HITACHI
Inspire the Next

Job Management Partner 1/Automatic Job Management System 3

System Design (Work Tasks) Guide

3020-3-S04-04(E)

Relevant program products

For details about the applicable OS versions, and the service packs and patches required for JP1/Automatic Job Management System 3, see the *Release Notes*.

For Windows Server 2008:
P-2A12-3K97 Job Management Partner 1/Automatic Job Management System 3 - Manager version 09-00
P-2A12-3397 Job Management Partner 1/Automatic Job Management System 3 - Agent version 09-00
P-2A2C-6L97 Job Management Partner 1/Base version 09-00
For Windows 7, Windows Server 2008 and Windows Vista:
P-2A12-3497 Job Management Partner 1/Automatic Job Management System 3 - View version 09-00
For Windows Server 2003 and Windows Server 2003(x64):
P-2412-3K97 Job Management Partner 1/Automatic Job Management System 3 - Manager version 09-00
P-2412-3397 Job Management Partner 1/Automatic Job Management System 3 - Agent version 09-00
P-242C-6L97 Job Management Partner 1/Base version 09-00
For Windows Server 2003, Windows Server 2003(x64), and Windows XP Professional:
P-2412-3497 Job Management Partner 1/Automatic Job Management System 3 - View version 09-00
For HP-UX(IPF):
P-1J12-2792 Job Management Partner 1/Automatic Job Management System 3 - Manager version 09-00
P-1J12-2992 Job Management Partner 1/Automatic Job Management System 3 - Agent version 09-00
P-1J2C-6L92 Job Management Partner 1/Base version 09-00
For Solaris 9(SPARC), and Solaris 10(SPARC):
P-9312-2792 Job Management Partner 1/Automatic Job Management System 3 - Manager version 09-00
P-9312-2992 Job Management Partner 1/Automatic Job Management System 3 - Agent version 09-00
P-9D2C-6L92 Job Management Partner 1/Base version 09-00
For AIX:
P-1M12-2792 Job Management Partner 1/Automatic Job Management System 3 - Manager version 09-00
P-1M12-2992 Job Management Partner 1/Automatic Job Management System 3 - Agent version 09-00
P-1M2C-6L92 Job Management Partner 1/Base version 09-00

Trademarks

■ Restrictions

Information in this document is subject to change without notice and does not represent a commitment on the part of Hitachi. The software described in this manual is furnished according to a license agreement with Hitachi. The license agreement contains all of the terms and conditions governing your use of the software and documentation, including all warranty rights, limitations of liability, and disclaimers of warranty.

Material contained in this document may describe Hitachi products not available or features not available in your country.

Printed in Japan.

■ Edition history

Edition 1 (3020-3-S04-03(E)): November 2009

Edition 1.4 (3020-3-S04-04(E)): November 2010

■ Copyright

Summary of amendments

The following table lists changes in this manual (3020-3-S04-04(E)) and product changes related to this manual.

Changes	Location
Descriptions have been changed. For details, see Appendix C.	*Appendix C*

In addition to the above changes, minor editorial corrections have been made.

Preface

This manual describes how to design work tasks for Job Management Partner 1/Automatic Job Management System 3 (abbreviated hereafter to *JP1/AJS3*). Read this manual in conjunction with the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*, which describes JP1/AJS3 functionality.

Intended readers

This manual is intended for:

- Those who wish to operate an automatic job execution system with JP1/AJS3 and those who design automatic job execution systems.
- Those who operate an automatic job execution system with JP1/AJS3.

Organization of this manual

This manual organized into the following chapters. The manual is a common reference for all supported operating systems. Any platform-dependent differences in functionality are noted in the manual.

1. Overview of Work Task Design

Chapter 1 provides an overview of designing work tasks that will be automated with JP1/AJS3. The chapter also includes the sequence of design steps and considerations during design.

2. Job Definition and Job Execution Order Considerations

Chapter 2 describes the considerations necessary for constructing the jobs and jobnets that are required for automating work tasks with JP1/AJS3.

3. Operation Calendar and Execution Schedule Considerations

Chapter 3 describes the considerations necessary for setting a calendar and execution schedule for JP1/AJS3 operation.

4. Execution Registration Method Considerations

Chapter 4 describes how to register jobnets for execution in JP1/AJS3.

5. Monitoring Method Considerations

Chapter 5 describes the considerations necessary for using JP1/AJS - View and JP1/AJS3 Console to monitor jobnets.

6. Access Permission Considerations

Chapter 6 describes the considerations necessary for setting the access permissions for jobnets and for associating JP1 users and OS users.

7. Cautionary Notes on Designing Work Tasks

Chapter 7 provides cautionary notes on designing work tasks.

8. Definition Pre-Check

Chapter 8 describes the procedure for the definition pre-check performed before actual JP1/AJS3 operation starts. The chapter also describes check items and includes cautionary notes.

Related publications

This manual is part of a related set of manuals. The manuals in the set are listed below (with the manual numbers):

About JP1/AJS:

- *Job Management Partner 1/Automatic Job Management System 3 Overview* (3020-3-S02(E))
- *Job Management Partner 1/Automatic Job Management System 3 System Design (Configuration) Guide* (3020-3-S03(E))
- *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1* (3020-3-S05(E))
- *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 2* (3020-3-S06(E))
- *Job Management Partner 1/Automatic Job Management System 3 Administration Guide* (3020-3-S07(E))
- *Job Management Partner 1/Automatic Job Management System 3 Troubleshooting* (3020-3-S08(E))
- *Job Management Partner 1/Automatic Job Management System 3 Operator's Guide* (3020-3-S09(E))
- *Job Management Partner 1/Automatic Job Management System 3 Command Reference* 1 (3020-3-S10(E))
- *Job Management Partner 1/Automatic Job Management System 3 Command Reference* 2 (3020-3-S11(E))
- *Job Management Partner 1/Automatic Job Management System 3 Linkage Guide* (3020-3-S12(E))
- *Job Management Partner 1/Automatic Job Management System 3 Messages 1* (3020-3-S13(E))

- *Job Management Partner 1/Automatic Job Management System 3 Messages 2* (3020-3-S14(E))
- *Job Management Partner 1/Automatic Job Management System 3 - Definition Assistant Description, Operator's Guide and Reference* (3020-3-S17(E))
- *Job Management Partner 1/Automatic Job Management System 3 - Web Operation Assistant Description, Operator's Guide and Reference* (3020-3-S18(E))
- *Job Management Partner 1/Automatic Job Management System 3 for Enterprise Applications Description, User's Guide and Reference* (3020-3-S29(E))
- *Job Management Partner 1/Automatic Job Management System 2 for Oracle E-Business Suite Description, User's Guide and Reference* (3020-3-F27(E))

About JP1:

- *Job Management Partner 1/Base User's Guide* (3020-3-R71(E))
- *Job Management Partner 1/Base Messages* (3020-3-R72(E))
- *Job Management Partner 1/Base Function Reference* (3020-3-R73(E))
- *Job Management Partner 1/Integrated Management - Manager Overview and System Design Guide* (3020-3-R76(E))
- *Job Management Partner 1/Integrated Management - Manager Configuration Guide* (3020-3-R77(E))
- *Job Management Partner 1/Integrated Management - Manager Administration Guide* (3020-3-R78(E))
- *Job Management Partner 1/Integrated Management - Manager GUI Reference* (3020-3-R79(E))
- *Job Management Partner 1/Integrated Management - Manager Command and Definition File Reference* (3020-3-R80(E))
- *Job Management Partner 1/Integrated Management - Manager Messages* (3020-3-R81(E))
- *Job Management Partner 1/Script Description and Reference* (3020-3-K55(E)), for Windows systems
- *Job Management Partner 1/File Transmission Server/FTP Description, Reference, and Operator's Guide* (3020-3-S36(E)), for Windows systems
- *Job Management Partner 1/File Transmission Server/FTP Description, Reference, and Operator's Guide* (3020-3-S37(E)), for UNIX systems
- *Job Management Partner 1/Software Distribution Description and Planning Guide* (3020-3-S79(E)), for Windows systems

- *Job Management Partner 1/Software Distribution Setup Guide* (3020-3-S80(E)), for Windows systems
- *Job Management Partner 1/Software Distribution System Administrator's Guide Volume 1* (3020-3-S81(E)), for Windows systems
- *Job Management Partner 1/Software Distribution System Administrator's Guide Volume 2* (3020-3-S82(E)), for Windows systems
- *Job Management Partner 1/Software Distribution Automatic Installation Tool Description and Reference* (3020-3-S83(E)), for Windows systems
- *Job Management Partner 1/Software Distribution Administrator Kit Description and Operator's Guide* (3020-3-S84(E))
- *Job Management Partner 1/Software Distribution Client Description and User's Guide* (3020-3-S85(E)), for UNIX systems
- *Job Management Partner 1/Software Distribution SubManager Description and Administrator's Guide* (3020-3-L42(E)), for UNIX systems
- *Job Management Partner 1/Software Distribution Manager Description and Administrator's Guide* (3000-3-841(E))
- *Job Management Partner 1/NQSEXEC System Administrator's Guide* (3020-3-F30(E))
- *Job Management Partner 1/Consolidated Management 2/Extensible SNMP Agent Description, Operator's Guide and Reference* (3020-3-L04(E)), for UNIX systems
- *Job Management Partner 1/Open Job Entry Description, User's Guide and Reference* (6190-3-365(E)), for VOS3 systems
- *Job Management Partner 1/Open Job Entry Description, User's Guide and Reference* (9000-3-365(E)), for MVS systems
- *Job Management Partner 1/Open Job Entry Description, User's Guide and Reference* (9000-3-366(E)), for OSIV/MSP systems
- *Job Management Partner 1/Open Job Entry for Midrange Computer Description and User's Guide* (9000-3-367(E))

Conventions: Abbreviations

This manual uses the following abbreviations for product names:

Abbreviation		Full name or meaning
JP1/AJS3	JP1/AJS3 - Manager	Job Management Partner 1/Automatic Job Management System 3 - Manager

Abbreviation		Full name or meaning
	JP1/AJS3 - Agent	Job Management Partner 1/Automatic Job Management System 3 - Agent
	JP1/AJS3 - View	Job Management Partner 1/Automatic Job Management System 3 - View
JP1/AJS2	JP1/AJS2 - Manager	Job Management Partner 1/Automatic Job Management System 2 - Manager
	JP1/AJS2 - Agent	Job Management Partner 1/Automatic Job Management System 2 - Agent
	JP1/AJS2 - View	Job Management Partner 1/Automatic Job Management System 2 - View
JP1/AJS2 - Advanced Manager		Job Management Partner 1/Automatic Job Management System 2 - Advanced Manager#
JP1/AJS2 - Client Toolkit		Job Management Partner 1/Automatic Job Management System 2 - Client Toolkit#
JP1/AJS3 - Definition Assistant		Job Management Partner 1/Automatic Job Management System 3 - Definition Assistant
JP1/AJS3 - Web Operation Assistant		Job Management Partner 1/Automatic Job Management System 3 - Web Operation Assistant
JP1/AJS3 for Enterprise Applications		Job Management Partner 1/Automatic Job Management System 3 for Enterprise Applications
JP1/AJS2 for Oracle E-Business Suite		Job Management Partner 1/Automatic Job Management System 2 for Oracle E-Business Suite
NNM	HP NNM	HP Network Node Manager Software version 7.5 or earlier
		HP Network Node Manager Software Starter Edition version 7.5 or earlier
JP1/FTP		Job Management Partner 1/File Transmission Server/FTP
JP1/IM	JP1/IM - Manager	Job Management Partner 1/Integrated Management - Manager
	JP1/IM - View	Job Management Partner 1/Integrated Management - View

Abbreviation		Full name or meaning
	JP1/IM - Central Console	Job Management Partner 1/Integrated Manager - Central Console#
	JP1/IM - Central Scope	Job Management Partner 1/Integrated Manager - Central Scope#
JP1/OJE		Job Management Partner 1/Open Job Entry
JP1/OJE for Midrange Computer		Job Management Partner 1/Open Job Entry for Midrange Computer
JP1/SES		Job Management Partner 1/System Event Service
JP1/OJE for VOS3		VOS3 Job Management Partner 1/Open Job Entry
MSCS		Microsoft(R) Cluster Server
Excel		Microsoft(R) Excel
		Microsoft(R) Office Excel
Exchange Server		Microsoft(R) Exchange 2000 Enterprise Server
		Microsoft(R) Exchange 2000 Server
		Microsoft(R) Exchange Server
IE		Microsoft(R) Internet Explorer(R)
Microsoft Mail		Microsoft(R) Mail
MSMQ		Microsoft(R) Message Queue Server
Outlook	Outlook 2000	Microsoft(R) Outlook(R) 2000
	Outlook 2002	Microsoft(R) Outlook(R) 2002
	Outlook 2003	Microsoft(R) Outlook(R) 2003
	Outlook 2007	Microsoft(R) Outlook(R) 2007
	Outlook Express	Microsoft(R) Outlook(R) Express
Microsoft SQL Server		Microsoft(R) SQL Server
		Microsoft(R) SQL Server Enterprise Edition
Windows 7		Microsoft(R) Windows(R) 7 Enterprise
		Microsoft(R) Windows(R) 7 Professional

Abbreviation		Full name or meaning
		Microsoft(R) Windows(R) 7 Ultimate
Windows Server 2003	Windows Server 2003	Microsoft(R) Windows Server(R) 2003, Enterprise Edition Operating System
		Microsoft(R) Windows Server(R) 2003, Standard Edition Operating System
	Windows Server 2003 (x64)	Microsoft(R) Windows Server(R) 2003, Enterprise x64 Edition
		Microsoft(R) Windows Server(R) 2003, Standard x64 Edition
Windows Server 2008		Microsoft(R) Windows Server(R) 2008 Datacenter
		Microsoft(R) Windows Server(R) 2008 Enterprise
		Microsoft(R) Windows Server(R) 2008 Standard
Windows Vista		Microsoft(R) Windows Vista(R) Business
		Microsoft(R) Windows Vista(R) Enterprise
		Microsoft(R) Windows Vista(R) Ultimate
Windows XP Professional		Microsoft(R) Windows(R) XP Professional Operating System
AIX		AIX 5L 5.3
		AIX V6.1
HP-UX	HP-UX (IPF)	HP-UX 11i V2(IPF)
		HP-UX 11i V3(IPF)
Solaris		Solaris 9(SPARC)
		Solaris 10(SPARC)
SAP BW		SAP Business Information Warehouse
SAP R/3		SAP R/3(R)

#: Version 7

- In this manual, *JP1/AJS* is sometimes used generically, referring to JP1/AJS3 and JP1/AJS2.

- *Windows* is sometimes used generically, referring to Windows 7, Windows Server 2008, Windows Vista, Windows Server 2003, and Windows XP Professional.
- *UNIX* is sometimes used generically, referring to HP-UX, Solaris, and AIX.

This manual also uses the following abbreviations:

Abbreviation	Full name or meaning
ACL	Access Control List
DB	Database
DBMS	Database Management System
DNS	Domain Name System
EUC	Extended UNIX Code
FQDN	Fully Qualified Domain Name
FTP	File Transfer Protocol
GUI	Graphical User Interface
IPF	Itanium(R) Processor Family
ISAM	Indexed Sequential Access Method
LAN	Local Area Network
MAPI	Messaging Application Programming Interface
MIB	Management Information Base
MIME	Multipurpose Internet Mail Extensions
NAT	Network Address Translator
NFS	Network File System
NIC	Network Interface Card
OS	Operating System
RDB	Relational Database
SNMP	Simple Network Management Protocol
SMTP	Simple Mail Transfer Protocol
SUP	Service Using Program
TCP/IP	Transmission Control Protocol/Internet Protocol

Abbreviation	Full name or meaning
UAC	User Account Control
UNC	Universal Naming Convention
WAN	Wide Area Network
WOW64	Windows On Windows 64
WSDL	Web Services Description Language

JP1 program reorganization in version 8

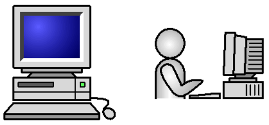
The following changes have been made to the JP1 product suite in version 8:

- JP1/AJS2 - Advanced Manager has been eliminated, and the database provided by JP1/AJS2 - Advanced Manager has been integrated into JP1/AJS2 - Manager in JP1 Version 8.
- JP1/AJS2 - Client Toolkit has been eliminated.
- JP1/AJS2 - View is provided only in the Windows version.

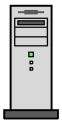
Conventions: Diagrams

This manual uses the following conventions in diagrams:

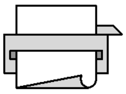
● Computer (terminal)



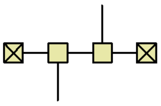
● Computer



● Printer

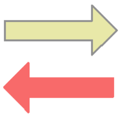


● Network

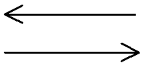


● Communication lines

● Data flow



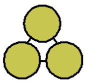
● Control flow



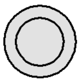
● Screen

● Program

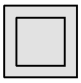
● Jobnet



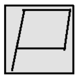
● Job



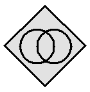
● Event job



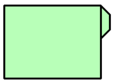
● Judgment job



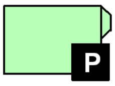
● OR job



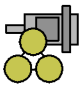
● Job group



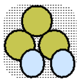
● Planning group



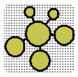
● Jobnet connector



● AJS3 unit monitoring object



● Business scope



● Process or operation flow

Conventions: Fonts and symbols

Font and symbol conventions are classified as:

- General font conventions
- Conventions in syntax explanations

These conventions are described below.

General font conventions

The following table lists the general font conventions:

Font	Convention
Bold	Bold type indicates text on a window, other than the window title. Such text includes menus, menu options, buttons, radio box options, or explanatory labels. For example, bold is used in sentences such as the following: • From the **File** menu, choose **Open**. • Click the **Cancel** button. • In the **Enter name** entry box, type your name.
Italics	Italics are used to indicate a placeholder for some actual text provided by the user or system. Italics are also used for emphasis. For example: • Write the command as follows: `copy` *source-file target-file* • Do *not* delete the configuration file.
`Code font`	A code font indicates text that the user enters without change, or text (such as messages) output by the system. For example: • At the prompt, enter `dir`. • Use the `send` command to send mail. • The following message is displayed: `The password is incorrect.`

Examples of coding and messages appear as follows (although there may be some exceptions, such as when coding is included in a diagram):

```
MakeDatabase
...
StoreDatabase temp DB32
```

In examples of coding, an ellipsis (...) indicates that one or more lines of coding are not shown for purposes of brevity.

Conventions in syntax explanations

Syntax definitions appear as follows:

S`tore`**D**`atabase [temp|perm]` (*database-name* `...`)

The following table lists the conventions used in syntax explanations:

Example font or symbol	Convention
`StoreDatabase`	Code-font characters must be entered exactly as shown.
database-name	This font style marks a placeholder that indicates where appropriate characters are to be entered in an actual command.
`SD`	Bold code-font characters indicate the abbreviation for a command.
`Perm`	Underlined characters indicate the default value.
`[ ]`	Square brackets enclose an item or set of items whose specification is optional.

Example font or symbol	Convention
|	Only one of the options separated by a vertical bar can be specified at the same time.
. . .	An ellipsis (...) indicates that the item or items enclosed in ( ) or [ ] immediately preceding the ellipsis may be specified as many times as necessary.
()	Parentheses indicate the range of items to which the vertical bar (|) or ellipsis (...) is applicable.

Conventions for mathematical expressions

This manual uses the following symbols in mathematical expressions:

Symbol	Meaning
x	Multiplication sign
/	Division sign
↑ ↑	The calculation result is rounded up to the next whole number. Example: The result of ↑ 34 / 3 ↑ is 12.
～ (tilde)	The item shown before this symbol must be specified in accordance with the conventions shown for angle brackets, double parentheses, and double angle brackets (below).
< > (angle brackets)	Indicates the characters and lexical elements that can be specified. <characters> One or more Kanji characters, katakana characters, upper-case alphabetic characters, lower-case alphabetic characters, or numeric characters <numeric> 0, 1, 2, 3, 4, 5, 6, 7, 8, or 9 <alphabetic character> A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z, \, #, or @ <alphanumeric character> Alphabetic or numeric character <symbolic name> No more than eight alphanumeric characters beginning with an alphabetic character <unsigned integer> One or more numeric characters <hexadecimal character> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, or F <file name> A system-determined name assigned to a file <path> The directories contained in the path, with each name separated by a forward slash (/) or backslash (\). The path notation is OS-dependent.

Symbol	Meaning
(( )) (double parentheses)	Indicates the range of specifiable values.
<< >> (double angle brackets)	Indicates the default assumed by the system when a value is unspecified. Example: If you do not specify *days-to-keep-form* ~<numeric> ((0 to 365)) <<`365`>>, 365 is assumed as the number of days to keep the form.
MAX	Choose the largest of the calculation results. Example: The result of MAX (3 x 6, 4 + 7) is 18.

Conventions: KB, MB, GB, and TB

This manual uses the following conventions:

- 1 KB (kilobyte) is 1,024 bytes.
- 1 MB (megabyte) is 1,024^2 bytes.
- 1 GB (gigabyte) is 1,024^3 bytes.
- 1 TB (terabyte) is 1,024^4 bytes.

Conventions: Meaning of "directory" and "folder"

As a general rule, Windows folder names are used in this manual if they are identical to UNIX directory names.

Conventions: Meaning of "member of the Administrators group"

The term *member of the Administrators group* in this manual refers to a user who is a member of the Administrators group on the local PC only. The privileges of local users, domain users, and Active Directory users are no different as long as these users are members of the Administrators group on the local PC.

Conventions: Version numbers

The version numbers of Hitachi program products are usually written as two sets of two digits each, separated by a hyphen. For example:

- Version 1.00 (or 1.0) is written as 01-00.
- Version 2.05 is written as 02-05.
- Version 2.50 (or 2.5) is written as 02-50.
- Version 12.25 is written as 12-25.

The version number might be shown on the spine of a manual as *Ver. 2.00,* but the same version number would be written in the program as *02-00*.

Default installation folders of JP1/AJS3 for Windows

The default installation folders of JP1/AJS3 for Windows are as follows:

Default installation folders of JP1/AJS3 - Manager:

system-drive`\Program Files`#1`\HITACHI\JP1AJS2`

and

system-drive`\Program Files`#1`\HITACHI\JP1AJS2CM`

Default installation folder of JP1/AJS3 - Agent:

system-drive`\Program Files`#1`\HITACHI\JP1AJS2`

Default installation folder of JP1/AJS3 - View:

system-drive`\Program Files`#2`\HITACHI\JP1AJS2V`

#1

For 64-bit versions of Windows Server 2008 and Windows Server 2003 (x64), replace `Program Files` with `Program Files (x86)`.

#2

For 64-bit versions of Windows 7, Windows Server 2008, Windows Vista, and Windows Server 2003 (x64), replace `Program Files` with `Program Files (x86)`.

Online manual

JP1/AJS3 - View comes with an online manual that you can read in either of the following browsers:

- Microsoft Internet Explorer version 6.0 or later
- Windows Internet Explorer Version 7.0 or later

The online manual has the same contents as the following manuals:

- *Job Management Partner 1/Automatic Job Management System 3 Overview*
- *Job Management Partner 1/Automatic Job Management System 3 System Design (Configuration) Guide*
- *Job Management Partner 1/Automatic Job Management System 3 System Design (Work Tasks) Guide*
- *Job Management Partner 1/Automatic Job Management System 3 Configuration*

Guide 1

- *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 2*
- *Job Management Partner 1/Automatic Job Management System 3 Administration Guide*
- *Job Management Partner 1/Automatic Job Management System 3 Troubleshooting*
- *Job Management Partner 1/Automatic Job Management System 3 Operator's Guide*
- *Job Management Partner 1/Automatic Job Management System 3 Command Reference 1*
- *Job Management Partner 1/Automatic Job Management System 3 Command Reference 2*
- *Job Management Partner 1/Automatic Job Management System 3 Linkage Guide*
- *Job Management Partner 1/Automatic Job Management System 3 Messages 1*
- *Job Management Partner 1/Automatic Job Management System 3 Messages 2*

In JP1/AJS3 - View, you can view the manual by choosing **Help** and then **Contents**. You can also press the **F1** key to view the manual contents. Your Web browser must be associated with a file that has the extension `htm`; otherwise, the online manual will not be displayed correctly. If this happens, associate the `htm` file with the Web browser.

Cautionary note

Depending on the OS settings, the online manual might appear in the active window of the browser when you launch the manual from the **Start** menu.

Organization of JP1/AJS3 manuals and choosing the right manuals

There are fourteen JP1/AJS3 manuals. The following table summarizes their contents.

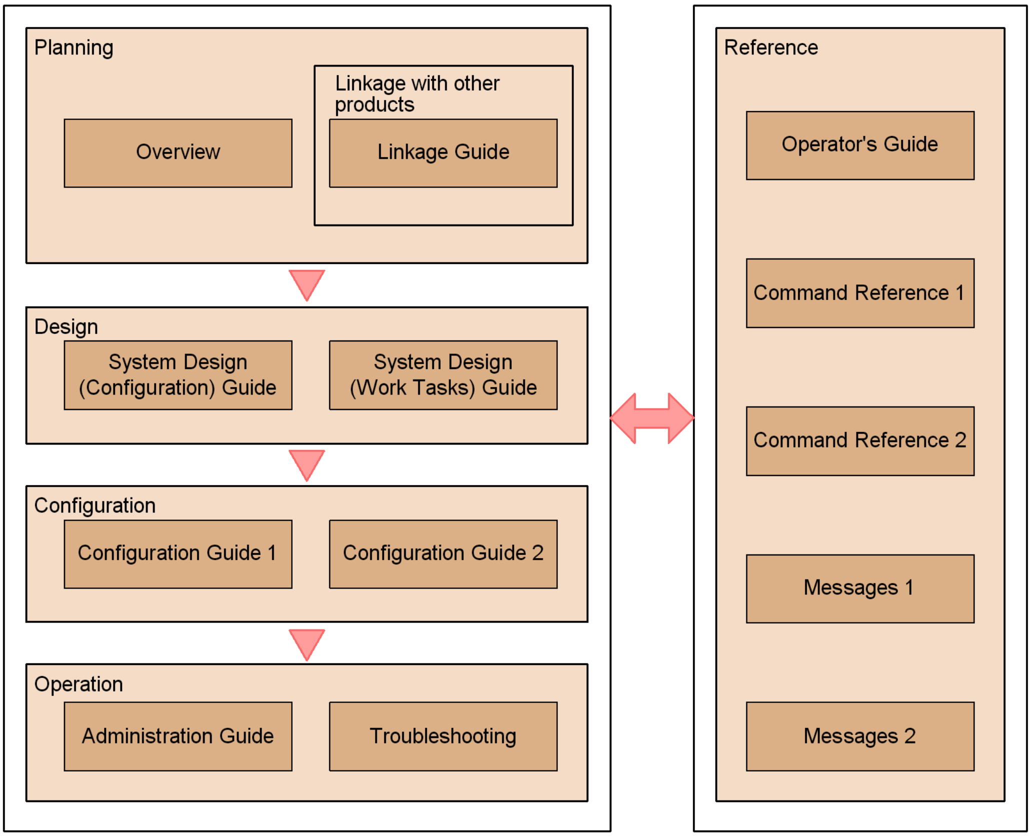
Note that *Job Management Partner 1/Automatic Job Management System 3* is not listed in the table.

No.	Manual	Contents
1	*Overview* (3020-3-S02(E))	• JP1/AJS3 features • Description of functions
2	*System Design (Configuration) Guide* (3020-3-S03(E))	• Information that must be considered when designing a system • Cautionary notes on designing a system

No.	Manual	Contents
3	*System Design (Work Tasks) Guide* (3020-3-S04(E))	• Information that must be considered when constructing jobs and jobnets • Cautionary notes on designing jobs and jobnets
4	*Configuration Guide 1* (3020-3-S05(E))	• Installation and setup procedures • Environment setup procedure by operation type
5	*Configuration Guide 2* (3020-3-S06(E))	• Description of environment setting parameters
6	*Administration Guide* (3020-3-S07(E))	• Information required to operate a system • Know-how useful for JP1/AJS3 operation
7	*Troubleshooting* (3020-3-S08(E))	• How to troubleshoot errors • Data required when an error occurs
8	*Operator's Guide* (3020-3-S09(E))	• How to operate JP1/AJS3 - View • How to operate JP1/AJS3 Console View • Description of windows and dialog boxes
9	*Command Reference 1* (3020-3-S10(E))	• Command syntax
10	*Command Reference 2* (3020-3-S11(E))	• Syntax of commands used for setup and special operations • Syntax and coding examples of information definition files
11	*Linkage Guide* (3020-3-S12(E))	• Description of functions that can be used when linked with other products and the setup method
12	*Messages 1* (3020-3-S13(E))	• Messages output by JP1/AJS3 (messages beginning with KAVC to KAVT)
13	*Messages 2* (3020-3-S14(E))	• Messages output by JP1/AJS3 (messages beginning with KAVU to KNAD)

Use the following illustration and table as a guide to determine the manuals you need to read.

Organization of JP1/AJS3 manuals



Purpose	Required reading	Read as necessary
To learn about JP1/AJS3's functionalities	• *Overview* (3020-3-S02(E))	• *Linkage Guide* (3020-3-S12(E))
To configure a system (including installation and setup) that automatically runs jobs	• *System Design (Configuration) Guide* (3020-3-S03(E)) • *Configuration Guide 1* (3020-3-S05(E))	• *Configuration Guide 2* (3020-3-S06(E)) • *Linkage Guide* (3020-3-S12(E))
To design work tasks that will be automated (including job definitions and schedule definitions)	• *System Design (Work Tasks) Guide* (3020-3-S04(E))	• *Operator's Guide* (3020-3-S09(E))

Purpose	Required reading	Read as necessary
To learn about monitoring and maintaining a running system.	• *Administration Guide* (3020-3-S07(E))	• *Troubleshooting* (3020-3-S08(E)) • *Messages 1* (3020-3-S13(E)) • *Messages 2* (3020-3-S14(E))
To learn about what action to take for problems that occur during operation.	• *Troubleshooting* (3020-3-S08(E))	• *Messages 1* (3020-3-S13(E)) • *Messages 2* (3020-3-S14(E))
To learn about operating JP1/AJS3	• *Operator's Guide* (3020-3-S09(E))	• *Command Reference 1* (3020-3-S10(E)) • *Command Reference 2* (3020-3-S11(E))

Regular expressions available in JP1/AJS3

Regular expressions can be used in some items in dialog boxes and commands. For details about regular expressions in Windows, see the *Job Management Partner 1/Base User's Guide*. For details about regular expressions in UNIX, see your UNIX documentation.

The regular expressions that you can use when executing an event job on a Windows host depend on the JP1/Base settings. For details on setting regular expressions for event job execution, see the explanation about extending the available regular expressions in the *Job Management Partner 1/Base User's Guide*.

Searching may take a long time if you often use the regular expression `.*` (which means match any character or characters). In long messages, use `.*` only where necessary. In UNIX, you can use `[^ ]*` (repeat characters other than space characters) instead of `.*` when you want to find a match other than space characters. Using `[^ ]*` reduces the search time.

About NNM linkage

JP1/AJS3 supports linkage with the following products:

- HP Network Node Manager Software version 6 or earlier
- HP Network Node Manager Starter Edition Software version 7.5 or earlier

In this manual, these products are indicated as *HP NNM*.

