/FSUNod
```

Linux32/64

```
/etc/opt/FJSVod
```

Files (provided with the Enterprise Edition)

config

gwconfig

inithost/initial_hosts

queue_policy (*1)

nsconfig

irconfig

Files (provided with the Standard-J Edition products)

config

inithost/initial_hosts

queue_policy (*1)

nsconfig

irconfig

*1 This is not valid for Linux (64 bit).

Note that files other than the above-mentioned files cannot be customized as the CORBA Service environment definition. Do not edit files other than those listed above using an editor or similar.

Notes

The Environment definition comes into context when there is an abnormal termination of the CORBA Service. The CORBA Service may not start normally if a resource such as a movement environment file is destroyed.

When resources are destroyed, there is a possibility that the CORBA Service will not start normally or that one of the following messages is displayed:

Message numbers: od10400, od10402, od10404, od10406, od10504, od10509, od10510

To restore the destroyed resources, restart the CORBA Service.

After an emergency reconstruction of the application environment, it is advisable to backup the resources.

For the procedure of creating a backup copy, refer to the "Maintenance (Resource Backup)" chapter in the Operator's Guide.

A.1 config

Overview

The config file contains definitions relating to CORBA Service operating environments.

File Name

Windows32/64

(Default installation path)

```
C:\Interstage\ODWIN\etc\config
```

Solaris32/64

(Default installation path)

```
/etc/opt/FSUNod/config
(Windows® Client: C:\Interstage\ODWIN\etc\config)
```

Linux32/64

```
/etc/opt/FJSVod/config
(Windows® Client: C:\Interstage\ODWIN\etc\config)
```

File Contents

Format

In the config file, values are specified in the following format.

Parameter name = value set

Lines beginning with a hash sign (#) are regarded as comment lines. Blank lines are ignored for analysis.

# comment

Example:

# comment

period_receive_timeout = 72

Parameters

Parameter values can be modified for the following operating environments:

- Operating environment relating to host information
- Operating environment relating to network environment
- Operating environment of application resources
- Operating environment of timeout monitoring

- Operating environment of security function
- Operating environment of maintenance function

Notes

- The CORBA Services use one connection for each machine on which the server application is running.
- The parameters for which values can be changed are shown in the following table.
- A parameter with "Required parameter" indicated in the "Remarks" column cannot be omitted. (The Solaris and Linux versions do not have required parameters.)
- If a non-numerical string (such as "abc") is specified for a numerical parameter (such as "period_receive_timeout"), it is the same as setting "0".

Table A.1 Operating Environment Relating to Host Information

Parameter	Initial value Default value Specifiable value	Meaning	Remarks
IIOP_hostname	- - -	If more than one IP address (or host name) has been set for the machine, specify this parameter when you operate the CORBA server application using the limited IP address. If you specify an IP address (or host name), the specified IP address is incorporated when the server application object reference is generated. The IP address is used at connection from clients. Also, the CORBA service does binding with the specified IP address only. If this parameter is omitted, binding is done for all IP addresses. However, which of the two IP address, Ipv4 or Ipv6, (or whether the IP addresses of both) is bound depends on the IP-version parameter setting.	Valid only for servers function. (*1) (*2)
IIOP_port	8002 Windows32/64 Required parameter Solaris32/64 Linux32/64 - -	Port number used by CORBA Service. Always specify this parameter when specifying a value other than the default (8002).	Windows32/64 Required parameter Solaris32/64 Linux32/64 (*3)

*1 Connection requests can be received from only a particular LAN card on a machine on which more than one LAN card is mounted.

If a host name is specified, name solution is performed according to the IP-version value.

If IP-version is v4-dual, name solution under IPv4 is given priority.

If IP-version is v6, name solution under IPv6 is given priority.

If a link local IPv6 address is specified in a Windows version, scope-id must also be specified.

Example: fe80::1234:5678:9abc:def0%4

*2 Do not specify this parameter unless required for a special purpose as explained in (*1). If an incorrect host name is specified, Interstage fails to start.

*3 In Solaris and Linux systems, if this value becomes invalid, the value set in /etc/services becomes valid.

Table A.2 Operating Environment Relating to Network Environment

Parameter	Initial value Default value Specifiable value	Meaning	Remarks
con_accept	all all all, localhost	Specify "localhost" for this parameter when receiving client connections is limited to your own host. With this specification, only the connections from your own host are received. If, however, "all" is specified, connections from all machines are received. Specify "localhost" when you do not want to permit connection requests from other hosts for system security reasons.	Valid only for server function.
IP-version	v4-dual v4-dual v4, v4-dual, v6	Set the IP version to be operated: - v4: Only IPv4 is used to run CORBA applications (IPv6 is not used.) - v4-dual: IPv4 and IPv6 are used to run CORBA applications. When the CORBA service operates as a server, both IPv4 and IPv6 are accepted. When the CORBA service operates as a client, IPv4 is used preferentially. - v6: IPv4 and IPv6 are used to run CORBA applications. When the CORBA service operates as a server, both IPv4 and IPv6 are accepted. When the CORBA service operates as a client, IPv6 is used preferentially. If "v4-dual" or "v6" is specified in an environment that does not support IPv6, "v4" will be set.	
read_interval_ timeout	30 30 0~100000000	Read response time for socket. If read cannot be completed during the read response time, a system exception (COMM_FAILURE) is reported to applications. The value represents the actual time (in seconds). If 0 is specified, time monitoring is not performed. Monitoring based on this parameter begins when message receive processing begins. For instance, if no packet is received in reply message wait state, monitoring based on this parameter is not performed. Instead, monitoring based on period_receive_timeout is performed. If at least one package is received, monitoring based on read_interval_timeout is performed because receive processing begins.	
write_interval_timeout	30 30 0~100000000	Write response time for socket. If write cannot be completed during the read response time, a system exception (COMM_FAILURE) is reported to applications.	

Parameter	Initial value	Meaning	Remarks
	Default value		
	Specifiable value		
		The value represents the actual time (in seconds). If 0 is specified, time monitoring is not performed. Monitoring based on this parameter begins when message transmission processing begins.	
tcp_nodelay	no	Configure settings to enable or disable the TCP_NODELAY function. If "yes" is specified, the Nagle algorithm is disabled when the message is sent. If "no" is specified, the Nagle algorithm is enabled. If the Nagle algorithm is enabled, the efficiency of the network for buffering sent data is improved. If the Nagle algorithm is disabled, the efficiency of the network is reduced because the sent data is not buffered. This may cause a reduction in the overall communication performance. However, it may also result in a reduction in the time lag that occurs when data is sent and received, and an improvement in the response performance.	Valid only for server function.
	no		
	yes, no		

Operating Environment of Application Resources (such as Process/Thread Concurrency, Number of Used Connections)

The values that can actually be specified for these parameters are restricted by OS resources.

Parameter	Initial value	Meaning	Remarks
	Default value		
	Specifiable value		
max_exec_instance	512 (*6)	Maximum number of threads (or processes) used for server application request execution.	Valid only for server functions. (*1) (*2)
	256		
	16~1000000		
max_IIOP_local_init_con	256	Maximum number of connections to the server host that are used by a client application.	(*1) to (*7)
	256		
	1~1000000		
max_IIOP_local_init_requests	4096	Maximum number of requests that a client application can send simultaneously.	
	4096		
	1~1000000		
max_IIOP_resp_con	8 (*6)	Maximum number of connections that can be set up with a client application.	Valid only for servers function. (see Cautions) (*2)
	8		
	1~500000		
limit_of_max_IIOP_resp_con	0	Maximum number to which max_IIOP_resp_con can automatically be increased. If 0 is specified, the following value is used: max_IIOP_resp_con x 1.3 (decimal portion is discarded)	Valid only for the server function. (*2) (*4)
	0		
	0~1000000		

Parameter	Initial value	Meaning	Remarks
	Default value		
	Specifiable value		
max_IIOP_resp_con_ extend_number	0	Number of connections that can automatically be increased from max_IIOP_resp_con. If 0 is specified, the following value is used: (limit_of_max_IIOP_resp_con - max_IIOP_resp_con) / max_IIOP_resp_con (decimal portion is rounded up)	Valid only for the server function. (*4) (*5)
	0		
	0~1000000		
max_IIOP_resp_ requests	128 (*6)	Maximum number of requests that the server host can receive simultaneously.	Valid only for the server function. (*2)
	128		
	1~500000		
limit_of_max_IIOP_ resp_requests	0	Maximum number to which max_IIOP_resp_requests can automatically be increased. If 0 is specified, the following value is used: max_IIOP_resp_requests x 1.3 (decimal portion is discarded)	Valid only for the server function. (*2) (*4)
	0		
	0~1000000		
max_IIOP_resp_ requests_extend_ number	0	Number of requests that can automatically be increased from max_IIOP_resp_requests. If 0 is specified, the following value is used: (limit_of_max_IIOP_resp_requests - max_IIOP_resp_requests) / max_IIOP_resp_requests (decimal portion is rounded up)	Valid only for the server function. (*4) (*5)
	0		
	0~1000000		
max_processes	20 (*6)	Maximum number of processes (number of clients + servers started)	Valid only for server functions. (*2) (*3).
	16		
	0~1000000		
max_impl_rep_entries	512	Maximum number of registrations in the Implementation Repository.	Valid only for servers function.
	256		
	100~1000000		
number_of_common_ buffer	0	Specify the number of default buffers to be used for queue control by the CORBA Service. It is used in the CORBA Service communication, excluding the CORBA application in which "Buffer Number: the number of communication buffers" is specified in the WorkUnit definition in the CORBA application, which is operated as a WorkUnit. Specify the maximum number of requests processed at the same time on the server machine. If 0 is specified, the following values are set: max_IIOP_resp_requests x 0.2 (decimal portion is discarded)	Valid only for servers function. (*2)
	0		
	0~500000		

Parameter	Initial value	Default value	Specifiable value	Meaning	Remarks
limit_of_number_of_ common_buffer	0	0	0~1000000	Maximum number to which number_of_common_buffer can automatically be increased. If 0 is specified, the following value is used: limit_of_max_IIOP_resp_requests	Valid only for the server function. (*2 and 4)
number_of_common_ buffer_extend_number	0	0	0~1000000	Number of buffers that can automatically be increased from number_of_common_buffer. If 0 is specified, the following value is used: (limit_of_number_of_common_buffer - number_of_common_buffer) / number_of_common_buffer (decimal portion is rounded up)	Valid only for the server function. (*4) (*5)
max_bind_instances	0	0	0~1000000	Number of binding server process and object relationships that can be registered in the CORBA Service. If 0 is specified, the following values are set: 1) If 1024 x max_processes < 65535 the value set is 1024 x max_processes 2) If 1024 x max_processes >= 65535 the value set is 65535	Valid only for the server function. (*2) (*7)

- *1 To estimate what value to set, calculate:

  Windows32/64

  Number of registered applications(*1) x maximum process concurrency(*2) x maximum number of concurrent threads(*3) + Number of connected clients(*4) + 64

  Solaris32/64 Linux32/64

  Number of registered applications(*1) x maximum process concurrency(*2) x maximum number of concurrent threads(*3) + Number of connected clients(*4) + 172

  - *1 Number of applications registered using the *OD_impl_inst* command
  - *2 proc_conc_max set using the *OD_impl_inst* command
  - *3 thr_conc_maximum value set using the *OD_impl_inst* command
  - *4 Number of connected clients corresponding to the scale-value set using the *isgendef* command.

- *2 With the server function, the parameter setting and the consumption amount can be checked using the odprtcurparam command.
- *3 This also includes:
  - the CORBA Services (CORBA Service, Naming Service, Interface Repository server and Interface Repository cache server). When making the estimate, add the amount used for the application to the amount used for the Interstage service (the amount used for the Interstage service is 20).
  - the CORBA Services commands. If executing more than one command at a time, add the number of commands to the value being specified.
- *4 Automatic Expansion

  The CORBA service supports two types of parameters for automatic expansion: limit_of_parameter-name and parameter-name_extend_number. For instance, limit_of_max_IIOP_resp_con and max_IIOP_resp_con_extend_number are available as the expansion parameters for the max_IIOP_resp_con parameter.

Each type of element is defined by parameter_name for the initial value and limit_of_parameter-name for the maximum value, and can be expanded in divisions of parameter-name_extend_number as needed.

An example is shown below:

Example

```
max_IIOP_resp_con = 100
limit_of_max_IIOP_resp_con = 140
max_IIOP_resp_con_extend_number = 2
```

With the above parameters specified, max_IIOP_resp_con can be increased up to twice to 120 and 140 from the initial value of 100.

If MANUAL is specified for definition item AutoConfigurationMode in the isconfig.xml file, the parameters related to automatic expansion are ignored and no elements are increased. For details on isconfig.xml, refer to the Interstage Application Server Operator's Guide.

- *5 The size that can be increased in single expansion processing is restricted by the initial size.

  If expansion is set in such a way that the single expansion size exceeds the initial size, the expansion count is corrected to the same value assumed when 0 is specified.

  If the expansion count exceeds the difference between the initial value and the maximum value for automatic expansion, the expansion count is corrected to the difference between initial value and the maximum value for automatic expansion.

  Example