Note that linkage with the following products is not supported:

- HP Network Node Manager i Software v8.10

Contents

Preface i

1. Overview of Work Task Design 1

2. Job Definition and Job Execution Order Considerations 9

7. Cautionary Notes on Designing Work Tasks 167

8. Definition Pre-Check 249

Appendixes 261

Chapter

1. Overview of Work Task Design

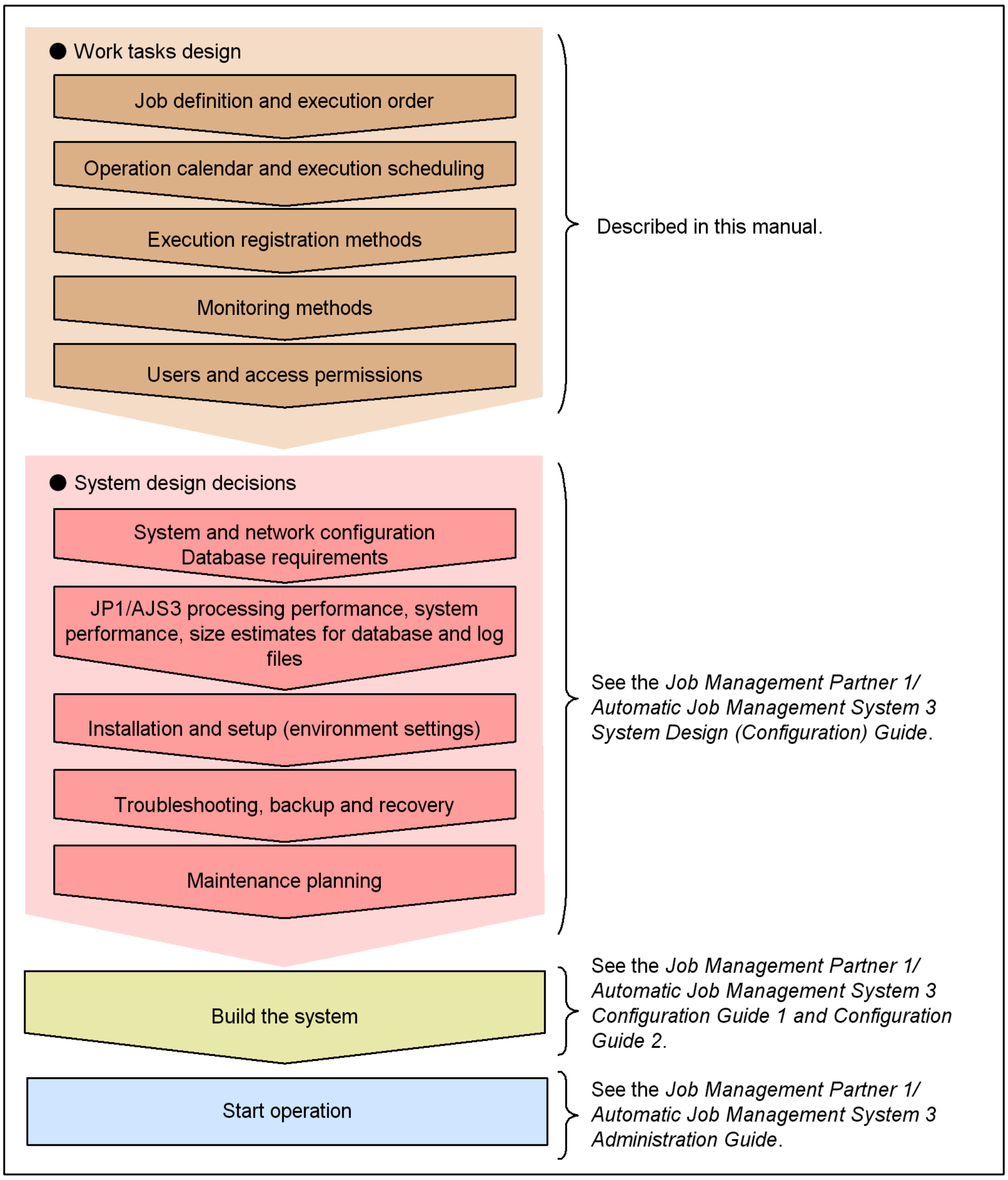
This chapter gives an overview of designing work tasks that will be automated with JP1/AJS3.

1.1 Flow of work task design

The design flow for deploying JP1/AJS3 can be broadly categorized as work task design for automating job execution under JP1/AJS3, and system design for installing and running JP1/AJS3 efficiently.

The following figure shows the basic design steps when you deploy JP1/AJS3:

Figure 1-1: JP1/AJS3 design steps


● Work tasks design
Job definition and execution order
Operation calendar and execution scheduling
Execution registration methods
Monitoring methods
Users and access permissions
Described in this manual.
● System design decisions
System and network configuration
Database requirements
JP1/AJS3 processing performance, system performance, size estimates for database and log files
Installation and setup (environment settings)
Troubleshooting, backup and recovery
Maintenance planning
See the *Job Management Partner 1/ Automatic Job Management System 3 System Design (Configuration) Guide*.
Build the system
See the *Job Management Partner 1/ Automatic Job Management System 3 Configuration Guide 1 and Configuration Guide 2*.
Start operation
See the *Job Management Partner 1/ Automatic Job Management System 3 Administration Guide*.

Note: The sequence might differ depending on the customer's environment.

This manual describes the matters to be considered when planning work task automation.

For information about system design, see the *Job Management Partner 1/Automatic Job Management System 3 System Design (Configuration) Guide*. For information about JP1/AJS3 operation, see the *Job Management Partner 1/Automatic Job Management System 3 Administration Guide*.

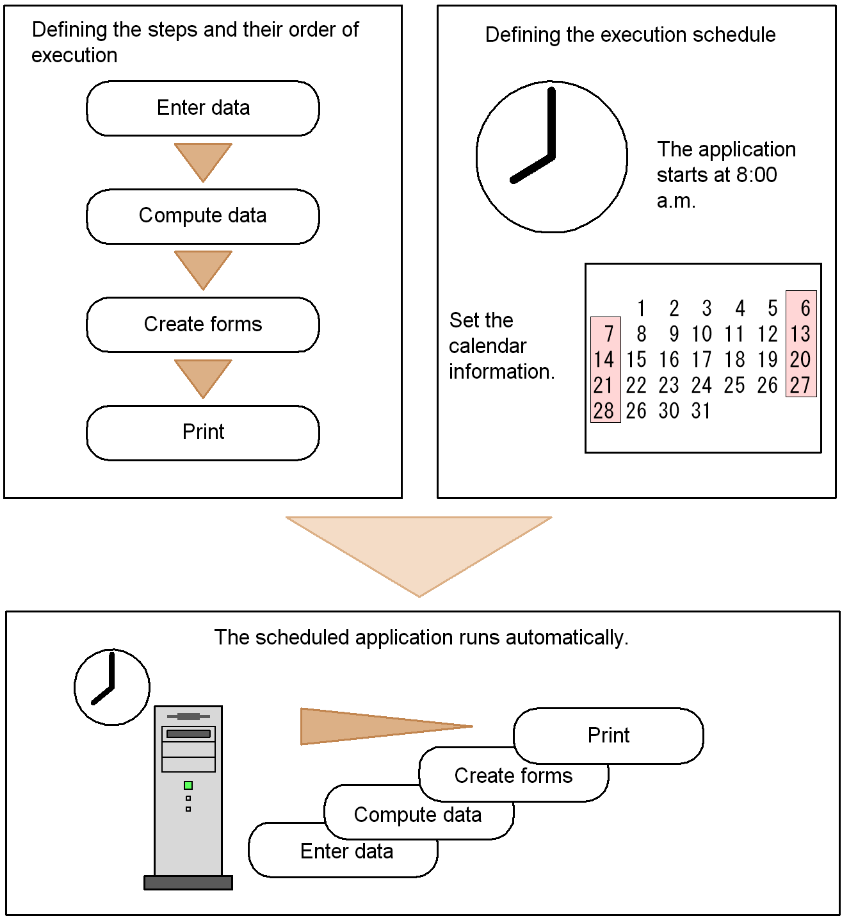
1.2 Automating work tasks

To use JP1/AJS3 to automate work tasks, you must define the following information:

- The content of each work task and its execution order
- The calendar or schedule determining when and under what conditions each work task is executed

The following figure gives an overview of work task automation:

Figure 1-2: Overview of work task automation


Defining the steps and their order of execution
Enter data
Compute data
Create forms
Print
Defining the execution schedule
The application starts at 8:00 a.m.
Set the calendar information.
1 2 3 4 5 6
7 8 9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 26 30 31
The scheduled application runs automatically.
Print
Create forms
Compute data
Enter data

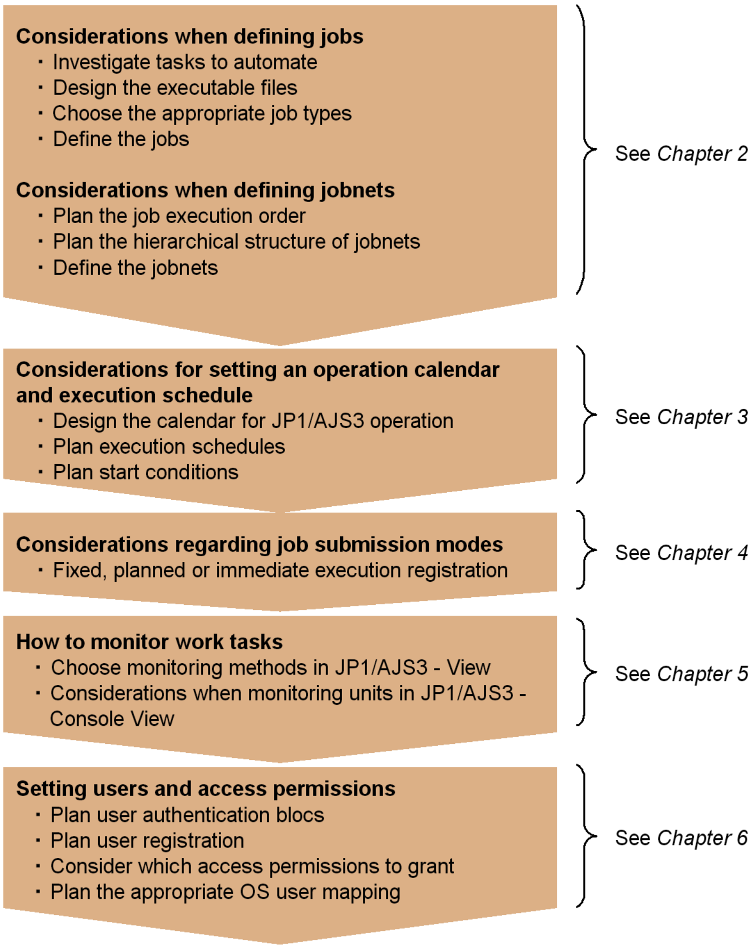
When you define work tasks for automation by JP1/AJS3, due consideration must be given to these aspects.

1.3 Flow of work task automation

JP1/AJS3 automates work tasks based on the individual components that make up each task, such as commands, application programs, and shell scripts. Each component is defined as a *job*. When assigned an execution schedule, these jobs collectively form a *jobnet*, which can be executed automatically at a date and time determined by one or more *schedule rules*.

The following figure shows the flow of work task automation in JP1/AJS3.

Figure 1-3: Flow of work task automation


Considerations when defining jobs
・Investigate tasks to automate
・Design the executable files
・Choose the appropriate job types
・Define the jobs
Considerations when defining jobnets
・Plan the job execution order
・Plan the hierarchical structure of jobnets
・Define the jobnets
See Chapter 2
Considerations for setting an operation calendar and execution schedule
・Design the calendar for JP1/AJS3 operation
・Plan execution schedules
・Plan start conditions
See Chapter 3
Considerations regarding job submission modes
・Fixed, planned or immediate execution registration
See Chapter 4
How to monitor work tasks
・Choose monitoring methods in JP1/AJS3 - View
・Considerations when monitoring units in JP1/AJS3 - Console View
See Chapter 5
Setting users and access permissions
・Plan user authentication blocs
・Plan user registration
・Consider which access permissions to grant
・Plan the appropriate OS user mapping
See Chapter 6

1.4 Key decisions when planning work task automation

The following table lists the key questions to be decided when you plan work task automation, and the relevant chapters in this manual.

Table 1-1: Work task automation decisions and relevant chapters

Item	Key decisions	Relevant chapter
1	Defining jobs	*2.1 Job definition considerations*
2	Defining jobnets	*2.2 Jobnet definition considerations*
3	Considerations for setting an operation calendar and execution schedule	*3. Operation Calendar and Execution Schedule Considerations*
4	How to register jobnets for execution	*4. Execution Registration Method Considerations*
5	How to monitor work tasks	*5. Monitoring Method Considerations*
6	Setting users and access permissions	*6. Access Permission Considerations*
7	Consideration of cautionary notes associated with work task design	*7. Cautionary Notes on Designing Work Tasks*

For information about JP1/AJS3 functions, see the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.

Chapter

2. Job Definition and Job Execution Order Considerations

In JP1/AJS3, users can automate work tasks by defining them as jobs and jobnets, and setting schedules and conditions that govern when the task should start.

This chapter describes the considerations necessary for constructing the jobs and jobnets that are required for automating work tasks with JP1/AJS3.

2.1 Job definition considerations

The individual operations in an automated work task are carried out in the form of jobs. A job is a group of commands, shell scripts, or application programs, and is the smallest building block of any automated task. Considerations include how best to pare down the steps involved in a work task, and the design of executable programs to achieve the required processing.

2.1.1 Investigating tasks to automate

You must investigate the work tasks that you want to automate with JP1/AJS3. In addition to repetitive work tasks executed on a fixed daily or monthly schedule, for example, JP1/AJS3 supports the automation of work tasks whose processing changes dynamically depending on the result of the preceding process. Non-regular work tasks which are not executed at a specific date or time can be set to execute automatically in response to a particular event, such as when a file is updated or a specific event is received.

To automate work tasks with JP1/AJS3, users should be familiar with the functionality that JP1/AJS3 has to offer. For information about JP1/AJS3 functions, see the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.

To help users get started with the work task automation process, this manual gives an introduction to the specific JP1/AJS3 functions they are most likely to use. For an overview of the JP1/AJS3 functions for automating work tasks, see *2.3.2 JP1/AJS3 functions useful for work task automation*.

2.1.2 Designing executable files

JP1/AJS3 processes work tasks based on *executable files*, such as batch files and shell scripts. This section identifies the considerations involved in designing executable files to carry out work tasks.

When designing executable files, consider the following points:

(a) Use a format supported by JP1/AJS3

Processing cannot be automated in JP1/AJS3 unless it complies with a supported executable file format. The following table lists the types of executable files that can be used with JP1/AJS3.

Table 2-1: Executable file types supported by JP1/AJS3

Host type	Supported executable files
Windows host	• `.exe` files • `.com` files • `.cmd` files • Batch files (`.bat`) • Script files (`.spt`) created in JP1/Script • Data files associated with an application program (as indicated by the extension)
UNIX host	• Shell scripts • Executable files

(b) Define one command per executable file

We recommend that you define only one command in each executable file.

Dividing work task processing in this way allows you to see which processing ends normally and which does not. Because you can identify the specific process that caused a work task to terminate abnormally, you can then either rerun the work task from that process, or rerun just the process that ended abnormally.

(c) Choose processing that does not generate on-screen messages or require a response

JP1/AJS3 runs executable files in the background. Therefore, do not use an executable file that displays a window or message and then waits for a user response. However, JP1/AJS3 can use executable files if, for example, the **Yes** button is automatically selected on a displayed message dialog box and the process proceeds.

(d) Choose processing that outputs a return code

JP1/AJS3 judges whether processing has ended normally or abnormally based on the return code of the executable file. Therefore, create executable files whose return codes reflect the processing result.

For processing by batch file:

> Use the `jp1exec` or `jp1exit` command to ensure that the executable programs in the batch file output return codes that reflect the processing result. To enable analysis of the cause when a command ends abnormally, ensure that the return code that caused the abnormal end is output unaltered.
>
> For details on the `jp1exit` and `jp1exec` commands, see *2. Commands* in the manual *Job Management Partner 1/Automatic Job Management System 3 Command Reference 1*.

For processing by shell script:

With shell scripts, ensure that a return code that reflects the processing result is output, as in the following example:

```
   :
RC=$?
exit $RC
```

Set the threshold values that determine the ranges for *end with warning* or *abnormal end* results.

(e) Choose processing with a short execution time

If the execution time for a single process is too long, you might not be able to determine whether the work task is proceeding normally, or processing has stopped due to an error of some kind. As a guide, make sure that no individual process takes longer than two hours.

(f) Use meaningful file and folder names

We recommend that you create a naming convention for executable files and the folders and directories where executable files are saved. Using consistent names with work tasks, processes, and other elements used with JP1/AJS3 makes it easier to manage work tasks as a whole. One effective method is to define a set of identifiers that indicate work task names, process names, processing cycles, execution locations, and so on.

One-byte alphanumeric characters and the following symbols can be used in the names of work tasks and processes used with JP1/AJS3.

```
! # $ % + @ - (hyphen) . (period) _ (underscore)
```

Note that some symbols have special meanings in command interpreters such as UNIX shells. Because symbols might cause unpredictable results, avoid using them for work task and process names.

In addition, do not use the following characters or symbols:

- Periods (.) or at marks (@) at the beginning of work task or process names
- Machine-dependent characters

(g) Save executable files on the host where the processing is to be executed

Save the executable files (such as commands and batch files) on the host where you intend the processing to be executed.

(h) Other points

- When specifying commands and input or output files in executable files, specify an absolute path including the folder names.

- To enable analysis of the cause when a command ends abnormally, output an arbitrary message to the standard error output by, for example, using an `echo` command. It is useful for this message to include information such as the cause of the error and how to recover from it, the work task name, and notes on rerunning the work task. You can view these messages in the Execution Result Details dialog box in JP1/AJS3 - View. For details about the Execution Result Details dialog box, see *15.3.35 Execution Result Details dialog box* in the *Job Management Partner 1/Automatic Job Management System 3 Operator's Guide*.
- As a general rule, if you use environment variables, you should define them in the executable file.
- When using a shell script, declare the shell used at the head of the file. For details about the supported shells, also see *2.5.3 OS user environment when a job is executed* in the *Job Management Partner 1/Automatic Job Management System 3 System Design (Configuration) Guide*.

2.1.3 Job type considerations

Because there are many different types of jobs, you must order the jobs that you are defining according to the type of process required. JP1/AJS3 provides the following job types:

- Standard job
- OR job
- Judgment job
- Event job
- Action job
- Custom job

The characteristics of each type of job are described below.

(a) Standard job

This type of job executes processing by specifying an executable file. The following table lists the types of executable files that can be specified in each type of standard job.

Table 2-2: Executable files specifiable in a standard job

No.	Job type	Description	Specifiable executable files
1	Unix job	Process executed on a UNIX host.	• Executable file • Shell script

No.	Job type	Description	Specifiable executable files
2	PC job	Process executed on a Windows host.	• `.exe` file • `.com` file • `.cmd` file • `.pif` file • `.bat` file • `.spt` file# (script file created with JP1/Script) • Data file that has a file type (extension) associated with the application
3	QUEUE job	Process executed by submitting a job to a specified queue. This job type is used for linking with other systems (such as JP1/NQSEXEC).	• Executable file • Shell script • `.exe` file • `.com` file • `.cmd` file • `.pif` file • `.bat` file • `.spt` file# (script file created with JP1/Script) • Data file that has a file type (extension) associated with the application

#

To execute an `.spt` file, JP1/Script must also be installed on the host running the job.

(b) OR job

For an *OR job*, you define a number of jobs (*event jobs*) which will monitor the system for a specific event. You define these event jobs as the preceding jobs of the OR job. If any one of the monitored events occurs, the OR job executes its succeeding job.

For details, see *3.1.1(1)(b) OR job* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.

(c) Judgment job

A judgment job checks whether a given condition is satisfied.

The judgment can be based on the following conditions:

- The return code of the preceding job
- Whether a specific file exists
- The result of a comparison between variables

For details, see *3.1.1(1)(c) Judgment job* in the manual *Job Management Partner 1/ Automatic Job Management System 3 Overview*.

(d) Event job

An *event job* monitors an event occurring in the system. You can define an event job as the start condition for a job flow or jobnet, so that the particular job or jobnet will be executed only when the monitored event occurs.

The following table describes each event job.

Table 2-3: Types of event jobs

No.	Event job name	Description
1	Receive JP1 event job	This job terminates when it receives a specific event from JP1/ Base.
2	Monitoring files job	This job terminates when a specific file is created, deleted, or updated.
3	Receive mail job	This job terminates when it receives a specific email.
4	Monitoring log files job	This job is linked with the log file trapping of JP1/Base, and terminates when specific information is written to the specified log file.
5	Monitoring event log job	This job is linked with the log file trapping of JP1/Base, and terminates when specific information is written to the Windows event log file.
6	Interval control job	This job terminates when the specified time period elapses.
7	Receive MQ message job	This job terminates when it receives a specific message from TP1/Message Queue or MQSeries.
8	Receive MSMQ message job	This job terminates when it receives a specific message from MSMQ.

Note

JP1/AJS3 must be linked with the appropriate program to use a Receive mail job, Receive MQ message job, or Receive MSMQ message job. For details, see the *Job Management Partner 1/Automatic Job Management System 3 Linkage Guide*.

Operation of event jobs is independent of JP1 user permissions and the authority level defined for the job (by owner, JP1 resource group, or by the user executing the job). In Windows, because event job operation is governed by the account rights to the JP1/ AJS3 service, JP1/AJS3 service rights must be set in advance.

(e) Action job

An *action job* executes a specific process. You can combine an action job with an event job to execute a process (action) when an event occurs. Typical actions might be to send a JP1 event, an email message, or a status notification.

The following table describes each action job.

Table 2-4: Types of action jobs

No.	Action job name	Description
1	Send JP1 event job	Sends a JP1 event to the event service of JP1/Base.
2	Send mail job	Sends an email message.
3	OpenView Status Report job	Sends a status to HP NNM.
4	Send MQ message job	Sends a message to TP1/Message Queue or MQSeries.
5	Send MSMQ message job	Sends a message to MSMQ.

Note

JP1/AJS3 must be linked with the appropriate program to use a Send mail job, Send MQ message job, Send MSMQ message job, or OpenView Status Report job. For details on program linkage, see the *Job Management Partner 1/ Automatic Job Management System 3 Linkage Guide*.

(f) Custom job

A *custom job* is used to link an external program with JP1/AJS3 to execute processing.

For details, see *3. Adding Custom Jobs* in the *Job Management Partner 1/Automatic Job Management System 3 Linkage Guide*.

2.1.4 Job definition considerations

This section describes the considerations that apply when you define the content of a job.

For information about the items that can be defined, see the explanations of the Define Details dialog boxes for each job in *15. Windows and Dialog Boxes* in the *Job Management Partner 1/Automatic Job Management System 3 Operator's Guide*.

(a) Execution agent

You can specify the name of the agent that is to execute a job. An *execution agent* is a logical name representing the agent host to which the job is to be distributed. Based on the execution agent information defined in the manager host, the manager maps the execution agent specified in the job to the physical host name of the agent host, and distributes the job accordingly.

By specifying an execution agent group, you can distribute the processing load among a group of execution agents. The manager distributes jobs among the execution agents according to their assigned priorities.

You can specify an execution agent for the following job types:

- PC jobs
- Unix jobs
- Event jobs#
- Action jobs
- Custom jobs

#

Event jobs do not support operations that use execution agent groups. For details, see *7.6 Notes on using event jobs*.

(b) Execution priority

You can set the priority of the processes that start when JP1/AJS3 runs an executable file. Determine whether you need to set execution priorities for jobs. For example, you might want to prioritize the processing of a certain executable file so that it finishes earlier.

JP1/AJS3 assigns a low default execution priority when none is specified for a job. This is to prevent some jobs from monopolizing system resources and taking down the entire JP1/AJS3 system, for example when a job executed from JP1/AJS3 goes into a loop. However, jobs with a low execution priority also have a lower CPU priority. This means that while the CPU is being monopolized by processes with a high execution priority, jobs executed from JP1/AJS3 can be forced to wait for long periods until the CPU becomes available. As a result, jobs might take a long time to finish, or job processes might remain in a stopped state. If one of these job processes places a lock on a resource while waiting for the CPU to become available, this will affect the execution of other processes that are waiting for the same resource to become available.

Such problems tend to occur more markedly in system configurations that have relatively high CPU usage, such as single-processor systems and configurations where multiple processes with high execution priorities are processed at the same time. Determine whether your system environment or manner of operation requires you to raise job execution priorities.

(c) End judgment

Consider which method to use to judge whether a process has ended successfully. You can judge the execution results of standard jobs. JP1/AJS3 provides five methods of end judgment:

- Always end normally

- Always end abnormally
- End normally if a specified file exists on the execution target host
- End normally if a specified file is updated while the job is running
- Set thresholds (warning threshold and abnormal threshold) and compare them with a job's return code

For judgment based on a threshold, you can evaluate whether a job ended normally, ended with a warning, or ended abnormally, by locating the job's return code relative to the threshold. The job ends normally if the return code at termination falls below the warning threshold, and ends with a warning if the return code exceeds the warning threshold but falls below the abnormal threshold. If the return code exceeds the abnormal threshold, the job ends abnormally.

Return values are interpreted as unsigned integers. For example, `-1` is handled as `4,294,967,293` in Windows, and as `255` in UNIX.

(d) Transfer files

With a standard job, if the executable files required to execute the job are not already present in the agent host, you can transfer the files from the manager host or the host where the command is executed to the agent host. These files must be in text format.

(e) Target service

To use a queueless job, specify **Queueless Agent** as the job's execution target service. The default is **Standard**.

Queueless jobs offer improved performance compared with normal queued jobs, and more jobs can be executed within a given period of time. For information on queueless jobs, see *10.5 Queueless jobs* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.

However, because queueless jobs cannot be associated with an execution agent or agent group, we recommend the use of ordinary queued jobs in most circumstances.

(f) Timeout period

If you set a timeout period for a job, the job will be canceled when a specific length of time has passed since the job started. Decide the timeout period to impose in situations where the processing does not complete for some reason. By timing-out job execution, you can investigate the cause of the problem, perform processing to notify the system administrator, and execute specific processing to be performed only in the event of abnormal termination.

(g) End delay monitoring

By setting the time required to execute a job, you can monitor for end delays based on how much time has passed since the job started. For details, see *5. Monitoring Method Considerations*.

(h) JP1 resource groups

Consider the user access permissions you need to assign to each job. For details, see *6. Access Permission Considerations*.

(i) Cautionary notes

- Do not use a space character at the end of a comment in a job definition. All space characters after a comment are treated as invalid.
- Do not use machine-dependent characters in any part of a job definition, because these characters might not be displayed correctly.

2.2 Jobnet definition considerations

You can define a jobnet by grouping a set of jobs that achieve a work task, and assigning an execution order to the jobs. When constructing a jobnet, you need to consider the order of job execution, and the hierarchical structure of the jobnet to facilitate job management in JP1/AJS3.

2.2.1 Job execution order considerations

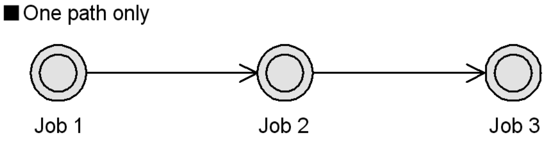
After you have given thought to the construction of each job, you then need to consider the execution order (job flow) of the jobs that make up each work task. A chart indicating the desired job flow and hierarchical structure can be a useful tool for defining jobs and jobnets.

The following is an example of job flow creation.

(1) Work task with only one processing path

The following figure shows an example of creating a job flow for three jobs, Job 1, Job 2, and Job 3, designed to execute in a specific order. This job flow has only one processing path.

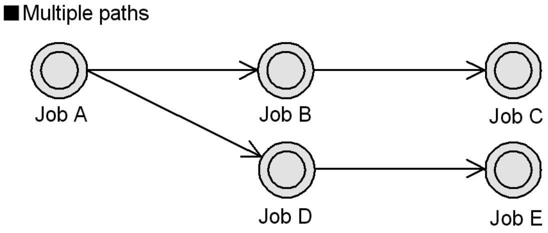
Figure 2-1: Job flow with a single processing path



(2) Work task with more than one processing path

The following figure shows an example of creating a job flow with more than one processing path.

Figure 2-2: Job flow with multiple processing paths



Here, when Job A is executed, processing diverges into two paths: Job A-Job B-Job C, and Job A-Job D-Job E.

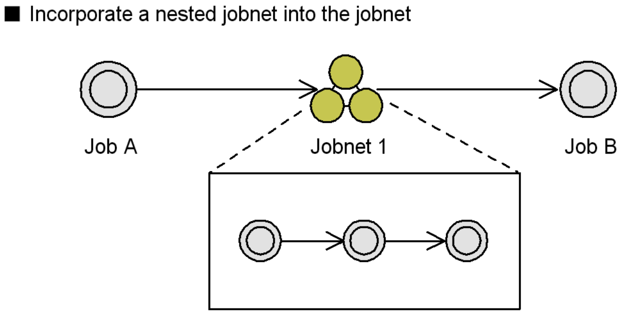
(3) Work task with nested jobnets

You can incorporate a jobnet into a job flow. Examples of using a nested jobnet are shown below.

(a) Nesting a jobnet

The following figure shows an example of incorporating a jobnet into a job flow.

Figure 2-3: Example of nested jobnet



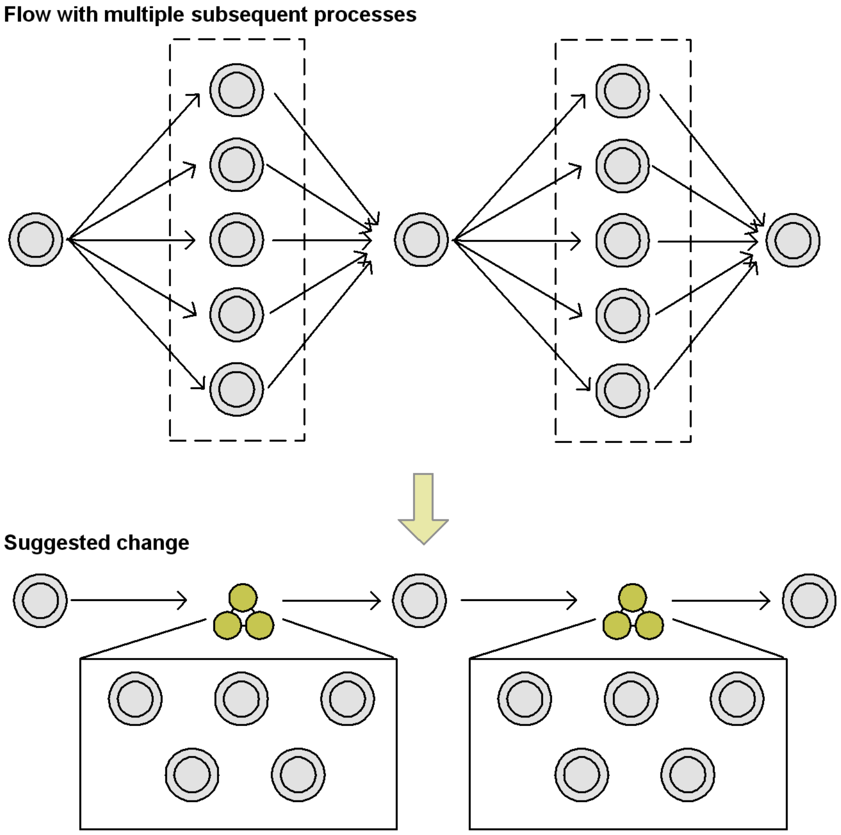
Job A is executed first, followed by the jobs defined in Jobnet 1 when Job A finishes. After Jobnet 1 finishes processing, Job B is executed.

(b) Grouping multiple jobs

It is not advisable to create jobs flows with multiple succeeding jobs. Multiple jobs brought into one jobnet are easier to administer.

The following figure shows an example of grouping multiple jobs in a nested jobnet.

Figure 2-4: Grouping multiple jobs in nested jobnet

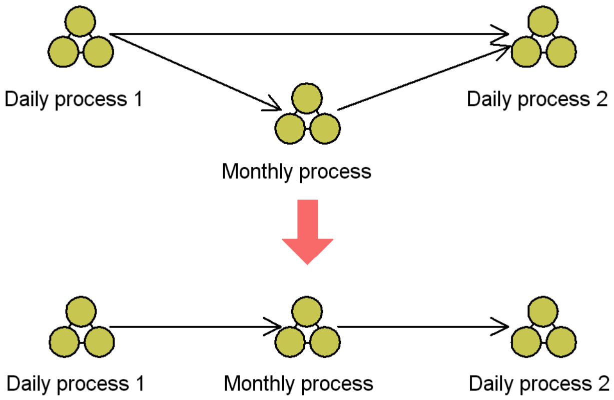


(c) Merging two paths into one

You can use nested jobnets to merge two processing paths into one.

The following figure shows an example of merging the two paths Daily Process 1-Daily Process 2, and Daily Process 1-Monthly Process-Daily Process 2, into a single flow.

Figure 2-5: Incorporating a jobnet into a job flow

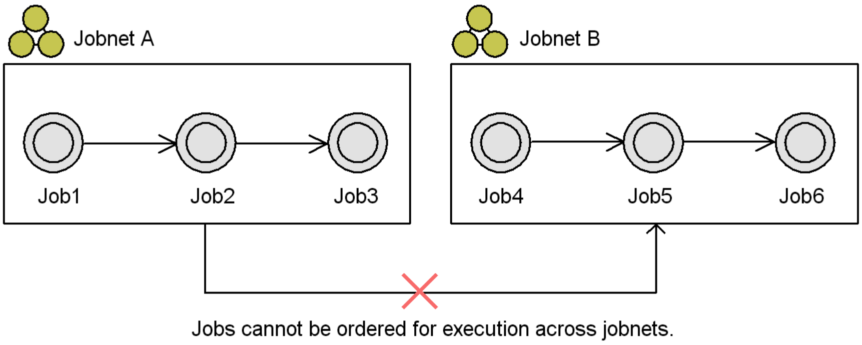


Daily Process 1 and Daily Process 2 are executed daily, and Monthly Process is executed only once a month. Because JP1/AJS3 skips jobnets that are not scheduled for execution on that day, you can incorporate all these jobnets into a single path.

(4) Sequencing jobs in different work tasks

In principle, jobs in different work tasks cannot be ordered in JP1/AJS3. When you need to associate jobs in different work tasks to create a new workflow, you can do so by splitting the jobnet, or by integrating the jobnets that contain each job.

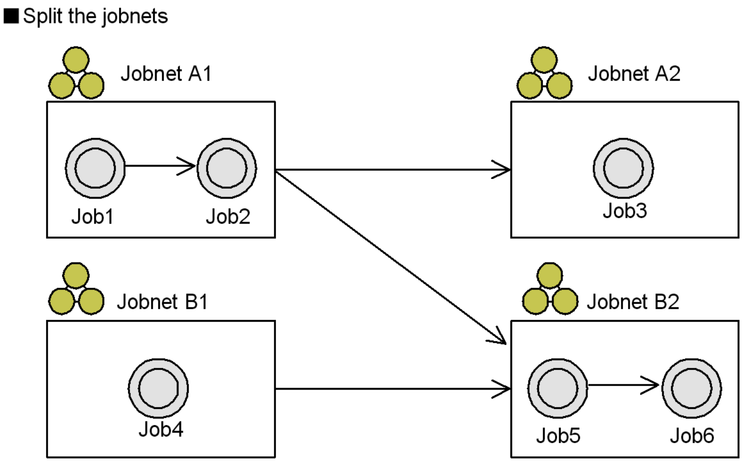
Figure 2-6: Sequencing jobs in different work tasks



(a) Splitting the jobnet

The following figure shows an example of splitting the jobnet shown in *Figure 2-6*.

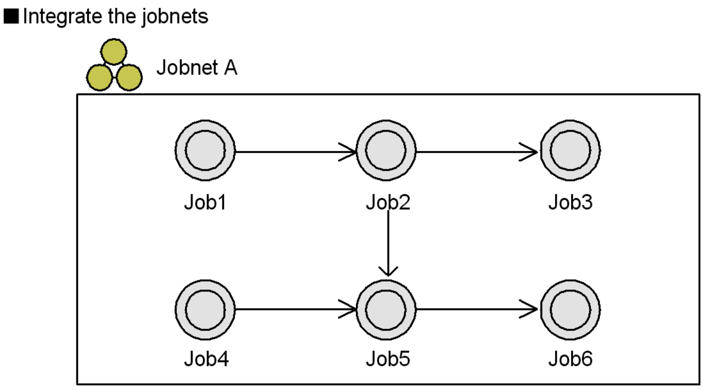
Figure 2-7: Splitting the jobnet



(b) Integrating the jobnets

The following figure shows an example of integrating the jobnets.

Figure 2-8: Integrating the jobnets



(5) Concepts for building jobnet hierarchies

The objectives and advantages of creating jobnet hierarchies are:

- Jobs and jobnets are easier to monitor.
- The job or jobnet location is easier to specify when changes are made.
- By assigning access permissions (JP1 resource groups) at the work task level, you can restrict access by those responsible for other work task groups.

- An appropriate hierarchical structure will help prevent poor start performance of jobnets and jobs.

Supplementary note

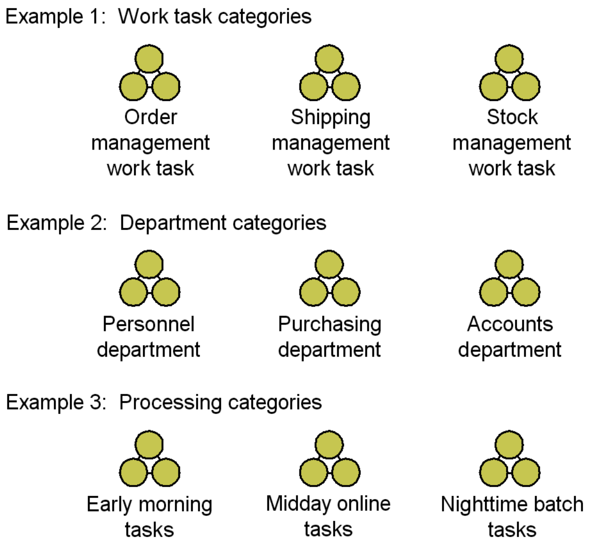
> Monitoring and other operations might be difficult if there are too many levels in the hierarchy. We recommend that you limit the number of levels to between 3 and 5.

We recommend the following method of creating a jobnet hierarchy:

1. Establish the highest-level jobnets.

   As shown in the following figure, establish the jobnet that is to occupy the highest level in each chosen category.

   Figure 2-9: Example of establishing highest-level jobnets

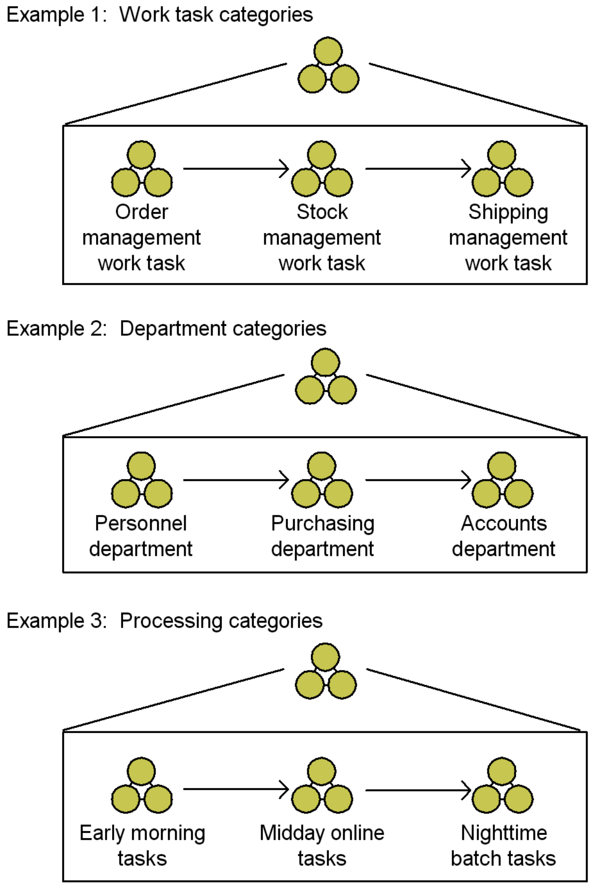

Example 1: Work task categories
Order management work task
Shipping management work task
Stock management work task
Example 2: Department categories
Personnel department
Purchasing department
Accounts department
Example 3: Processing categories
Early morning tasks
Midday online tasks
Nighttime batch tasks

   If there is no mutual order among jobnets, you can manage each of them as a highest-level jobnet.

2. If there is an order among the jobnets established in each of the chosen categories, group them into a single jobnet and assign it a hierarchy level.

   The following figure shows an example of hierarchization when there is an order among jobnets.

Figure 2-10: Hierarchization of ordered jobnets



Example 1: Work task categories

Order management work task
Stock management work task
Shipping management work task

Example 2: Department categories

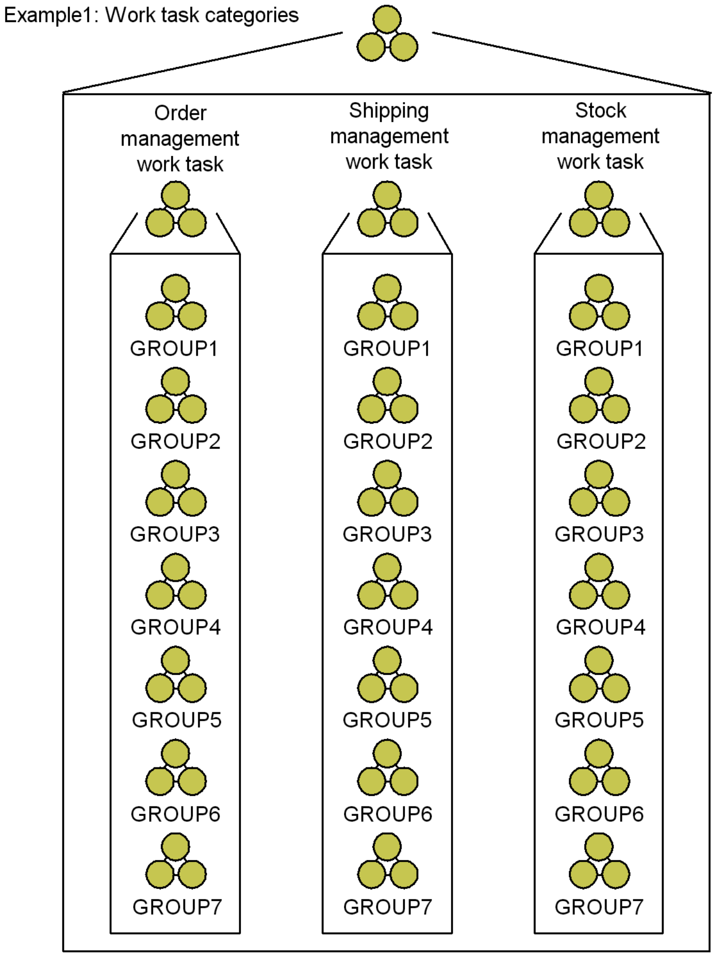
Example 3: Processing categories

3. Establish a jobnet for each processing cycle.

   If jobnets are executed in different execution cycles under the same work task, group the jobnets that have the same execution cycle.

   The following figure shows an example of grouping jobnets according to their processing cycle.

Figure 2-11: Example of establishing jobnets by processing cycle


Example1: Work task categories
Order management work task
Shipping management work task
Stock management work task
GROUP1
GROUP2
GROUP3
GROUP4
GROUP5
GROUP6
GROUP7

Decide rules like those shown in the following table for the jobnets of each processing cycle. We recommend that you include these rules as comments in the definition.

Table 2-5: Rules when jobnets are grouped by processing cycle

Jobnets in each processing cycle	Rule
GROUP1	Group of *daily*, *weekly*, *monthly*, *six-monthly*, or *yearly* work tasks that are mutually related
GROUP2	Group of *daily work tasks* executed in isolation

Jobnets in each processing cycle	Rule
GROUP3	Group of *weekly work tasks* executed in isolation
GROUP4	Group of *monthly work tasks* executed in isolation
GROUP5	Group of *six-monthly work tasks* executed in isolation
GROUP6	Group of *yearly work tasks* executed in isolation
GROUP7	Group of *irregular work tasks* executed in isolation

4. Establish the lowest-level jobnets.

   Next, at the lowest level of the jobnets that have been grouped according to their processing cycles, establish jobnets assigned the names listed below. We recommend that you use names consisting of one-byte alphanumeric characters. This will enable command-line execution of jobs and jobnets, and use of regular expressions with JP1/IM's automated action functionality.

   *xxxxxx*DN

   *xxxxxx*WN

   *xxxxxx*MN

   *xxxxxx*HN

   *xxxxxx*YN

   *xxxxxx*RN

   Legend:

   D: Indicates a jobnet executed in a "daily" cycle.

   W: Indicates a jobnet executed in a "weekly" cycle.

   M: Indicates a jobnet executed in a "monthly" cycle.

   H: Indicates a jobnet executed in a "six-monthly" cycle.

   Y: Indicates a jobnet executed in a "yearly" cycle.

   R: Indicates a jobnet executed at irregular intervals.

   N: Indicates a jobnet.

5. Create jobs.

   Finally, create jobs assigned the following names:

   *xxxxxx*DJ

*xxxxxx*WJ

*xxxxxx*MJ

*xxxxxx*HJ

*xxxxxx*YJ

Legend:

D: Indicates a job executed in a "daily" cycle.

W: Indicates a job executed in a "weekly" cycle.

M: Indicates a job executed in a "monthly" cycle.

H: Indicates a job executed in a "six-monthly" cycle.

Y: Indicates a job executed in a "yearly" cycle.

J: Indicates a job.

2.2.2 Corrective action when job execution fails

When you have decided the order of job execution, you must then think about the corrective action to take when a job fails to execute properly.

Consider the following:

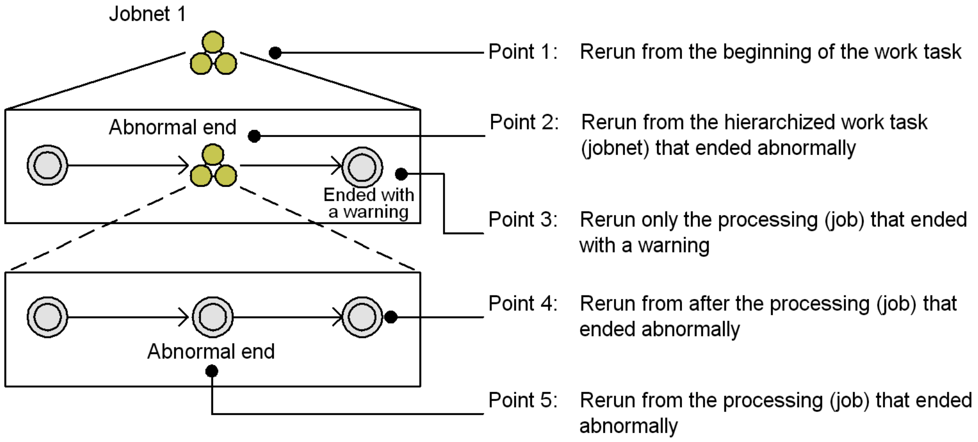
- What processing to execute automatically after a job ends abnormally
- How to run the job again

JP1/AJS3 judges whether processing has ended normally or abnormally based on the return code of the executable file. This allows processing, such as sending an email informing the system administrator of the problem, to be executed automatically when an abnormal return code is output.

If you choose to rerun a failed job, you must consider whether to re-execute all the jobs from the beginning of the job flow, or just some of them.

The following figure lists the considerations for re-executing a job.

Figure 2-12: Points from which jobs can be re-executed


Jobnet 1
Abnormal end
Ended with a warning
Abnormal end
Point 1: Rerun from the beginning of the work task
Point 2: Rerun from the hierarchized work task (jobnet) that ended abnormally
Point 3: Rerun only the processing (job) that ended with a warning
Point 4: Rerun from after the processing (job) that ended abnormally
Point 5: Rerun from the processing (job) that ended abnormally

Whichever method you decide on, you can put a job on hold temporarily before rerunning it. Because execution will not proceed until the "on hold" status is released, you can temporarily alter the execution schedule or suspend execution.

If you choose to re-execute only some jobs, consider whether you want to re-execute the job that ended abnormally, or re-start execution from the next job.

2.2.3 Jobnet definition considerations

This section describes the matters you need to consider when defining jobnets.

For details of the information you can define in a jobnet, see the description of the Define Details dialog box for a root jobnet in *15. Windows and Dialog Boxes* in the *Job Management Partner 1/Automatic Job Management System 3 Operator's Guide*.

(a) Execution agent

You can specify the name of the agent that is to execute a jobnet. An *execution agent* is a logical name representing the agent host to which a job or jobnet is distributed. Based on the execution agent information defined in the manager host, the manager maps the execution agent specified in the job or jobnet to the physical host name of the agent host, and distributes the jobs accordingly.

By specifying an execution agent group, you can distribute the processing load among a group of execution agents. The manager distributes jobs among the execution agents according to their assigned priorities.

(b) Concurrent execution and schedule option

For jobnets that are executed periodically, consider whether you want JP1/AJS3 to run multiple instances of the process concurrently if the process has not completed by the time the next scheduled start time arrives. For details, see *3.3.3 Concurrent execution* *and schedule option* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.

(c) Number of logs to keep

Consider how many generations of execution logs you need to keep for the jobnet. For details, see *4.2.3 Jobnet generation management* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.

(d) Timeout period

When a jobnet fails to execute by its scheduled start time, consider how long you want the jobnet to wait to execute before it times out. Jobnets that time out while waiting to execute are placed in *Skipped so not executed* status. For details, see the description of timeout periods in *3.1.1(2) Jobnets* in the manual *Job Management Partner 1/ Automatic Job Management System 3 Overview*.

(e) Jobnet monitoring

By setting the time required to execute a jobnet, you can monitor for end delays based on how much time has passed since the jobnet started. For details, see *5. Monitoring Method Considerations*.

(f) Execution order control

You can assign an execution order to the jobnets at the highest level of the hierarchy to facilitate job management. For details, see *2.2.4 Using jobnet connectors to control the order of root jobnet execution*.

(g) JP1 resource group

Consider the access permissions you need to assign to each jobnet. For details, see *6. Access Permission Considerations*.

(h) Cautionary notes

- Do not use a space character at the end of a comment in a jobnet definition. All space characters after a comment are treated as invalid.
- Do not use machine-dependent characters in any part of a jobnet definition, because these characters might not be displayed correctly.

2.2.4 Using jobnet connectors to control the order of root jobnet execution

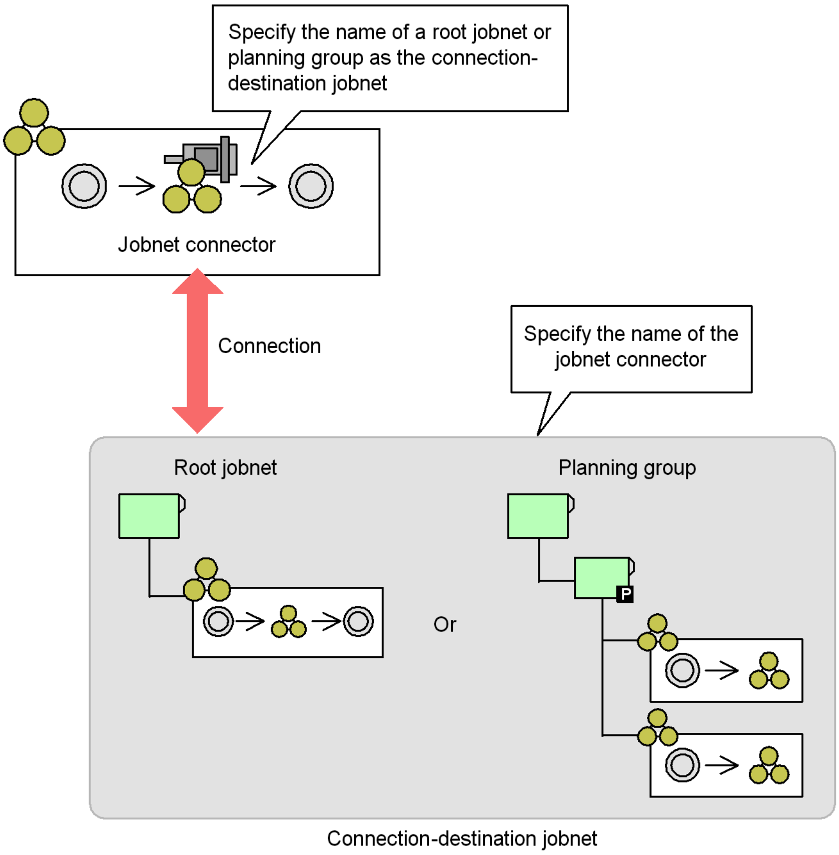
Unlike other units such as jobs and nested jobnets, you cannot arrange root jobnet units in a order. However, you can control their execution order through a unit called a *jobnet connector*.

(1) Overview of using jobnet connectors to control the root jobnet execution order

A jobnet connector links to a root jobnet and controls the order in which it executes. By forcing the root jobnet to wait for another root jobnet to end, or synchronizing its startup process with that of another root jobnet, the jobnet connector can control the execution timing of the root jobnet and its place in the execution order.

The following figure shows the relationship between the jobnet connector and the root jobnet whose execution order it controls.

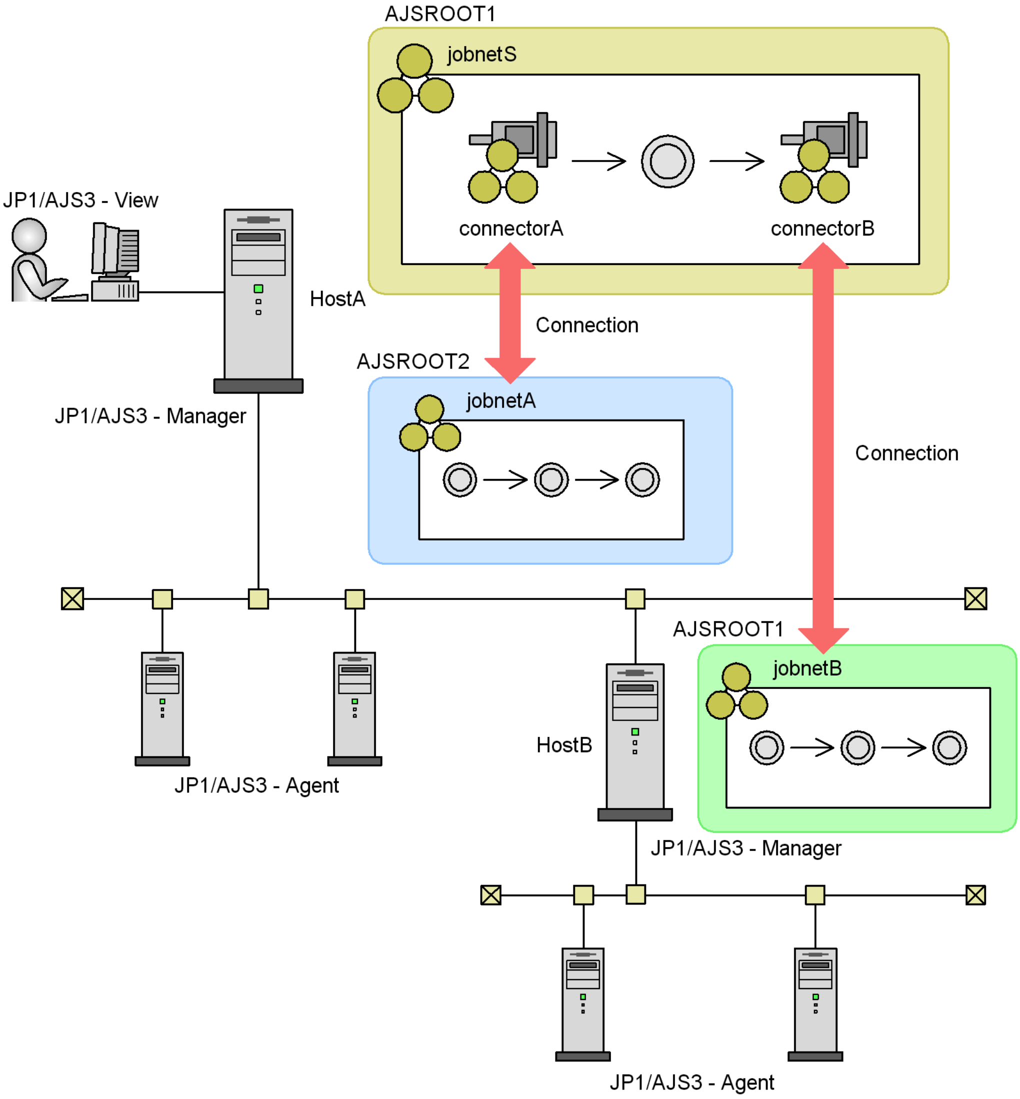
Figure 2-13: Relationship between jobnet connector and root jobnet


Specify the name of a root jobnet or planning group as the connection-destination jobnet
Jobnet connector
Connection
Specify the name of the jobnet connector
Root jobnet
Planning group
P
Or
Connection-destination jobnet

The jobnet connector is defined as a unit under a jobnet.

A jobnet connector can control the execution order of a root jobnet, or a root jobnet directly under a planning group. It can also control root jobnets that are managed by different scheduler services, as shown in the figure below.

Figure 2-14: Execution order control of root jobnets managed by different scheduler services



The root jobnet associated with the jobnet connector is called the *connection-destination jobnet*.

To use jobnet connectors to control the execution order of root jobnets:

1. Create the jobnets and define the jobnet connectors.

2. Define the connections between the jobnet connectors and the jobnets whose

execution order they control.

Specify the name of the connection-destination jobnet in the definition of the jobnet connector. If the jobnet whose execution order you are controlling is a root jobnet, specify a root jobnet name. If the target is a root jobnet in a planning group, specify the planning group name.

In the definition of the connection-destination jobnet, specify the corresponding jobnet connector name.

Supplementary note

You can simplify the task of manually entering the definitions by using the **Save as Jobnet Connector** or **Auto-create Jobnet Connector** menu commands in JP1/AJS3 - View. However, you must manually enter the definitions if the jobnets in question are managed by different scheduler services.

3. Register the jobnets for execution.

For details on how to define jobnets and jobnet connectors, see *5. Defining Jobnets* in the *Job Management Partner 1/Automatic Job Management System 3 Operator's Guide*. For details on how to register a jobnet for execution, see *7. Executing Jobnets* in the *Job Management Partner 1/Automatic Job Management System 3 Operator's Guide*.

Cautionary notes

- You cannot use jobnet connectors with versions of JP1/AJS - View earlier than 08-10, or when connecting to a version of JP1/AJS - Manager earlier than 08-10.
- If you are using version 08-10 of JP1/AJS - View, or connecting to version 08-10 of JP1/AJS - Manager, you cannot use jobnet connectors to control the execution order of root jobnets that are under the control of different scheduler services. This capability is offered with version 08-50 or later.
- A jobnet connector must be defined under a root jobnet or nested jobnet.
- You must specify a root jobnet or planning group as the connection target jobnet for a jobnet connector.
- Do not define a jobnet connector under a connection-destination jobnet. If you do so, an error will occur when the jobnet is registered for execution.
- Do not assign a start condition to a jobnet for which a jobnet connector is defined, or to the connection-destination jobnet. If you do so, an error will occur when the jobnet is registered for execution.
- You cannot define a jobnet connector as a dependent unit or recovery unit.
- JP1/AJS3 allows you to define a jobnet connector under a dependent jobnet

or recovery jobnet. However, we recommend that you avoid doing so due to the complicated relationships this type of arrangement creates.

- You cannot create a new jobnet connector or delete an existing one while execution is suspended.

(2) Rules governing connections between jobnet connectors and connection-destination jobnets

When a jobnet with a jobnet connector and its connection-destination jobnet are registered for execution, a relationship is established between generations that share an execution date. You can control the execution order of related root jobnets, for example by having one wait for the other to finish, or having them start simultaneously.

(a) Connection rules

Relationships are established between generations of a jobnet for which a jobnet connector is defined and those of its connection-destination jobnet, when both of the following conditions are satisfied:

- The generations share the same execution date.
- The generations are not part of a connected relationship with another generation.#

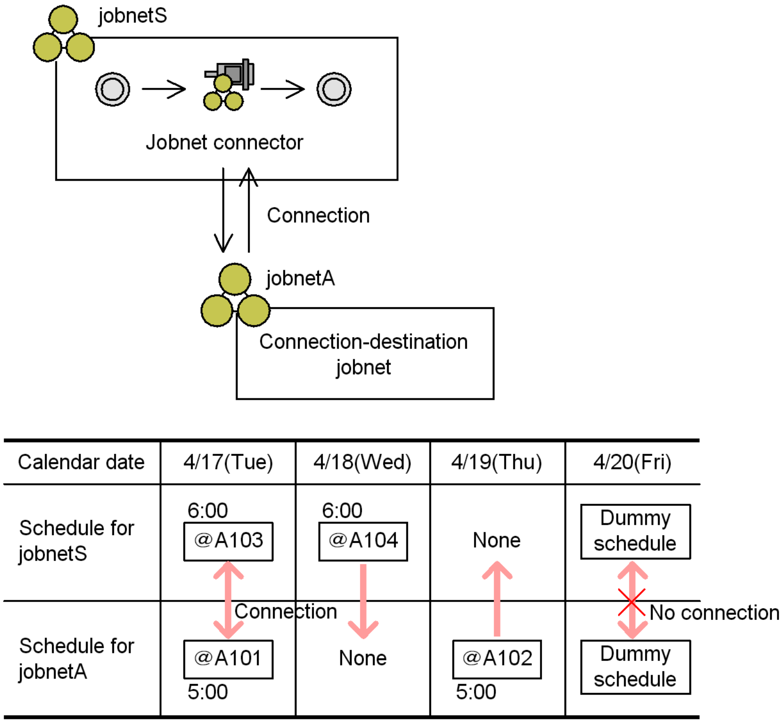
#

The relationship between the jobnets is not formalized until both have started, or one has skipped execution due to being placed in *skipped so not executed* status.

When a jobnet is scheduled for execution more than once on the same day, a relationship is established between generations with the earliest scheduled execution times among those that meet the above conditions. However, no relationship is formalized while either jobnet's schedule is still a dummy schedule.

The following figure shows an example of the connections established between a root jobnet for which a jobnet connector is defined (jobnet S) and its connection-destination jobnet (jobnet A) once both have been registered for execution.

Figure 2-15: Example of connections between jobnet connector and connection-destination jobnet



Calendar date	4/17(Tue)	4/18(Wed)	4/19(Thu)	4/20(Fri)
Schedule for jobnetS	6:00 @A103	6:00 @A104	None	Dummy schedule
Schedule for jobnetA	@A101 5:00	None	@A102 5:00	Dummy schedule

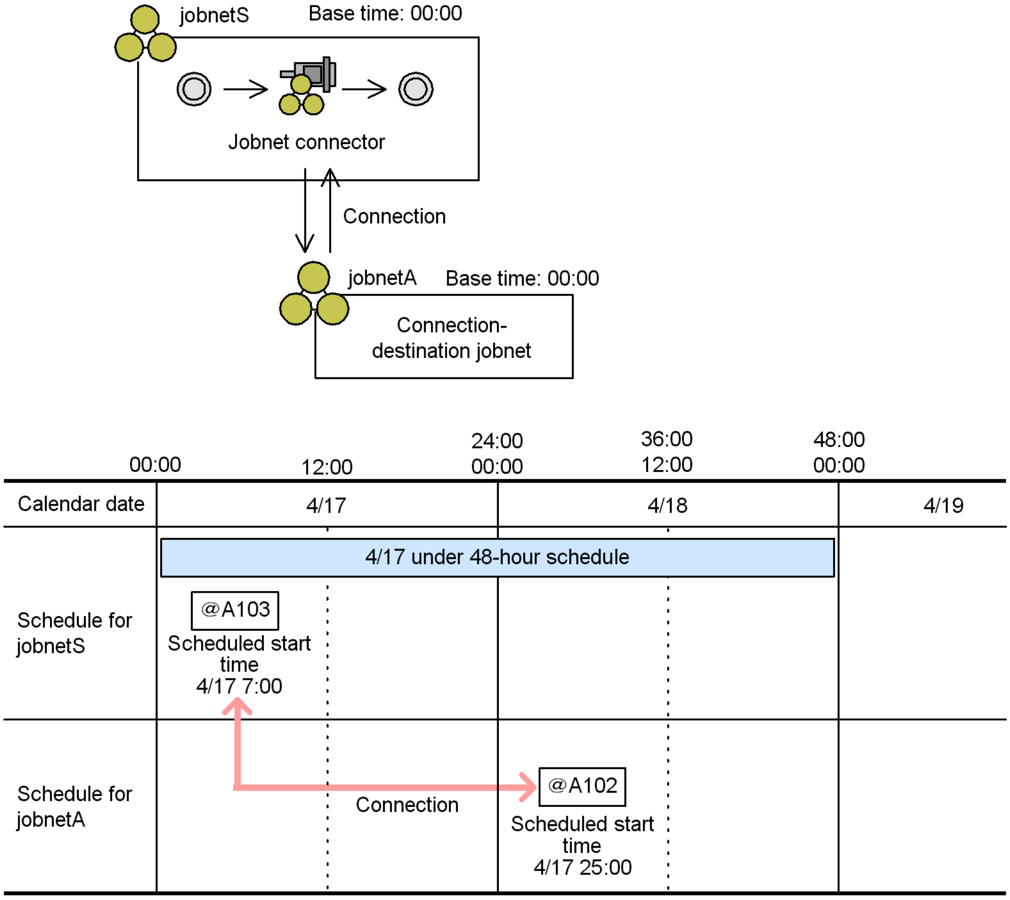
A connection is established between @A103 of jobnetS and generation @A101 of jobnetA because they share the same execution date. No connection is established between generation @A104 of jobnetS and generation @A102 of jobnetA because they are scheduled for execution on different dates. Although jobnetS and jobnetA are both tentatively scheduled for execution on 4/20 (Fri), no connection is established while the schedule is still a dummy schedule.

(b) Using the 48-hour schedule

If you are using the 48-hour schedule to calculate schedules, connections are established between generations with the same execution date.

The following figure shows an example of the connections established under a 48-hour schedule.