```
max_IIOP_resp_con = 100
limit_of_max_IIOP_resp_con = 300
max_IIOP_resp_con_extend_number = 1
```

  With the above parameters specified, max_IIOP_resp_con_extend_number is corrected to 2.

- *6 The default values are changed as listed in Table A.3 Changed Values in the following cases:

  Windows32/64

  - The default values are changed in Enterprise Edition/Standard-J Edition.

  Solaris32/64

  - The default values are changed when standard or custom installation (or installation using the GUI installer) is performed with Enterprise Edition/Standard-J Edition (installation with the pkgadd command is not performed).

  Linux32/64

  - The default values are changed when standard or custom installation (or installation using the GUI installer) is performed with Enterprise Edition/Standard-J Edition (installation with the rpm command is not performed).

Table A.3 Changed Values

Parameter name	Changed value
max_IIOP_resp_con	512
max_IIOP_resp_requests	2048
max_processes	512
max_exec_instance	16384

- *7 Set a value that is greater than the total of:

  The number of objects registered by the C++ CORBA application issuing the CORBA::ORB::bind_object function call + the number of EJB application Session Beans plus Entity Bean EJB object instances called from the other JavaVM.

Cautions

For max_IIOP_local_init_con, specify the maximum number of connections to the server host that are used by each application.

The values max_IIOP_resp_con specify the number of connections between applications used on the hosts.

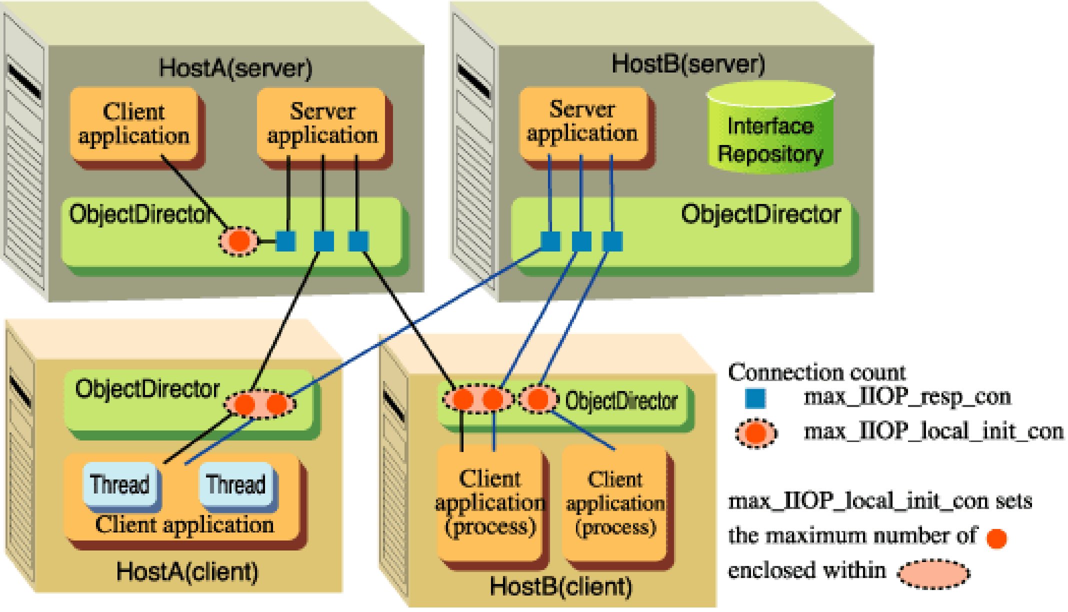
As a general rule, connections between applications are generated in units of client application processes. For example, even if several requests are issued concurrently from client applications to a single server application, the number of connections will be one.

If the SSL coordinated function is used, SSL connections and non-SSL connections must be counted separately. For example, if SSL connections and non-SSL connections are used for connection to the server machine from a client application, the number of connections is two.

Since one connection is used for each when executing the commands and when operating the Interface Repository, increase the number of connections when necessary. When starting more than one command at a time, add the number of the commands to specify the value.

The following figure shows the counting of connections between applications used on hosts.

Figure A.1 max_IIOP_resp_con/max_IIOP_local_init_con Connection Count



The following paragraphs explain how to count the number of connections for each parameter.

max_IIOP_local_init_con

Specify the maximum number of connections to the server application (host unit) from the client application (process unit) on a host in which a client application runs.

- Formula for estimating the max_IIOP_local_init_con value (when an Interface Repository is going to be run):

  max_IIOP_local_init_con = 256 or ['maximum number of server hosts to which one client application is connected'], whichever is greater

- Formula for estimating the max_IIOP_local_init_con value (when an Interface Repository is going to be run and the SSL coordinated function is used):

  max_IIOP_local_init_con = 256 or ['maximum number of server hosts to which one client application is connected' x 2], whichever is greater

max_IIOP_resp_con

Specify a value calculated by adding up the total number of client application processes connecting on hosts where server applications are running. This number of connections will also need to be added in cases where the client application and the server application are connecting on the same host.

- Formula for estimating the max_IIOP_resp_con value (when an Interface Repository is going to be run):

  max_IIOP_resp_con = number of client application processes connecting + 2

- Formula for estimating the max_IIOP_resp_con value (when an Interface Repository is going to be run and the SSL coordinated function is used):

  max_IIOP_resp_con = (number of client application processes connecting x 2) + 2

About max_IIOP_local_init_requests and max_IIOP_resp_requests

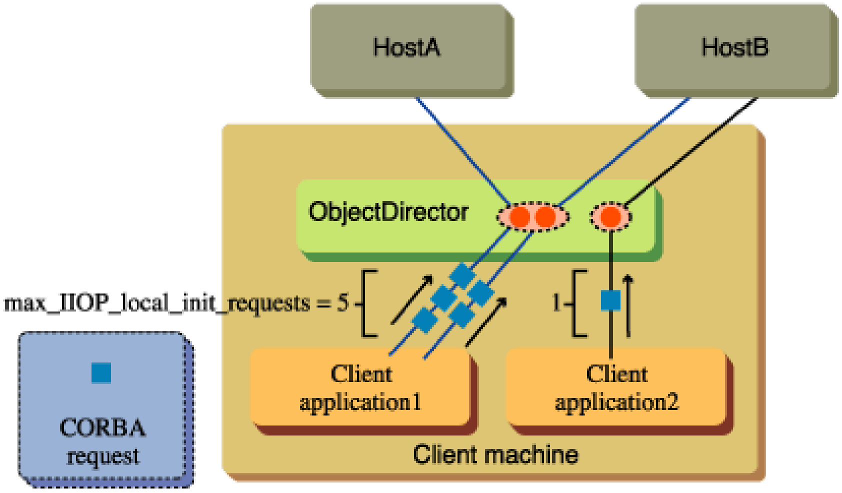
For the CORBA service, max_IIOP_local_init_requests must be set according to the number of requests that a client application sends simultaneously. Similarly, max_IIOP_resp_requests must be set according to the number of requests that a server application receives simultaneously.

max_IIOP_local_init_requests

Specify the maximum number of requests that that a client application can send simultaneously. In the following example, client application 1 sends five requests simultaneously and application 2 sends one request simultaneously. Thus, max_IIOP_local_init_requests must be set to 5 or a greater value.

If the calculated value is 4,096 or less, the default value of 4,096 can be used as is. In this example, the calculated value is less than 4,096 and therefore max_IIOP_local_init_requests need not be changed from default 4,096.

Figure A.2 max_IIOP_local_init_requests



max_IIOP_resp_requests

Specify the maximum number of requests that the CORBA server application can receive simultaneously.

This value is the number of requests that are processed simultaneously by the CORBA server application after the requests issued from individual client machines reach the server machine. Thus, the total number of requests issued simultaneously from individual client machines must be estimated.

In the following example, a total of nine requests issued from individual client machines reach the server machine. max_IIOP_resp_requests must be set to 9 or more.

Figure A.3 max_IIOP_resp_requests

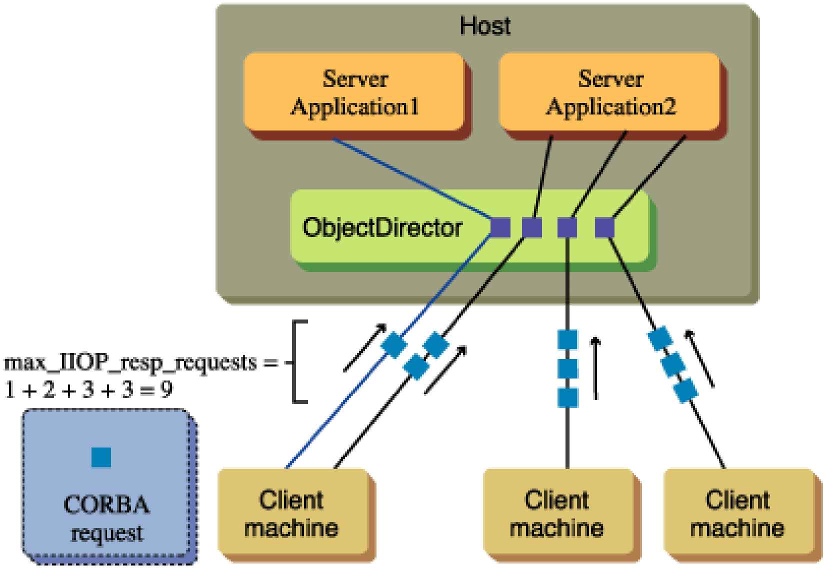


Table A.4 Operating Environment of Timeout Monitoring

Parameter	Initial value	Meaning	Remarks
	Default value		
	Specifiable value		
period_client_ idle_con_timeout	96 (480 sec)	The monitoring time (connection maintenance time after request reply is completed) in client no communication status (no request to server is transmitted). Connection with the server is broken if a request is not sent to the server before this time has been exceeded (*2). This value multiplied by 5 becomes the actual value (in seconds). If this value is set to zero, no timeouts will occur.	
	96 (480 sec)		
	0~20000000		
period_idle_con_ timeout	120 (600 sec)	The monitoring time (connection maintenance time after request reply is completed) of no communication status (no request from client is transmitted) in the server. The connection with the client is cut when there is no request transmission from the client even if this time is exceeded, and the memory resource used to process the request is released. This value multiplied by 5 becomes the actual value (in seconds). If this value is set to zero, no timeouts will occur.	Valid only for server functions.
	1 (5 sec)		
	0~20000000		
period_receive_ timeout	72 (360 sec)	The response time between the time of request and the time of response in the client. The client is notified of the timeout when there is no reply from the server even if this time is exceeded. This value multiplied by 5 becomes the actual value (in seconds).	
	72 (360 sec)		
	0~20000000		

Parameter	Initial value	Meaning	Remarks
	Default value		
	Specifiable value		
period_server_ timeout	120 (600 sec) (*1)	This parameter works differently between server applications other than the Persistent type and other application types. For server applications other than the Persistent type, the parameter specifies the monitoring time from the start of an application to the completion of the CORBA_ORB_init method. If the CORBA_ORB_init method is not completed within this time, a system exception (NO_IMPLEMENT) is posted to the client. For client applications and server applications of the Persistent type, the parameter specifies the monitoring time from the issuance of a CORBA_ORB_init method to its completion. This value multiplied by 5 becomes the actual value (in seconds).	Valid only for server functions.
	120 (600 sec)		
	0~20000000		

*1 The Interface Repository may not start if the initial value is reduced.

*2 Connection with the server is re-established the next time a request is sent.

In process mode, however, connection with the server is broken/re-established the next time a request is sent, not when this time is exceeded.

Remarks

The timeout parameter must be set with consideration to the timeout applied to a linkage application. For details, refer to "CORBA Application Timeouts" in the OLTP Server User's Guide (provided with Enterprise Edition products).

Windows32/64 Solaris32/64

Table A.5 Operating Environment of Security Function

Parameter	Initial value	Meaning	Remarks
	Default value		
	Specifiable value		
http_proxy	proxy_host	Host name of HTTP proxy server.	(*1)
	null (no value is set)		
	-		
http_proxy_port	8080	Port number used by HTTP proxy server.	(*1)
	0		
	-		
http_proxy_use	No	Specifies the use of HTTP proxy server function: "yes": Used "no": Not used	(*1)
	No		
	yes, no		
UNO_IIOP_ssl_use	No	Specifies the use of the SSL coordinated function: "yes": Used "no": Not used	(*2)
	No		
	yes, no		
UNO_IIOP_ssl_port	4433	Port number for SSL communication.	

Parameter	Initial value	Meaning	Remarks
	Default value		
	Specifiable value		
	4433		
	-		

*1 These parameters are referred to when HTTP tunneling is used via an HTTP proxy server with pre-installed runtime (an executive environment except for Portable-ORB). http_proxy and http_proxy_port are valid when http_proxy_use = yes is specified. Specify the host name and port number of the HTTP proxy server used by the Web browser.

*2 SSL connections and non-SSL connections must be counted separately. Take this into consideration this when estimating the 'max_IIOP_resp_con' and 'max_IIOP_local_init_con' parameters.

Solaris32/64 Linux32/64

Table A.6 Operating Environment of Security Function (Resource Protection)

Parameter	Initial value	Meaning	Remarks
	Default value		
	Specifiable value		
iss_use	No	Specifies whether the resource protection function is valid. If "yes" is specified, only users (or the root) that belong to the iss_group group can start CORBA applications.	(*1) (*2)
	No		
	yes, no		
iss_group	root(0)	Specifies the group ID for operating applications when the resource protection function is valid (iss_use = yes).	
	root(0)		
	-		

*1 If "enhanced security mode" is selected for the security settings when the application is installed, the initial values change as follows:

Parameter	Initial value
iss_use	Yes
iss_group	"Interstage application group name" specified when the application is installed

*2 To change the resource protection function settings, it is recommended that the issetsecuritymode command is used. For details, refer to the "Enhancing Security (Protecting Interstage Resources)" appendix in the Security System Guide.

*3 Specify a group that has already been registered in the system.

*4 The execution of the CORBA application is limited to users that belong to the group specified as the iss_group or the root. Other general users cannot execute the CORBA application. For this reason, note the effective users of the application (refer to "OD_impl_inst" in the "CORBA Service Operation Commands" chapter of the Reference Manual (Command Edition).

Table A.7 Operating Environment of Maintenance Function

Parameter	Initial value	Meaning	Remarks
	Default value		
	Specifiable value		
access_log_policy	start (Default value is recommended.)	Specifies if an access log is to be kept when the CORBA Service starts. - "start": a log is to be kept when the CORBA Service starts. - "standby": no log is to be kept.	Valid only for server functions (*1)
	start		
	start, standby		

Parameter	Initial value	Meaning	Remarks
	Default value		
	Specifiable value		
access_log_size	3000000	Maximum size of access log (bytes)	Valid only for server functions (*1)
	3000000		
	1~2147483467 (Maximum value for long type)		
access_log_level	send_stex: recv_stex: send_userex: recv_userex: close_resp_info	The key word at the access log collection level is specified. The delimiter is a colon (":"). Blanks cannot be specified. When "all" is specified, it is considered that all the collection levels are specified.	Valid only for server functions (*1)
	send_stex: recv_stex: send_userex: recv_userex: close_resp_info		
	-		
error_log_policy	start (Default value is recommended.)	Specifies if an error log is to be kept when the CORBA Service starts. - "start": a log is to be kept when the CORBA Service starts. - "standby": no log is to be kept.	(*1)
	start		
	start, standby		
error_log_size	3000000	Maximum size of error log (bytes)	(*1)
	3000000		
	1~2147483467 (Maximum value for long type)		
info_log_policy	start (Default value is recommended.)	Specifies whether to collect log information when the CORBA service starts. - "start": a log is to be kept when the CORBA Service starts. - "standby": no log is to be kept.	(*1)
	start		
	start, standby		
info_log_size	3000000	Maximum size of information log file (bytes)	(*1)
	3000000		
	1~2147483467 (Maximum value for long type)		
logging	no	Specifies if an internal log is to be kept. - "no": no log is to be kept - "yes": a log is to be kept.	(*2)
	no		
	yes, no		
log_file_size	10000000	Maximum size of internal log (bytes) If "logging = yes" is to be specified, do not omit this parameter.	(*2)
	-1		
	4096~2147483467		

Parameter	Initial value	Meaning	Remarks
	Default value		
	Specifiable value		
process_log_policy	start (Default value is recommended.)	Specifies if a process log is to be kept when the CORBA Service starts. - "start": a log is to be kept when the CORBA Service starts. - "standby": no log is to be kept.	(*1)
	start		
	start, standby		
process_log_size	3000000	Maximum size of process log (bytes)	(*1)
	3000000		
	1~2147483467 (Maximum value for long type)		
log_file_path	-	Specify the absolute path to the log file output destination. The following log files will be output to the location identified by the path specified by this parameter: - Access log - Error log - Information log - Process log - Internal log If the path specified here does not exist, the CORBA service cannot start. Paths longer than 128 bytes are not valid. If a path longer than 128 bytes is specified, this parameter is not effective. Additionally, the path cannot include white space or the equals sign. (=). If a path including these characters is specified, only values before the white space or equal sign are valid. In Windows® system, both "\" and "/" are used as a separator of folders.	(*1) (*2)
	-		
	-		
snap_size	40000	Upper limit for snapshot size (in bytes)	Valid only for server functions
	40000		
	1~2147483467 (Maximum value for long type) (*3)		
snap_use	yes	Specifies collection of snapshot: - "yes": collected - "no": not collected	Valid only for server functions
	yes		
	yes, no		
trace_file_synch_ level	stop	Specifies trace file output timing. Multiple files can be specified. (Separator is "&"). - none: It is output to trace file only when the odformtrace command is used.	Valid only for server functions
	stop		
	-		

Parameter	Initial value	Meaning	Remarks
	Default value		
	Specifiable value		
		- exit: The trace information of the completed application is output to the trace file when the application ends normally. - vanish: The trace information of the completed application is output to the trace file when the application ends abnormally. - stop: The trace information of all applications is output to the trace file when the CORBA service has ended. - loop: It is output to the trace file when size of the trace information acquired on memory has exceeded the trace_size_per_process.	
trace_size_per_ process	10000	The maximum of the trace information size for each process. (in byte unit)	Valid only for server functions
	10000		
	1024~100000000		
trace_size_of_daemon	0	The maximum size (in bytes) of the trace information for the CORBA service daemon process. If 0 is specified, the following value is set. If the result exceeds 100000000, 100000000 is set. If a value smaller than the trace_size_per_process value is specified, it will be corrected to the trace_size_per_process value. trace_size_per_process x 32	
	0		
	1024~100000000		
trace_use	yes (Default value is recommended.)	Specifies acquisition of the trace information: - yes: Acquire - no: Not acquire	Valid only for server functions
	yes		
	yes, no		

*1 The access log, process log, error log, and information log are output to the path specified by log_file_path. If log_file_path is not specified, they are collected in the following directories.

It is necessary to add the following log file sizes to the disk area.

Windows32/64

Storage Directory (Default installation path)