Figure 2-16: Example of connections between jobnet connector and connection-destination jobnet (48-hour schedule)



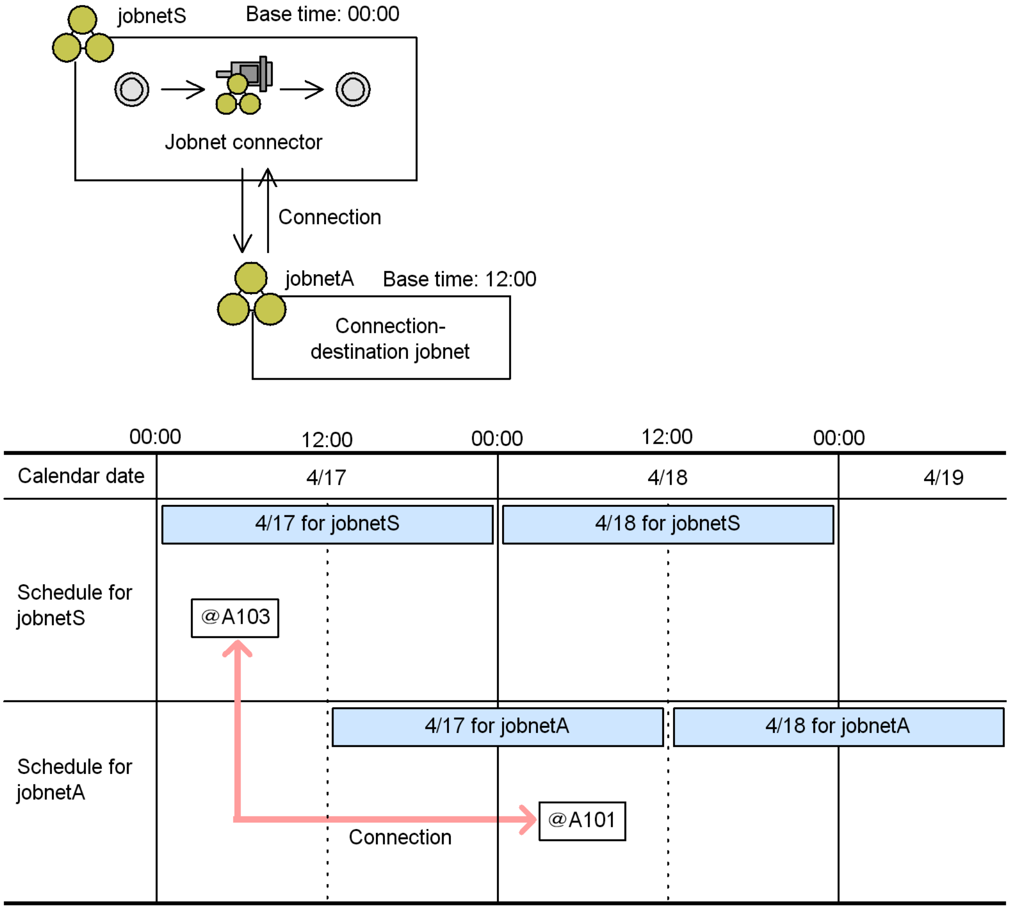
A connection is established between generation @A103 of jobnetS and generation @A102 of jobnetA, even though they are scheduled to execute on different calendar dates. This is because the two generations share an execution date under the 48-hour schedule.

(c) Behavior when base times differ

If a root jobnet with a jobnet connector has a different base time from its connection-destination jobnet, a connection is still established between generations that share an execution date.

The following figure shows an example of the connections established between jobnets with different base times.

Figure 2-17: Example of connections between jobnet connector and connection-destination jobnet (different base times)



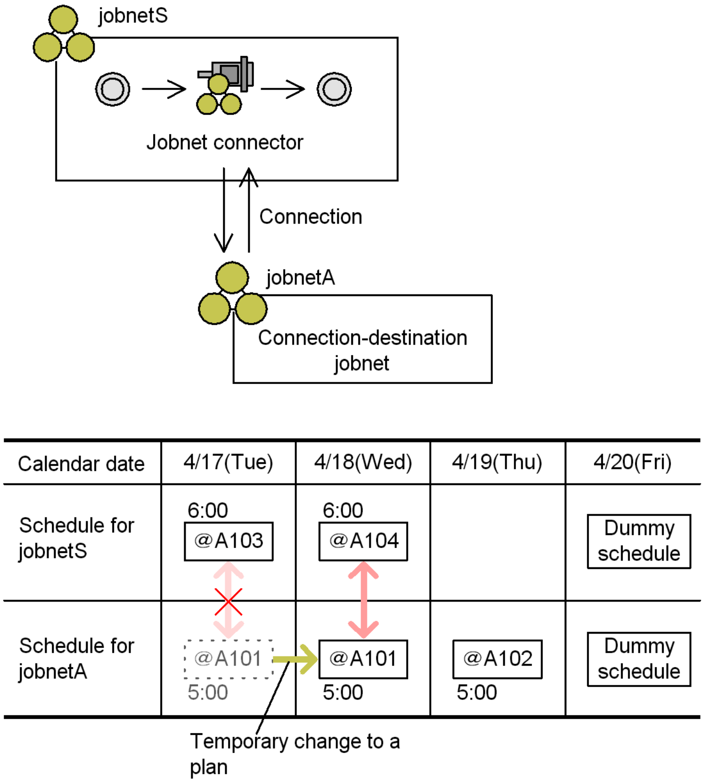
Although generation @A103 of jobnetS and generation @A102 of jobnetA are scheduled to execute on different calendar dates, a connection is established between the jobnets because both have an execution date of 4/17 with respect to their base time. In this manner, when you control the execution order of root jobnets that have different base times, connections might be established between jobnets that are scheduled to execute on different calendar dates. For this reason we recommend that you coordinate the base times of jobnet connectors and their connection-destination jobnets.

(d) Effects when the execution date of a generation changes

Connections between generations of a root jobnet with a jobnet connector and its connection-destination jobnet might be lost if either jobnet has its registration canceled or execution suspended before execution.

If a temporary plan change or a change to a schedule definition shifts a generation's execution date, this change will also affect the connections between generations. The following figure shows an example of how connections are affected when changes are made to a generation's scheduled execution date.

Figure 2-18: Connections when the scheduled execution date changes



Calendar date	4/17(Tue)	4/18(Wed)	4/19(Thu)	4/20(Fri)
Schedule for jobnetS	6:00 @A103	6:00 @A104		Dummy schedule
Schedule for jobnetA	@A101 5:00	@A101 5:00	@A102 5:00	Dummy schedule

Generation @A103 of jobnetS and generation @A101 of jobnetA were connected because they shared an execution date. However, when a temporary plan change shifts the scheduled execution date of @A101 back by a day to 4/18, the connection with @A103 is severed. Instead, @A101 establishes a connection with @A104, which is scheduled for execution on 4/18.

(e) When a jobnet is scheduled for execution more than once in one day

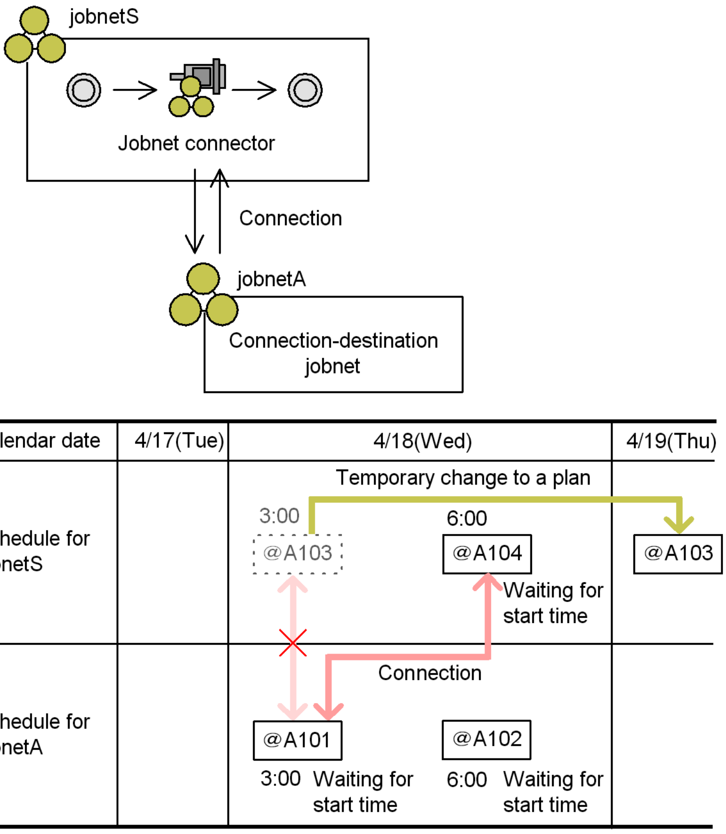
Connections are established between generations that have the same execution date, even if the jobnet is scheduled for execution more than once on a given day. However, because this can result in complicated relationships if the execution schedule of a jobnet changes, we recommend that you schedule jobnets related by a jobnet connector to execute only once per day.

Cautionary note

If execution is delayed for a jobnet with a jobnet connector that is scheduled to execute multiple times in one day, and *Synchronous* is specified as the method of execution order control, the next scheduled generation of the connection-destination jobnet might not be generated at all. Because the relationships of a delayed jobnet can be complicated, we recommend that you schedule jobnets with a jobnet connector to execute only once per day. For details, see *(3)(b) Synchronous execution*.

The following figure shows an example of temporarily rescheduling a jobnet that is scheduled to execute more than once per day.

Figure 2-19: Effect of temporarily rescheduling a jobnet scheduled to execute multiple times per day

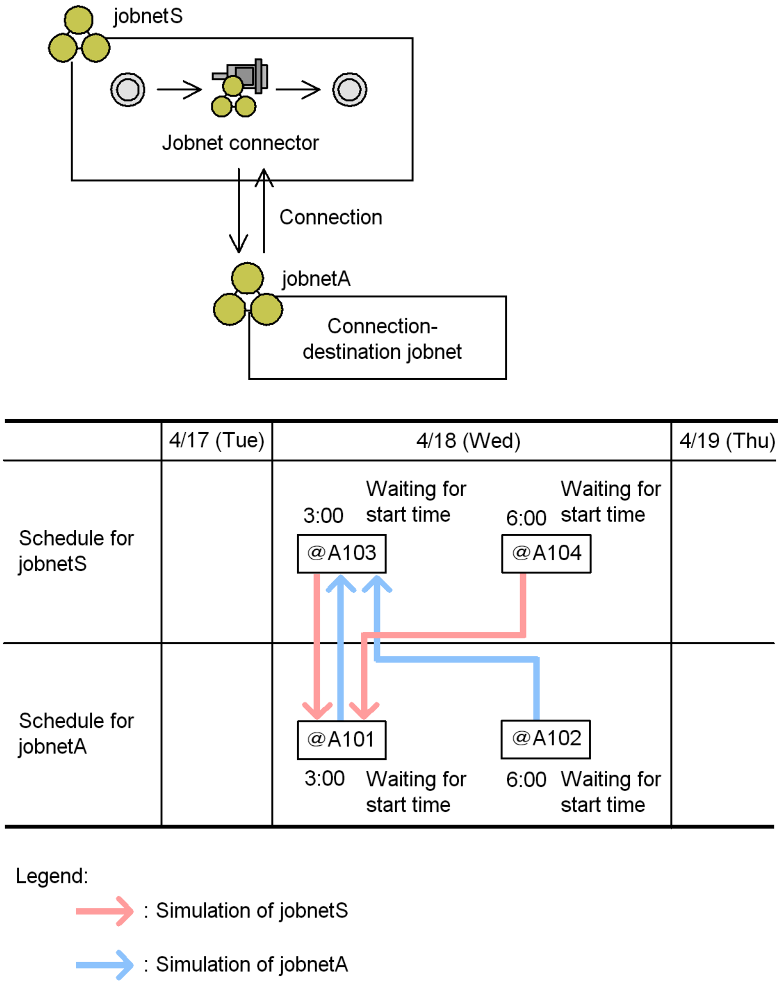


Of the generations scheduled to execute on the same day, connections are established in order, starting from those with the earliest scheduled execution times. Accordingly, generation @A103 of jobnetS establishes a connection with generation @A101 of jobnetA. At this stage, if a change to the schedule plan for jobnetS pushes the execution date of generation @A103 back to 4/19 (Thu), the connection between generation @A103 of jobnetS and generation @A101 of jobnetA is severed, and generation @A101 of jobnetA establishes a new connection with generation @A104 of jobnetS.

If you use JP1/AJS3 - View or the `ajsshow` command to simulate the connections between jobnets scheduled for execution more than once per day, the simulation will show multiple connections for unexecuted generations.

The following figure shows an example of such a situation. This simulation assumes that all generations are in *Wait for start time* status.

Figure 2-20: Example of connection simulation for jobnets scheduled to execute multiple times per day



In the simulation, because jobnetS and jobnetA share an execution date, generation @A103 of jobnetS connects to the generation of jobnetA with the earliest scheduled execution time, in this case @A101. However, because the connection between generation @A103 of jobnetS and generation @A101 of jobnetA remains unfixed, the simulation also connects generation @A101 to generation @A104 of jobnetS. The simulation similarly connects generation @A101 of jobnetA to generation @A103 of jobnetS, and generation @A102 of jobnetA to generation @A103 of jobnetS.

(f) Behavior when time zones differ

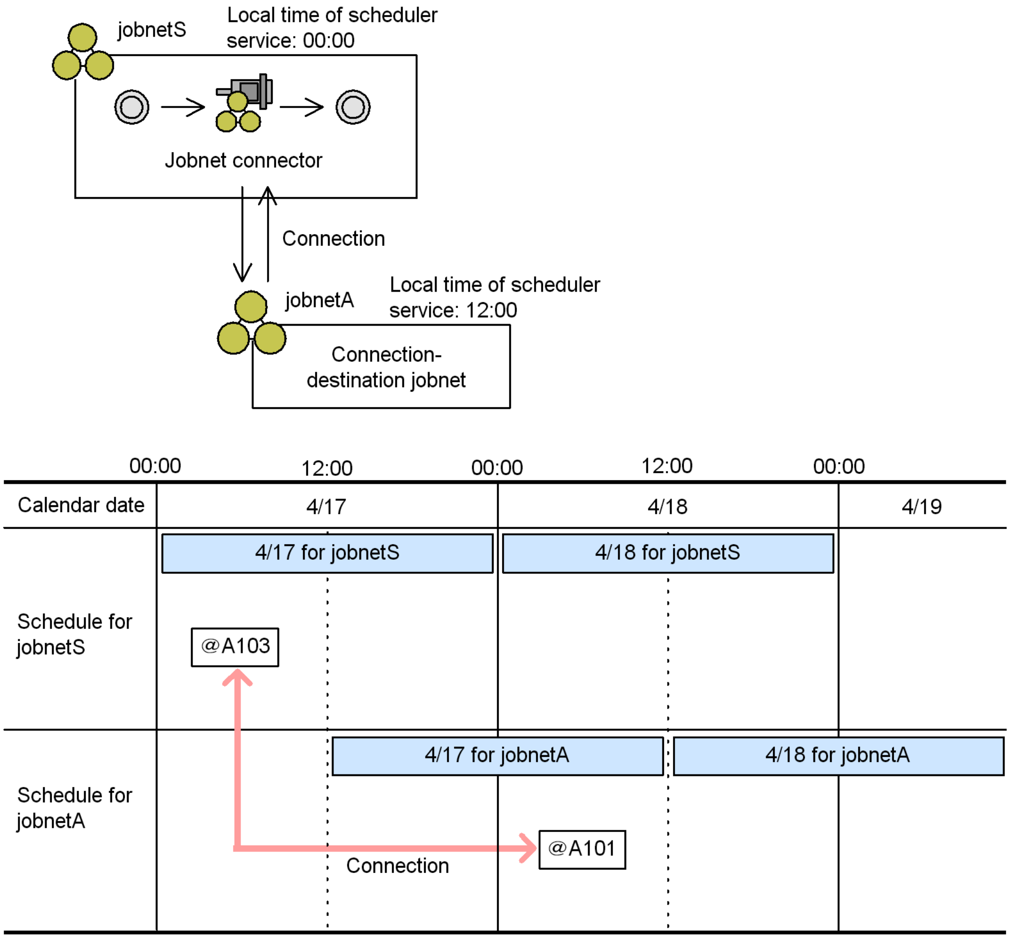
When a root jobnet with a jobnet connector and its connection-destination jobnet are scheduled under different time zones, connections will still be established between generations if their execution dates are the same. However, for the sake of simplicity, we recommend that jobnet connectors and their connection-destination jobnets use the same time zone.

(g) Behavior when scheduler services have different local times

When the scheduler service governing a root jobnet with a jobnet connector uses a different local time setting from the scheduler service of the connection-destination jobnet, connections will still be established between generations if their execution dates are the same.

The following figure shows an example of the connections established between jobnets when their scheduler services use different local time settings.

Figure 2-21: Example of connections between jobnet connector and connection-destination jobnet (when scheduler services have different local time settings)



Although generation @A103 of jobnetS and generation @A101 of jobnetA are scheduled to execute on different calendar days, a connection is established between them because both are scheduled for execution on 4/17 at their respective local times. To avoid a situation in which connections are established between generations scheduled for execution on different days, ensure that the scheduler services responsible for the connection-destination jobnet and the jobnet connector are set to the same local time.

(3) Methods for controlling the execution order of root jobnets

The method of execution order control governs whether execution of the root jobnet is synchronized with execution of the jobnet connector. Specify the control method in the settings of the connection-destination jobnet. The default is **Asynchro**.

The following describes the behavior at root jobnet execution under each control method.

(a) Asynchronous execution

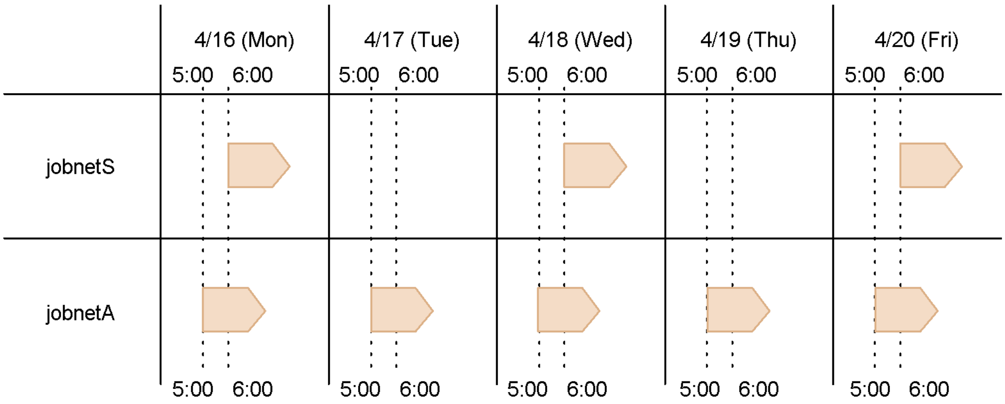
If **Asynchro** is set as the method of execution order control, the connection-destination jobnet executes according to its own schedule regardless of any connections with other jobnets and the status of the jobnet connector.

The following figure shows how the respective jobnets are executed under the asynchronous method of execution order control. Here, jobnetS is the jobnet with the jobnet connector, and jobnetA is the connection-destination jobnet.

Figure 2-22: Operation under asynchronous execution order control

	4/16 (Mon)	4/17 (Tue)	4/18 (Wed)	4/19 (Thu)	4/20 (Fri)
Schedule for jobnetS	6:00 @A106		6:00 @A107		6:00 @A108
Schedule for jobnetA	@A101 5:00	@A102 5:00	@A103 5:00	@A104 5:00	@A105 5:00

■ Execution schedule when **Asynchro** is specified



(b) Synchronous execution

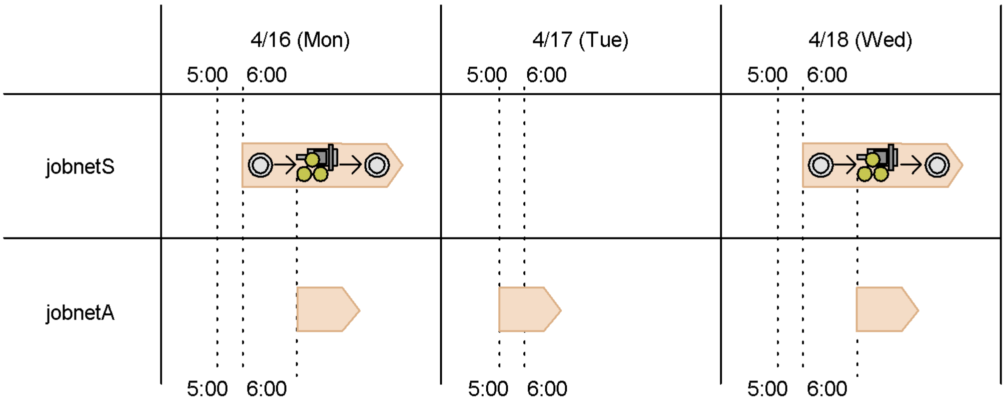
If **Synchro** is set as the method of execution order control, generations in a connected relationship wait until the jobnet connector has executed before starting. In the absence of a corresponding generation with the same execution date, a generation will execute according to its own schedule.

The following figure shows how the respective jobnets are executed under the synchronous method of execution order control. Here, jobnetS is the jobnet with the jobnet connector, and jobnetA is the connection-destination jobnet.

Figure 2-23: Operation under synchronous execution order control

	4/16 (Mon)	4/17 (Tue)	4/18 (Wed)
Schedule for jobnetS	6:00 @A106		6:00 @A107
Schedule for jobnetA	@A101 5:00	@A102 5:00	@A103 5:00

■ Execution schedule when **Synchro** is specified



JobnetS and jobnetA have scheduled start times of 6:00 and 5:00, respectively. The connected relationship between the generations that are scheduled for execution on 4/16 (Mon) and 4/18 (Wed) means that on these days, jobnetA starts in synchronization with the jobnet connector, instead of at 5:00 as scheduled. The generation of jobnetA scheduled for execution on 4/17 (Tue), which lacks a connection to a generation of jobnetS, starts at 5:00 as originally scheduled.

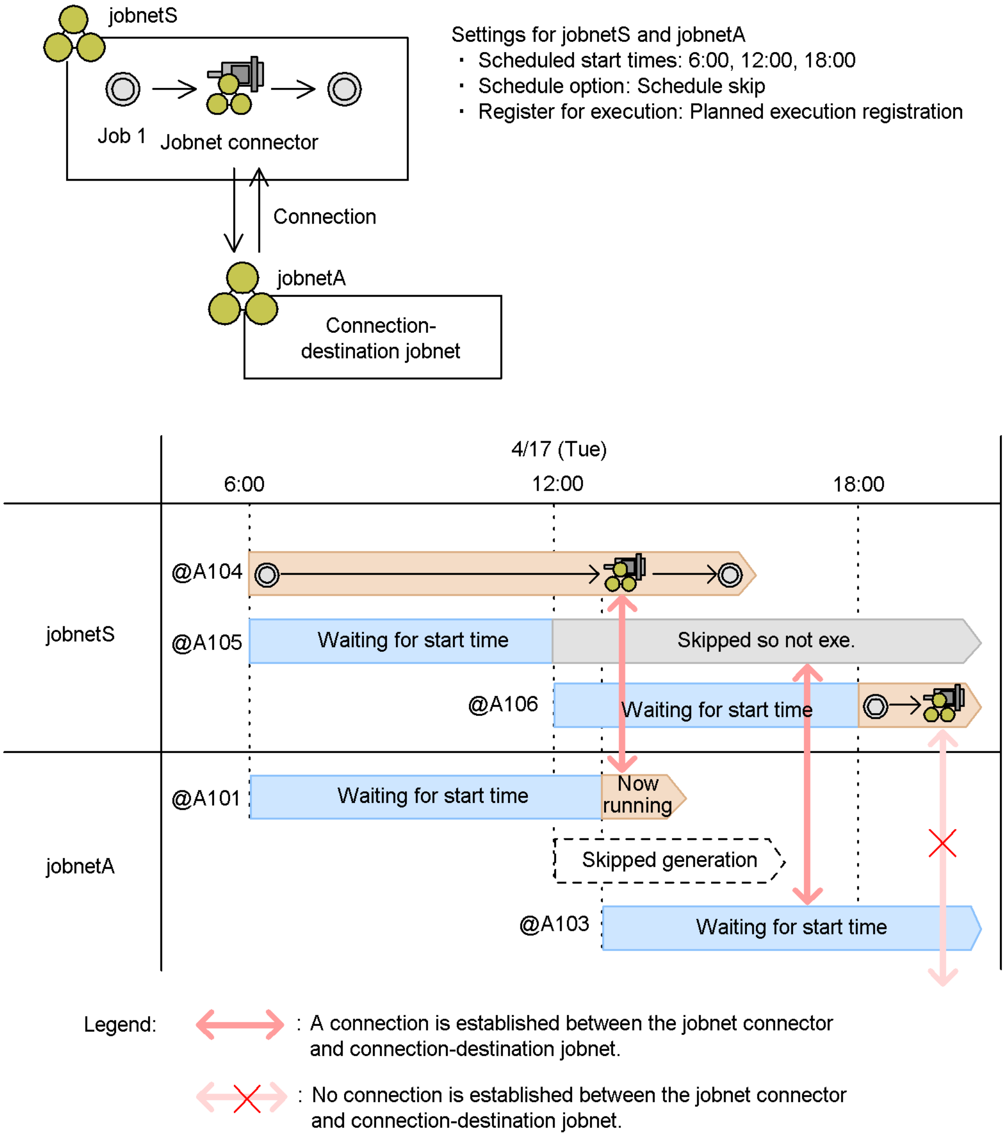
The synchronous execution setting applies only to the first synchronized execution, not to any subsequent re-executions. To apply synchronous execution to re-executed generations, you can use the function for temporarily changing the method of execution order control.

Cautionary note

If execution is delayed for a jobnet with a jobnet connector that is scheduled to execute multiple times per day, and **Synchro** is specified as the method of execution order control, the next scheduled generation of the connection-destination jobnet might not be generated at all. Because the relationships of a delayed jobnet can be complicated, we recommend that you schedule jobnets with a jobnet connector to execute only once per day.

The following figure shows an example in which jobnets are executed more than once per day with the execution order control method set to **Synchro**.

Figure 2-24: Example of executing jobnets multiple times per day based on synchronous execution order control


jobnetS
Job 1
Jobnet connector
Connection
jobnetA
Connection-destination jobnet
Settings for jobnetS and jobnetA
・ Scheduled start times: 6:00, 12:00, 18:00
・ Schedule option: Schedule skip
・ Register for execution: Planned execution registration
4/17 (Tue)
6:00
12:00
18:00
jobnetS
@A104
@A105
Waiting for start time
Skipped so not exe.
@A106
Waiting for start time
jobnetA
@A101
Waiting for start time
Now running
Skipped generation
@A103
Waiting for start time
Legend:
: A connection is established between the jobnet connector and connection-destination jobnet.
: No connection is established between the jobnet connector and connection-destination jobnet.

In this example, the scheduled execution start times of the root jobnet for which the jobnet connector is defined (jobnetS) and the connection-destination jobnet (jobnetA) are 6:00, 12:00, and 18:00. Schedule skip is set as the schedule option, and planned execution registration is used as the execution registration method.

The following describes the behavior of jobnetS and jobnetA with respect to generation @A104 of jobnetS (6:00 start time), when the preceding "Job 1" of the jobnet connector has not completed by the time jobnetS is next scheduled for execution (12:00).

1. Generation @A105 of jobnetS scheduled for execution at 12:00 enters *Skipped so not executed* status.

   Because *schedule skip* is set as the schedule option for jobnetS, generations whose execution overlaps with that of an earlier scheduled generation enter *Skipped so not executed* status and their execution is skipped.

   Because generation @A101 has not started by 12:00, the generation of jobnetA scheduled for execution at that time is removed from the dummy schedule.

2. After Job 1 has finished executing, the jobnet connector associated with generation @A104 of jobnetS is executed.

   Because synchronous execution is specified, generation @A101 of jobnetA also begins executing. At the same time, JP1/AJS3 creates the next scheduled generation of jobnetA (scheduled for 18:00) in *Wait for start time* status. This generation is assigned the execution ID @A103. At this point, a connection is established between generation @A105 of jobnetS, which was placed in *Skipped so not executed* status, and generation @A103 of jobnetA. Because generation @A103 of jobnetA is connected to a jobnet in *Skipped so not executed* status, it remains in *Wait for start time* status and is not executed. For an explanation of the status transitions undergone by the jobnet connector and connection-destination jobnet, see *(4) Transitions of jobnet connector and connection-destination jobnet statuses*.

3. At 18:00, the generation of jobnetS scheduled for execution at that time (@A106) is executed.

   At this stage, generation @A106 of jobnetS has no connection to any other generation. Hence, generation @A106 of jobnetS waits in *Now running* status until generation @A103 of jobnetA has finished and it is clear whether any generations of jobnetA can serve as a connection target.

   If you change to asynchronous execution by using the function for temporarily changing the execution order control method, generation @A106 will be executed regardless of whether any connected generations exist. For details on this feature, see *(c) Temporarily changing the method of execution order control*.

If generation @A105 of jobnetS is re-executed from its *Skipped so not executed* status before 18:00, the connected generation @A103 of jobnetA is also executed. If a new generation of jobnetA is created after generation @A103 has finished, a connection is established between generation @A106 of jobnetS and this new generation, and generation @A106 is executed. If no new generation is created, the jobnet connector associated with generation @A106 of jobnetS enters *Bypassed* status.

(c) Temporarily changing the method of execution order control

JP1/AJS3 provides a feature for temporarily changing the method of execution order control. You can use this feature to synchronize re-execution of a connection-destination jobnet with a jobnet connector, or to apply a different method of execution order control to a particular generation.

To access this feature, choose **Operations** and then **Change Execution Order Method** in the following windows:

- Daily Schedule (Hierarchy) window
- Monthly Schedule window
- Jobnet Monitor window

Supplementary note

You cannot change the method of execution order control for a dummy schedule.

(4) Transitions of jobnet connector and connection-destination jobnet statuses

This section describes the status transitions that apply to jobnet connectors and their connection-destination jobnets. For details on the statuses that these units can acquire, see *6.1.1 Status levels of jobnets, jobs, and jobnet connectors* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.

(a) Jobnet connector status transitions

When you register a jobnet with a jobnet connector for execution, the root jobnet associated with the jobnet connector begins executing. At the point when all preceding units in the jobnet have ended, the status of the jobnet connector changes according to the status of the connection-destination jobnet.

The following table lists the status transitions that apply to jobnet connectors.

Table 2-6: Jobnet connector status transitions

Status of connection-destination jobnet	Status of jobnet connector
Not registered	Running + Abend
Not sched. to exe.	Bypassed
Wait for start time	Now running
Being held	Now running

Status of connection-destination jobnet	Status of jobnet connector
Skipped so not exe.	Running + Abend
Ended normally	Ended normally
Ended with warning	Ended with warning
Ended abnormally	Running + Abend
Interrupted	Running + Abend
Killed	Running + Abend
Shutdown	Running + Abend
Invalid exe. seq.	Running + Abend
Now running	Now running
Running + Warning	Running + Warning
Running + Abend	Running + Abend

The jobnet connector immediately enters *Ended abnormally* status if the definition of the jobnet connector is invalid. Possible reasons are as follows:

- No connection-destination unit is specified.
- A non-existent unit is specified as the connection-destination jobnet.
- The unit specified as the connection-destination jobnet is neither a root jobnet nor a planning group.
- Execution order control is disabled in the definition of the unit specified as the connection-destination jobnet.
- The **Jobnet Connector** is specified incorrectly in the definition of the unit specified as the connection-destination jobnet.
- The **Connection range** setting of the jobnet connector is different from that of the connection-destination jobnet.
- One of the following problems renders the definition invalid when **Other service** is specified for **Connection range**.
    - JP1/AJS3 cannot connect to the host specified in **Connection host**.
    - The scheduler service specified in **Connection service** does not exist.
    - The host specified in **Connection host** for the connection-destination jobnet is different from that of the jobnet connector.
    - The scheduler service specified in **Connection service** for the

connection-destination jobnet is different from that of the jobnet connector.

(b) Connection-destination jobnet status transitions

When a connection-destination jobnet is registered for execution, its behavior at execution changes according to the status of connected generations, as well as which execution order control method (synchronous or asynchronous) is in effect.

The following table lists the status transitions that apply to connection-destination jobnets.

Table 2-7: Connection-destination jobnet status transitions

Status of jobnet connector	Status of connection-destination jobnet	
	Synchronous	Asynchronous
Not registered	Wait for start time	Now running
Not sched. To exe.	Wait for start time	Now running
Wait for prev. to end	Wait for start time	Now running
Not executed + Ended	Wait for start time	Now running
Bypassed	Now running	Now running
Now running	Now running	Now running
Ended normally	Wait for start time#	Now running
Ended with warning	Wait for start time#	Now running
Ended abnormally	Wait for start time#	Now running
Killed	Wait for start time#	Now running
Shutdown	Wait for start time#	Now running
Unknown end status	Wait for start time#	Now running

#

If the jobnet connector has already terminated as a result of user intervention by the time the connection-destination jobnet starts, the connection-destination jobnet will stay in *Wait for start time* status until the jobnet connector is executed. You can resolve this issue by re-executing the jobnet connector, or by temporarily changing the method of execution order control for the connection-destination jobnet to *asynchronous*.

The connection-destination jobnet immediately enters *Ended abnormally* status if the definition of the jobnet contains an error, such as one of those listed below. However, if the jobnet is subject to asynchronous control and **Other service** is specified for **Connection range**, the jobnet executes as normal without entering *Ended abnormally* status even if the jobnet definition contains an error. For this reason, we recommend that you use the definition pre-check function to check the definition of the connection-destination jobnet before you register it for execution.

- Execution order control is enabled, but no jobnet connector name is specified.
- A non-existent unit is specified as the jobnet connector.
- The unit specified as the jobnet connector is not a jobnet connector.
- The specified jobnet connector is associated with a different connection target.
- The **Connection range** setting of the jobnet connector is different from that of the connection-destination jobnet.
- One of the following problems renders the definition invalid when **Other service** is specified for **Connection range**:
    - JP1/AJS3 cannot connect to the host specified in **Connection host**.
    - The scheduler service specified in **Connection service** does not exist.
    - The host specified in **Connection host** for the jobnet connector is different from that of the connection-destination jobnet.
    - The scheduler service specified in **Connection service** for the jobnet connector is different from that of the connection-destination jobnet.

(c) When communication between scheduler services fails

When the execution order of root jobnets governed by different scheduler services is controlled, communication between the respective scheduler services takes place at the following times:

- When the jobnet connector starts executing
- When the connection-destination jobnet starts executing
- When the status of the connection-destination jobnet changes

This subsection describes how the statuses of the jobnet connector and the connection-destination jobnet are affected when a communication error occurs at each of these stages. For details about how to troubleshoot errors, see *2.7.5 Troubleshooting problems related to jobnet connectors* in the manual *Job Management Partner 1/ Automatic Job Management System 3 Troubleshooting*.

■ If a communication error occurs when the jobnet connector starts executing

When the jobnet connector starts executing, connections are established with generations of the connection-destination jobnet. For information about this process, see *(2) Rules governing connections between jobnet connectors and connection-destination jobnets*.

If a communication error occurs during this process, the jobnet connector enters *Ended abnormally* status. The status of the connection-destination jobnet remains unchanged.

■ If a communication error occurs when the connection-destination jobnet starts executing

When synchronous control is used for the connection-destination jobnet, connections are established with generations of the jobnet connector when the connection-destination jobnet starts executing. For information about this process, see *(2) Rules governing connections between jobnet connectors and connection-destination jobnets*.

If a communication error occurs during this process, the connection-destination jobnet enters *Ended abnormally* status. The status of the jobnet connector remains unchanged.

■ If a communication error occurs when the status of the connection-destination jobnet changes

The jobnet connector remains unaware of the change to the connection-destination jobnet's status. Consequently, the jobnet connector does not undergo a corresponding status change.

(5) Re-execution of jobnet connectors and connection-destination jobnets

The following describes the re-execution of jobnet connectors and connection-destination jobnets.

(a) Re-executing jobnet connectors

In the same manner as with a job, you can re-execute a jobnet connector only after it has finished executing. Note that because a jobnet connector does not take a hold attribute, you cannot have it immediately placed in *Held* status upon re-execution.

(b) Re-executing connection-destination jobnets

A connection-destination jobnet can be re-executed in the same manner as a standard jobnet. However, after a jobnet controlled by the synchronous method has been executed once in synchronization with a jobnet connector, subsequent executions run according to the asynchronous method. Consequently, such a connection-destination jobnet will not synchronize with the jobnet connector at re-execution. You can synchronize re-execution with a jobnet connector by changing the execution order control method to synchronous by using the feature provided for this purpose.

(c) Effect on re-execution when connections between generations are severed

A generation that is in a connected relationship with another generation might be removed from the schedule for some reason. This might be due to a cancellation of registration, execution cancellation, or changes to the number of logs to keep. In this case, you will be unable to re-execute the generation that was in the connected relationship with the deleted generation. The following table shows how JP1/AJS3 behaves when one of a pair of connected generations is deleted, and describes what action the user must take.

Table 2-8: Handling re-execution when a connected relationship is severed

Generation	Behavior at re-execution	Action to take
The remaining generation is the jobnet connector	Upon re-execution, the jobnet connector enters *Ended abnormally*# status.	Resume the job flow from the unit succeeding the jobnet connector.
The remaining generation is the connection-destination jobnet	When you re-execute the connection-destination jobnet after changing the execution order control method to synchronous, the jobnet enters *Ended abnormally*# status.	Temporarily change the execution order control method to asynchronous, and re-execute the connection-destination jobnet.

#

The jobnet connector or connection-destination jobnet will enter *Ended abnormally* status only if you re-execute it after it has already finished. You will be able to re-execute the unit without it entering *Ended abnormally* status if you do so while it is still running.

(6) Jobnet connector polling

When you control the execution order of root jobnets governed by different scheduler services, jobnet connectors and connection-destination jobnets under synchronous control check the status of their counterparts by regular polling. This polling takes place during the following periods:

- The time from when the jobnet connector starts until the jobnet connector has ended
- The time from when the connection-destination jobnet reaches its start time until it starts

The jobnet connector or connection-destination jobnet will end abnormally if polling fails for some reason, such as an interruption to the scheduler service or a communication error, and it cannot confirm the status of its counterpart.

2.3 Tips on work task automation

This section gives some tips on work task automation that fall outside the categories discussed previously in this chapter.

2.3.1 Processing with a distributed load

JP1/AJS3 allows you to specify a group of execution agents as the agent host on which to execute a job or jobnet. This allows the processing load to be distributed among the execution agents in the group. The patterns of load distribution are as follows:

- Jobs are distributed evenly among a number of execution agents.
- Individual execution agents are assigned different limits for the number of concurrently executable jobs.
- When the number of jobs executing concurrently at a specific execution agent reaches the limit, execution is distributed to the other execution agents.

You can specify an execution agent group to execute the following units:

- Root jobnets
- Nested jobnets
- PC jobs
- Unix jobs
- Action jobs
- Custom jobs

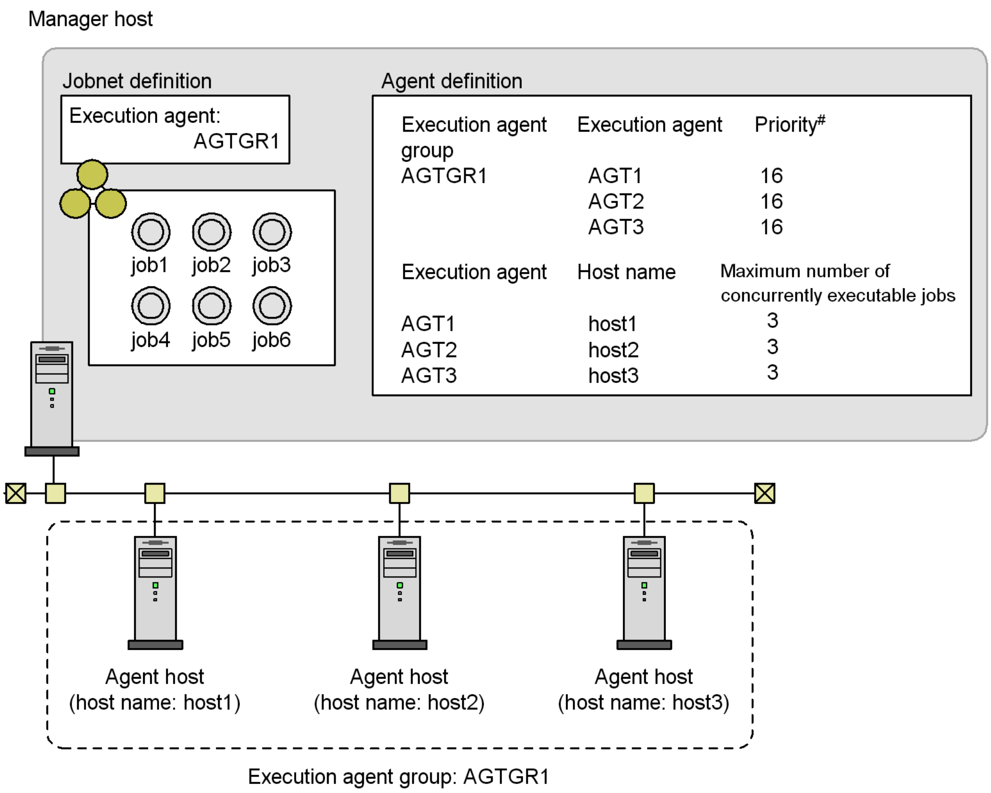
This section describes how load distribution processing takes place in each pattern. In this example, the agent group AGTGR1 is specified as the execution agent group for a jobnet containing six jobs (job1 to job6).

(1) Equal load distribution

In this pattern, the agents in the execution agent group have the same priority and the same maximum number of concurrently executable jobs.

The following figure shows an example of the pattern in which the load is distributed evenly among execution agents:

Figure 2-25: Example of equal load distribution



Execution agent group	Execution agent	Priority#
AGTGR1	AGT1	16
	AGT2	16
	AGT3	16

Execution agent	Host name	Maximum number of concurrently executable jobs
AGT1	host1	3
AGT2	host2	3
AGT3	host3	3

#: Specify the execution agent priority in the range 1 to 16, where 16 is the highest priority.
The default is 16 (highest priority).

In this example, the job execution order is determined as follows, where the priorities of the three execution agents (AGT1, AGT2, and AGT3) in execution agent group AGTGR1 are 16, and a maximum of three jobs can be executed concurrently by each agent.

1. Determine the execution agent for job1.

   Ratio: *number of active jobs* / *maximum number of concurrently executable jobs*:

   - Execution agent AGT1: 0 / 3
   - Execution agent AGT2: 0 / 3
   - Execution agent AGT3: 0 / 3

   As all execution agents have the same load and the same priority, AGT1, which is defined first in the agent group AGTGR1, becomes the execution agent for job1.

2. Determine the execution agent for job2.

   Because job1 has been assigned to execution agent AGT1, the ratio (*active jobs* / *maximum concurrent jobs*) is now:

   - Execution agent AGT1: 1 / 3
   - Execution agent AGT2: 0 / 3
   - Execution agent AGT3: 0 / 3

   Execution agents AGT2 and AGT3 now have the same load and the same priority. Therefore, AGT2, which is defined earlier in the agent group AGTGR1, becomes the execution agent for job2.

3. Determine the execution agent for job3.

   Because job2 has been assigned to execution agent AGT2, the ratio (*active jobs* / *maximum concurrent jobs*) is now:

   - Execution agent AGT1: 1 / 3
   - Execution agent AGT2: 1 / 3
   - Execution agent AGT3: 0 / 3

   Execution agent AGT3 now has the lightest load. Therefore, job3 is assigned to execution agent AGT3.

4. Determine the execution agent for job4.

   Now that job1 through job3 have been assigned to execution agents, the ratio (*active jobs* / *maximum concurrent jobs*) is now:

   - Execution agent AGT1: 1 / 3
   - Execution agent AGT2: 1 / 3
   - Execution agent AGT3: 1 / 3

   Once again, all execution agents have the same load. Therefore, AGT1 becomes the execution agent for job4 because it is defined first in the agent group AGTGR1.

5. Determine the execution agent for job5.

   Because job4 has been assigned to execution agent AGT1, the ratio (*active jobs* / *maximum concurrent jobs*) is now:

   - Execution agent AGT1: 2 / 3
   - Execution agent AGT2: 1 / 3
   - Execution agent AGT3: 1 / 3

   Execution agents AGT2 and AGT3 now have the same load. Therefore, AGT2 becomes the execution agent for job5 because it is defined earlier in the agent group AGTGR1.

6. Determine the execution agent for job6.

   Because job5 has been assigned to execution agent AGT2, the ratio (*active jobs* / *maximum concurrent jobs*) is now:

   - Execution agent AGT1: 2 / 3
   - Execution agent AGT2: 2 / 3
   - Execution agent AGT3: 1 / 3

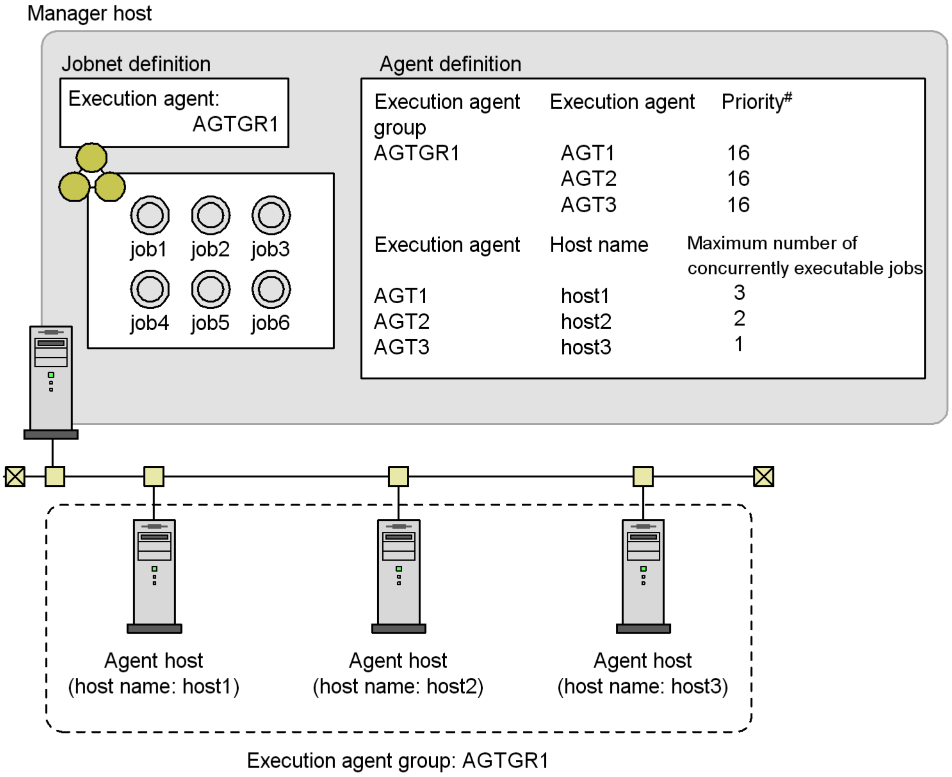
   Execution agent AGT3 now has the lightest load. Therefore, job6 is assigned to execution agent AGT3.

(2) Distribution among agents assigned different job execution limits

To distribute loads by using different job execution limits at each execution agent, set the maximum number of concurrently executable jobs to a different value, but set the same priority for each execution agent in the group.

The following figure shows an example where the execution agents are assigned a different number of concurrently executable jobs:

Figure 2-26: Example of distribution between execution agents with different job execution limits



Execution agent group	Execution agent	Priority#
AGTGR1	AGT1	16
	AGT2	16
	AGT3	16

Execution agent	Host name	Maximum number of concurrently executable jobs
AGT1	host1	3
AGT2	host2	2
AGT3	host3	1

#: Specify the execution agent priority in the range 1 to 16, where 16 is the highest priority.
The default is 16 (highest priority).

In this example, the job execution order is determined as follows, where the maximum number of concurrently executable jobs of the three execution agents (AGT1, AGT2, and AGT3) in execution agent group AGTGR1 are 3, 2, and 1.

1. Determine the execution agent for job1.

   Ratio: *number of active jobs* / *maximum number of concurrently executable jobs*:

   - Execution agent AGT1: 0 / 3
   - Execution agent AGT2: 0 / 2
   - Execution agent AGT3: 0 / 1

   As all execution agents have the same load and the same priority, AGT1, which is defined first in the agent group AGTGR1, becomes the execution agent for job1.

2. Determine the execution agent for job2.

   Because job1 has been assigned to execution agent AGT1, the ratio (*active jobs* / *maximum concurrent jobs*) is now:

   - Execution agent AGT1: 1 / 3
   - Execution agent AGT2: 0 / 2
   - Execution agent AGT3: 0 / 1

   Execution agents AGT2 and AGT3 now have the same load and the same priority. Therefore, AGT2, which is defined earlier in the agent group AGTGR1, becomes the execution agent for job2.

3. Determine the execution agent for job3.

   Because job2 has been assigned to execution agent AGT2, the ratio (*active jobs* / *maximum concurrent jobs*) is now:

   - Execution agent AGT1: 1 / 3
   - Execution agent AGT2: 1 / 2
   - Execution agent AGT3: 0 / 1

   Execution agent AGT3 now has the lightest load. Therefore, job3 is assigned to execution agent AGT3.

4. Determine the execution agent for job4.

   Now that job1 through job3 have been assigned to execution agents, the ratio (*active jobs* / *maximum concurrent jobs*) is:

   - Execution agent AGT1: 1 / 3
   - Execution agent AGT2: 1 / 2
   - Execution agent AGT3: 1 / 1

   Once again, all execution agents have the same load. Therefore, AGT1 becomes the execution agent for job4 because it is defined first in the agent group AGTGR1.

5. Determine the execution agent for job5.

   Because job4 has been assigned to execution agent AGT1, the ratio (*active jobs* / *maximum concurrent jobs*) is now:

   - Execution agent AGT1: 2 / 3
   - Execution agent AGT2: 1 / 2
   - Execution agent AGT3: 1 / 1

   Although AGT2 and AGT3 have the same load, AGT3 has now reached its maximum number of concurrently executable jobs. Therefore, job5 is assigned to execution agent AGT2.

6. Determine the execution agent for job6.

   Because job5 has been assigned to execution agent AGT2, the ratio (*active jobs / maximum concurrent jobs*) is now:

   - Execution agent AGT1: 2 / 3
   - Execution agent AGT2: 2 / 2
   - Execution agent AGT3: 1 / 1

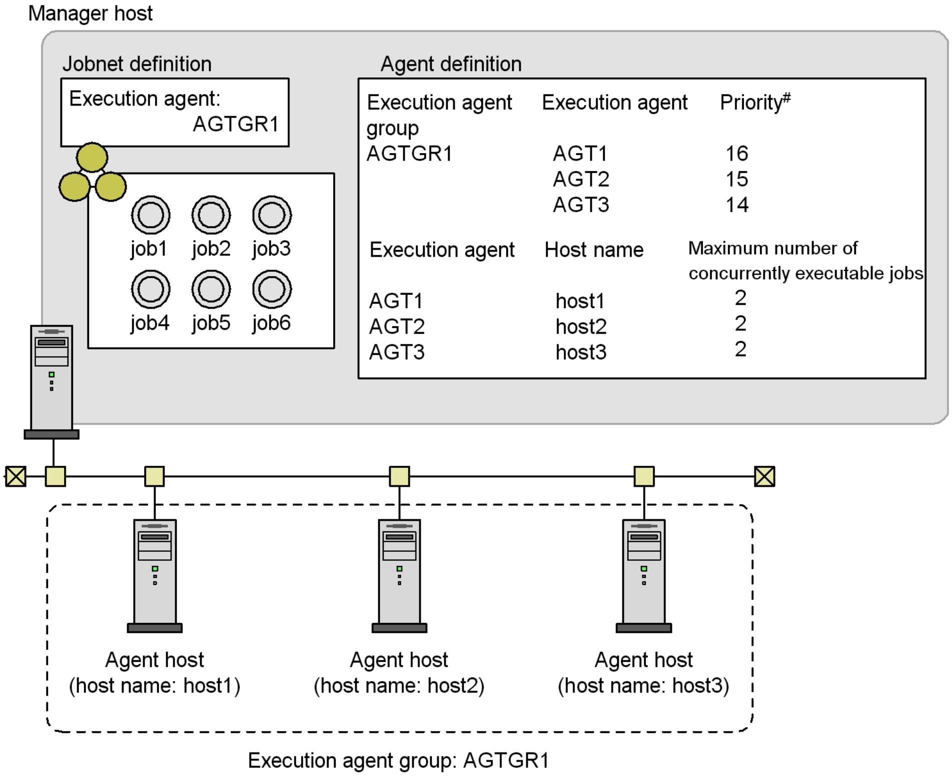
   As AGT2 and AGT3 have both reached their maximum number of concurrently executable jobs, job6 is assigned to execution agent AGT1.

(3) Job execution distributed to other execution agents when the limit is reached

To have JP1/AJS3 distribute jobs to another execution agent in the group when the number of concurrently executed jobs reaches a limit at an agent, you must specify the priority of each agent in the group. You can also assign different job execution limits to each execution agent as required.

The following figure shows an example where jobs are distributed to another execution agent when the number of concurrently executed jobs has reached the limit.

Figure 2-27: Example of distribution to another execution agent when the number of jobs executed reaches the limit



#: Specify the execution agent priority in the range 1 to 16, where 16 is the highest priority. The default is 16 (highest priority).

In this example, the job execution order is determined as follows, where the priorities of the three execution agents (AGT1, AGT2, and AGT3) in execution agent group AGTGR1 are 16 (highest), 15 (second highest), and 14 (third highest), and a maximum of two jobs can be executed concurrently by each agent.

1. Determine the execution agent for job1.

   Ratio: *number of active jobs* / *maximum number of concurrently executable jobs*:

   - Execution agent AGT1: 0 / 2
   - Execution agent AGT2: 0 / 2
   - Execution agent AGT3: 0 / 2

   As AGT1 has the highest priority, AGT1 becomes the execution agent for job1.

2. Determine the execution agent for job2.

   Because job1 has been assigned to execution agent AGT1, the ratio (*active jobs / maximum concurrent jobs*) is now:

   - Execution agent AGT1: 1 / 2
   - Execution agent AGT2: 0 / 2
   - Execution agent AGT3: 0 / 2

   Although AGT2 and AGT3 have a lower load, AGT1 has highest priority, so AGT1 becomes the execution agent for job2.

3. Determine the execution agent for job3.

   Because job2 has been assigned to execution agent AGT1, the ratio (*active jobs / maximum concurrent jobs*) is now:

   - Execution agent AGT1: 2 / 2
   - Execution agent AGT2: 0 / 2
   - Execution agent AGT3: 0 / 2

   Although AGT1 has the highest priority, it has now reached its maximum number of concurrently executable jobs. Therefore, job3 is assigned to execution agent AGT2, which has the next highest priority.

4. Determine the execution agent for job4.

   Because job3 has been assigned to execution agent AGT2, the ratio (*active jobs / maximum concurrent jobs*) is now:

   - Execution agent AGT1: 2 / 2
   - Execution agent AGT2: 1 / 2
   - Execution agent AGT3: 0 / 2

   As with job3, execution agent AGT2 is determined for job4.

5. Determine the execution agent for job5.

   Because job4 has been assigned to execution agent AGT2, the ratio (*active jobs / maximum concurrent jobs*) is now:

   - Execution agent AGT1: 2 / 2
   - Execution agent AGT2: 2 / 2
   - Execution agent AGT3: 0 / 2

   As AGT1 and AGT2 have both reached their maximum number of concurrently executable jobs, job5 is assigned to execution agent AGT3.

6. Determine the execution agent for job6.

   Because job5 has been assigned to execution agent AGT3, the ratio (*active jobs / maximum concurrent jobs*) is now:

   - Execution agent AGT1: 2 / 2
   - Execution agent AGT2: 2 / 2
   - Execution agent AGT3: 1 / 2

   As AGT1 and AGT2 have both reached their maximum number of concurrently executable jobs, job6 is assigned to execution agent AGT3.

2.3.2 JP1/AJS3 functions useful for work task automation

This section describes the JP1/AJS3 functionality that you can use to automate work tasks.

The following table lists tasks that you might want to perform with JP1/AJS3 and the functionality to use in each case.

Table 2-9: JP1/AJS3 functions for realizing work task automation

Item	Task you want to perform in JP1/AJS3	Relevant JP1/AJS3 function	Relevant section
1	Prepare processing that changes dynamically depending on the results of the preceding job.	Judgment job	*2.4.3 Dynamically changing a process depending on the result of a preceding job (example of defining a jobnet that uses a judgment job)*
2	Receive JP1 events from a host of your choice.	JP1 event reception monitoring job	*2.4.4 Executing an event-driven process (example of defining a jobnet that uses an event job)*
3	Detect when a file is created, deleted, or updated.	File monitoring job	
4	Monitor specific data output to user log files or syslog.	Log file monitoring job, or a combination of the JP1/Base log file trap function and a JP1 event reception monitoring job	
5	Monitor messages output to the Windows event log.	Monitoring Event Log job, or a combination of the JP1/Base event log trap function and a Receive JP1 Event job	

Item	Task you want to perform in JP1/AJS3	Relevant JP1/AJS3 function	Relevant section
6	Execute jobs with a regular time interval between them, or execute jobnets at a regular interval.	Interval Control job	
7	Execute a jobnet at regular intervals.	Set the start condition and define an Interval Control job. • Start condition • Interval Control job	
8	Use the information of the preceding event job to execute subsequent processing (job or jobnet).	Inheriting of event job reception information	
9	Execute processing (of a job or jobnet) by using a received file with an unspecified filename transferred into a specified folder.	• Start condition • Monitoring Files job • Inheriting event job reception information	
10	Send JP1 events to a host of your choice.	Send JP1 Event job	*2.4.5 Sending a JP1 event at completion of the preceding job or when an event occurs (example of defining a jobnet that uses a Send JP1 event job)*
11	Execute a recovery job or jobnet when an error occurs during job execution.	Set a recovery job or recovery jobnet.	*2.4.6 Executing a specific process when a job ends abnormally (example of defining a jobnet that uses a recovery job or recovery jobnet)*
12	Put subsequent executions of a jobnet on hold when the jobnet ends abnormally	Set the hold method in the jobnet properties. • Hold if prev. = 'abend' • Hold if prev. = 'warning' or 'abend'	*15.3.8 Define Details - [Jobnet] dialog box (for a root jobnet)* in the *Job Management Partner 1/ Automatic Job Management System 3 Operator's Guide*
13	Change the definition of a jobnet during operation.	Create a jobnet based on the new definition, and register it for release in the schedule at the date and time when you want the definition to change.	*8.3 Switching a jobnet definition while the jobnet is registered for execution* in the *Job Management Partner 1/Automatic Job Management System 3 Administration Guide*

Item	Task you want to perform in JP1/AJS3	Relevant JP1/AJS3 function	Relevant section
14	Group together several root jobnets with different schedules so that their tasks can be performed as a single jobnet.	Planning group	*10.1 Using a planning group to change the plans for root jobnets* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*
15	Execute processing (job or jobnet) in response to mail sent from a particular person or mail with a particular subject line.	• Linkage with a mail system • Receive Mail job	*2. Linking Mail Systems* in the *Job Management Partner 1/Automatic Job Management System 3 Linkage Guide*
16	Receive email notification when a job or jobnet has completed, or when there is an error in the system.	• Linkage with a mail system • Send Mail job	
17	Monitor the operating status of JP1/AJS3 and job execution statuses by using HP NNM.	• Linkage with HP NNM • OpenView Status Report Job	*A. Monitoring Jobnets Using HP NNM* in the *Job Management Partner 1/ Automatic Job Management System 3 Linkage Guide*
18	Automate a work task whose parameters change with each execution, without having to change the jobnet definition.	Specify a macro variable and inherited information when registering the jobnet for execution.	*4.1.2 Specifying macro variable values during registration for execution* in the manual *Job Management Partner 1/ Automatic Job Management System 3 Overview*

2.4 Jobnet definition examples

This chapter provides some examples of jobnet definition, based on the characteristics of each units, and keeping in mind the specific requirements of the application and its processes.

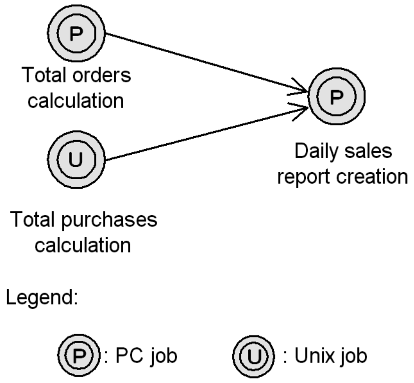
The descriptions in this chapter assume that the user is defining jobnets through a GUI. For details on the parameters you can specify in a unit definition, and how to use each type of unit, see *5. Defining Jobnets* in the *Job Management Partner 1/Automatic Job Management System 3 Operator's Guide*.

2.4.1 Executing a process in a specified file (example of defining a jobnet consisting of standard jobs)

Use standard jobs when defining a jobnet in which processes are performed by file execution.

In the example below, standard jobs are used to define a jobnet in which the total daily orders are calculated and output to a file by the program `juchu.exe`, the total purchases are calculated and output to a file by the program `shiire.exe`, and a daily sales report is prepared and printed, using sales figures calculated from these results, by the program `nippou.exe`. Host A (Windows) is used for calculating the total received orders and preparing the daily sales report. Host B (UNIX) is used for calculating the total purchases.

Figure 2-28: Example of defining a jobnet that uses standard jobs



The *total orders calculation* and *daily sales report creation* jobs are executed on a Windows host. Therefore, use PC jobs for these processes. Use a Unix job for the *total purchases calculation* since this job is executed on a UNIX host.

For the *total orders calculation* job, specify `hostA` in **Exec-agent**, and `juchu.exe` in **File name**.

For the *total purchases calculation* job, specify `hostB` in **Exec-agent**, and `shiire.exe` in **File name**.

For the *daily sales report creation* job, specify `hostA` in **Exec-agent**, and `nippou.exe` in **File name**.

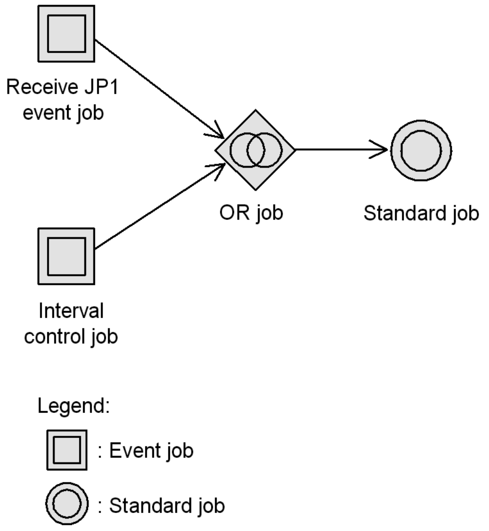
The *total orders calculation* and *total purchases calculation* jobs do not need to run in any particular order, but the sales data must be calculated from the results of these two jobs, giving the job flow shown above.

2.4.2 Executing a process when one of multiple conditions is satisfied (example of defining a jobnet that uses an OR job)

Use an OR job when defining a jobnet in which a process is performed when one of multiple conditions is satisfied.

In the example below, an OR job is used to define a jobnet that monitors the system for 10 minutes for receipt of a JP1 event. If the JP1 event is received, the succeeding job is executed immediately. If the JP1 event is still not received after 10 minutes, the succeeding job is executed at that point.

Figure 2-29: Example of defining a jobnet that uses an OR job



Only event jobs can be defined as the preceding jobs of an OR job. In this example, we want to watch for a JP1 event to be received, so we define a Receive JP1 event job. We use an Interval control job to monitor for elapse of the 10-minute time period, and define the wait time as 10 minutes.

If any event monitored by the event job defined as the preceding job of an OR job occurs, the succeeding job is executed, and other event jobs stop monitoring events and enter the *Bypassed* status. Thus, in this example, once the event being monitored by the Receive JP1 event job is received, the Interval control job stops monitoring for the interval to elapse. At completion of the succeeding job that was triggered by event reception, the jobnet also ends.

When the succeeding job performs an end judgment on the return value of the OR job, the OR job's return value is the same as the return value of the event job executed as the preceding job.

Cautionary notes

- If execution of an event job is stopped, the event job enters the *Not sched. to exe.* status. Thereafter, when the preceding job of the event job terminates, the status of the event job changes from *Not sched. to exe.* to *Bypassed*, and the succeeding OR job is executed.

  If the status of an event job has changed to *Killed* or *Ended abnormally* because of killing a jobnet or a timeout, other event jobs also stop monitoring events and enter the *Bypassed* status. In this case, however, the OR job is not executed because the preceding job of the OR job includes the abnormally terminated event job.

- When you restart execution from the event job defined as the first or second preceding job of an OR job, check whether the preceding jobs include an event job that has been terminated and placed in the *Bypassed* status. If there is such an event job, the OR job is executed immediately after the restart. In this situation, if, for example, you want the event job to monitor events, place all event jobs that directly precede the OR job in the *Ended abnormally* status. Then, re-execute the root jobnet with **From abnormally ended job** selected in **Rerun Method**.

2.4.3 Dynamically changing a process depending on the result of a preceding job (example of defining a jobnet that uses a judgment job)

Use a judgment job for a jobnet in which processing is to change dynamically depending on the results of the preceding job, on whether a file exists, or on the information inherited from the preceding job.

(1) Example of using a judgment job

JP1/AJS3 provides judgment jobs in which the next process is selected according to the end result of the preceding process. Judgment takes place according to the following two patterns:

- Judgment based on return code
- Judgment based on whether a file exists

The following provides examples of using each type of judgment job.