```
C:\Interstage\ODWIN\var
```

Solaris32/64

Storage Directory (Default installation path)

```
/var/opt/FSUNod
```

Linux32/64

Storage Directory

```
/var/opt/FJSVod
```

Table A.8 Log File Name and File Size

Log name	Log file name	Log file size
Access log	accesslog accesslog.old	access_log_size x 2
Process log (If server library (ODSV.DLL, libOM.so) is linked.)	proclog proclog.old	process_log_size x 2
Process log (If client library (ODWIN.DLL) is linked.)	proclogcl proclogcl.old	process_log_size x 2
Error log (If server library (ODSV.DLL, libOM.so) is linked.)	errlog errlog.old	error_log_size x 2
Error log (If client library (ODWIN.DLL) is linked.)	errlogcl errlogcl.old	error_log_size x 2
Information log	infolog infolog.old	info_log_size x 2
Information log If client library (ODWIN.DLL) is linked	infologcl infologcl.old	info_log_size x 2

Notes

1. The log file storage directory requires the write access permission of the administrator group in order to collect the following log files:

    - Access log
    - Process log (if the library for a server has been linked)
    - Error log (if the library for a server has been linked)
    - Information log (if the library for a server has been linked)

    The log file storage directory requires write access permission of the group including the user who executes the client application to collect these log files:

    - Process log (if the library for a client has been linked)
    - Error log (if the library for a client has been linked)
    - Information log (if the library for a client has been linked)

    Without the access permission above, the output of the log files fails. At this time, in some cases, no particular error message may be displayed. Therefore, if log files are to be collected, confirm that the access permission to the log file storage directory is correctly set up before starting the operation.

2. When logging=yes is specified, a great deal of time is required for output processing to the internal log file. This results in deterioration of CORBA service and the CORBA application response performance. A great deal of time is also required for starting

the Interface Repository or Naming Service. If it takes one minute or more to start the Interface Repository or Naming Service, Interstage fails to start.

When logging=yes is specified, an internal log is output to the path specified in log_file_path. If log_file_path is not specified, an internal log data is output to the following path. The file name is common regardless of the log_file_path value.

- Windows®:
    - Path : C:\Interstage\ODWIN\var
    - File : log (log.old), appNNNN.log (appNNNN.old) (NNNN: alphanumeric characters) for each server application
- Solaris:
    - Path : /var/opt/FSUNod
    - File : log (log.old)
- Linux:
    - Path : /var/opt/FJSVod
    - File : log (log.old)

When the pre-installed type Java library is used, logging is output to the following files as well as those described above:

Under the work directory (to the position that is indicated with user.dir of the Java VM system property)

JVxxxxxxxxxx.log (where xxxxxxxxxx is a number)

For applet applications, user.dir can be changed using the Java VM start option.

The log_file_path value has no effect.

Table A.9 Operating Environment Regarding the Compatibility of Previous Versions

Parameter	Initial value Default value Specifiable value	Meaning	Remarks
msg_output_ compatible	no no od10301, od10605, od10924, od10925, od10926, od10941, od11101, od60003, no	Specify the message numbers to output to the system log. When multiple numbers are specified, use an ampersand ("&") to separate them. If the following messages are not going to be output, specify "no". - od10301 - od10605 - od10924 - od10925 - od10926 - od10941 - od11101 - od60003	

A.2 gwconfig

Overview

The gwconfig file defines the system requirements of the HTTP-IIOP gateway started by the Web Server when HTTP tunneling is used.

When an item that relates to timeout monitoring of the CORBA Service is corrected, it is necessary to correct the definition of similar items.

The gwconfig file does not need to be corrected when not changing from an initial value.

File Name

Windows32/64

(Default installation path)

```
C:\Interstage\ODWIN\etc\gwconfig
```

Solaris32/64

(Default installation path)

```
/etc/opt/FSUNod/gwconfig
```

Linux32/64

```
/etc/opt/FJSVod/gwconfig
```

File Contents

In the gwconfig file, values are specified in the following format.

Format

Parameter name=value set

Lines beginning with a hash sign (#) are regarded as comment lines. Blank lines are ignored for analysis.

# comment

Sample Statement

timeout_response=60

Parameters

The parameters for which the values can be changed are shown in the following table.

Table A.10 Modifiable gwconfig Parameters

Parameter	Initial value	Specifiable value	Meaning
timeout_response	360	0~100000000	Reply waiting time for request. Standby time from request transmission in HTTP-IIOP gateway to reply (seconds). The timeout is notified to the client if there is no reply from the method of the server in this time. Change this parameter to below the period_receive_timeout value (the response time between the time of request and the time of response by the client defined with the config file) when you change period_receive_timeout for the CORBA Service on the client side. If 0 is specified, the timeout will not occur.

Parameter	Initial value	Specifiable value	Meaning
timeout_session	180	0~2147483647	Session hold time (no communication monitoring time between clients). Hold time in HTTP-IIOP gateway for each handling of the client (seconds). The administrative information for the client is deleted when there is no new request transmission in this time from the client and there is no reply waiting from the server. If 0 is specified, the timeout will not occur.
timeout_connection	60	0~2147483647	Connection hold time (no communication monitoring time between servers). The monitoring time in the HTTP-IIOP gateway is not communicated. The connection with the server is cut when there is no new request transmission in this time from the client and there is no replay waiting from the server. If 0 is specified, the timeout will not occur.
logmode	5	1~5	Collection of internal log for HTTP-IIOP gateway. When an internal log is collected, 1 is set. (see Note) Specify from the following: - 5: Internal log information is not collected. - 3: Request data and information at error occurrence are collected. - 2: In addition to information collected in 3, reply data and internal processing information are collected. - 1: In addition to information collected in 2, trace information is collected.
max_log_file_size	1048576	1048576~10485760	Log file size (in bytes).

Notes

- The internal log of the HTTP-IIOP gateway is output as follows.
  - Windows®: C:\Interstage\ODWIN\var\httpgw*.log
  - Solaris: /opt/FSUNod/var/httpgw*.log
  - Linux: /opt/FJSVod/var/httpgw*.log
- It is necessary to stop the Web Server to stop internal log collection.

Solaris32/64

- When the Web server is Interstage HTTP Server, an internal log is created for each communication process. The file name is httpgwprocess-number_N.log (N is a number).
- When the Web server is Interstage HTTP Server, set write permission for the user or group name that executes a server process of Interstage HTTP Server in the directory to which the internal log of the HTTP-IIOP gateway is output. If write permission is not available, message od40102 is output to the system log.

Linux32/64

- When the Web server is Interstage HTTP Server, an internal log is created for each communication process. The file name is httpgwprocess-number_N.log (N is a number).
- Set write permission for the user or group name that executes a server process of Interstage HTTP Server in the directory to which the internal log of the HTTP-IIOP gateway is output. If write permission is not available, message od40102 is output to the system log.

Notes

- Modified definition information becomes valid the next time the Web server is started.
- Storage directory of gwconfig file

  The storage directory of the gwconfig file is specified with the OD_HTTPGW_HOME or OD_HOME environment variable. If both variables are specified, OD_HTTPGW_HOME is given priority. It stores the file in the etc directory under the specified directory.
- A line feed must be placed at the end of the last line.
- If a non-numeric character is used in a specified value, the numeric part before the non-numeric character will be regarded as a valid specification.
- If the Web server is Interstage HTTP Server on Solaris or Linux, the timeout_session and timeout_connection parameters are ignored.

A.3 inithost/initial_hosts

Overview

The inithost/initial_hosts file is where host information of the naming service and the interface repository are defined. Because the naming service and the interface repository contain the location of applications and interface information, they are necessary for linkage with CORBA applications.

In the inithost/initial_hosts file, specify the host name where the service is present (or an IP address), and a CORBA services port number (default value is 8002). Up to 16 combinations of the host name and the port number can be specified.

Inquiries of the service are made in the order of definition, and if the service referenced does not exist, an inquiry is made to the host defined in the next line. Note that the inquiries will be performed to those hosts whose names were successfully resolved on the CORBA service startup. If the attempt has failed on all hosts, then the inquiries will be performed to those hosts whose name resolution has failed.

When the naming service and the interface repository are operated on a local host, setting of the host name and port number is not necessary.

File Name

Windows32/64

(Default installation path)

```
C:\Interstage\ODWIN\etc\inithost
```

Solaris32/64

(Default installation path)

```
/etc/opt/FSUNod/initial_hosts
(Windows® Client: C:\Interstage\ODWIN\etc\inithost)
```

Linux32/64

```
/opt/FJSVod/etc/initial_hosts
```

```
(Windows® Client: C:\Interstage\ODWIN\etc\inithost)
```

File Contents

In the inithost/initial_hosts file, values are specified in the following format.

Format

Hostname port number

Sample Statement

hostname 8002

Parameters

The parameters for which the values can be changed are shown in the following table.

Table A.11 Modifiable inithost Parameters

Parameter	Initial value	Meaning
Host name	No value	Specify the host name (or an IP address) where the naming service or the interface repository is operating. A host name can be up to 64 bytes long. (Note 1)
Port number	No value	Specify the port number of the CORBA service defined in the host where the above service is running.

Note

It must be possible to resolve (convert to an IP address) the host name set in the object reference registered in the initial service and the Naming Service on the server machine. To refer to information about the object reference that has been registered, execute the OD_or_adm command (-l option) and odlistns command (-l option) on the server machine.

It must be set to the same host name on the local host side and on the server side which operates the service.

Local host side:

- Windows:

  lmhosts or hosts of Windows ®System directory\system32\drivers\etc.

- Solaris/Linux:

  /etc/hosts or NIS+ etc.

Server side:

- Host name definition on server side

Notes

- Modification of definition information

  When definition information is modified, it becomes valid after the CORBA service is next started.

- Set up the inithost file using the isinit command and ismodifyservice command:

  - Comment out, or delete, the host name of Interface Repository Service and Naming Service specified in the inithost/initial_hosts file when you execute an isinit command and an ismodifyservice command.

  - The setup of the inithost/initial_hosts file becomes possible after an isinit command and an ismodifyservice command are executed.

  - Even when a host name is set up in the inithost/initial_hosts file, it gives priority to the host name set up with the isinit command and the ismodifyservice command. And, it doesn't need to set up the host name set up with the isinit command and the ismodifyservice command in the inithost/initial_hosts file.

  - When environment setting was performed by the isinit command and ismodifyservice command, remote host operation of the naming service using the inithost/initial_hosts file cannot be performed. This is because when the isinit command and ismodifyservice command are used, the remote host name of the naming service is set in the initial service (meaning the inithost/initial_hosts file cannot be used).

    - Remote host operation of the naming service using the inithost/initial_hosts file can be performed only during operation of the CORBA service client, in which neither the isinit command nor the ismodifyservice command is included.

- Unnecessary host information definition

  If a non-existent host (or a host unable to communicate) is specified as the host name when the inithost file in Windows is edited, the operation of the client application, IDL compiler, and so on may slow down. Delete unnecessary host names.

Remarks

Setting of the host name and the port number can be performed by the odsethost command also.

A.4 queue_policy

This is not valid for Linux (64 bit) and Windows (64 bit).

Overview

queue_policy is the file used as the queue policy by the queue control function.

A "queue_policy.default" file is provided as a sample. Edit the sample file, and make a "queue_policy" file when you use the queue control function.

File Name

Windows32/64

(Default installation path)

```
C:\Interstage\ODWIN\etc\queue_policy
```

Solaris32/64

(Default installation path)

```
/etc/opt/FSUNod/queue_policy
```

Linux32/64

```
/opt/FJSVod/etc/queue_policy
```

File Contents

The queue_policy file is divided into three sections

- [QUEUEGROUP] section
- [QUEUE] section
- [GUARANTY] section.

The [QUEUEGROUP] and [QUEUE] sections can be updated using the odsetque command.

Change the [GUARANTY] section by using an editor. If the [GUARANTY] section is undefined, refer to "odsetque" in the "CORBA Service Operation Commands" chapter of the Reference Manual (Command Edition) for details of the maximum value.

Format

    [GUARANTY]

Queue name = maximum limit for queue

Sample Statement

```
[GUARANTY]
queue1   = 64
```

Parameters

The parameters for which the values can be changed are shown in the following table.

Table A.12 Modifiable queue_policy Parameters

Parameter	Specifiable value	Meaning
Queue name (*1)	-	Specifies the queue name registered using the odsetque command.
Maximum limit for queue	1~2147483467 (Maximum value for long type)	Specifies the queue threshold value (this may not be omitted).

*1 The queues and set values used by the CORBA service are shown in the following table.

Table A.13 Queues Used by CORBA Services

Queue name	Value set (upper limit value of the queue)
SYSTEM_GLOBAL	Cannot be edited
OD_ORB_QUEUE	Cannot be edited
COS_NAMING_QUE	A value equal to or greater than max_IIOP_resp_con
INTERFACE_REP_QUE	A value equal to or greater than max_IIOP_resp_con

Notes

- When definition information is modified, it becomes valid only after restarting the CORBA service.
- When registration is carried out using the odsetque command, queue information is not added to [GUARANTY]. If you set a maximum limit, you will need to add a new definition.

A.5 nsconfig

Overview

nsconfig is the file which sets the operating environment for the Naming Service.

File Name

Windows32/64

(Default installation path)

```
C:\Interstage\ODWIN\etc\nsconfig
```

Solaris32/64

(Default installation path)

```
/etc/opt/FSUNod/nsconfig
```

Linux32/64

```
/opt/FJSVod/etc/nsconfig
```

File Contents

In the nsconfig file, values are specified in the following format.

Format

Parameter name = value set

Insert " = " (space + equal sign + space) between the parameter name and the value.

Lines beginning with a hash sign (#) are regarded as comment lines. Blank lines are ignored for analysis.

# comment

Sample Statement