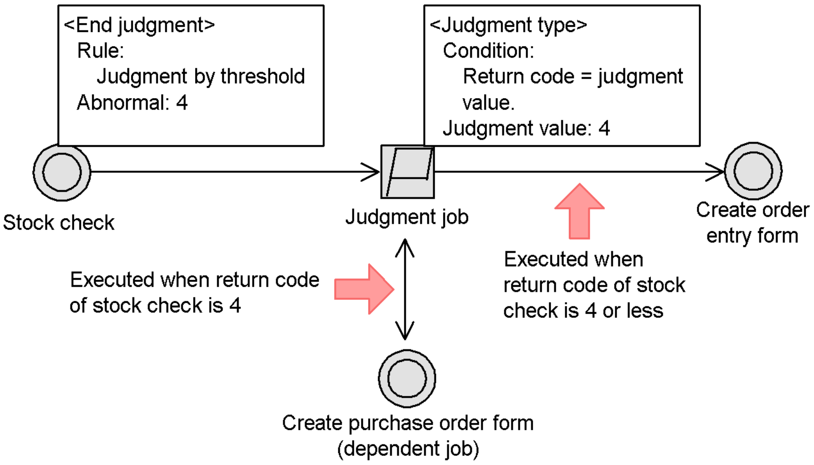
We also explain how to rerun a subordinate job if it ends abnormally.

(a) Judgment based on return code

The following figure shows an example of using a judgment job that works with return codes.

The jobnet created in this example creates a purchase order form if the stock level is low, or an order entry form if there is adequate stock, depending on the execution result of a job that checks the stock level.

Figure 2-30: Example of using a judgment job that works with return codes


<End judgment>
Rule:
Judgment by threshold
Abnormal: 4
<Judgment type>
Condition:
Return code = judgment value.
Judgment value: 4
Stock check
Judgment job
Create order entry form
Executed when return code of stock check is 4
Executed when return code of stock check is 4 or less
Create purchase order form (dependent job)

The stock check job returns 3 or a lower value if there is adequate stock, 4 if the stock level is low, and 5 or higher if there is a stock shortage.

By creating a processing flow like that in the figure above, you can execute the following processing:

- When there is adequate stock

  The return code is 3 or lower. Therefore, the *create purchase order form* job is skipped and the *create order entry form* job is executed.

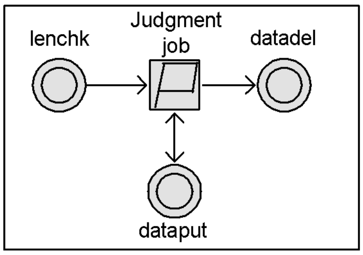
- When the stock level is low

  The return code is 4. Therefore, the *create purchase order form* job is executed first, followed by the *create order entry form* job.

(b) Judgment based on file presence or absence

The following figure shows an example of using a judgment job based on the presence or absence of a file.

Figure 2-31: Example of using a judgment job based on the presence or absence of a file



Specification of the executable file for the processing "lenchk"

If the number of records is 0, create chkfile.
If the number of records is not 0, delete chkfile.

Setting in the "lenchk" processing dialog box

Rule of **End judgment** :
"Judgment by threshold"

Setting in the judgment job dialog box

Judgment type : "File"
Condition : "File does not exist"
File name : "chkfile"

- If the number of records is 0 (if there is a chkfile), "dataput" is skipped and "datadel" is executed.
- If the number of records is not 0 (if there is no chkfile), "dataput" is executed and then "datadel" is executed.

By creating a processing flow like that shown in the figure above, you can also detect abnormalities in other processing (commands) executed in the executable file `lenchk`, and execute the subordinate job `dataput`.

Note that judgments based on the presence or absence of files are used to check files at the host where the work task manager (JP1/AJS3 - Manager) is installed. Depending on the environment settings (files on a Windows network drive or UNIX NFS mount), you might also be able to check the presence or absence of files at other hosts.

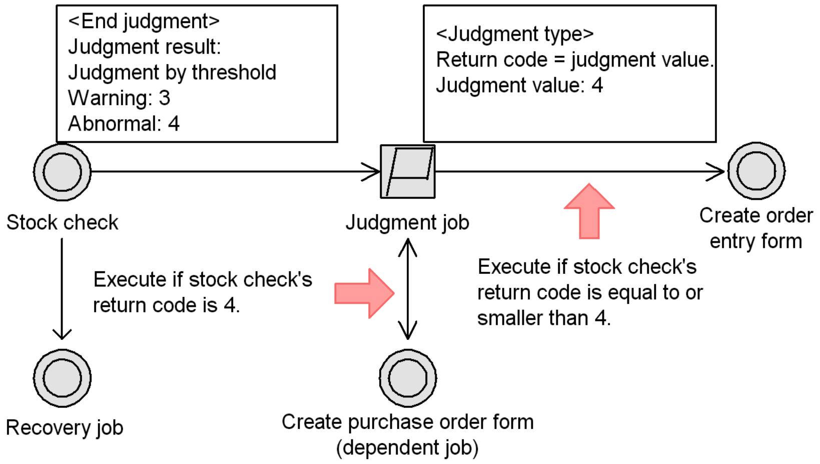
(c) Rerunning subordinate jobs

Subordinate jobs are rerun differently than normal jobs. For precautions on rerunning subordinate jobs, see *4.5.11 Rerunning a job or jobnet* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.

(2) Example of defining a jobnet that uses a judgment job

In the example below, a judgment job is used to define a jobnet that creates a purchase order form if the stock level is slightly down, or an order entry form if there is adequate stock, depending on the execution result of a job that checks the stock level. If there is a definite stock shortage, the stock check job ends abnormally and a recovery job is executed. For details on recovery jobs, see *2.4.6 Executing a specific process when a job ends abnormally (example of defining a jobnet that uses a recovery job or recovery jobnet)*.

Figure 2-32: Example of defining a jobnet that uses a judgment job


<End judgment>
Judgment result:
Judgment by threshold
Warning: 3
Abnormal: 4
<Judgment type>
Return code = judgment value.
Judgment value: 4
Stock check
Judgment job
Create order entry form
Execute if stock check's return code is 4.
Execute if stock check's return code is equal to or smaller than 4.
Recovery job
Create purchase order form (dependent job)

In this example, the judgment depends on the return value of the preceding job. The *stock check* job returns 3 or a lower value if there is adequate stock, 4 if the level is slightly down, and 5 or higher if there is a stock shortage.

First, set **Judgment by threshold** as the end judgment of the *stock check* job. Set the warning threshold to 3 and the abnormal threshold to 4. With these settings, the *stock check* job is judged by its return value, and its status changes as follows:

- Return value is 3 or lower: *Ended normally*
- Return value is 4: *Ended with warning*
- Return value is 5 or higher: *Ended abnormally*

Set the judgment job to execute a dependent job if the *stock check* job's return code equals the judgment value of 4.

Accordingly, the succeeding job of the *stock check* job is as follows:

- When the return value of the *stock check* job is 3 or lower:

  The return value does not match the judgment value of 4 set in the judgment job. Therefore, the dependent job *create purchase order form* is skipped and the *create order entry form* job is executed.

- When the return value of the *stock check* job is 4:

  The return value matches the judgment value set in the judgment job, so the dependent job *create purchase order form* is executed first. At normal termination of the dependent job, the *create order entry form* job is executed.

- When the return value of the *stock check* job is 5 or higher:

A recovery job is executed. Because no succeeding job is executed in the event of an abnormal termination, the *create order entry form* job is not executed after the recovery job.

You can define two or more judgment jobs in succession. This allows you to define a number of processing paths equivalent to the number of defined judgment jobs, each depending on the end result of the preceding job.

The following explains how to use multiple judgment jobs.

(3) Example of using a succession of judgment jobs

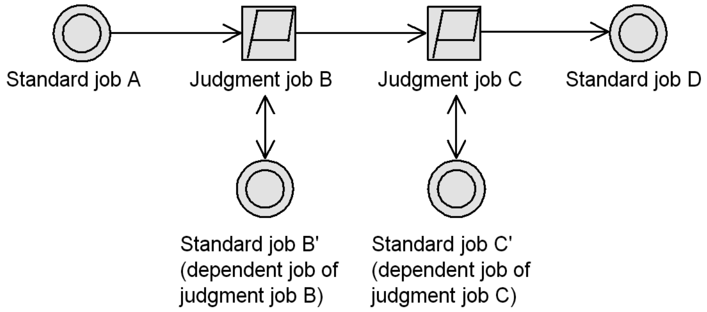
You can define two or more judgment jobs in succession.

When you define multiple judgment jobs based on a return code, the second and subsequent judgment jobs evaluate the return code set in the preceding job of the first judgment job, not the return code of the dependent job.

Conversely, when you define multiple judgment jobs based on the presence or absence of a file, the second and subsequent judgment jobs evaluate the file information current at the time they make their evaluation, not the file information current when the first judgment job performed its evaluation.

An example of defining a succession of judgment jobs is shown below.

Figure 2-33: Example of using a succession of judgment jobs


Standard job A
Judgment job B
Judgment job C
Standard job D
Standard job B' (dependent job of judgment job B)
Standard job C' (dependent job of judgment job C)

Suppose that each unit is defined as follows:

- Standard job A: **End judgment** is set to **Judgment by threshold**, and `5` is entered in the **Warning** field
- Judgment job B: The judgment type is **Return code = judgment value.**, and `0` is specified for **Judgment value**.
- Judgment job C: The judgment type is **Return code = judgment value.**, and `4` is specified for **Judgment value**.

When you execute this jobnet, the return value of standard job A determines the succeeding job, as follows:

- When standard job A returns 0

  The return code matches the condition set for judgment job B. Therefore, standard job B' (the dependent job of judgment job B) is executed. Next, judgment job C evaluates the return code. Because the return code does not match the condition set for judgment job C, C's dependent job (standard job C') is not executed. Instead, standard job D is executed.

- When standard job A returns 4

  Because this return code does not match the condition set for judgment job B, B's dependent job (standard job B') is not executed. Judgment job C now evaluates the return code. Because it matches the condition set for C, C's dependent job (standard job C') is executed, followed by standard job D.

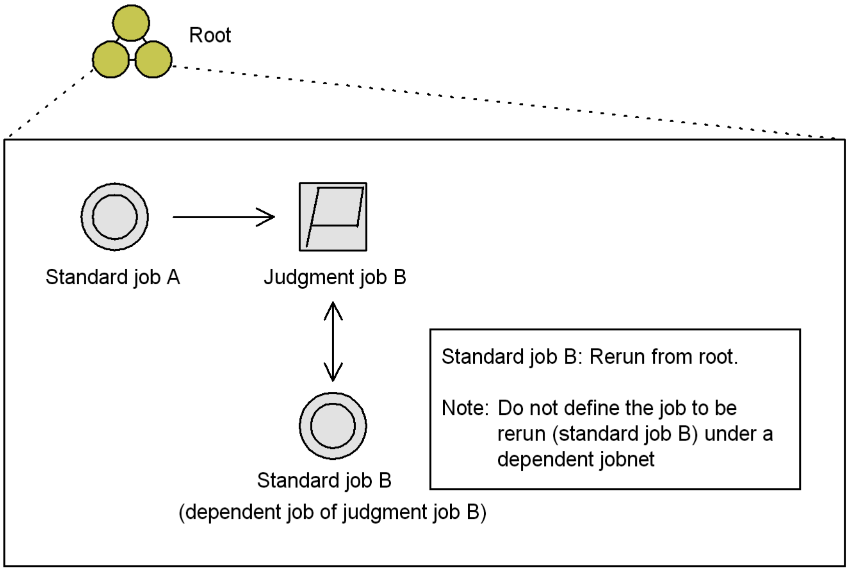
- When standard job A returns a value other than 0 or 4

  Because the return code matches neither of the conditions for judgment jobs B and C, neither of their dependent jobs (standard jobs B' and C') is executed. Only standard job D is executed.

(4) Cautionary notes

If you want to rerun a root jobnet or the preceding unit based on the result of a judgment job, define the job that you want to rerun as a dependent job as shown in the following figure:

Figure 2-34: Example of rerunning a root jobnet from the result of a judgment job



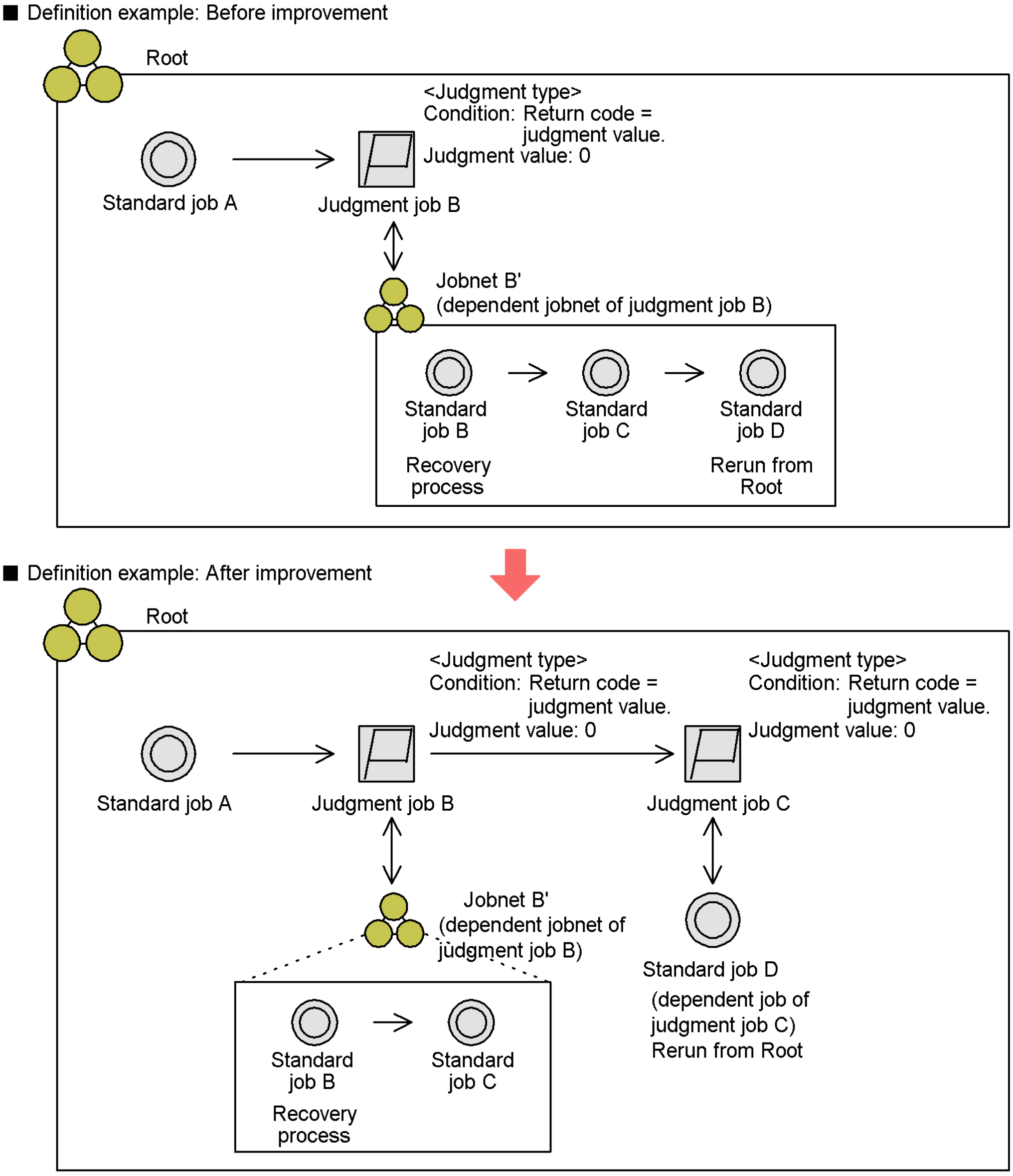
Standard job B: Rerun from root.

Note: Do not define the job to be rerun (standard job B) under a dependent jobnet

If you define the job that you want to rerun under a dependent jobnet, the dependent jobnet might execute again depending on the termination timing of the rerun job.

In the first of the two definition examples in the figure below, recovery and rerun processing are defined in a single dependent jobnet. If you want to implement a recovery process before rerunning a job, you must define a dependent jobnet that performs the recovery process and a dependent job that reruns the root jobnet, as shown by the second improved example.

Figure 2-35: Definition examples incorporating a recovery process before re-execution

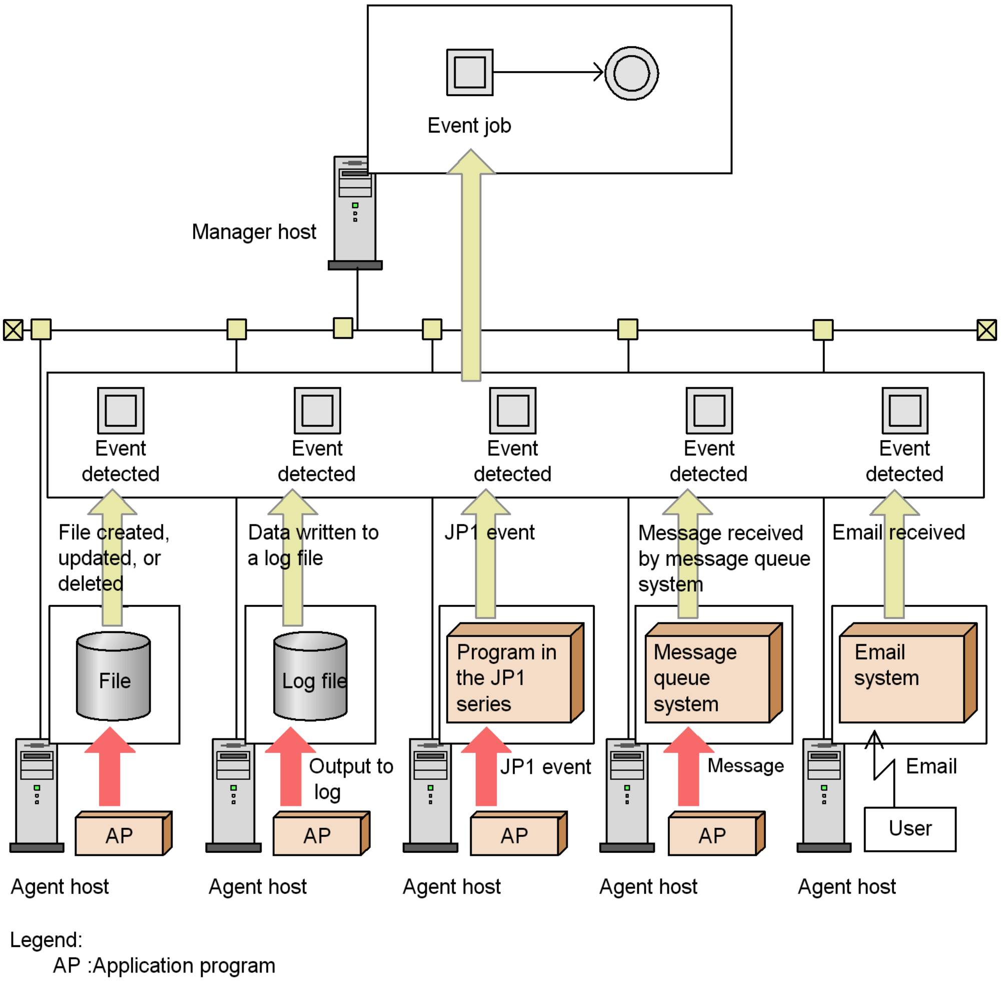


2.4.4 Executing an event-driven process (example of defining a jobnet that uses an event job)

Use an event job when defining a jobnet in which a process is triggered by an occurrence such as event receipt or file update.

An event job can be used to monitor events on any manager host or agent host on the network. The following figure shows the types of events that can be monitored.

Figure 2-36: Types of events that can be monitored by an event job



Timing of event detection by an event job

There might sometimes be a time lag between the execution time of an event job and the time that it is ready to monitor events. Events occurring during this time lag will not be detected. You should bear this in mind when defining a jobnet that uses an event job.

For Receive JP1 event jobs, you can use the find events prior to execution setting to solve this problem (see *(1) Executing a process on receipt of a JP1 event (Receive JP1 event job)* below.)

Using randomly occurring events to trigger a jobnet

When an event occurs more than once at irregular times, we recommend that you set a start condition for the jobnet triggered by that event. For details on setting a start condition, see *3.4 Defining a start condition* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.

If you know that the event you are monitoring occurs in a predictable manner, you can define an event job at the start of the jobnet, rather than defining a start condition.

Setting a timeout period for an event job

When a timeout period is set for an event job, the period is counted at the host on which the job is executed. If a power failure or other problem occurs at the target host and event monitoring is resumed after the host is restarted, the timeout period is counted again, beginning from the restart time. To suspend monitoring by the event job at an absolute time, counted from the execution start time regardless of the host status, define an event job as the jobnet start condition and set the valid range of start condition to absolute time. For details on start conditions, see *3.4 Defining a start condition* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.

You must also specify what status the event job is to be placed in after the timeout period specified for the job has elapsed. Select *Killed*, *Ended normally*, *Ended with warning*, or *Ended abnormally* status (the default is *Killed* status).

By setting an event job in this way, you can choose to cancel or continue the jobnet after the event job's timeout period elapses.

Passing event information

You can define the event information received by an event job as a variable (macro variable) that is passed to the succeeding job or jobnet. For details on passing event information, see *(6) Passing information received by an event job* below.

An example of an event job defined in a jobnet is given below. For additional information on defining an event job in a jobnet, see *7.6 Notes on using event jobs*

JP1/AJS3 must be linked with the appropriate program to use a Receive mail job. For details, see the *Job Management Partner 1/Automatic Job Management System 3 Linkage Guide*.

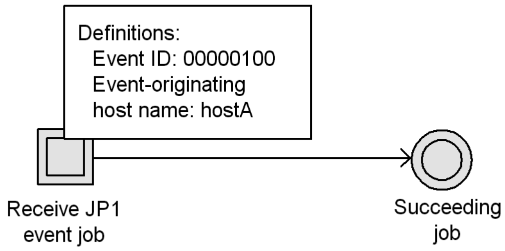
(1) Executing a process on receipt of a JP1 event (Receive JP1 event job)

A *JP1 event* is an event managed by JP1/Base, issued whenever some event occurs in a JP1 program. Because JP1 events contain information such as a severity level (error, warning, or notice) or message, you can execute a succeeding job or jobnet on receipt of an error or warning, or only when a specific message is received. Using the JP1/Base event converter, you can also execute a succeeding job or jobnet at termination of an external application program.

To execute a Receive JP1 event job, you must first start the JP1/Base event service. (In the API settings file, set `keep-alive` for the JP1/Base event service.) If this service is inactive, the Receive JP1 event job is placed in the *Now running* status until the service starts.

The following figure shows an example of defining a jobnet that executes a certain job when the Receive JP1 event job receives a JP1 event from hostA, which differs from the host where the jobnet was defined.

Figure 2-37: Example of defining a jobnet that uses a Receive JP1 event job



A Receive JP1 event job can link with a jobnet defined on another host.

On hostA, define a Send JP1 event job that sends an event with the event ID of `100`. To monitor any JP1 events issued by hostA, set hostA in the Receive JP1 event job as the event-originating host name. In addition, to execute the succeeding job when an expected JP1 event is received, set `100` as the event ID.

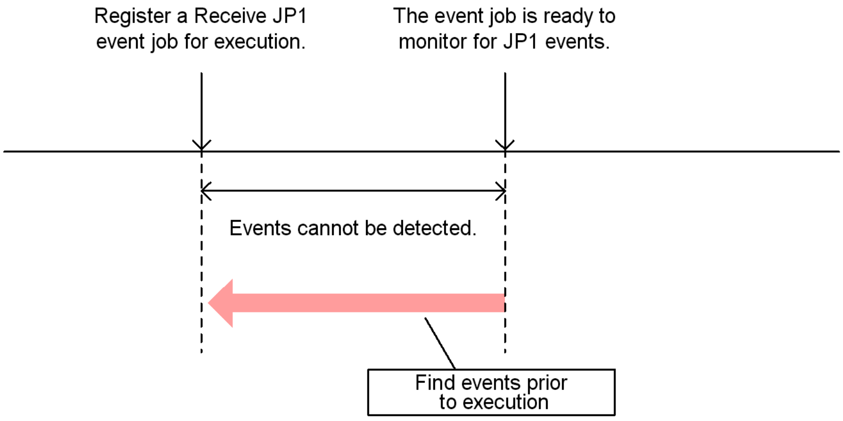
After these settings have been made, if the Send JP1 event job in the jobnet is executed on hostA, the jobnet stops monitoring JP1 events and executes the succeeding job.

Note that the system will not detect any JP1 event occurring during the time lag after execution of the Receive JP1 event job until it is actually able to monitor for receipt of JP1 events. One solution to this problem is the *find events prior to execution* setting.

- Find events prior to execution

  This feature searches for events occurring before an executed Receive JP1 event job is in ready state. Any matching event is identified as having occurred.

Figure 2-38: Find events prior to execution


Register a Receive JP1 event job for execution.
The event job is ready to monitor for JP1 events.
Events cannot be detected.
Find events prior to execution

Cautionary note

The find events prior to execution feature cannot be used with version 06-71 or earlier of JP1/AJS2 - Manager or JP1/AJS2 - Agent. If you are using an earlier version, remember that when you use a Receive JP1 event job, there might be a time lag before the job can begin monitoring JP1 events.

(2) Executing a process at file update (Monitoring files job)

Use a Monitoring files job when defining a jobnet in which file update or file creation triggers a job.

The monitoring options that you can specify for a Monitoring files job are described below.

Monitoring options

Specify the file status required as the condition to end monitoring and initiate the job. You can set any of the following four monitoring options:

- Creation of a file of the specified file name (**Create**)
- Deletion of a file of the specified file name (**Delete**)
- Change in size of a file of the specified file name (**Change size**)
- Change in last update time of a file of the specified file name (**Final time write**)

Supplementary notes

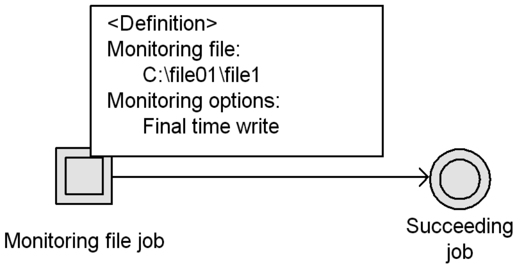
- When creation of a file with the specified name is being monitored, that file might already exist at the time the Monitoring files job starts. You can specify whether the job should assume that the monitoring condition has been satisfied if a file with the specified name already exists.

- You can specify more than one monitoring condition. For example, if you want to execute a succeeding job when a file is deleted or updated, you can specify the **Delete** and **Final time write** options. Note, however, that the **Change size** and **Final time write** options are mutually exclusive.
- For details on when a monitoring condition is or is not satisfied, see *7.6.2(1) Events monitored by the Monitoring Files job*.
- If any of the **Create**, **Change size**, and **Final time write** monitoring conditions are satisfied, the system performs a *close check* to make sure the monitored file is not being accessed by any process other than the Monitoring files job. If another process is accessing the file, the condition is judged to be unsatisfied, and another close check is performed when the next monitoring interval occurs. If the file is not being accessed by a process other than the Monitoring files job, the condition is judged to have been satisfied.

  The close check prevents the job from assuming that the condition is satisfied before the transmission (for example, copying) of a monitored file has been completed.

In the example below, a Monitoring files job is used to define a jobnet that monitors the write time to a particular file (file name: File1) and executes a succeeding job when the file is updated.

Figure 2-39: Example of defining a jobnet that uses a Monitoring files job



Specify the file to monitor and the monitoring option in the definition of the Monitoring files job. In the **Monitoring file** field, enter the path for File1, and in the **Monitoring option** area, select **Final time write**.

In this example, the Monitoring files job ends and the triggering condition is satisfied when the monitored file closes (no further applications are accessing the file) and the last update time changes.

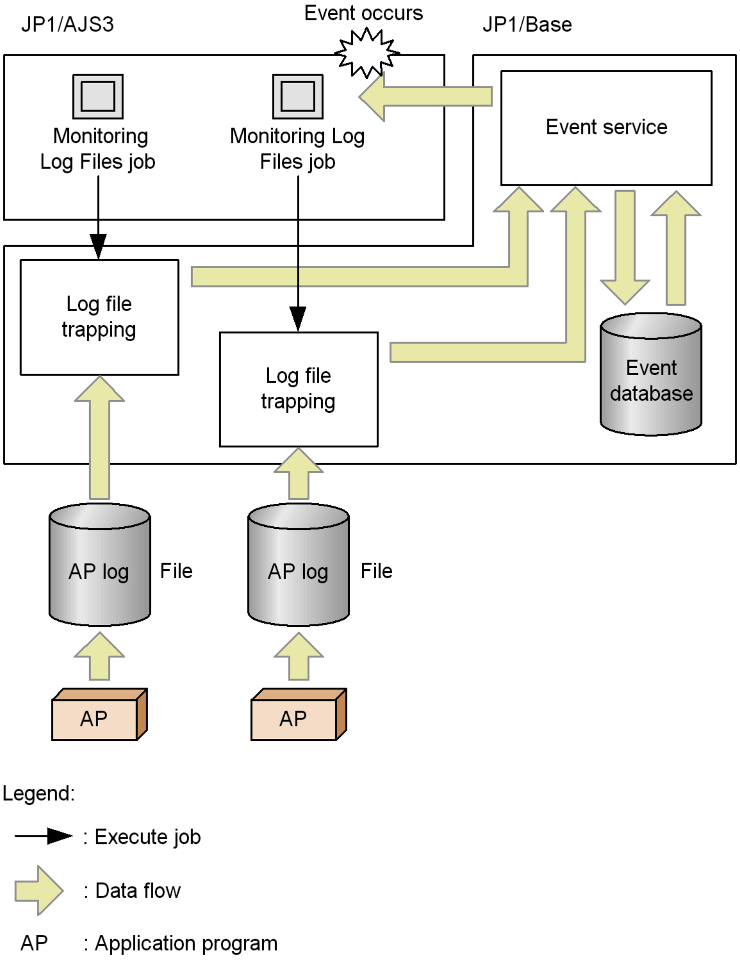
(3) Executing a process at log file update (Monitoring log files job)

A Monitoring log files job utilizes the JP1/Base log file trapping facility. JP1/Base log file trapping converts the log file records output by an application into JP1 events, and registers them in an event database. By defining a Monitoring log files job, you can execute a job or jobnet when a log file is updated.

To execute a Monitoring log files job, you must first start the JP1/Base log-file trap management service and JP1/Base event service. If these services are inactive, the Monitoring log files job is placed in the *Now running* status until the services start. For details on JP1/Base log file trapping, see the *Job Management Partner 1/Base User's Guide*.

The following figure shows how a Monitoring log files job operates.

Figure 2-40: Monitoring log files job



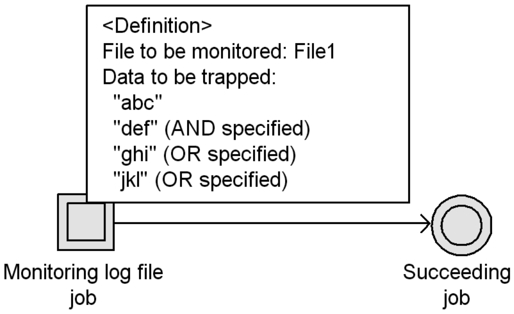
A Monitoring log files job monitors for output of specific data to a log file on the host specified as the monitoring target. In this job, you specify the log file and the character string to be monitored. Regular expressions can be used to specify the character string (see the *Job Management Partner 1/Base User's Guide* for regular expressions in Windows, or your UNIX documentation for regular expressions in UNIX).

As the monitoring interval, you can specify a value from 1 to 86,400 (seconds). Only log files in text format can be monitored. You can monitor up to eight log files. Do not specify log files that are mounted and unmounted at run-time. If you attempt to monitor this type of log file, the monitoring process might not work properly or the system might wrongly assume that a new log file has been created and start reading it from the top.

An example of setting a Monitoring log files job as the triggering condition is shown below.

In this example, the Monitoring log files job is used to define a jobnet that executes a succeeding job when log data containing a specific character string is written to the log file (file name: File1).

Figure 2-41: Example of defining a jobnet that uses a Monitoring log files job



Specify the log file name and the character strings to be monitored in the definition of the Monitoring log files job. In the **File to be monitored** field, enter the path for File1. In the **Data to be trapped** area, enter the desired character strings as shown in the figure.

In this example, the Monitoring log files job ends when data matching any of three conditions (a string containing "abc" and "def", a string containing "ghi", or a string containing "jkl") is written to File1 and the log data is retrieved from the log file. This satisfies the triggering condition and the succeeding job is executed.