```
file_sync = yes
trace_level = update
bl_how_many=65536
ogl_how_many = 256 (*1)
ext_intf=yes
cn_userexception_log_use = yes
cn_userexception_log_size = 2000000
```

*1 This is not valid for Linux (64 bit).

Parameters

The parameters for which the values can be changed are shown in the following table.

Table A.14 Modifiable nsconfig Parameters

Parameter	Initial value	Meaning
	Specifiable value	
file_sync	yes	Specifies whether synchronous file writing should be performed when the Naming Service is being updated. - yes: Enable synchronous file writing. - no: Disable synchronous file writing. When updating large volumes of data in the course of an initial build, for example, you can increase the processing speed by setting this parameter to no. While the Naming Service is in operation, this parameter should be set to yes for increased reliability.
	yes, no	
trace_level	update	Specifies the trace level when collecting an automatic trace of method executions: - update: Collect only the updated log. - all: Collect all the traces.
	update, all	
bl_how_many	65536	Specifies the maximum numbers of bindings returned by NamingContext::list and BindingIterator::next_n
	0~65536	
ogl_how_many (*1)	256	Specifies the maximum value of the load balance object group list.

Parameter	Initial value	Meaning
	Specifiable value	
	(Default value is recommended.)	Note that if the list for the load balance object group is created in the naming service, memory is allocated according to the setting of this parameter. This directly affects the memory usage, therefore, specify only the minimum value required, while taking the overall memory capacity into consideration.
	128~256	
ext_intf	yes	Specifies whether the extended functions provided by Naming Service are used. yes: Enable the naming service extended function. no: Disable the naming service extended function. If no is specified, naming context extended interface (NamingContexExt interface) cannot be used. Specify no, when working with V2.X or earlier clients.
	yes, no	
cn_userexception_log_use	yes	Specifies whether or not the user exception log is collected. yes: The user exception log is collected no: The user exception log is not collected
	yes, no	
cn_userexception_log_size	2000000	Specifies the user exception log file size.
	1000~2000000	

*1 This is not valid for Linux (64 bit).

Note

When a value is changed, the new value does not take effect until the next time the Naming Service is started.

A.6 irconfig

Overview

The irconfig file specifies backup and logging settings and other operating environment parameters of the Interface Repository.

File Name

Windows32/64

(Default installation path)

```
C:\Interstage\ODWIN\etc\irconfig
```

Solaris32/64

(Default installation path)

```
/etc/opt/FSUNod/irconfig
```

Linux32/64

```
/opt/FJSVod/etc/irconfig
```

File Contents

In the irconfig file, values are specified in the following format.

Format

Parameter name = value set

Lines beginning with a hash sign (#) are regarded as comment lines. Blank lines are ignored for analysis.

# comment

Sample Statements

```
auto backup = no(yes)
auto backup path =
logging = no(yes)
logging memory size = 512
logfile path =
sync = no
select cache obj =
```

Parameters

The parameters for which the values can be changed are shown in the following table.

Table A.15 Modifiable irconfig Parameters

Parameter	Initial value	Meaning
	Specifiable value	
auto backup	no	Specifies whether to take a backup automatically when the Interface Repository is started. - yes: Enable automatic backup. - no: Disable automatic backup. Note: The backup is only taken once, when the Interface Repository is started.
	yes, no	
auto backup path	-	Specifies the storage location for the backup data. If "auto backup" is set to "yes", the path must be specified. If no path is specified, the backup will not be taken. Note that the storage location requires an amount of free space equal to or greater than the size of the database created.
	-	
auto recovery	no	Specifies whether recovery is to be performed (based on the backup data), for example if an error is found in the database due to a shutdown during transaction processing. - yes: Enable automatic recovery. - no: Disable automatic recovery. Note that if you want to use this function you will need to set "auto backup=yes", and to specify the auto backup path.
	yes, no	
ir_timeout	1800 (sec)	Specifies the response time needed to get a response from the interface repository in the IDL compile (IDLc) and the interface information import (odimportir). If "0" is specified, the response time is not monitored.
	0~100000000(sec)	
iss_use Solaris32/64 Linux32/64	no	Specifies whether the resource protection function is valid. - yes: Enable the resource protection function. - no: Disable the resource protection function.
	yes, no	

Parameter	Initial value Specifiable value	Meaning
		If "yes" is specified, the Interface Repository can only be operated by the database administrator (Default: root). Note: If "enhanced security mode" is selected for the security settings when the application is installed, the initial value is "yes".
logging	no yes, no	If this parameter is set to "yes", log information will be collected when problems arise. - yes: Collect log information. - no: Do not collect log information. The log collected can be output to a file using the irlogdump command.
logging memory size	512(KB) 1~4096(KB)	Specifies the size of the shared memory storing the log information. If logging is set to "no", this value has no meaning.
logfile path	- -	This parameter is used to specify the full path for the directory containing the log information output by the irlogdump command when logging has been set to "yes". If no path is specified, the information will be stored in the same directory where the CORBA Service is running (refer to "A.1 config"). If logging is set to "no", this value has no meaning.
select cache obj	- -	Specifies the objects to be cached when the Interface Repository is started. The objects to be cached should be listed by repository Id in a text file, whose file name is specified by the full path. If no file name is specified, all registered objects are cached. Refer to the specification method description below for examples of creating the files, and additional notes.
sync	no yes, no	Specifies whether to enable synchronous mode. - yes: Enable synchronous mode. - no: Disable synchronous mode. If this parameter is set to "no", the throughput of the renewal management in the Interface Repository improves. This is because data access is asynchronous (that is, one write is not synchronized with another) and therefore, if during renewal the database goes down, it cannot be detected. If this parameter is set to "yes", the system will run in sync mode to guarantee that Write operations are carried out one transaction at a time. This option should be set when reliability in the system is required. (Destruction of a database can be recognized.) In synchronous mode, the throughput of the renewal management declines and therefore a timeout may occur. The timeout period is set by period_receive_timeout of config.

Specifying Objects to be Cached

Restricting the objects that are cached when the Interface Repository is started improves startup performance when large numbers of objects are registered in the Interface Repository.

However, when objects are specified for caching, the reference performance for uncached objects is degraded, and this affects operations.

In addition, if the Interface Repository is started with objects specified for caching, no further objects can be added to or updated (IDLc, tdc, odimportir) in the Interface Repository. This should only be done during production runs (when no additions to or updating of the Interface Repository is done).

To cache objects, specify the Interface Repository ID of the objects to be cached in a text file.

Only the ModuleDef object or InterfaceDef object that directly includes the Repository object (route object) need be specified for the repository ID. All objects included in the specified object are then cached.

When an association is established with another module through inheritance or scope reference, that module must also be specified for caching.

Refer to information on Interface Repository Service Programming in the Distributed Application Development Guide (CORBA Service Edition) for details of the types of objects managed by the Interface Repository service and the inclusion/inheritance of Interface Repository objects. (The Distributed Application Development Guide (CORBA Service Edition) is only provided with Enterprise Edition products.)

The odlistdir command can be used to display the inclusive relationships of objects registered in an Interface Repository.

The following are examples of coding files to specify objects to be cached:

```
IDL:testmodule1:1.0
IDL:testmodule2:1.0
IDL:testmodule3:1.0
```

Enter only one repository ID for an object to be cached per line. Comments cannot be used.

Notes

- When a value is changed, the new value does not take effect until the next time the Naming Service is started.
- When setting up the environment using the ismodifyservice command, if "auto backup" has been set to "yes", a backup will be taken when the database is empty. A database which has been backed up in this state cannot be used.
- The auto backup function takes a backup when the interface has just been started up. This means that interface definition information updated after the interface is started up will not be backed up. If you need to back up information updated after the startup, restart Interstage (Interface Repository). Alternatively, back up using the odbackupsys command.
- The odbackupsys command cannot be used to back up files that specify cached objects. Back up these files with the copy commands.
- If the database administrator of the interface repository is other than the default of root, and "iss_use=yes" is set in the irconfig file, log information of the interface repository (database access function) is not collected even if the log information collection function is enabled ("logging=yes" in the irconfig file) (Log information of the cache server is collected).

  If "iss_use=yes" is set in the irconfig file, the *irlogdump* command (output/control of log information) must be executed with the administrator authority (root).

Appendix B Component Transaction Service Environment Definition

This appendix explains the Component Transaction Service environment definition file.

This definition file is supported with the following products:

- Interstage Application Server Enterprise Edition

Windows32/64

This definition file is C:\INTERSTAGE\etc\sysdef.

Solaris32/64

This definition file is /var/opt/FSUNtd/etc/sysdef.

Linux32/64

This definition file is /var/opt/FJSVtd/etc/sysdef.

This definition file can be modified only while the Component Transaction Service is stopped.

The environment definition of the Component Transaction Service is described in the following format.

Note

Upon completion of structuring the application environment, creating a backup copy of the resources is recommended for the possible crash of resources.

For the procedure of creating a backup copy, see Maintenance (Backup of Resources) in the Operator's Guide.

B.1 Syntax

The standard file description format is set up according the following structure. If a syntax error occurs in the file, the contents of the file are ignored.

- Statement
- Section
- Comment line
- Blank line

B.1.1 Statement

A statement is a line for setting the information, and is specified in the following format:

Keyword: Setup information \n

A statement consists of a keyword, a colon :, and the setup information. The statement description rules are as follows:

- To omit a statement, either delete the applicable statement, or omit only the setup information.
- A statement and a comment cannot exist on the same line.

The components of a statement are detailed as follows:

Keyword

The keyword is a registered identifier, and conforms to the following rules:

- Keywords consist of alphanumeric characters and spaces, and must begin with an alphanumeric character.
- Keywords are not case sensitive.

- Multiple spaces in succession a keyword are treated as a single space.
- Spaces or tabs at the beginning of the line are ignored.

Colon

The colon separates the keyword from the setup information, and conforms to the following rules:

- An en-size colon must be specified.
- Spaces or tabs before or after the colon are ignored.

Setup Information

The setup information defines the settings for the keyword, and conforms to the following rules:

- Any colons : included in the setup information are treated as literal characters.
- Setup information ends with a space, tab, line feed "\n", or EOF.
- Setup information is case sensitive.
- Only one character string can be specified in setup information.
- Spaces or tabs must be enclosed in double quotation marks.
- There can be only one setup information item per statement. To include more than one setup information item, write a new statement.
- All statements are analyzed in the same way.

Examples

The following examples show valid statements:

```
keyword:Information\n
KEYWORD:Information\n
KeyWord:Information\n
Keyword:"Information Area"\n
```

The following examples show invalid statements causing syntax errors:

```
# Two types of setup information are specified.
Keyword: Information Area\n
# A statement and a comment are written on the same line.
Keyword: Information #This is a statement\n
# The ending double quotation mark is missing.
Keyword: "START  Information.\n
# Keyword and setup information are specified on two lines.
Keyword: "START  Information.\n
Information END" \n
```

Specifying an unregistered keyword also causes a syntax error.

B.1.2 Section

A section is a group of statements described in the following format:

```
section-name \n
keyword:  setup information \n
keyword:  setup information \n
```

A section is comprised of a "section name" and multiple statements, and conforms to the following rules:

- A section begins with the "section name" identifier, and ends when a new section is defined, or with an EOF.
- To omit a section, delete the entire section, or make it a comment.
- A section cannot contain the "section name" identifier only.

- Nothing else can exist on the same line as the "section name" identifier, including comments.
- "Section name" must be enclosed in square brackets .
- The section name consists of alphanumeric characters and spaces, and must begin with an alphanumeric character.
- The section name is not case sensitive.

The following example shows a valid section:

```
Section\n
Keyword1: Information\n
Keywordn: Information\n
```

The following examples show invalid sections, causing syntax errors:

# section-name and a comment are written on the same line.

```
Section  # This is a Section\n
Keyword: Information\n
```

# The Section end bracket is missing.

```
Section\n
Keyword: Information.\n
```

Specifying an unregistered section name also causes a syntax error.

B.1.3 Comment Line

A comment line is used by the programmer to insert non-executable yet useful information, and is described in the following format:

```
# comment \n
```

A comment is preceded by the single byte hash character (#). All text following the # is ignored.

B.1.4 Blank Line

A blank line can be inserted, but is ignored during analysis.

B.2 Control Statement of System Environment Definition

This section details the control statements that can be specified in the system environment definition file.

B.2.1 [SYSTEM ENVIRONMENT] Section

System Scale:

System scale

Using Session Information Management Object:

Presence of use of SMO

Name of Session Information Management Object:

Name of SMO

Using Authentication Server Object: Windows32/64 Solaris32/64

Presence specification of start of attestation object

Name of Authentication Server Object: Windows32/64 Solaris32/64

Name of attestation object

Authentication Server Object Trace: Windows32/64 Solaris32/64

Trace collection existence specification of attestation object

Host of InfoDirectory: Windows32/64 Solaris32/64

Host name of InfoDirectory

Port of InfoDirectory: Windows32/64 Solaris32/64

Port number of InfoDirectory

Number of Number of Maximum WRAPPER Hold Session Windows32/64 Solaris32/64

system maximum reservation sessions

Number of Communication Buffer

Number of communication buffers of the transaction application

Using Interface Check

Specification of whether the Interface Check function is used or not

IP version Solaris32/64

The version of the network to be used

The following parameters are filled in on each line.

System Scale

Please specify the scale of the system.

If the system is initialized by using the Interstage integration command, an appropriate system scale is set, and you do not have to modify the scale.

Determine the system scale based on the number of the clients connected. The following values can be specified according to the system scale.

- Windows32/64

  "small"Small scale (Number of clients connected: 1 to 5)

  "moderate"Inside scale (Number of clients connected: 6 to 10)

  "large"Large scale (Number of clients connected: 11 to 50)

  "super"Super-large scale (Number of clients connected: 51 to 100)

- Solaris32/64 Linux32/64

  "small"Small scale (Number of clients connected: 1 to 50)

  "moderate"Inside scale (Number of clients connected: 51 to 100)

  "large"Large scale (Number of clients connected: 101 to 500)

  "super"Super-large scale (Number of clients connected: 501 to 1000)

Using Session Information Management Object

Specifies whether or not the session information management function is used.

- "YES" : SMO is used.
- "NO" : SMO is not used.

When this statement is omitted, "NO" is set by default.

Name of Session Information Management Object

Specify the SMO name when registering the Naming Service.

A value of up to 256 bytes can be specified at the OD_or_adm command -n option, however, the naming context cannot be specified.

Default value is "ISTD::SMO".

If NO is specified in the statement "Using Session Information Management Object", this statement is ignored.

Using Authentication Server Object

Windows32/64 Solaris32/64

Specifies whether or not Authentication Object starts up.

- "YES" : The attestation object is started.
- "NO" : The attestation object is not started.

If this statement is omitted, "NO" is set by default.

Name of Authentication Server Object

Windows32/64 Solaris32/64

The name when naming service of the attestation object is registered with the server that starts the attestation object is specified.

The value can be specified is an alphanumeric character within 255 bytes. In the server on which the Authentication Object runs, specify the name that was specified when the Authentication Object Naming Service was registered.

Specify the name in up to 255 alphanumeric characters. You cannot specify a naming context.

If this statement is omitted, "ISTD::ASO" is set.

This statement is ignored if "NO" is set for the Using Authentication Server Object statement. The default value is "ISTD::ASO"

When "NO" is specified for the Using Authentication Server Object statement, this statement is disregarded.

Authentication Server Object Trace

Windows32/64 Solaris32/64

Determines whether the Authentication Server Object is traced.

- "ENABLE" : The trace is gathered.
- "DISABLE" : The trace is not gathered.

When this statement is omitted, "DISABLE" is set as a default value.

When "DISABLE" is specified for the Using Authentication Server Object statement, this statement is disregarded.

Host of InfoDirectory

Windows32/64 Solaris32/64

Host name of InfoDirectory

The host name of the server of InfoDirectory is specified in the server that executes the server or the access control by which the attestation object is started.

The value that can be specified is limited to 255 bytes.

The default value is assumed to be host's host name.

Port of InfoDirectory

Windows32/64 Solaris32/64

Port number of InfoDirectory

The port number of InfoDirectory is specified in the server that starts the attestation object and the server which executes the access control.

The value that can be specified is 1-65535.

The default value is 389.

Number of Maximum WRAPPER Hold Session

Windows32/64 Solaris32/64

The number of session reservations in the entire system maximum reservation session number system is specified.

When 0 is specified, the session is not controlled.

This definition becomes effective when the PSYS Name statement and the Number of Maximum Session statement specified in the WRAPPER section are defined.

The value that can be specified is 0-1000.

The default value is 0.

Number of Communication Buffer: Number of Communication Buffers of Transaction Application

Represents the number of communication buffers that are used for communication of the transaction application. The size of the communication buffer is given as a product of the number of communication buffers specified in this statement and 4096 bytes.

A value in the range of 500 to 10000 can be specified.

Multiply the number of communications happening at a single time by the number of clients connected concurrently, then estimate the number of communication buffers based on the product.

Specify the sufficient number of the communication buffers.

The following default values are set based on the system scale:

Small : 500

moderate : 1000

large : 1500

super : 2000

Using Interface Check: Specification of Whether the Interface Check Function is Used or Not

Specifies whether or not the Interface Check function is used.

- "YES": The Interface Check function is used.
- "NO": The Interface Check function is not used.

When this statement is omitted, "NO" is set.

IP Version: The Version of the Network to be Used

Solaris32/64

The version of the network to be used is specified.

- "v6": An IPv6 network is used.
- "v4": IPv4 conventional network is used.

When this statement is omitted, "v4" is set.

B.2.2 [WRAPPER] Section

Windows32/64 Solaris32/64

Enter the [WRAPPER] section in the following format:

[WRAPPER]

PSYS Name:

Communication path for load control name

Number of Maximum Session:

Number of communication sessions at maximum

PSYS Name:

Communication path for load control name

The DPCF communication path name which the load control targets is specified in the alphanumeric character of eight characters or less.

Number of Maximum Session:

Maximum number of simultaneous communication sessions.

The maximum values of the number of sessions which can be the reservation in the DPCF communication path set by the PSYS Name statement it are specified.

The value that can be specified is 0-1000. If a session exceeds this number, it is subject to load restrictions.

Notes

- Define all of each statement of the SYSTEM ENVIRONMENT section and the WRAPPER section when you use the load control function.
- The PSYS Name statement and the Number of Maximum Session statement of the WRAPPER section are specified by the pair.
- When the load is controlled to two or more DPCF communication path, the PSYS Name statement and the Number of Maximum Session statement are specified for two or more WRAPPER sections by the pair.
- Specify the value specified for the Number of Maximum Session statement when the DPCF communication path name which uses the OSI-TP communication function or the TCP/IP communication function for the PSYS Name statement is specified below the number of priority conversation connections specified by the IDCM network definition. When bigger value than the number of priority conversation connections is specified, the load might not be controlled.

Appendix C Database Linkage Service Environment Definition Windows32/64 Solaris32 Linux32/64

The Database Linkage Service environment definition is specified in the following format:

<Parameter> = <Setup value>

C.1 config File Windows32/64 Solaris32 Linux32/64

Overview

The config file is the definition file maintaining the information that applies to the OTS system at the startup of the system.

Note

- To reflect modifications to the config file in the OTS system, restart the OTS system.
- To reflect modifications to the config file in the Interstage Management Console, restart the Interstage Management Console.

File Name

The config file is placed in the following location.

Windows32/64

(Default installation path)

```
C:\Interstage\ots\etc\config
```

Solaris32

(Default installation path)

```
/opt/FSUNots/etc/config
```

Linux32/64

```
/opt/FJSVots/etc/config
```

File Contents

Format

```
Key name = value set
```

Example

```
OBSERVE_CYCLE_TIME=6 (*1)
TRAN_TIME_OUT=300
2PC_TIME_OUT=60
COM_RETRY_TIME=2 (*1)
COM_RETRY_MAX=3 (*1)
RECOVER_RETRY_TIME=30 (*1)
```

```
RECOVER_RETRY_MAX=60 (*1)
RESOURCE_TRANMAX=5
OTS_TRACE_SIZE=4096 (*1)
RESOURCE_TRACE_SIZE=4096 (*1)
RECOVERY_TRACE_SIZE=4096 (*1)
OBSERVE_TRACE_SIZE=4096 (*1)
DATABASE_RETRY_TIME=5 (*1)
DATABASE_RETRY_MAX=5 (*1)
MEM_RETRY_TIME=5 (*1)
MEM_RETRY_MAX=5 (*1)
RSCSTOP_CHECK_COUNT=100 (*1)
OTS_VERSION=5 (*1)
JTS_VERSION=5 (*1)
TRACE_MODE=1
TRACE_LEVEL=1
JAVA_VERSION=14
PATH=C:\Interstage\JDK6\bin\java.exe (For Windows(R))
PATH=/opt/FJSVawjbk/jdk6/bin/java (For Solaris/Linux)
```

*1 This item is not defined in the config file that is created at the time of the Interstage installation. When this item is specified, the value will be enabled. We recommend that the default value not be changed.

Notes

- For a description on timeout, refer to the OLTP Server User's Guide.
- All the items of a config file can be omitted. A default value for each item becomes effective if it is omitted.

Key

- OBSERVE_CYCLE_TIME

  Specify the Database Linkage Service observe cycle time in seconds. If this time is less then observe cycle of the system increases which leads to low performance. On the other hand, if the observe cycle time is greater, abnormality detection is delayed. Set the parameters by taking these two points into consideration.

  A value from 1 to 60 can be specified. The default value is 5.

- TRAN_TIME_OUT

  Specify the Database Linkage Service transaction timeout from begin to commit, in seconds.

  The default value is 300.

  If the timeout is specified in the application set_timeout method, the application's value will override TRAN_TIME_OUT. Specify a value from 1 to the maximum value permitted for 'long'.

- 2PC_TIME_OUT

  Specify the commit timeout for the resource manager, between phase 1 and phase 2 for the 2PC 2-phase commit of the Database Linkage Service transaction, in seconds.

  Specify a value from 1 to the maximum value permitted for 'long'. The default value is 60.

  #### Note

  If the CORBA Service client side non-communication monitoring time (the value set for "period_client_idle_con_timeout" in the CORBA Service operating environment file raised to the power of 5) is not 0, the value specified for this parameter must be lower than that value.

- COM_RETRY_TIME

  If a communication abnormality occurs during the transaction process, set the retry time for that communication.

  A value from 1 to 600 can be specified. The default value is 2.

- COM_RETRY_MAX

  If a communication abnormality occurs during the transaction process, set the maximum retry time for that communication.

Specify a value from 1 to the maximum value permitted for 'long'. The default value is 3.

- RECOVER_RETRY_TIME

  If a communication abnormality occurs in the Database Linkage Service recovery process, set the retry time for that communication.

  A value from 1 to 600 can be specified. The default value is 30.

- RECOVER_RETRY_MAX

  If a communication abnormality occurs during the Database Linkage Service recovery process, set the maximum retry time for that communication.

  Specify a value from 1 to the maximum value permitted for 'long'. The default value is 60.

- RESOURCE_TRANMAX

  Specify the maximum multiple level of transactions for each resource control program.

  Specify a value from 1 to the maximum value permitted for 'long'. The default value is 5.

  Note

  Set up the degree of thread multiplex of an OTS system, and the degree of maximum multiplex of the transaction of 1 resource control program to maintain the following relations.

```
the degree of thread multiplex of an OTS system =< the degree of maximum
multiplex of the transaction of 1 resource control program
```

- OTS_TRACE_SIZE

  Specify the Database Linkage Service trace log size, in Kb.

  A value from 128 to the maximum value permitted for 'long' can be specified. The default value is 4096.

- RESOURCE_TRACE_SIZE

  Specify the resource manager trace log size, in Kb.

  A value from 128 to the maximum value permitted for 'long' can be specified. The default value is 4096.

- RECOVERY_TRACE_SIZE

  Specify the recovery process trace log size, in Kb.

  A value from 128 to the maximum value permitted for 'long' can be specified. The default value is 4096.

- OBSERVE_TRACE_SIZE

  Specify the observe process trace log size, in Kb.

  A value from 128 to the maximum value permitted for 'long' can be specified. The default value is 4096.

- DATABASE_RETRY_TIME

  Specify the retry time if recoverable errors, such as insufficient resources occur while accessing the Database Linkage Service database system.

  A value from 1 to 600 can be specified. The default value is 5.

- DATABASE_RETRY_MAX

  Specify the maximum retry time if recoverable errors, such as insufficient resources occur while accessing the Database Linkage Service database system.

  A value from 128 to the maximum value permitted for 'long' can be specified. The default value is 5.

- MEM_RETRY_TIME

  Specify the retry time in seconds if recoverable errors, such as insufficient resources occur while processing the Database Linkage Service.

  A value from 1 to 600 can be specified. The default value is 5.

- MEM_RETRY_MAX

  Specify the maximum retry time if recoverable errors, such as insufficient resources occur while processing the Database Linkage Service.

  Specify a value from 1 to the maximum value permitted for 'long'. The default value is 5.

- RSCSTOP_CHECK_COUNT

  The resource control program is usually stopped in transaction processing and the meeting frequency by which the transaction completion is matched to the watch synchronization of OBSERVE_CYCLE_TIME is specified.

  When the completion of the transaction during the OBSERVE_CYCLE_TIME*RSCSTOP_CHECK_COUNT second cannot be completed in the meeting time, the stop of the resource control program is usually switched from the stop to the compulsion stop.

  The range that can be pointed out can specify the value that can be specified with 1 to the maximum value for 'long'.

  When this parameter is omitted, 100 is set.

- OTS_VERSION

  Specify the version of the OTS. This value is usually not changed.

  When this parameter is omitted, 5 is set.

- JTS_VERSION

  Specify the version of the JTS. This value is usually not changed.

  When this parameter is omitted, 5 is set.

- JAVA_VERSION: Version of JDK/JRE

  This is the Java version used by the JTS resource management program.

  The default value of this parameter is 14. Do not change this value.

- PATH: Path of JDK/JRE

  Specify the full path to the java command which the resource control program for JTS uses. Specify the path containing the java executable.

  The initial value is the path for the JDK/JRE 6 version. This value is usually not changed.

  Notes

  - When the resource control program for JTS is used, specifying this parameter is mandatory.
  - Specify the path for JDK/JRE bundled with Interstage Application Server.

- TRACE_MODE

  The output form of the trace output in the environment using JTS is specified.

  The value of 1 to 3 can be specified. The meaning is as follows.

  1 : Only when abnormalities occur, a trace file is output to the installation directory / var subordinate of OTS. Usually, please choose 1.

  2 : Regardless of the state of a system, trace is always output to the installation directory / var subordinate of OTS. Since it is always output, be careful of file size.

  3 : A file output is not carried out. However, a trace file can be output by using the *otsgetdump* command.

- TRACE_LEVEL

  The mode of the trace output in the environment using JTS is specified.

  The numbers from 1 to 5 can be put in and such fine trace can be output that a number is large. Usually, 1 is specified at the time of employment. Since performance is also influenced, it does not usually change.

  Moreover, trace is not output when 0 is specified This is the default when the a number outside the range is specified.

C.2 Setup Information File Windows32/64 Solaris32 Linux32/64

Overview

The setup information file is specified when the *otssetup* command sets the OTS system operating environment. If the setup processing is done using the Interstage integration command (isinit command), the Interstage operating environment definition needs to customize the definition.

The setup information file must be created before the *otssetup* command is entered. Save the setup information file once created, because it can be reused when performing the setup processing next time.

File Contents

Format

```
Parameter name = value set
```

Example

```
MODE=SYS
LOGFILE=c:\ots\logfile (For Windows(R))
LOGFILE=/dev/rdsk/c0t0d0s0 (For Solaris)
LOGFILE=/dev/raw/raw1 (For Linux)
TRANMAX=10
PARTICIPATE=4
OTS_FACT_THR_CONC=5
OTS_RECV_THR_CONC=2
JTS_RMP_PROC_CONC=5
JTS_RMP_THR_CONC=16
HOST=otshost
PORT=8002
LOCALE=EUC
```

Note

All of the items in bold are mandatory. Any or all of the rest can be omitted. A default value becomes effective when an item is omitted.

C.2.1 MODE

Either the host in which the OTS system works, or the host in which only the Resource Management Program works, can be specified.

- SYS: The host in which the OTS system and the Resource Management Program work.
- RMP: The host in which only the Resource Management Program works.

Specify both "SYS" and "RMP" in upper case letters.

When "SYS" is specified, setup of the OTS system operating environment and the Resource Management Program operating environment will be performed, and the system log file will be created. When "RMP" is specified, setup of the Resource Management Program operating environment only will be performed. The OTS system cannot be activated in an environment where setup was performed by specifying "RMP".

Corresponds to the "OTS Setup mode" of the Interstage operating environment definition file.

When "RMP" is specified, it is necessary to refer to the Naming Service of the host in which the OTS system works in order to correctly operate the Resource Management Program. Carry out the setup using either of the following methods:

- Specify "RMP" and at the same time specify "HOST", "PORT" , and "LOCALE" and carry out setup. In this case, the Naming Service of not the OTS system, but that of the host in which "RMP" was specified will be used. It is then possible to operate separately both the Naming Service of the host in which "RMP" was specified, and that of the host in which the OTS system works.

- After initializing Interstage using *isinit* type3 (Setup Naming Service of the host in which NS Use, NS Host, NS Port Number, and NS work), specify "RMP", and set up using the *otssetup* command. In this case, the Naming Service of the host in which the OTS system works, and that of the host in which "RMP" is specified, will be shared.

Note

- Sharing the Naming Service of hosts in which "SYS" is specified is not possible. For each Naming Service, one host must exist in which "SYS" is specified. When "RMP" is specified, it is possible to share the Naming Service among multiple hosts.
- When "RMP" is specified, it is not possible to start the OTS system using the *otsstart* command. Starting the Resource Management Program, using the *otsstartrsc* command, can only be performed.

C.2.2 LOGFILE

Specify the path to the OTS system log file.

Windows32/64

Specify using the absolute path including the drive name. Character strings must be specified. However, use of both upper and lower case characters is possible.

Solaris32 Linux32/64

Specify the character strings excluding spaces beginning with "/".

This item is valid only when "MODE" is "SYS".

The maximum length is 255 characters.

Corresponds to the "OTS path for system log" of the Interstage operating environment definition file.

Point Linux32/64

The following shows how to create a raw device:

1. Using the parted or fdisk command of the operating system, create the raw device partition.
2. Identify the udev block device name that corresponds to the disk partition.

   An example of the parted command execution is shown below:

   For RHEL5

```
# parted /dev/sda
(parted) p
      :

Number Start  End     Size    Type      File system  Flags
1     32.3kB  107MB   107MB   primary ext3          boot
2     107MB   9656MB  9550MB  primary               lvm
3     9656MB  10.7GB  1078MB  primary               lvm

(parted) q
# udevinfo -q path -n /dev/sda3
/block/sda/sda3
# udevinfo -q env -p /block/sda/sda3 | grep ID_PATH
ID_PATH=pci-0000:00:10.0-scsi-0:0:0:0
```

   For RHEL6

```
# parted /dev/sda
(parted) p
      :
```

```
Number Start   End     Size    Type      File system  Flags
1      1049kB  211MB   210MB   primary   ext4         boot
2      211MB   32.4GB  32.2GB  primary   ext4
       :
8      77.5GB  78.5GB  974MB   logical

(parted) q
# udevadm info --query=path --name=/dev/sda8
/devices/pci0000:00/0000:00:1f.2/host0/target0:0:0/0:0:0:0/block/sda/sda8
# udevadm info --query=property --path=/devices/pci0000:00/0000:00:1f.2/host0/
target0:0:0/0:0:0:0/block/sda/sda8 | grep ID_PATH
ID_PATH=pci-0000:00:1f.2-scsi-0:0:0:0
```

3. Edit the udev settings file (/etc/udev/rules.d/60-raw.rules) and bind the created partition.

   For RHEL5

```
ACTION=="add", KERNEL=="sda3", ENV{ID_PATH}=="pci-0000:00:10.0-scsi-0:0:0:0", RUN+="/bin/
raw /dev/raw/raw1 %N"
```

   For RHEL6

```
ACTION=="add", KERNEL=="sda8", ENV{ID_PATH}=="pci-0000:00:1f.2-scsi-0:0:0:0", RUN+="/bin/
raw /dev/raw/raw1 %N"
```

4. So that the raw device access privileges will be set correctly using udev, if necessary edit the permissions rules file in /etc/udev/rules.d/ that will be added.

Notes

- Specify the partition for the block device that will bind the raw device. Hard disk devices without a partition ID (such as /dev/sdg) contain disk labels (partition tables), and therefore should not be used as raw devices.
- The entry that is specified in the setup information file log file must be the device name that was bound to the character device using the raw command.

C.2.3 TRANMAX

Specify the maximum number of the transaction.

It is necessary to surely specify. Moreover, when "RMP" is set as MODE, set up the same value as the OTS system (system by which MODE is "SYS") which cooperates.

Windows32/64

A value from 1 to 256 can be specified.

Solaris32 Linux32/64

A value from 1 to 1024 can be specified.

Corresponds to the "OTS maximum Transaction" of the Interstage operating environment definition file.

C.2.4 PARTICIPATE

Specify the maximum number of resources permitted to participate in 1 transaction.

Valid only when "SYS" is specified in "MODE".

An integer value in the range of 2 to 32 can be specified. The default value is 4.

Corresponds to the "OTS Participate" of the Interstage operating environment definition file.

C.2.5 OTS_FACT_THR_CONC

Specify the thread concurrency of the OTS system.

Valid only when "SYS" was specified in "MODE".

A value from 1 to 31 can be specified.

The default value is 5.

As many Concurrent interfaces of begin, commit, and rollback and UserTransaction interfaces as specified can be concurrently operated.

Corresponds to the "OTS Multiple degree" of the Interstage operating environment definition file.

When maximum is exceeded, a warning message (ots9013) is output and 31 is set up automatically.

Note

Since the degree of thread multiplex of an OTS system is tuned up so that a transaction processing performance may be pulled out to the maximum extent, it does not need to change a default value.

When you change, set up to maintain the following relations.