(4) Executing a process on receipt of a Windows event log record (Monitoring event log job)

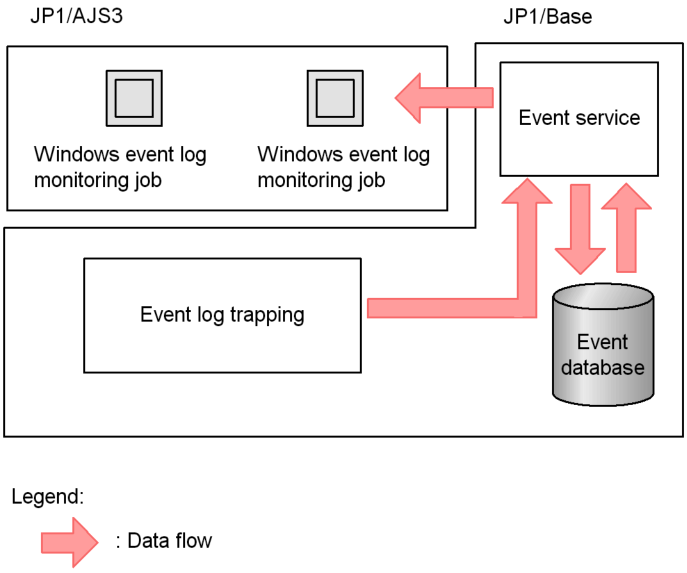
A Monitoring event log job utilizes the JP1/Base event log trapping facility, which converts Windows event log records into JP1 events and registers them in an event database. By defining a Monitoring event log job, you can execute a job or jobnet when a Windows event is logged.

To execute a Monitoring event log job, you must first start the JP1/Base event log trapping service and the JP1/Base event service. If these services are inactive, the Monitoring event log job is placed in the *Now running* status until the services start. In addition, to use a Monitoring event log job, you must also set the event log trapping behavior in the JP1/Base action definition file.

For details on JP1/Base event log trapping and the action definition file, see the *Job Management Partner 1/Base User's Guide*.

The following figure shows how a Monitoring event log job operates.

Figure 2-42: Event log trapping



A Monitoring event log job works as follows.

Log type

The types of logs that you can specify are listed below.

- **Application**
- **Security**
- **System**
- **DNS Server**
- **Directory Service**
- **File Replication Service**

Event types

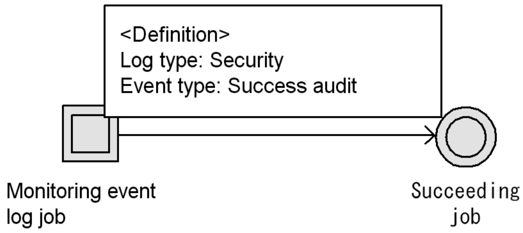
You can specify the following types of events:

- **Information**
- **Warning**
- **Error**
- **Success audit**
- **Failure audit**

An example of defining a Monitoring event log job is shown below.

In this example, the Monitoring event log job is used in a jobnet that executes a succeeding job when a Windows event reporting successful authentication on the security system is logged to the Windows event log.

Figure 2-43: Example of defining a jobnet that uses a Monitoring event log job



Before executing a Monitoring event log job, you must set the event log trapping behavior in the JP1/Base action definition file. In this example, the log type is `Security`, and the attribute name (with Type set) is `Audit_success`.

You must also set the log type and event type to be monitored. Set **Security** in **Log type**, and **Success audit** in **Event type**.

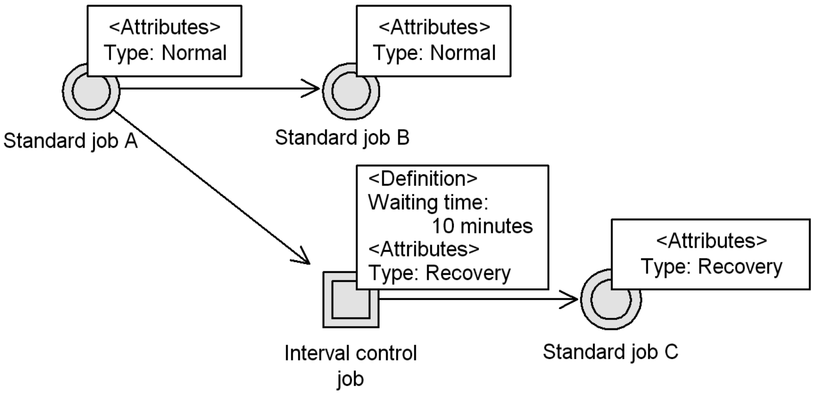
With these settings, the Monitoring event log job ends when the specified Windows event is output and the event log data is retrieved. This satisfies the triggering condition and the succeeding job is executed.

(5) Executing a process after a specified wait time (Interval control job)

Use an Interval control job when defining a jobnet in which a job is executed after a specified wait time.

In the example below, an Interval control job is used to define a jobnet that executes a recovery job 10 minutes later if the succeeding job ends abnormally.

Figure 2-44: Example of defining a jobnet that uses an Interval control job



If the preceding job ends abnormally, the next process is executed 10 minutes later. Therefore, use an Interval control job to monitor the time period. Set 10 minutes in **Waiting time**. Because the Interval control job will be executed at abnormal termination, set **Recovery** in the job **Type**. Likewise, set **Recovery** in **Type** for standard job C (the Interval control job's succeeding job).

With these settings, if standard job A ends abnormally, the Interval control job (recovery job) monitors the time until 10 minutes has passed, then standard job C (the succeeding job) is executed. If standard job A ends normally, standard job B is executed.

The **Waiting time** specified in an Interval control job refers to the wait time of the interval control process, not the time required to execute the Interval control job. Depending on the communication status or other factors, there might be a difference between the specified wait time and the actual interval.

(6) Passing information received by an event job

The event information received by an event job can be defined as a variable (macro variable) that is inherited by the succeeding job or jobnet. This inherited information is called *passing information*.

To pass event information to a succeeding job, you must define a macro variable in the event job and specify the same macro variable in the succeeding job.

In the event job, specify the macro variable in the following form:

Macro variable definition

`?AJS2`*macro-variable-name*`?:`*passing-information-name*

In `?AJS2`*macro-variable-name*`?`, specify a character string of up to 64 characters, using uppercase alphabetic characters (A to Z), numerals (0 to 9), and periods (.). In *passing-information-name*, you can specify only the passing information supported for the particular type of event job. For details, see *A. Information Passed by Event Jobs* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.

At execution of a macro variable, the passed information is expanded in the command line on the target host that executes the succeeding job of the event job. You should define a macro variable only if you are sure that the information to be passed is of a form that can be handled as a command argument when the succeeding job is executed.

The following explains how the information received by event jobs is passed to the various types of succeeding jobs.

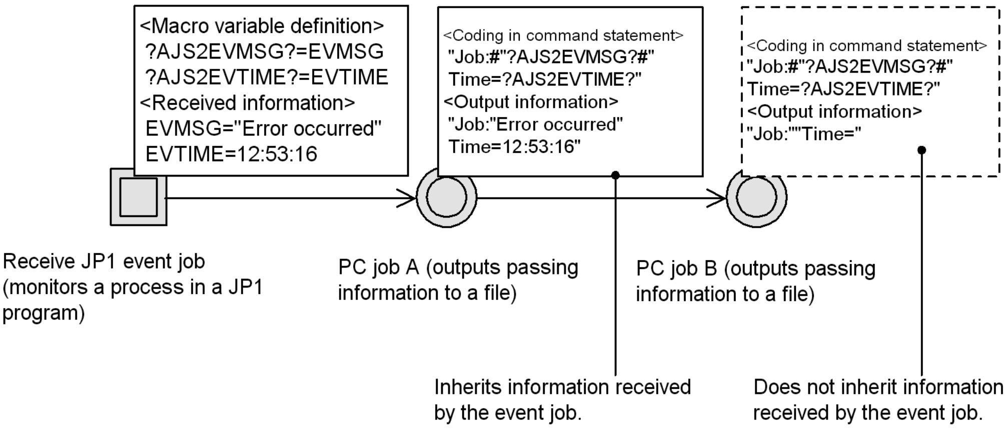
To make event job information available to an entire jobnet, use a start condition. (For details on start conditions, see *3.4 Defining a start condition* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.)

(a) Standard job or action job defined after an event job

The information received by an event job can be passed in a macro variable to a standard job (PC job or Unix job) or to an action job defined as the succeeding job. To enable the received event information to be passed, specify the macro variable using items that can accept a character string, such as command text, a script file name, parameter name, or environment variable.

The following figure shows an example of a PC job defined after an event job.

Figure 2-45: Example of a PC job defined after an event job


<Macro variable definition>
?AJS2EVMSG?=EVMSG
?AJS2EVTIME?=EVTIME
<Received information>
EVMSG="Error occurred"
EVTIME=12:53:16
<Coding in command statement>
"Job:#"?AJS2EVMSG?#"
Time=?AJS2EVTIME?"
<Output information>
"Job:"Error occurred"
Time=12:53:16"
<Coding in command statement>
"Job:#"?AJS2EVMSG?#"
Time=?AJS2EVTIME?"
<Output information>
"Job:""Time="
Receive JP1 event job (monitors a process in a JP1 program)
PC job A (outputs passing information to a file)
PC job B (outputs passing information to a file)
Inherits information received by the event job.
Does not inherit information received by the event job.

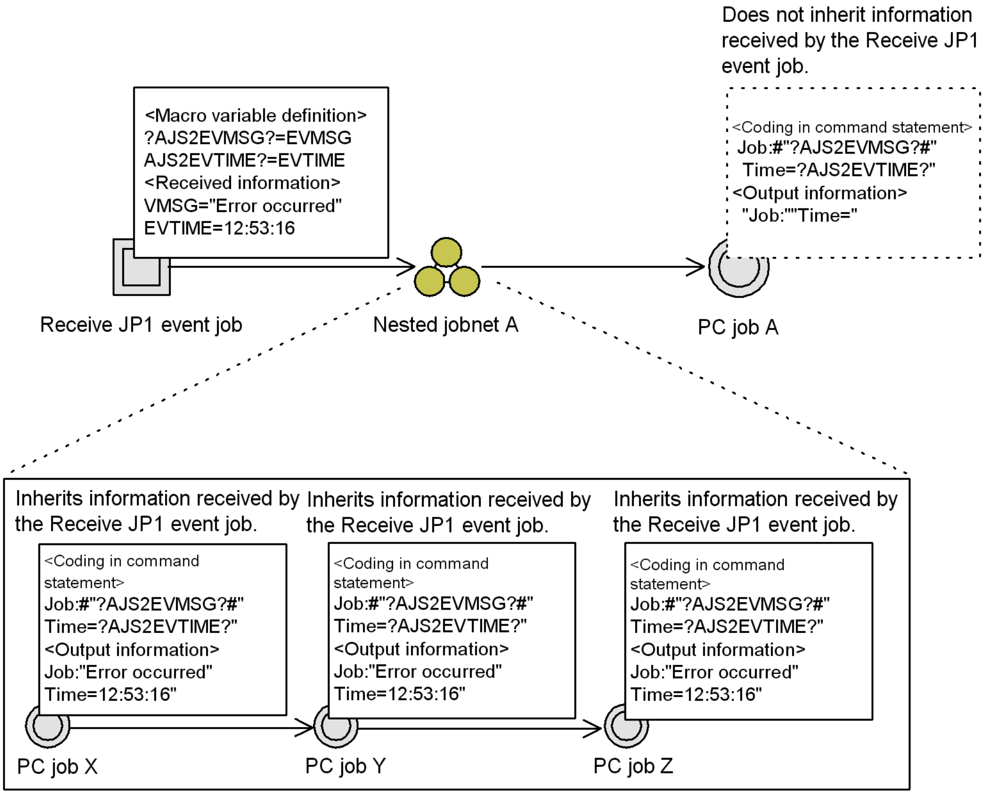
When a standard job (PC job or Unix job) or action job is defined as the succeeding job of an event job, the received information is passed only to the job immediately following the event job. In the above example, the event information received by the Receive JP1 event job is passed to PC job A. The information is not passed to PC job

B, even if PC job A ends abnormally.

(b) Nested jobnet defined after an event job

By specifying a macro variable in a nested jobnet defined as the succeeding unit of an event job, you can pass the received event information to the standard jobs (PC jobs or Unix jobs) and action jobs defined in the nested jobnet. The following figure shows an example of a nested jobnet defined after an event job.

Figure 2-46: Example of a nested jobnet defined after an event job

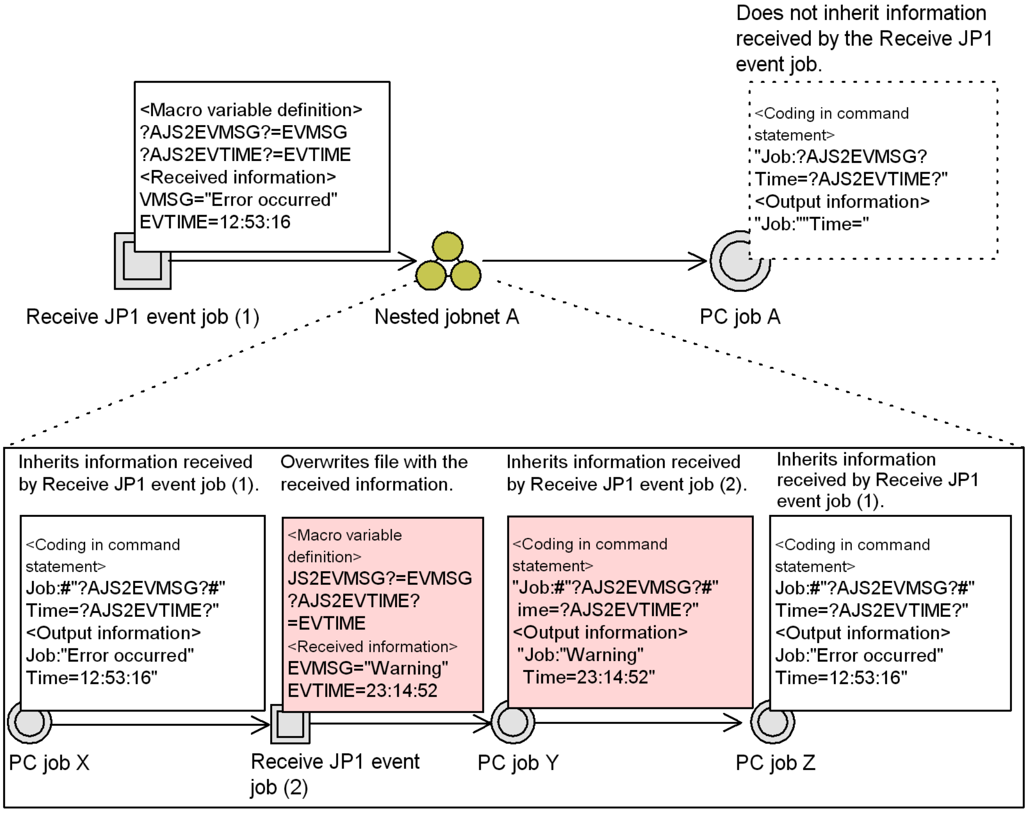


Note: Assume that each standard job inherits the macro variable and outputs data to file.

When a nested jobnet is defined as the succeeding unit of an event job, the received information is passed to all standard jobs (PC jobs or Unix jobs) and action jobs in the nested jobnet. In the above example, the event information received by the Receive JP1 event job is passed to PC jobs X, Y, and Z defined in nested jobnet A.

If the same event job with the same macro variable is also defined within the nested jobnet, the information received by the nested event job takes precedence. The following example shows the same event job and macro variable defined as a preceding job and also within the nested jobnet.

Figure 2-47: Example of the same event job defined in a nested jobnet



Note: Assume that each PC job inherits the macro variable and outputs data to file.

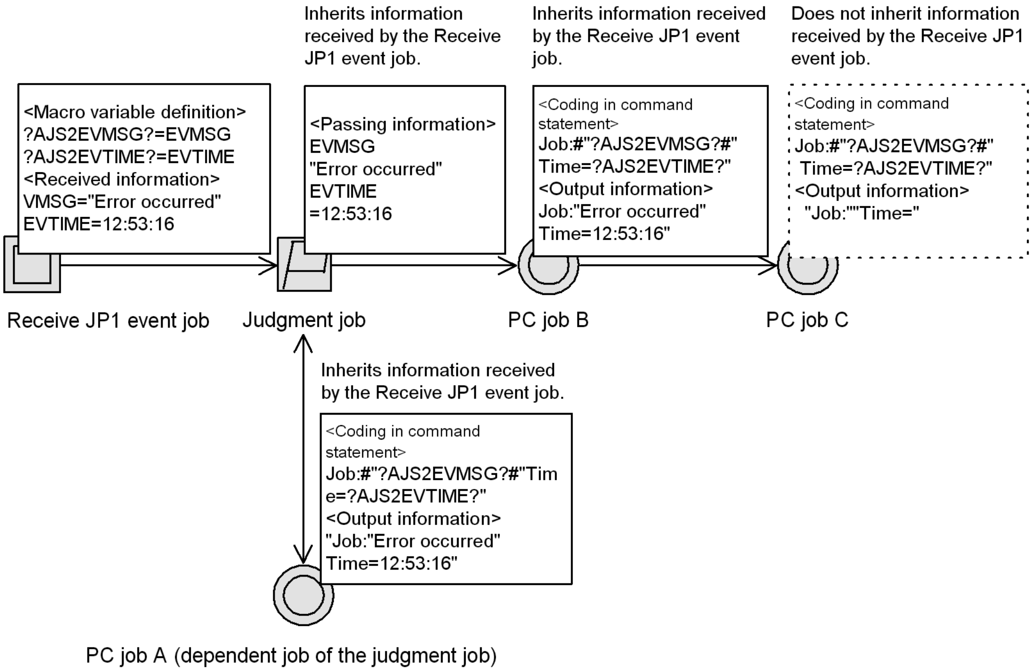
In this example, PC job X inherits event information from Receive JP1 event job (1), whereas PC job Y inherits the information received by Receive JP1 event job (2). The macro variable defined in Receive JP1 event job (1) is specified in PC job Z, so the information received by Receive JP1 event job (1) is passed to PC job Z.

(c) Judgment job defined after an event job

By specifying a macro variable in the dependent job of a judgment job that follows an event job, you can pass the information received by the event job to both the dependent job and the succeeding job of the judgment job.

The following figure shows an example of a judgment job defined after an event job.

Figure 2-48: Example of a judgment job defined after an event job


Inherits information received by the Receive JP1 event job.
Inherits information received by the Receive JP1 event job.
Does not inherit information received by the Receive JP1 event job.
<Macro variable definition>
?AJS2EVMSG?=EVMSG
?AJS2EVTIME?=EVTIME
<Received information>
VMSG="Error occurred"
EVTIME=12:53:16
<Passing information>
EVMSG
"Error occurred"
EVTIME
=12:53:16
<Coding in command statement>
Job:#"?AJS2EVMSG?#"
Time=?AJS2EVTIME?"
<Output information>
Job:"Error occurred"
Time=12:53:16"
<Coding in command statement>
Job:#"?AJS2EVMSG?#"
Time=?AJS2EVTIME?"
<Output information>
"Job:""Time="
Receive JP1 event job
Judgment job
PC job B
PC job C
Inherits information received by the Receive JP1 event job.
<Coding in command statement>
Job:#"?AJS2EVMSG?#"Time=?AJS2EVTIME?"
<Output information>
"Job:"Error occurred"
Time=12:53:16"
PC job A (dependent job of the judgment job)

Note: Assume that each PC job inherits the macro variable and outputs data to file.

When a judgment job is defined as the succeeding job of an event job, the received information is passed to both the dependent job and the succeeding job of the judgment job. In the above example, the event information received by the Receive JP1 event job is passed to PC job A (the dependent job) and to PC job B (the succeeding job).

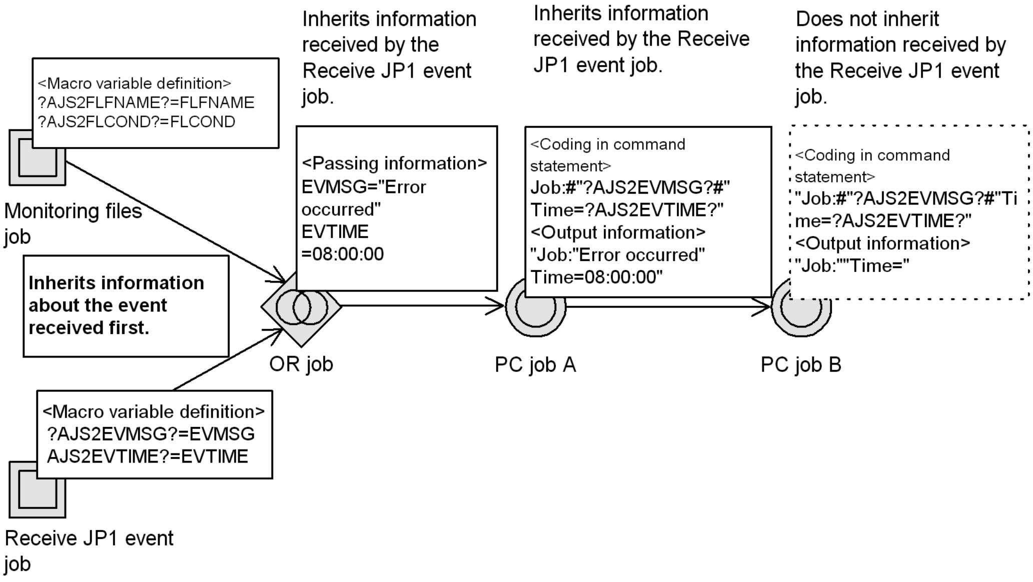
(d) OR job defined after an event job

By specifying a macro variable in the succeeding job of an OR job that follows an event job, you can pass the information received by the event job to the succeeding job of the OR job.

The following figure shows an example of an OR job defined after an event job.

Figure 2-49: Example of an OR job defined after an event job

- Event occurring at **8:00**: JP1 event
  (Event information EVMSG ="Error occurred" EVTIME=08:00:00)
- Event occurring at **10:00**: File size changed
  (Event information FLFNAME=C:\ACCEPT\DATA.dat FLCOND=s)


Inherits information received by the Receive JP1 event job.
Inherits information received by the Receive JP1 event job.
Does not inherit information received by the Receive JP1 event job.
<Macro variable definition>
?AJS2FLFNAME?=FLFNAME
?AJS2FLCOND?=FLCOND
Monitoring files job
<Passing information>
EVMSG="Error occurred"
EVTIME =08:00:00
<Coding in command statement>
Job:#"?AJS2EVMSG?#" Time=?AJS2EVTIME?"
<Output information>
"Job:"Error occurred" Time=08:00:00"
<Coding in command statement>
"Job:#"?AJS2EVMSG?#"Time=?AJS2EVTIME?"
<Output information>
"Job:""Time="
Inherits information about the event received first.
OR job
PC job A
PC job B
<Macro variable definition>
?AJS2EVMSG?=EVMSG
AJS2EVTIME?=EVTIME
Receive JP1 event job

Note: Assume that each PC job inherits the macro variable and outputs data to file.

When an OR job is defined as the succeeding job of an event job, the received information is passed to the succeeding job of the OR job. In the above example, events are being monitored by two types of event jobs: a Monitoring files job and a Receive JP1 event job. A JP1 event occurs at 8:00. The information received by the Receive JP1 event job on detecting this event is passed to the OR job's succeeding job, and the Monitoring files job is placed in *Bypassed* status.

Cautionary notes

- When a macro variable is specified in the command line of a succeeding job, the information will not be passed correctly if it contains a space or single quotation mark (').
- Do not pass data containing an escape order to the command line. Enclose the macro variable with double quotation marks ("). This prevents unpredictable behavior should a space be included in the passed information.
- When passing information is used in the command line of a job, any double

quotation marks (") included in the information are ignored. For example, AB"C is passed to the succeeding job in the form ABC. This might cause the job to execute incorrectly, depending on the command line constraints of the particular operating system. If the information contains a special character, use an environment variable to pass the information, rather than expanding the macro variable in the command line. Note, however, that by utilizing the feature for enabling double quotation marks, information containing double quotation marks (") can be passed as is. For details on this feature, see *4.3.7(4) Passing event data containing double quotation marks* in the *Job Management Partner 1/Automatic Job Management System 3 System Design (Configuration) Guide*. See also *6.3.4 Passing event data containing double quotation marks* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1* and *14.3.4 Passing event data containing double quotation marks* in the *Job Management Partner 1/ Automatic Job Management System 3 Configuration Guide 1*. This feature is not available in JP1/AJS2 - Manager 06-71 or earlier versions.

Supplementary notes

- When a single succeeding job follows multiple event jobs, it inherits the information received by all the event jobs. However, if you define the same macro variable for all the event jobs, the received event job information will be overwritten each time. The succeeding job will be able to reference only the most recently received information.
- When two or more macro variables with the same name are defined in the passing information of an event job, the information defined first is passed to the succeeding job. For example, if the first macro variable is ?AJS2111?:EVID (which assigns an event ID to ?AJS2111?), and the second macro variable is ?AJS2111?:EVMSG (which assigns message information to ?AJS2111?), the information assigned to this macro variable will be an event ID (EVID).
- If there is no information to be passed from the event job, or if the event job did not execute, the character string representing the macro variable name is passed to the macro variable defined in the succeeding job. For example, if the macro variable name is ?AJS2111?, the string ?AJS2111? is passed.

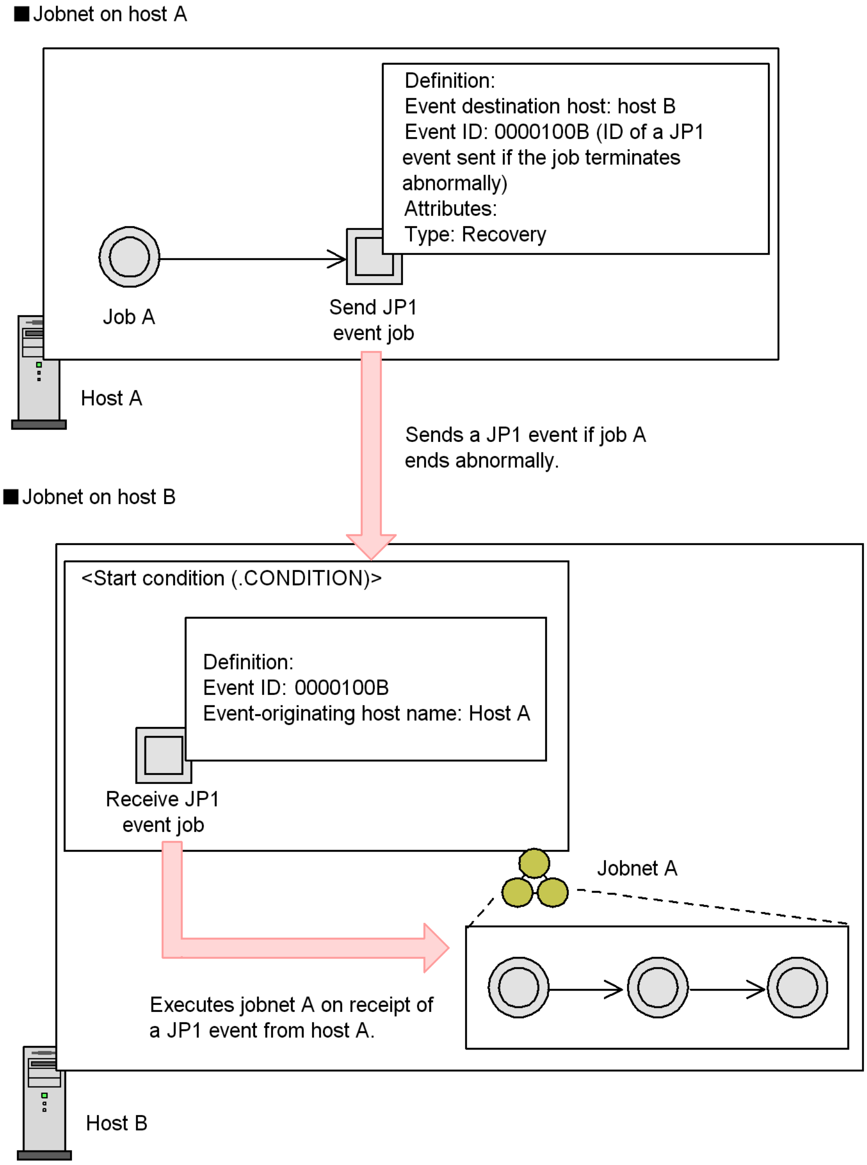
2.4.5 Sending a JP1 event at completion of the preceding job or when an event occurs (example of defining a jobnet that uses a Send JP1 event job)

Use a Send JP1 event job when defining a jobnet in which a JP1 event is sent when the preceding job completes or when an event occurs.

To execute a Send JP1 event job, you must first start the JP1/Base event service on both the sending host and the receiving host.

In the example below, a Send JP1 event job is used to define a jobnet in which a JP1 event is sent to host B if a job (job A) ends abnormally on host A. On receipt of the JP1 event, a jobnet to run after abnormal end (jobnet A) is executed on host B.

Figure 2-50: Example of defining a jobnet that uses a Send JP1 event job


■Jobnet on host A
Definition:
Event destination host: host B
Event ID: 0000100B (ID of a JP1 event sent if the job terminates abnormally)
Attributes:
Type: Recovery
Job A
Send JP1 event job
Host A
Sends a JP1 event if job A ends abnormally.
■Jobnet on host B
<Start condition (.CONDITION)>
Definition:
Event ID: 0000100B
Event-originating host name: Host A
Receive JP1 event job
Jobnet A
Executes jobnet A on receipt of a JP1 event from host A.
Host B

On host A, specify a Send JP1 event job as the succeeding job of job A. The Send JP1 event job is executed if job A ends abnormally. Therefore, set **Recovery** in the job **Type**. For details, see *2.4.6 Executing a specific process when a job ends abnormally (example of defining a jobnet that uses a recovery job or recovery jobnet)*. If job A ends abnormally, a JP1 event will be sent to host B. Therefore, specify host B in **Event destination host**. The JP1 event ID sent to host B is 0000100B. The JP1 event ID can be set to any user-specified value.

On host B, as the start condition to execute jobnet A, define a Receive JP1 event job for receiving the JP1 event from host A. (For details on start conditions, see *3.4 Defining a start condition* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*. In the Receive JP1 event job, set host A in **Host name** and specify the event ID sent from host A as 0000100B.

- Event arrival confirmation

  You can check whether a JP1 event was actually sent to a specified destination host by a Send JP1 event job. To perform this check, you can set a check interval (seconds) or a check count.

  Cautionary note

  > The event arrival confirmation facility is not supported in JP1/AJS2 - Manager 06-71 or earlier versions. If event arrival confirmation is specified for a Send JP1 event job distributed by JP1/AJS2 - Manager version 07-00 or later to a destination host running JP1/AJS2 - Agent version 06-71 or earlier, the event is sent but its arrival is not confirmed.

2.4.6 Executing a specific process when a job ends abnormally (example of defining a jobnet that uses a recovery job or recovery jobnet)

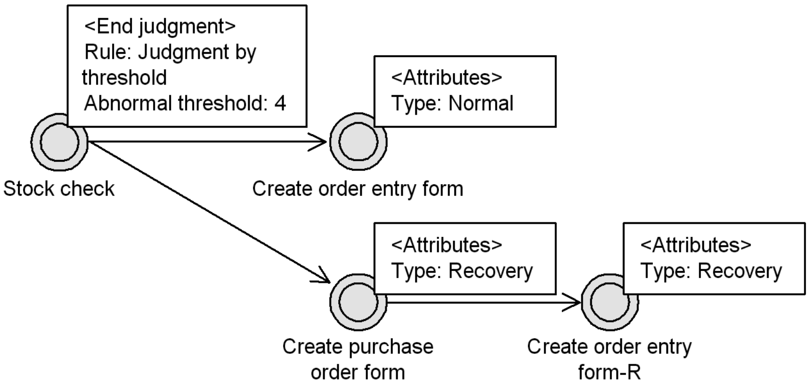
Use a recovery job or recovery jobnet when defining a jobnet in which a process is executed only if a job ends abnormally.

You can set a recovery job or recovery jobnet by entering **Recovery** in **Type** in the **Attributes** definition of the job or jobnet. Any job or jobnet can be set as a recovery job or jobnet.

(1) Example of defining a jobnet that uses a recovery job

In the example below, a recovery job is used in defining a jobnet that performs a stock check, and then creates an order entry form if there is adequate stock. If there is insufficient stock, the *stock check* job ends abnormally, a purchase order form is created, and then the order entry form is created.

Figure 2-51: Example of defining a jobnet that uses a recovery job



In this example, the *stock check* job returns 5 or a higher value if the execution result indicates a stock shortage.

First, set **Judgment by threshold** as the end judgment of the *stock check* job and set the abnormal threshold to 4. (That is, the *stock check* job ends normally if the return code is 4 or lower, or ends abnormally if the return code is 5 or higher.) If the *stock check* job ends normally, the *create order entry form* job will be executed next. Therefore, leave **Normal** (default) as the *create order entry form* job's **Type** setting. If the *stock check* job ends abnormally, the *create purchase order form* job will be executed next. Therefore, set **Recovery** in **Type** in the **Attributes** definition of the *create purchase order form* job. When this job ends, the *create order entry form-R* job will be executed. Set **Recovery** in **Type** for this job too (because only a recovery job or jobnet can be defined as the succeeding unit of a recovery job or jobnet).