```
the degree of thread multiplex of an OTS system =< the degree of multiplex
 of a resource control program (*1)
the degree of thread multiplex of an OTS system =< the degree of maximum
multiplex of the transaction of 1 resource control program
```

*1 Ask for the degree of multiplex in the resource control program for JTS by the following formulas.

```
the degree of process multiplex of the resource control program for JTS :
JTS_RMP_PROC_CONC * the degree of thread multiplex of the resource control
program for JTS : JTS_RMP_THR_CONC
```

C.2.6 OTS_RECV_THR_CONC

Specify the thread concurrency of the recovery program.

Valid only when "SYS" is specified in "MODE".

A value from 1 to 214748367 can be specified.

The default value is 2. Recovery up to the number can be operated simultaneously.

Corresponds to the "OTS Recovery" of the Interstage operating environment definition file.

C.2.7 JTS_RMP_PROC_CONC

Specify the process concurrency of the resource control program for JTS.

A value from 1 to 32 can be specified.

The default value is 5. It is recommended that you specify the number of the resource (databases or resource adapter etc.) to be used If the number is less than or equal to 5, modification of this setting is not necessary.

Corresponds to the "OTS JTSs RMP Multiple degree of Process" of the Interstage operating environment definition file.

When maximum is exceeded, a warning message (ots9013) is output and 31 is set up automatically.

Note

Since the degree of thread multiplex of an OTS system is tuned up so that a transaction processing performance may be pulled out to the maximum extent, it does not need to change a default value.

When you change, set up to maintain the following relations.

```
the degree of thread multiplex of an OTS system =< the degree of multiplex
 of a resource control program (*1)
the degree of thread multiplex of an OTS system =< the degree of maximum
multiplex of the transaction of 1 resource control program
```

*1 Ask for the degree of multiplex in the resource control program for JTS by the following formulas.

```
the degree of process multiplex of the resource control program for JTS :
JTS_RMP_PROC_CONC * the degree of thread multiplex of the resource control
program for JTS : JTS_RMP_THR_CONC
```

C.2.8 JTS_RMP_THR_CONC

Specify the thread concurrency of the resource control program for JTS.

A value from 1 to 2147483647 can be specified.

The default value is 16. Modification of this setting is not usually necessary.

Corresponds to the "OTS JTSs RMP Multiple degree of Thread" of the Interstage operating environment definition file.

Note

Since the degree of thread multiplex of an OTS system is tuned up so that a transaction processing performance may be pulled out to the maximum extent, it does not need to change a default value.

When you change, set up to maintain the following relations.

```
the degree of thread multiplex of an OTS system =< the degree of multiplex
of a resource control program (*1)
the degree of thread multiplex of an OTS system =< the degree of maximum
multiplex of the transaction of 1 resource control program
```

*1 Ask for the degree of multiplex in the resource control program for JTS by the following formulas.

```
the degree of process multiplex of the resource control program for JTS :
JTS_RMP_PROC_CONC * the degree of thread multiplex of the resource control
program for JTS : JTS_RMP_THR_CONC
```

C.2.9 HOST

Specify the name of the host in which the OTS system operates.

Valid only when "RMS" is specified in MODE.

Set up using a character string of a maximum of 64 characters, starting with an alphabetic character, consisting of alphanumerics, minus symbols, and periods. The string must not end with a minus symbol or period.

This statement can be omitted. When specifying this statement, specify also PORT and LOCALE concurrently.

Refer to C.2.1 MODE for information on how to apply this statement.

Corresponds to the "OTS Host" of the Interstage operating environment definition file.

Note

Do not use this statement if "type3" has been selected in the *isinit* command.

C.2.10 PORT

Specify the port number of CORBA Service of the node in which the OTS system works.

Valid only when "RMP" was specified in MODE.

A value from 1024 to 65535 can be specified.

This statement can be omitted. When this statement is specified, specify also HOST and LOCALE at the same time.

Refer to C.2.1 MODE for information on how to apply this statement.

Corresponds to the "OTS Port" of the Interstage operating environment definition file.

Note

Do not use this statement if "type3" has been selected in the *isinit* command.

C.3 RMP Property Windows32/64 Solaris32 Linux32/64

Overview

The RMP property file is a property file for the resource control program for JTS.

File Name

The RMP property file is placed in the following location.

Windows32/64

(Default installation path)

```
C:\Interstage\ots\etc\RMP.properties
```

Solaris32

(Default installation path)

```
/opt/FSUNots/etc/RMP.properties
```

Linux32/64

```
/opt/FJSVots/etc/RMP.properties
```

File Contents

Format

```
Parameter name = value set
```

Parameter

RecoveryTarget

Specify the resource definition name for a recovery target that is to be recovered at JTS startup. If the recovery target is not specified, recovery processing is not executed even after re-starting the JTS resource control program. When there is more than one recovery target, insert a blank between each target.

When there are three recovery targets:

```
RecoveryTarget=Oracle_resource1 Oracle_resource2 Oracle_resource3
```

- JavaPath

  Description is added when required. Usually, do not specify. When specified, there is no guarantee of operation.

- JavaCommandOption

  Description is added when required. Usually, do not specify. When specified, there is no guarantee of operation.

- ClassPath

  Windows32/64

  Description is added when required. Usually, do not specify. When specified, there is no guarantee of operation.

  Solaris32 Linux32/64

  When you use a cluster service function, set up the path to the class library which is needed in order to cooperate with a resource.

- Librarypath

  Specify the library path required for resource linkage.

  Windows32/64

  This is added to the PATH environment variable of the JTS resource management program.

  Solaris32 Linux32/64

  This is added to the LD_LIBRARY_PATH environment variable of the JTS resource management program.

- Environ

  Specify the environment variable required for resource linkage.

  When the ORACLE_HOME environment variable is specified:

  Windows32/64

  ```
  Environ ORACLE_HOME=C:\app\user\product\11.2.0\db
  ```

  Solaris32 Linux32/64

  ```
  Environ ORACLE_HOME=/u01/app/oracle/product/11.2.0/dbhome
  ```

C.4 Resource Definition File Windows32/64 Solaris32 Linux32/64

Overview

It is the file which defines the information for connecting with the resources (a database, resource adapter, etc.) with which OTS and JTS cooperate. It registers per resource using the otssetrsc command.

File Contents

Format

```
key name = value set
```

Example

Resource definition file for OTS

```
# Environment variable
ENVIRON ORACLE_SID=orac
ENVIRON ORACLE_HOME=/opt/oracle (For Solaris/Linux)
ENVIRON LD_LIBRARY_PATH=/opt/oracle/lib (For Solaris/Linux)

# Database system name to be used An OPENINFO character sequence, CLOSEINFO character sequence
NAME=oracle_rmp_thread
RMNAME=Oracle_XA
OPENINFO=Oracle_XA+Acc=p/system/manager+SesTm=0+Threads=true
CLOSEINFO=
THREADS=TRUE (For Solaris/Linux)
```

Resource definition file for JTS

```
# database1
 name=xads1
 rscType=JTS
 type=JDBC
 lookUpName=jdbc/XADataSource
 initialContextFactory=com.sun.jndi.fscontext.RefFSContextFactory
 providerURL=file:/tmp/JNDI
 user=dbuser
 password=dbpass
 logfileDir=c:\interstage\ots\var (For Windows(R))
 logfileDir=/opt/FSUNots/var (For Solaris)
 logfileDir=/opt/FJSVots/var (For Linux)
```

Note

Although the name of a key has the difference between a capital letter and a small letter by OTS and JTS, it has the same meaning.

Key

- ENVIRON

  The environment variable env passed to the database library which operates within the same process as a resource control program or a resource control program is set as a value data. It is omissible.

  Solaris32 Linux32/64

  Please specify the same environment variable as the environment variable to the database specified at the time of starting of the server application which uses a resource control program.

  Moreover, $ specification cannot be carried out to a resource definition file like

  ```
  LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/opt/oracle/lib.
  ```

  When the database to be used is Symfoware/RDB, specify the path name of the mandatory Symfoware/RDB library for environment variable LD_LIBRARY_PATH.

- NAME

  When registered by the *otssetrsc* command, it is registered as a resource definition name indicated here. It becomes possible to treat all the resource definition files registered at once by the resource definition name. It is necessary to describe a resource definition name within 32 characters.

  Since "JTSRMP" is a reservation word, it cannot be used for a resource definition name (it cannot be used even if it makes a part or all into a small letter).

Specify the "definition name" of the resource used as the candidate for connection registered by J2EE resource access definition in the resource definition file for JTS.

It is not omissible.

- RMNAME

  Windows32/64

  Specify the system name of a database as system_name.

  - In Oracle, it is "Oracle_XA".
  - In Symfoware/RDB, it is "RDBII".

  Windows32/64

  - In SQL Server, it is "MS_SQL_Server".

  Windows32/64 Solaris32

  - In MQD, it is "XA_MQD".

- OPENINFO

  When it opens the database which the vender of a database opens to open_data, a required open character sequence is specified.

  open_data is specified by 256 or less characters.

  Refer to the manual of each database about the contents to specify.

  Note

  - If there is no access authority over each database, starting of a resource control program will fail in the user name specified to be OPENINFO. Please refer to the manual of each database about required authority.

  Solaris32 Linux32/64

  - Since the starting of a resource control program may incorrectly-operate when the type in process mode and thread mode is different at the time of resource control program creation and operation (thread specification in a resource definition file), please be sure to apply in accordance with a type.

- CLOSEINFO

  When it closes the database which the vender of a database opens to close_data, a required close character sequence is specified.

  open_data is specified by 256 or less characters.

  Refer to the manual of each database about the contents to specify.

- THREADS Solaris32 Linux32/64

  A resource control program specifies process mode (FALSE) and thread mode (TRUE). It is set to FALSE (process mode) when it omits.

- OTS_RMP_PROC_CONC

  The multiplex number of the resource control program for OTS is specified. When it omits, it becomes the degree 5 of multiplex. Usually, it is not necessary to change. When specifying, specifying in 1-31 is possible.

  When maximum is exceeded, a warning message (ots9017) is outputted and 31 is set up automatically.

  Note

  Since the degree of multiplex of a resource control program is tuned up so that a transaction processing performance may be pulled out to the maximum extent, it does not need to change a default value.

  When you change, please set up the relation between the degree of thread multiplex of an OTS system, and the degree of multiplex of a resource control program as follows.

```
the degree of thread multiplex of an OTS system  =< the degree of multiplex
of a resource control program
```

- RSCTYPE

  The classification of a resource definition file is specified. When using OTS, "OTS" is specified, and "JTS" is specified when using JTS. "OTS" is specified when it omits.

- type

  When you connect with a database using JDBC, please specify "JDBC" or "DBMS" (the specification method in the old version). Please specify "JCA", when you connect with a resource adapter using J2 EE Connector Architecture. It is not omissible.

- lookupName

  When connecting with a database using JDBC, the name which bound the data sauce which a database offers is specified. Please specify the same value as the data sauce name set up by the J2EE resource access definition.

- initialContextFactory

  The initialContextFactoruy name used when referring to the bound data sauce is specified. Please specify the same value as the class name set up by the J2EE resource access definition. It is indispensable only when connecting with a database using JDBC.

- providerURL

  Provider URL used when referring to the bound data sauce is specified. Please specify the same value as the class name set up by the J2EE resource access definition. The initialContextFactoruy name used when referring to the bound data sauce is specified. It is indispensable only when connecting with a database using JDBC.

- USER Solaris32 Linux32/64

  A database administrator's user name is specified. When it omits, it comes to start by a superuser's authority. When -u option is specified at the time of otssetrsc command execution, the user name specified to be an option becomes effective. It is necessary to specify it as "GROUP" simultaneously.

- user

  In case it connects with a resource, it specifies, when a user name is required. Please specify the user name set up by J2EE resource access definition.

- password

  In case it connects with a resource, it specifies, when a password is required. Please specify the user name set up by J2EE resource access definition.

- GROUP Solaris32 Linux32/64

  A database administrator's group name is specified. When it omits, it comes to start by a superuser's authority. When -g option is specified at the time of *otssetrsc* command execution, the group name specified to be an option becomes effective. It is necessary to specify it as "USER" simultaneously.

- logfileDir

  When you conduct trouble investigation of the connected resource, please specify the directory which extracts a trace log. Please do not add separator to the last of a directory name. Usually, it does not specify.

Appendix D Event Service Environment Definition

This appendix describes the Event Service operation environment file and the methods for estimating the total number of the Event Channels, Suppliers, and Consumers.

Each file is stored as follows:

Storage directory

Windows32/64

(Default installation path)

```
C:\Interstage\eswin\etc
```

Solaris32/64

(Default installation path)

```
/etc/opt/FJSVes
```

Linux32/64

```
/etc/opt/FJSVes
```

Files

traceconfig

Note

Files other than the above-mentioned files cannot be customized as the Event Service environment definition. Do not edit files other than those listed above using an editor or similar.

D.1 traceconfig

Overview

The *traceconfig* file contains the definition related to the Event Service trace operating environment.

File Name

Windows32/64

(Default installation path)

```
C:\Interstage\eswin\etc\traceconfig
```

Solaris32/64

(Default installation path)

```
/etc/opt/FJSVes/traceconfig
```

Linux32/64

```
/etc/opt/FJSVes/traceconfig
```

File Contents

Format

In the *traceconfig* file, values are specified in the following format.

Parameter name = value set

Parameters

Parameter values can be modified for the following operating environments:

Notes

- The parameter value changes made in the environment definition file take effect from the next startup of the Event Service.

Table D.1 Parameter and Meaning

Parameter	Initial value	Meaning
	Default value	
	Specifiable range	
trace_size	1024	Specifies the size of the buffer used to collect trace information in kilobytes. (*1)
	512	
	512 - 102400	
trace_file_number	Windows32/64 Solaris32/64 50 Linux32/64 10	Specifies the maximum number of trace information files to be collected. If the number of trace information files exceeds the specified number, old trace information files are overwritten.
	Windows32/64 Solaris32/64 50 Linux32/64 10	
	Windows32/64 Solaris32/64 50 - 1000 Linux32/64 10 - 1000	
trace_auto	yes	Specifies whether to enable automatic collection of trace information.
	yes	- yes: Enables automatic collection of trace information. (*2)

Parameter	Initial value	Meaning
	Default value	
	Specifiable range	
	yes, no	- no: Disables automatic collection of trace information.
Linux32/64 trace_buffer	process	Specifies the method for collecting the internal trace. - process: The internal trace is collected by the process. - system: The internal trace is collected by the Event Service.
	process	
	process, system	

*1 Windows32/64 Solaris32/64

The size of trace information to be collected varies depending on the numbers of channels, consumers, suppliers, and communication frequency.

The size of the trace information buffer used for each type of processing is shown below.

Linux32/64

If the internal trace is collected by the process (trace_buffer = process), the size of trace information to be collected varies depending on the number of channels, consumers, suppliers, and communication frequency.

The size of the trace information buffer used for each type of processing is shown below.

If the internal trace is collected by the Event Service (trace_buffer = system), trace information used in start, communication, and stop processing is stored in one buffer. For this reason, the size for each must be calculated. Use the following to calculate these:

- Start processing

  Event channel start processing: 3.2 KB

  Supplier start processing (until a push method is issued): 1.0 KB

  Consumer start processing (until a pull method is issued): 1.0 KB

- Communication processing

  push method: 0.8 KB

  pull method (reception successful): 1.2 KB

  pull method (COMM_FAILURE[minor=0x464a09c1]): 1.0 KB

- Stop processing

  Event channel stop processing: 3.4 KB

  Supplier disconnect processing: 0.5 KB

  Consumer disconnect processing: 0.8 KB

The following shows an example of calculation for operation with the default value set for the trace information buffer size.

Example: One channel with the number of consumers equal to the number of suppliers

One cycle of push-pull communication requires 2.0 KB (0.8 KB + 1.2 KB) of buffer size.

Because the trace information buffer is used half by half cyclically, the buffer (1024 KB) can contain 256 sets of communication trace information.

("Trace information buffer size" / 2) / "buffer size required for one cycle of communication"

= (1024 KB / 2) / 2.0 KB = 256

Suppose one cycle of communication is performed every 40 seconds, then communication for about 2.8 hours can be logged.

256 x 40 s = 10240 s = about 2.8 h

The above example can collect trace information for about 2.8 hours until an event that triggers automatic collection of trace information occurs.

Specify the size of the trace information buffer so that it can collect trace information for at least five minutes.

If the default size of the trace information buffer is changed, utilization of shared memory increases as much as the buffer size (in units of kilobytes).

*2 If automatic collection of the trace information is enabled (trace_auto = yes), the trace file is output to the following file:

Windows32/64

```
C:\Interstage\eswin\var\ESLOGXXX (XXX is a decimal value of treble.)
```

Solaris32/64

```
/var/opt/FJSVes/ESLOGXXX (XXX is a decimal value of treble.)
```

Linux32/64

- When the internal trace is collected by the process (trace_buffer = process)

```
[Event Service daemon process log information]
/var/opt/FJSVes/ESLOGDUMPDAEMONXXX (XXX is a decimal value of treble.)

[Event Factory process log information]
/var/opt/FJSVes/ESLOGDUMPFACTORYXXX (XXX is a decimal value of treble.)

[Static Event Channel process log information]
/var/opt/FJSVes/ESLOGDUMP group nameXXX (XXX is a decimal value of treble.)

[Dynamic Event Channel process log information]
/var/opt/FJSVes/ESLOGDUMP implementation nameXXX (XXX is a decimal value of treble.)
```

- When the internal trace is collected by the Event Service (trace_buffer = system)

```
/var/opt/FJSVes/ESLOGXXX (XXX is a decimal value of treble.)
```

D.2 Estimating the Total Number of Suppliers and Consumers

Refer to the following formula when estimating the total number of Suppliers and Consumers that will connect to an event channel that uses the same unit at the time of the persistent channel operation.

```
Total for the maximum number of the event channel (see Note) + 10 < tranmax value of the unit
definition file
```

Note

For all event channels that use the same unit, set the following values for each event channel (if this value is less than 256, then it will assume the value of 256) and calculate the total.

Windows32/64

```
maximum number of connections (value specified for the -m option in the esmkchnl command (default:
16)) * 2
```

Solaris32/64 Linux32/64

```
maximum number of connections (value specified for the -m option in the esmkchnl command (default:
16)) + 16
```

Appendix E Interstage HTTP Server Environment Definition

The Interstage HTTP Server operating conditions can be tuned in two ways:

1. Make setting changes from the Interstage Management Console
2. Make settings changes in the Interstage HTTP Server environment definition file (httpd.conf).

To make settings using the Interstage Management Console, the environment is set as follows. For information on starting the Interstage Management Console, refer to the Operator's Guide.

- When setting the environment with the Standalone Server of the Interstage Management Console:
    1. Log in the Interstage Management Console of the Standalone Server.
    2. Configure the environment using the settings on the tab [System] > [Services] > [Web Server] > [Web Server name] > [Web Server Settings].
- When setting the environment with the Admin Server of the Interstage Management Console:
    1. Log in the Interstage Management Console of the Admin Server.
    2. Configure the environment using the settings on the tab [Application Management] > [Interstage] > [Interstage Application Server] > [Services] > [Web Server] > [FJapache(Server Group name or Server name)] > [Web Server Settings].

This appendix explains how to define the environment definition file (httpd.conf).

The environment definition file of the Interstage HTTP Server is stored in the following:

Windows32/64

(Default installation path)

```
C:\Interstage\F3FMihs\servers\(Web Server name)\conf\httpd.conf
```

Solaris32/64

(Default installation path)

```
/var/opt/FJSVihs/servers/(Web Server name)/conf/httpd.conf
```

Linux32/64

```
/var/opt/FJSVihs/servers/(Web Server name)/conf/httpd.conf
```

E.1 Timeout Value

This section described the timeout value.

Setting the Timeout Value

Set the timeout value by editing the following directive in the environment definition file (httpd.conf).

Timeout seconds

Specifies the time Interstage HTTP Server waits when a data packet is sent or received between clients. Specify a value from 0 to 65535 seconds.

When the packet cannot be received even if reaching at the specified time, the Interstage HTTP Server closes the connection. When traffic on the connected network increases, and the connection closes frequently, increasing this time will decrease the number of closed connections.

The initial value is 600, and the default value is 300.

Note

If the request does not arrive after the client TCP connection, the TCP connection will break when the specified time (in seconds) is reached.

SSLHandshakeTimeout seconds

Set the maximum wait time following the sending or receipt of a data package to or from the client for the established SSL connection process. Specify a value from 0 to 65535 seconds. When 0 is specified, the wait time is unlimited.

Interstage HTTP Server closes the connection if no data packages are received by the time the specified time is reached. The time taken for the established SSL connection process can be tuned.

There is no initial value for the timeout for the sending or receipt of data packages when the SSL connection is established. The default value is the value set for the Timeout directive.

Setting the HTTP Keep-Alive Function

To set the HTTP Keep-Alive function, edit the following directive in the environment definition file (httpd.conf).

KeepAlive On|Off

With the Interstage HTTP Server, a continuous between clients of a Web browser etc. can be kept.

If "Off" is specified, the connection is closed whenever a request is completed, and reconnected for the next request.

However, if "on" is specified the client response time improves as the connection is not closed after each request.

The initial value (and default value) is "On".

KeepAliveTimeout seconds

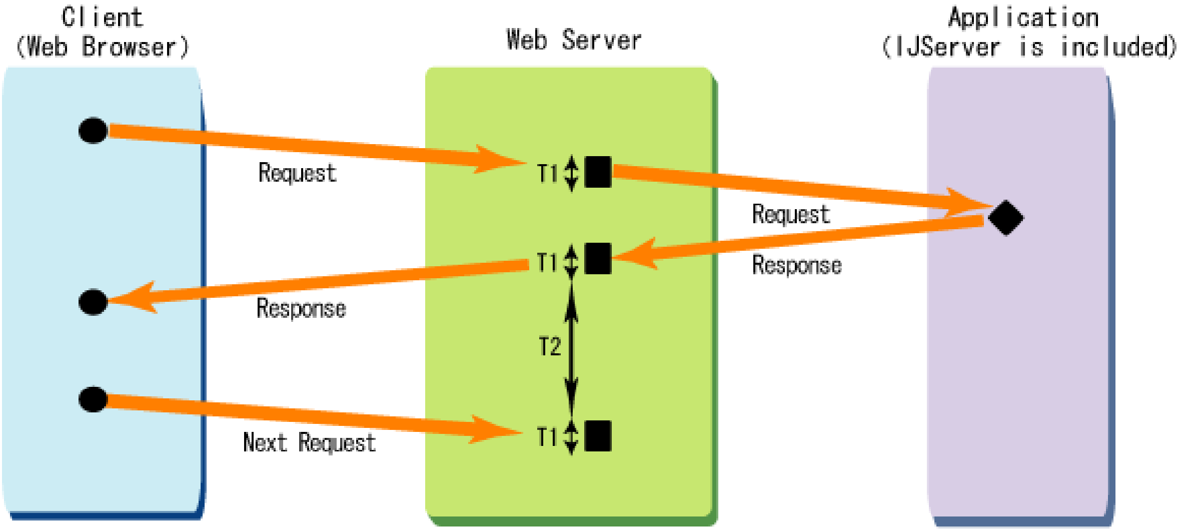
The number of seconds that Interstage HTTP Server will wait for a subsequent request before closing the connection is specified. This directive can be used only for "KeepAlive On". A value from 0 to 65535 can be specified for the connection keep-alive time. The connection is closed when there is no new request after this time elapses.

The initial value (and default value) is 15.

Configuration of a Timeout Value

The diagram below shows the configuration of a timeout value (for Timeout and KeepAliveTimeout).

Figure E.1 Timeout value (for Timeout and KeepAliveTimeout)



- T1 Timeout (initial value is 600 seconds)

  Specifies the time Interstage HTTP Server waits when a data packet is sent or received between clients.

  - Note

    In a POST or PUT request, when data is divided and sent or received in multiple fragments, this is the maximum time for an individual fragment.

- T2 KeepAliveTimeout (initial value is 15 seconds.)

  Specifies the number of seconds that Interstage HTTP Server will wait for a subsequent request before closing the connection.

E.2 Number of Clients that can be Connected Simultaneously

To set the number of clients that can be connected simultaneously, edit the following directive in the environment definition file (httpd.conf).

ThreadsPerChild number Windows32/64

This is the maximum number of requests that Interstage HTTP Server can accept from a Web browser at the same time.

The allowed range is from 1 to the value set for the ThreadLimit directive.

The initial value is 50, and the default value is 64.

Note

The higher the specified value, the greater the number of simultaneous access requests that can be accepted, but this also means more memory resources or temporary files are used, which may affect system performance.

Point

If there are more requests than specified with this value, they are saved in the following queue:

Client connection request that exceeds the value set for this directive	Queue	Maximum number of queues
First client connection request	Queue on the web server	1
Second and subsequent client connection requests	Queue in the operating system	Value of the ListenBacklog directive

ThreadLimit number Windows32/64

This is used to set the maximum number of clients for the ThreadsPerChild directive. The allowed range is from 1 to 15000. A value of less than 1 is treated as 1. Similarly, a value greater than 15000 is treated as 15000.

This directive should only be used if a value greater than 1920 needs to be set for the ThreadsPerChild directive.

There is no minimum value for the maximum number of communication threads. The default is 1920.

ListenBacklog number Windows32/64

If there are more requests than the simultaneous number of client connections set by the ThreadsPerChild directive, the maximum number of queues to be saved in the operating system is set. Any number from 1 to 200 can be specified.

The maximum number of queues waiting to be connected is 200 by default.

Note

If the connection request from the client exceeds the following value, this connection request will not be accepted and no status code will be returned.

```
Number of Clients that can be Connected Simultaneously (value of ThreadsPerChild directive) + Maximum
number of queues waiting for request processing (value of this directive + 1)
```

MaxClients number Solaris32/64 Linux32/64

This is the maximum number of requests that Interstage HTTP Server can accept from a Web browser at the same time.

The allowed range is from 1 to the value set for the ThreadLimit directive. A value of less than 1 is treated as 1. Similarly, a value greater than the value specified for the ServerLimit directive is treated as the ServerLimit directive value.

The initial value is 50, and the default value is 256.

Note

The higher the specified value, the greater the number of simultaneous access requests that can be accepted, but this also means that more memory resources or temporary files are used, which may affect system performance.

Point

If there are more requests than specified with this value, they are saved in a queue in the operating system. Use the ListenBacklog directive to set the number of queues waiting to be connected.

ServerLimit number Solaris32/64 Linux32/64

This is used to set the number of current client connections set for the MaxClients directive. The allowed range is from 1 to 20000. A value of less than 1 is treated as 1. Similarly, a value greater than 20000 is treated as 20000.

This directive should only be used if a value greater than 256 needs to be set for the MaxClients directive.

There is no minimum value for the maximum number of communication processes. The default is 256.

ListenBacklog number Solaris32/64 Linux32/64

If there are more requests than the simultaneous number of client connections set by the MaxClients directive, the value that corresponds to the following conditions will be the maximum value for the number of queues waiting to be connected in the operating system. Any number from 1 to 2147483647 can be specified.

The maximum number of queues waiting to be connected is 511 by default.

Condition	Maximum number of queues waiting for request processing
Value set by this directive <= Maximum number of TCP connections in standby (*1)	Value set by this directive
Value set by this directive > Maximum number of TCP connections in standby (*1)	Maximum number of TCP connections in standby (*1)

*1 The maximum value for TCP connections in standby is set in the operating system. Execute the commands shown below to check this. For details on setting the TCP connections in standby and the command, refer to the operating system documents.

	Maximum number of TCP connections in standby	Command execution examples
Solaris32/64	tcp_conn_req_max_q	/usr/sbin/ndd /dev/tcp tcp_conn_req_max_q
Linux32/64	/proc/sys/net/core/somaxconn	/sbin/sysctl -n net.core.somaxconn

Note

If the connection request from the client exceeds the following value, this connection request will not be accepted and no status code will be returned.

```
Number of Clients that can be Connected Simultaneously (value of ThreadsPerChild directive) + Maximum
number of queues waiting for request processing (value of this directive + n)
```

n: Value corresponding to the specifications of the operating system

Appendix F Environment Definition for Interstage Single Sign-on

This appendix explains how to tune the environment definition for Interstage Single Sign-on operation.

F.1 Tuning for Setting up Repository Server on One Server

This is not valid for Standard-J Edition on Windows (64 bit).

This is not valid for Standard-J Edition on Linux (64 bit).

The following example shows how to tune a repository server that is set up on one server.

Tuning Web Server (Interstage HTTP Server)

Tune the repository server by modifying the environment definition of the Web server (Interstage HTTP Server).

Refer to "Interstage HTTP Server Environment Definition" Chapter for details.

Windows32/64

ThreadsPerChild

In the ThreadsPerChild field, set the assumed maximum number of concurrent accesses or a greater value. (Initial value: 50)

Timeout (seconds)

In the Timeout field, specify the maximum response wait time (seconds) for data packets sent to, or received from, a client. (Initial value: 600)

Solaris32/64

MaxClients

In the MaxClients field, set the assumed maximum number of concurrent accesses or a greater value. (Initial value: 50)

Timeout (seconds)

In the Timeout field, specify the maximum response wait time (seconds) for data packets sent to, or received from, a client. (Initial value: 600)

Linux32/64

MaxClients

In the MaxClients field, set the assumed maximum number of concurrent accesses or a greater value. (Initial value: 50)

Timeout (seconds)

In the Timeout field, specify the maximum response wait time (seconds) for data packets sent to, or received from, a client. (Initial value: 600)

Example of Tuning

Windows32/64

The following example shows a system with a maximum of 256 concurrent users:

- **Interstage HTTP Server**

  ThreadsPerChild = 256 + a (*1)

  Timeout = 600 (*2)

  *1 To ensure stable system operation, specify a value from 10 to 100 in a.

*2 If the connection frequently stops because of high traffic in the connected network, increase the Timeout value.

Solaris32/64 Linux32/64

The following example shows a system with a maximum of 256 concurrent users:

- **Interstage HTTP Server**

  MaxClients = 256 + a (*1)

  Timeout = 600 (*2)

  *1 To ensure stable system operation, specify a value from 10 to 100 in a.

  *2 If the connection frequently stops because of high traffic in the connected network, increase the Timeout value.

Tuning TCP/IP Parameters

To use session management, tune the TCP/IP parameters on the Repository Server (update system). (*1)

For details, refer to "Tuning TCP/IP Parameters".

*1 The messages shown below are output if the session management server connection fails when the maximum number of simultaneous accesses to the Repository Server (update system) is increased. In this case, tune the TCP/IP parameters.

- sso00114
- sso00119

F.2 Tuning for Setting up Authentication Server on One Server

The following explains how to tune an authentication server that is set up on one server.

Tuning Web Server (Interstage HTTP Server)

Tune the authentication server by modifying the environment definition of the Web server (Interstage HTTP Server).

See Appendix E - "Web Server (Interstage HTTP Server) Environment Definition" for details.

Windows32/64

ThreadsPerChild

In the ThreadsPerChild field, set the assumed maximum number of concurrent accesses or a greater value. (Initial value: 50)

Timeout (seconds)

In the Timeout field, specify the maximum response wait time (seconds) for data packets sent to or received from a client. (Initial value: 600)

Solaris32/64

MaxClients

In the MaxClients field, set the assumed maximum number of concurrent accesses or a greater value. (Initial value: 50)

Timeout (seconds)

In the Timeout field, specify the maximum response wait time (seconds) for data packets sent to or received from a client. (Initial value: 600)

Linux32/64

MaxClients

In the MaxClients field, set the assumed maximum number of concurrent accesses or a greater value. (Initial value: 50)

Timeout (seconds)

In the Timeout field, specify the maximum response wait time (seconds) for data packets sent to or received from a client. (Initial value: 600)

Example of Tuning

Windows32/64

The following example shows a system with a maximum of 256 concurrent users:

- **Interstage HTTP Server**

  ThreadsPerChild = 256 + a (*1)

  Timeout = 600 (*2)

  *1 To ensure stable system operation, specify a value from 10 to 100 in a.

  *2 If the connection frequently stops because of high traffic in the connected network, increase the Timeout value.

The following example shows a system where up to a maximum of 500 concurrent usersare distributed and processed by five authentication servers:

- **Interstage HTTP Server**

  ThreadsPerChild = 100 + a (*3)

  Timeout = 600 (*4)

  *3 To ensure stable system operation, specify a value from 10 to 100 in a.

  *4 If the connection frequently stops because of high traffic in the connected network, increase the Timeout value.

Solaris32/64 Linux32/64

The following example shows a system with a maximum of 256 concurrent users:

- **Interstage HTTP Server**

  MaxClients = 256 + a (*1)

  Timeout = 600 (*2)

  *1 To ensure stable system operation, specify a value from 10 to 100 in a.

  *2 If the connection frequently stops because of high traffic in the connected network, increase the Timeout value.

The following example shows a system where up to a maximum of 500 concurrent users are distributed and processed by five authentication servers:

- **Interstage HTTP Server**

  MaxClients = 100 + a (*3)

  Timeout = 600 (*4)

  *3 To ensure stable system operation, specify a value from 10 to 100 in a.

  *4 If the connection frequently stops because of high traffic in the connected network, increase the Timeout value.

F.3 Tuning for Setting up Both Repository Server and Authentication Server on One Server

This is not valid for Standard-J Edition on Windows (64 bit).

This is not valid for Standard-J Edition on Linux (64 bit).

The following example shows how to tune a repository server and authentication server set up on one server.

Tuning the Web Server (Interstage HTTP Server)

Tune the repository server and authentication server by modifying the environment definition of the Web server (Interstage HTTP Server).

See Appendix E - "Web Server (Interstage HTTP Server) Environment Definition" for details.

Windows32/64

ThreadsPerChild

In the ThreadsPerChild field, set the value acquired by multiplying the assumed maximum number of concurrent accesses by 2 or set a greater value. (Initial value: 50)

Timeout (seconds)

In the Timeout field, specify the maximum response wait time (seconds) for data packets sent to or received from a client. (Initial value: 600)

Solaris32/64

MaxClients

In the MaxClients field, set the value acquired by multiplying the assumed maximum number of concurrent accesses by 2 or set a greater value. (Initial value: 50)

Timeout (seconds)

In the Timeout field, specify the maximum response wait time (seconds) for data packets sent to or received from a client. (Initial value: 600)

Linux32/64

MaxClients

In the MaxClients field, set the value acquired by multiplying the assumed maximum number of concurrent accesses by 2 or set a greater value. (Initial value: 50)

Timeout (seconds)

In the Timeout field, specify the maximum response wait time (seconds) for data packets sent to or received from a client. (Initial value: 600)

Example of Tuning

Windows32/64

The following example shows a system with a maximum of 256 concurrent users:

- **Interstage HTTP Server**

  ThreadsPerChild = 256 * 2 + a (*1)

  Timeout = 600 (*2)

  *1 To ensure stable system operation, specify a value from 10 to 100 in a.

  *2 If the connection frequently stops because of high traffic in the connected network, increase the Timeout value.

Solaris32/64 Linux32/64

The following example shows a system with a maximum of 256 concurrent users:

- **Interstage HTTP Server**

  MaxClients = 256 * 2 + a (*1)

  Timeout = 600 (*2)

  *1 To ensure stable system operation, specify a value from 10 to 100 in a.

*2 If the connection frequently stops because of high traffic in the connected network, increase the Timeout value.

Tuning TCP/IP Parameters

To use session management, tune the TCP/IP parameters on the Repository Server (update system). (*1)

For details, refer to "Tuning TCP/IP Parameters".

*1 The messages shown below are output if the session management server connection fails when the maximum number of simultaneous accesses to the Repository Server (update system) is increased. In this case, tune the TCP/IP parameters.

- sso00114
- sso00119

F.4 Tuning for Setting up the Business Server

This section explains the following topics:

- Tuning the Cache Size and Cache Count
- Tuning Web Server (Interstage HTTP Server)
- Tuning Microsoft(R) Internet Information Services

Tuning the Cache Size and Cache Count

When session management is used, authorization performance can be improved by caching the authentication information about the authenticated user in the business server. For efficient caching, the cache size and cache count must be set according to the number of simultaneous users in the system and the size of the authentication information about the user.

To configure the cache size and cache count settings, click [System] > [Security] > [Single Sign-on] > [Business system] > [Business system name] > [Settings] > [Detailed Settings[Show]] in the Interstage Management Console of the business server to perform the following:

- [Authentication information cache] - [Cache size]
- [Authentication information cache] - [Cache count]

Cache Size

The size of the authentication information is estimated according to the following formula and set in Kbytes.

Size of the authentication information = (150 + DN string length + user ID string length + authentication method string length + role size (*1) + size of the extended user information (*2)) * 1.4 bytes

*1 This is the sum of the role name string lengths that are set + 10 * the number of roles.

*2 This is the sum of the attribute name string lengths that are set + the sum of the attribute value string lengths + 10 * the number of extended user information.

Cache Count

The Authentication Information Cache is retained from the last time the user makes an access until the idle monitoring time elapses. Set the maximum assumed number of simultaneous accesses inside the idle monitoring time + Y(*1) for this value.

*1 Even if one user repeats sign on and sign off inside the idle monitoring time, it will use up the cache count, therefore a value that is slightly greater than the maximum assumed number of simultaneous accesses should be set.

Note

If the cache size, or the cache count that was set, is exceeded, continued use is still possible, however the sso03062 or sso03063 message is output to the system log.

The authorization performance may deteriorate, therefore take action according to the message.

Example of Tuning

An example of user information shown in the table below.

Item	String length	Value
DN	55	cn=Fujitsu Tarou,ou=User,ou=interstage,o=fujitsu,dc=com
User ID	5	Tarou
Authentication method	19	basicAuthOrCertAuth
Role name	5	Admin
Role name	6	Leader
Extended user information (attribute name)	4	Mail
Extended user information (attribute value)	20	tarou@jp.fujitsu.com
Extended user information (attribute name)	14	employeeNumber
Extended user information (attribute value)	6	100001

- **Role size**

  When two roles are set (the Admin and Leader roles in the above table) this size is calculated as follows:

  (Admin (5 characters) + Leader (6 characters)) + 10 * the number of roles (2) = 31

- **Size of the extended user information**

  When two attributes are set (the mail and employeeNumber attributes in the above table) this size is calculated as follows:

  (mail (4 characters) + employeeNumber (14 characters)) + (tarou@jp.fujitsu.com (20 characters) + 100001 (6 characters)) + 10 * number of extended user information (2) = 64

  Using the above example the size of authentication information is as follows:

  Size of authentication information = (150 + DN string length (55 characters) + user ID string length (5 characters) + authentication method string length (19 characters) + role size (31 characters) + size of the extended user information (64 characters)) * 1.4 bytes = approximately 454 bytes

  For [Cache Size], round off the above authentication information size and set 1K byte.

Tuning Web Server (Interstage HTTP Server)

See Appendix E - "Web Server (Interstage HTTP Server) Environment Definition" for details.

Windows32/64

ThreadsPerChild

In the ThreadsPerChild field, set the assumed maximum number of concurrent accesses or a greater value. (Initial value: 50)

Timeout (seconds)

In the Timeout field, specify the maximum response wait time (seconds) for data packets sent to or received from a client. (Initial value: 600)

Solaris32/64

MaxClients

In the MaxClients field, set the assumed maximum number of concurrent accesses or a greater value. (Initial value: 50)

Timeout (seconds)

In the Timeout field, specify the maximum response wait time (seconds) for data packets sent to or received from a client. (Initial value: 600)

Linux32/64

MaxClients

In the MaxClients field, set the assumed maximum number of concurrent accesses or a greater value. (Initial value: 50)

Timeout (seconds)

In the Timeout field, specify the maximum response wait time (seconds) for data packets sent to or received from a client. (Initial value: 600)

Tuning Microsoft(R) Internet Information Services

Windows32/64

To tune Microsoft(R) Internet Information Services, specify the following definition items in the property information of "Microsoft(R) Internet Information Services".

Maximum number of connections

In the maximum connection count field, set the assumed maximum number of concurrent accesses. (Default value: 1,000)

Appendix G Defining the Multi Server Management Environment Windows32/64 Solaris32 Linux32/64

This appendix explains how to tune the environment definition file for a Multi Server management operating environment.

G.1 Multi Server Management Definition File

Overview

The Multi Server management definition file is a file for storing definitions for a Multi Server management operating environment.

File Name

OS	Windows32/64	Solaris32 Linux32/64
File path	%IS_HOME%\jmx\etc\ssv.xml	/etc/opt/FJSVisjmx/ssv.xml

File Format

The format is XML. <ssv> is used as the root tag.

List of Tags and Attributes

Tag Name: site

Attribute name	Contents	Default value	Value range
server.limit	The maximum number of Managed Servers that belong to the Site.	100	1 to INT_MAX (*1)
servergroup.limit	The maximum number of Server Groups that belong to the Site.	100	1 to INT_MAX (*1)
server.limit.inservergroup	The maximum number of Managed Servers that belong to the Server Group.	100	1 to INT_MAX (*1)

*1 If the value range described in the Multi Server management definition file is exceeded, the default value is used.

Tag Name: ijserver

Attribute name	Contents	Default value	Value range
deploy.path	Temporary storage directory for the archive file to be deployed.	Windows32/64 %IS_HOME%\jmx\var\ssv_ijs Solaris32 Linux32/64 /etc/opt/FJSVisjmx/var/ssv_ijs	-

Appendix H Web Server (Sun Java System Web Server) Environment Definition

Solaris32

This appendix explains how to tune the environment definition file for a Sun Java System Web Server.

MaxProcs, RqThrottle

Specify the maximum number of processes that can be executed simultaneously (MaxProcs), and the maximum number of threads that can be processed simultaneously in each process (RqThrottle) in the Sun Java System Web Server.

MaxProcs and RqThrottle must be set so that they satisfy the following conditions:

```
MaxProcs * RqThrottle <= number of Servlet containers that can be processed simultaneously
```

For details, refer to the J2EE User's Guide. The relevant section is "Preparation for Servlet Service Operation" - "Sun Java System Web Server environment settings".

Appendix I Setting IPC Resources

Solaris32/64

To set the parameter values of IPC resources, use one of the following methods:

- Editing /etc/system Files
- Resource Control

Editing /etc/system Files

Using this method, make the necessary parameter value settings by editing files in /etc/system. The system must be rebooted for the changes to take effect. For more details on editing, refer to the Solaris documentation.

Resource Control

Modify parameter values using the following procedure:

1. Stop Interstage

   Stop Interstage, and also stop any services that are running in order to use the Interstage Management Console.

2. Modify the user.root project parameters

   Using the projmod command, modify the values of the user.root project parameters.

   Example

   ```
   projmod -s -K 'project.max-sem-ids=(privileged,155,deny)' user.root
   ```

   Set the required authorization level for making changes to "privileged" and set the action when a threshold is exceeded to "deny".

3. Modify the system project parameters

   Using the projmod command, modify the values of system project parameters.

   Example

   ```
   projmod -s -K 'project.max-sem-ids=(privileged,155,deny)' system
   ```

   Set the same values as those set for the user.root project.

4. Apply the changes

   Using the newtask command, apply the changes in the system.

   ```
   newtask -p user.root -c $$
   ```

5. Start Interstage

   Start Interstage, and if necessary, also start the services required to use the Interstage Management Console.

For more details on resource control, refer to the Solaris documentation.

Notes about Solaris 10 or later

In Solaris 10 and Solais 11, when tuning the system by editing the /etc/system file, set values for shmmax and shmmni according to the formula shown below.

```
project.max-shm-memory = shmmax * shmmni
```

Appendix J Portable-ORB Environment Definition

Use the *porbeditenv* command to set the Portable-ORB operating environment file.

To use Portable-ORB make the following environment settings:

- Config
- Initial Host
- Security
- Network

For details on the *porbeditenv* command, refer to 'porbeditenv' in the Reference Manual (Command Edition). For details on the Portable-ORB operating environment file, refer to Portable-ORB Operation Environment File Settings in the Distributed Application Development Guide (CORBA Service Edition).

When setting up the application environment, it is recommended that the resources are backed up. For details on backing up resources, refer to Maintenance (Resource Backup) in the Operator's Guide.