Accordingly, the succeeding job of the *stock check* job is as follows:

- When the return code is 4 or lower

  The check result shows adequate stock and the *stock check* job ends normally. The succeeding job (*create order entry form*) will be executed next.

- When the return code is 5 or higher

  The check result shows that stock is too low and the *stock check* job ends abnormally. The recovery job (*create purchase order form*) is executed. When this job ends normally, the succeeding job of the recovery job (*create order entry form-R*) is executed.

Note that when a recovery job or recovery jobnet is executed (because the preceding job ended abnormally), even if that recovery job or jobnet ends normally, the entire jobnet in which those units are defined is assumed to have ended abnormally. Also, if no preceding job is defined, the recovery job or jobnet is not be executed and is set to *Not executed* + *Ended* status.

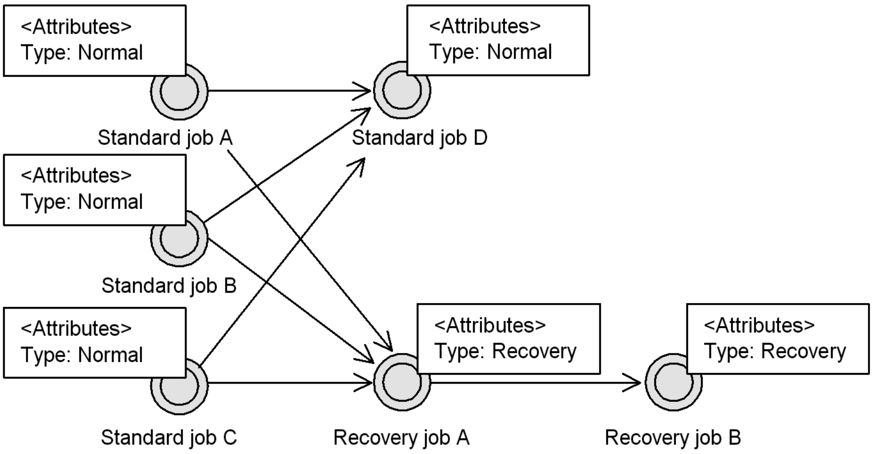
Supplementary note

When you define an OR job as a recovery job, you must define its preceding event jobs and its succeeding job as recovery jobs.

(2) Example of defining multiple preceding jobs for a recovery job

You can define multiple preceding jobs for a recovery job or recovery jobnet. In this case, the recovery job/jobnet will be executed only if all the preceding jobs end abnormally. An example of defining multiple preceding jobs for a recovery job is shown below.

Figure 2-52: Example of defining multiple preceding jobs



In this example, recovery job A is executed only if standard jobs A, B, and C all end abnormally. Recovery job B is executed following normal termination of recovery job A.

2.4.7 Controlling the execution order of a root jobnet (example of defining a jobnet that uses a jobnet connector)

Use jobnet connectors to define a process flow with characteristics like the following:

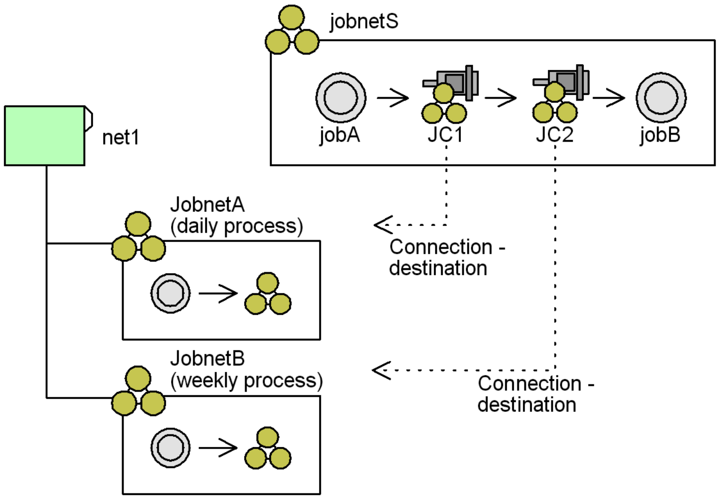
- A root jobnet *jobnetA* with a daily schedule is executed after a daily job *jobA*.
- A job *jobB* is executed after the root jobnet *jobnetA*.
- A root jobnet *jobnetB* with a weekly schedule is executed only on Sundays after *jobnetA*.

This example assumes that the root jobnets *jobnetA* and *jobnetB* have already been defined to execute on a daily and weekly schedule, respectively.

The following figure shows an example of defining jobnet connectors.

Figure 2-53: Example of defining jobnet connectors

■ Jobnet definition



■ Schedule definition

	Mon	Tue	Wed	Thu	Fri	Sat	Sun
jobnetS	🟦	🟦	🟦	🟦	🟦	🟦	🟦
jobnetA	🟦	🟦	🟦	🟦	🟦	🟦	🟦
jobnetB							🟦

First, create the root jobnet *jobnetS*, which incorporates the jobnet connectors. Then, create a job flow in which *jobA*, *jobnetA*, *jobnetB*, and *jobB* are executed sequentially. Associate the root jobnets *jobnetA* and *jobnetB* with the jobnet connectors *JC1* and *JC2*, respectively, and define a job flow in which *jobA*, *JC1*, *JC2*, and *jobB* are executed sequentially. Then, specify *jobnetA* as the connection-destination jobnet for *JC1*, and *jobnetB* as the connection-destination jobnet for *JC2*. Assign a schedule for daily execution to the root jobnet *jobnetS*.

Next, in the definitions of the connection-destination jobnets (*jobnetA* and *jobnetB*), specify the appropriate settings including the corresponding jobnet connector name and the method of execution order control. Use **Yes** as the **Exec. order control** setting for both root jobnets. As the jobnet connector name, specify *JC1* for *jobnetA* and *JC2* for *jobnetB*. When setting the execution order control method for each jobnet, specify whether the jobnet is to be executed in synchronization with the jobnet connector.

When you have finished defining the jobnets with jobnet connectors and their connection-destination jobnets, register them for execution. At this point, connections are established between generations that share an execution date according to the connection rules, and processing is executed in the defined order.

Chapter

3. Operation Calendar and Execution Schedule Considerations

After deciding the work tasks that you want to automate, you need to plan a calendar and execution schedule for work tasks in JP1/AJS3.

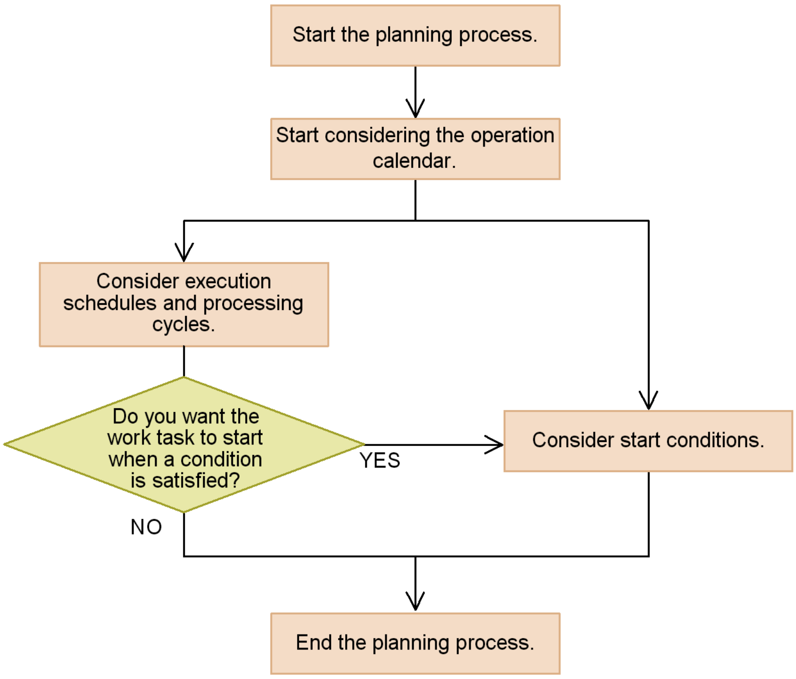
This chapter describes the considerations necessary for setting a calendar and execution schedule.

3.1 Flow of calendar and schedule planning

The flow for setting the schedule and calendar differs depending on the pattern of jobnet execution that you adopt.

The following figure shows the flow for setting the schedule and calendar based on the jobnet execution pattern.

Figure 3-1: Flow of calendar and schedule planning


Start the planning process.
Start considering the operation calendar.
Consider execution schedules and processing cycles.
Do you want the work task to start when a condition is satisfied?
YES
Consider start conditions.
NO
End the planning process.

3.2 Considerations when defining a calendar for JP1/AJS3 operation

You can create a JP1/AJS3 operational calendar for jobnet execution, in which open days and closed days such as the Sundays and holidays in an ordinary calendar are defined. When preparing the calendar, you need to consider the base day and base time settings, as well as which days to designate as open days and closed days.

Table 3-1: Matters to consider when creating a calendar

Consideration	Description
Open days and closed days	Consider the days on which jobnets will be executed (*open days*) and the days on which jobnets will not be executed (*closed days*).
Base day	Consider which day is to serve as the first day of the month. You can designate the base day as a specific date, or as the *n*th occurrence of a specific day of the week. For example: If the closing date for calculating salaries is the 20th of each month, by setting the 21st as the base day you can fix the period from the 21st of one month to the 20th of the next month as one month for the purpose of salary calculation.
Base time	Consider the time at which you want each day to start. For example: By setting 8:00 as the base time, processing that takes place at 1:00 on the following calendar day is treated as part of the current day.
Exclusive schedule	Consider whether you want a particular jobnet not to be executed if its execution schedule coincides with that of another jobnet.

For details about defining a calendar for JP1/AJS3 operation, see *3.2 Defining a calendar for JP1/AJS3 operation* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.

3.3 Considerations when defining a jobnet execution schedule

Plan the *schedule rules* that determine the execution schedule for a jobnet. These include the start time and processing cycle for the jobnet.

The following table lists and explains the considerations when you plan jobnet execution schedules.

Table 3-2: Matters to consider when defining a jobnet schedule

Consideration	Description
Execution start date and time	Consider the following matters, which determine how the start date and time of the jobnet are calculated: • The method of specifying the start date (Registered day, Absolute day, Relative day, Open day, Closed day) • The method of specifying the start time (Absolute time, Relative time) • Whether to set a base day and base time
Processing cycle	Consider whether to execute jobnets on a regular schedule. You can run a jobnet weekly, monthly, or yearly.
Method for closed day substitutions	Consider whether jobnets whose planned execution date according to the processing cycle falls on a closed day are to be executed on another day instead.

For details about schedule rules, see *3.3 Defining a schedule* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*. For details about how to set schedules for a variety of operation patterns, see *3.5 Setting schedules*. Use these references when you consider how to define a jobnet execution schedule.

3.4 Start condition considerations

JP1/AJS3 allows you to execute even irregular work tasks, which do not permit specification of a time of execution in advance, by specifying a *start condition* that stipulates the conditions under which the work task starts. As a start condition, you define an event job that monitors for an event like those listed below. You can associate more than one event job with a start condition.

- When a particular file is updated (Monitoring Files job)
- When a particular JP1 event is received (Receive JP1 Event job)
- When a specified period of time has elapsed (Interval Control job)
- When a particular character string is output to a log file (Monitoring Log Files job)
- When email is received (Receive Mail job#)

#: Linkage with a mail system is required.

For details about start conditions, see *3.4 Defining a start condition* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.

Consider the following when using a start condition to execute a jobnet.

(a) Valid range of start conditions

Consider the range in which the occurrence of an event defined as a start condition is to be monitored. You can specify the valid range as a number of executions or a specific time. Set the valid range of a start condition when defining the schedule rules for a jobnet.

Number of executions

> Specify the number of times that the jobnet can be executed from the time that monitoring for the start condition begins.

Period

> Monitoring of the start condition continues until the specified time arrives. You can specify the time as an absolute or relative time.

(b) Operation when multiple event jobs are used in a start condition

When multiple event jobs are defined, consider whether you want the start condition to be satisfied when all the events occur (an AND condition), or when any one of the defined events occurs (an OR condition).

(c) Concurrent execution of monitoring and execution generations

When you execute a jobnet with a start condition, generations which monitor for the occurrence of the events defined in the start condition (monitoring generations), and generations which are executed when the start condition is satisfied (execution generations), are both generated. For jobnets with start conditions, consider whether to allow concurrent execution for monitoring generations, and for execution generations.

Concurrent execution of execution generations

In situations where the start condition is satisfied multiple times, consider whether you want a new generation of the jobnet to run concurrently with the previous generation or to wait until the previous generation has ended. This behavior is determined by the concurrent execution setting in the jobnet definition.

If you disable concurrent execution, you can choose whether to hold any execution generations generated during execution of the previous generation.

Concurrent execution of monitoring generations

For jobnets with a start condition and a processing cycle, consider how you want a new monitoring generation to be executed if the previous monitoring generation is still running when its start time is reached. You can have the new monitoring generation execute concurrently, wait until the previous monitoring generation has completed, or skip execution altogether.

(d) When a jobnet with a start condition ends abnormally

Consider the behavior of the system after a jobnet with a start condition ends abnormally. You can select from the following behavior:

- Start execution of jobnet

  After ending abnormally, the jobnet starts at every subsequent occurrence of the event being monitored by the start condition.

- Hold start of jobnet

  After ending abnormally, the jobnet is placed in *Held* status, or remains in *Wait for start condition* status with future execution suspended.

- Stop monitoring of start conditions

  After the jobnet ends abnormally, the system stops monitoring for the start condition.

(e) Other considerations

Also consider the following:

- How often a condition will be satisfied within a given time frame# (how many event jobs are likely to be executed).

#: You can use this information when you estimate event job performance.

3.5 Setting schedules

You can adapt JP1/AJS3 to operate in a variety of ways, using functions such as the jobnet scheduling function.

3.5.1 Establishing schedules for applications that extend over two days

A work task executed late at night will often extend over more than one day. Sometimes the date change can lead to the system reporting a scheduling error. For example, suppose that you calculate the daily sales data from Monday to Friday at 1:00 a.m. on the following day, and Saturdays are closed days. If you set the start time for the calculation process to 1:00 on the following day, the calculation of Friday's sales data will be scheduled to begin at 1:00 on Saturday. However, because Saturday is a closed day, the calculation process cannot be executed.

By setting 25:00 as the start time, you can ensure that the calculation process is executed even when Saturday is a closed day. However, sometimes the root jobnet and the nested jobnet will be scheduled to execute on different days. As a result, the nested jobnet might fail to execute.

In such cases, you can change the length of time that JP1/AJS3 treats as 1 day, so that a process that runs over two days can fit within 1 day under JP1/AJS3. You can use one of the following methods to change the length of time that JP1/AJS3 treats as one day.

- Set 1 day to 48 hours, and calculate schedules based on a 48-hour day.

  With this method, you define the schedule of the root jobnet using the 48-hour schedule.

- Set the start time for 1 day to a time other than 0:00.

  With this method, you specify a time other than 0:00 as the base time.

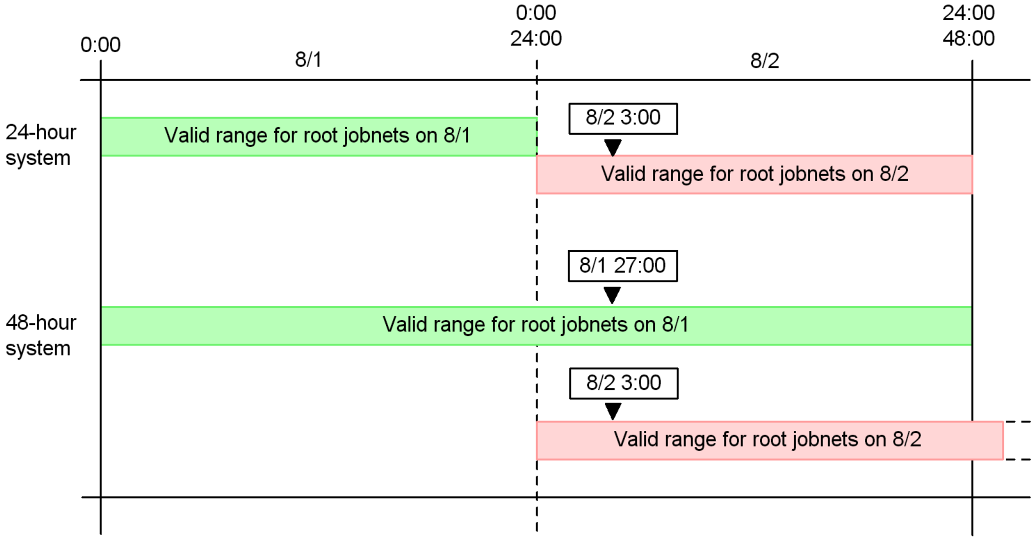
We recommend that you use the 48-hour schedule to define the schedule of the root jobnet. This method enables you to create schedules more easily. Each method is described below.

(1) Defining an application that extends over two days using a 48-hour schedule

If you define the schedule of the root jobnet using the 48-hour schedule, the time between 0:00 and 23:59 on the next day in the calendar is treated as part of the current day, from 24:00 to 47:59. For example, you can refer to 1 a.m. on Saturday as 25:00 on Friday. If Saturday is a closed day, as in the example above, you can use the 48-hour schedule and set the execution start time to 25:00 on Friday. This will allow the job to execute on schedule.

The following figure shows the difference between a 24-hour schedule and a 48-hour schedule.

Figure 3-2: Difference between a 24-hour schedule and a 48-hour schedule


0:00
0:00
24:00
24:00
48:00
8/1
8/2
24-hour system
Valid range for root jobnets on 8/1
8/2 3:00
Valid range for root jobnets on 8/2
48-hour system
8/1 27:00
Valid range for root jobnets on 8/1
8/2 3:00
Valid range for root jobnets on 8/2

If you specify the execution start time as an absolute time under the 48-hour schedule (base time 0:00), it can be expressed in two different ways. For example, a job that executes 3 a.m. can be expressed as 8/1 27:00 or 8/2 3:00. Although the actual time is the same, the scheduled execution date is different. This means that when the schedule is affected by closed days/open days, or by shifting the execution schedule, the application may not be executed.

Note that if you use the 24-hour schedule, you can still specify a time between 24:00 and 47:59. However, if for example you specify the time 8/1 27:00 under the 24-hour schedule, it is interpreted as 8/2 3:00, and the schedule will be set accordingly.

(a) Changing the time system from the 24-hour schedule to the 48-hour schedule

To set up a schedule for a jobnet based on the 48-hour schedule, you need to set up the environment for the scheduler service.

To change a schedule from the 24-hour schedule to the 48-hour schedule:

1. Change the value of the `ROOTJOBNETSCHEDULERANGE` environment setting parameter of the scheduler service to `00000030`.
2. Specify a base time of 0:00 in the details of the job group.

   If you do not specify a base time, the default is used.
3. Specify a time later than 24:00 as the execution start time for the jobnet.

For details about how to set environment setting parameters, see *4.2 Environment setting parameter settings* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1* (for Windows systems), or *13.2 Environment setting parameter settings* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1* (for UNIX systems).

(b) Reviewing schedules after changing to the 48-hour schedule

When you change the schedule of a root jobnet from the 24-hour to the 48-hour schedule, the behavior of the schedule for the root jobnet will change. You should therefore review existing schedules after changing them to the 48-hour schedule. You should change the schedule rules for a jobnet when they meet any of the following conditions:

1. You defined a time later than 24:00 as the execution start time for the root jobnet, and the base time is set to a value other than 0:00.

   This combination of factors causes the execution start date, the target date and time to which a delay time applies, and the event wait time to change for the root jobnet and nested jobnets. Check the schedule definitions.

2. You defined an exclusive schedule for a jobnet or planning group that is in the same hierarchy as a jobnet that meets the condition 1 above.

   This causes the execution start date of the exclusive schedule to change. Check the schedule definitions.

3. A jobnet that meets conditions 1 or 2 above contains a nested jobnet whose schedule does not depend on the upper-level jobnet.

   This causes the execution start date of the nested jobnet to change. Check the schedule definitions.

(2) Defining applications that extend over two days by changing the base time

Normally, JP1/AJS3 considers 1 day to be the 24 hours between 0:00 on a certain day and 0:00 on the next day. By shifting the time when 1 day starts, you can fit processes that run over two different days into one day. For example, if you set the base time to 8:00, JP1/AJS3 treats the 24 hours from 8:00 until 8:00 the next day as 1 day. The time from 0:00 until 7:59 on the current day is counted as part of the previous day.

Changing the base time can complicate the applications involved in schedule definition, such as setting execution start times and delay times. For example, you must specify the time in different ways depending on the type of execution start time. When the base time is 8:00, you can specify a start time of 1:00 on August 5 in one of the ways shown below.

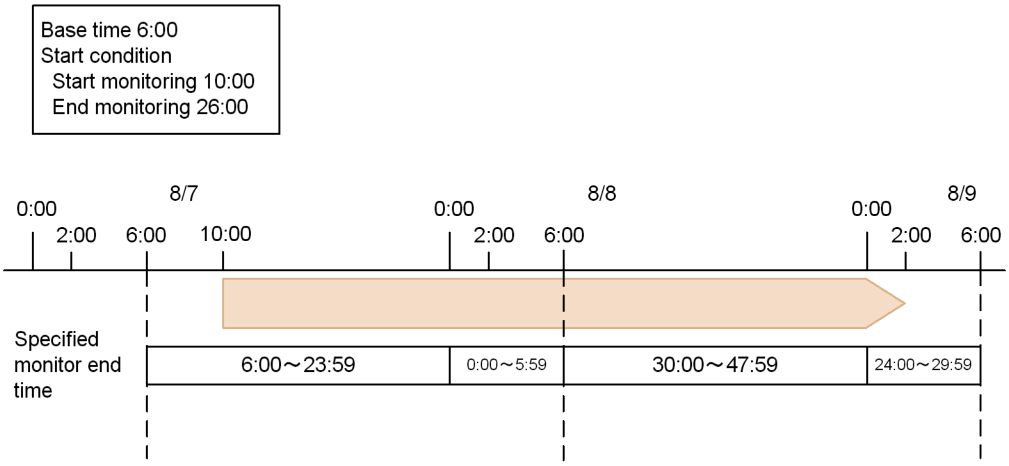
Table 3-3: Setting an execution start time

Type of execution start time	Type of execution start date	Time specified
Absolute time	Registered day, Absolute day, Relative day	8/5 1:00 (alternatively 8/4 25:00)
Absolute time	Open day, Closed day	8/4 1:00
Relative time	All	8/4 17:00

(a) When a start condition is defined for a jobnet

Imagine you are using the 24-hour schedule, and have set the base time to a value other than 0:00. If you specify a start condition for a jobnet, and an end time for the start condition that is later than 24:00 but earlier than the base time, the end time of the start condition is treated as falling on the day after next. The following figure shows an example where this happens.

Figure 3-3: End time of the start condition becomes the day after next



From this example, it is apparent that if you have set the base time to a value other than 0:00, specifying a time between 0:00 (24:00) and the base time is complicated. We therefore recommend that you use the 48-hour schedule to define the schedule.

3.5.2 Setting multiple execution start times

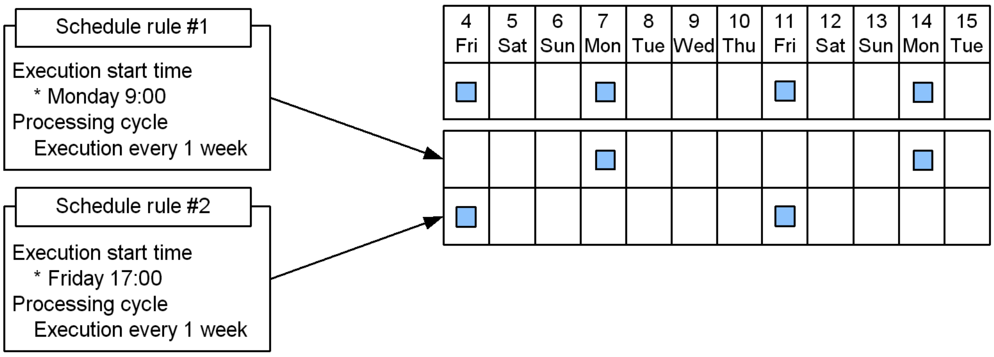
Sometimes, you may need to operate a jobnet according to a complex schedule that cannot be expressed by a single schedule rule. JP1/AJS3 allows you to create multiple schedule rules for a jobnet. You can define up to 144 schedule rules for a single jobnet. When multiple schedule rules are defined, the schedule is set with the earliest schedule rule.

For example, you may want to run a jobnet multiple times a day at fixed times, or for the jobnet to have different execution times on different days of the week. In such cases, you can define the schedule for the jobnet by creating multiple schedule rules. If more than one execution is scheduled for the same execution start date and time, as a result of setting multiple schedule rules, then the jobnet is executed only once.

Note that when you set multiple schedule rules, a jobnet may end up being scheduled for execution more than once at the same execution start time. In this case, the jobnet will be executed only once.

The following is an example of a schedule with multiple schedule rules.

Figure 3-4: Schedule with multiple schedule rules



This schedule is for a jobnet that will execute twice a week, at 9:00 on Monday and 17:00 on Friday.

Schedule Rule #1 defines Monday's schedule, and Schedule Rule #2 defines Friday's schedule. When you register the jobnet for execution, the execution schedule for the jobnet is calculated based on both of these schedule rules.

3.5.3 Defining a different schedule for some jobs in a jobnet

To execute some jobs in a jobnet according to different schedules, use nested jobnets.

You can set schedule rules for nested jobnets, as with root jobnets. You define schedule rules for nested jobnets by linking them with the schedule rules for the root jobnet. You can link multiple nested jobnet schedule rules to one root jobnet schedule rule.

If you do not define a schedule for a nested jobnet, the nested jobnet is executed according to the same schedule as the root jobnet.

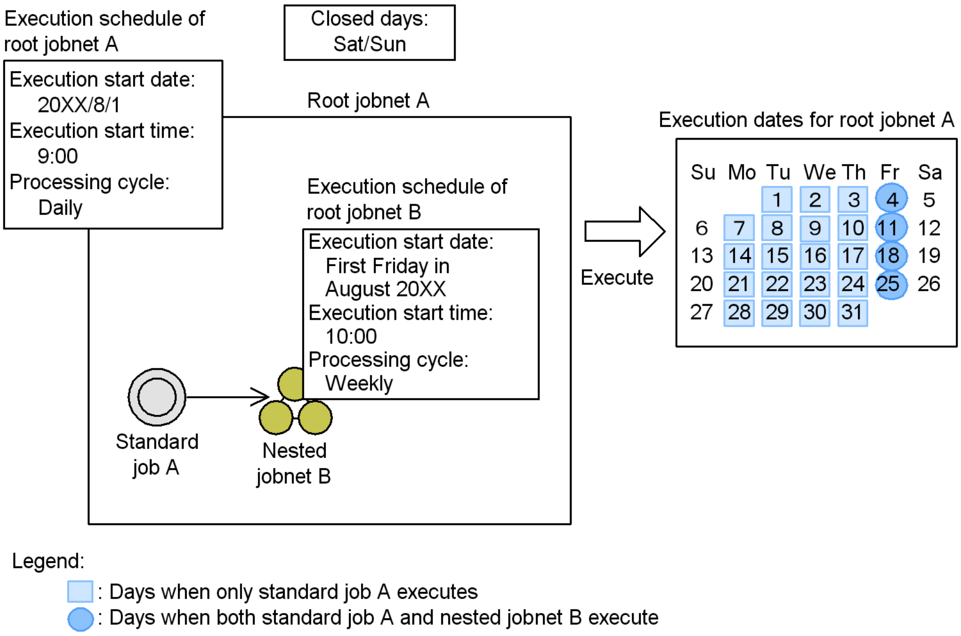
(1) Defining a schedule for a nested jobnet

First, create a nested jobnet under a root jobnet. Define the jobs that you want to execute according to a different schedule under the nested jobnet. Once you have defined the nested jobnet, set the schedule rules for the nested jobnet.

Now register your defined root jobnet for execution. The nested jobnet runs according to the schedule you defined. However, the nested jobnet is only executed when the execution conditions of the upper-level jobnet are met. Even if you have defined a schedule for the nested jobnet, it will not be executed on days when the root jobnet is not scheduled for execution.

The following figure shows the use of a schedule for a nested jobnet.

Figure 3-5: Using a schedule to execute a nested jobnet


Execution schedule of root jobnet A
Execution start date: 20XX/8/1
Execution start time: 9:00
Processing cycle: Daily
Closed days: Sat/Sun
Root jobnet A
Execution schedule of root jobnet B
Execution start date: First Friday in August 20XX
Execution start time: 10:00
Processing cycle: Weekly
Standard job A
Nested jobnet B
Execute
Execution dates for root jobnet A

Su	Mo	Tu	We	Th	Fr	Sa
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

Legend:

□: Days when only standard job A executes

●: Days when both standard job A and nested jobnet B execute

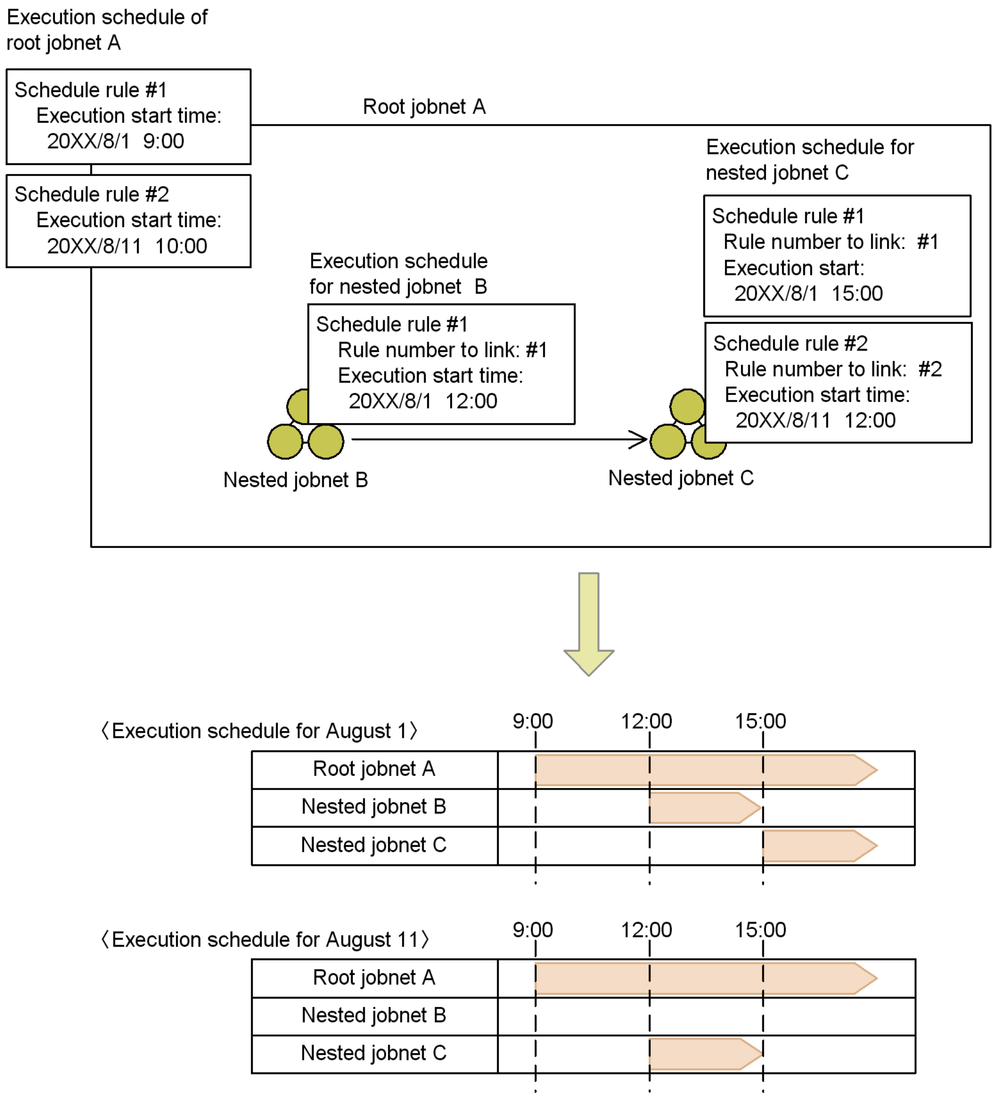
In this example, the nested jobnet B is executed only on Fridays. This means that nested jobnet B is not executed on Monday through Thursday, regardless of whether jobnet A is executed.

(2) Linking to the schedule rules of the root jobnet

You define schedule rules for a nested jobnet by linking them with the schedule rules of the root jobnet. When a schedule rule of the root jobnet comes into effect, the linked schedules of the nested jobnet also come into effect.

The following is an example of linking schedule rules.

Figure 3-6: Linking schedule rules of nested jobnets and root jobnets



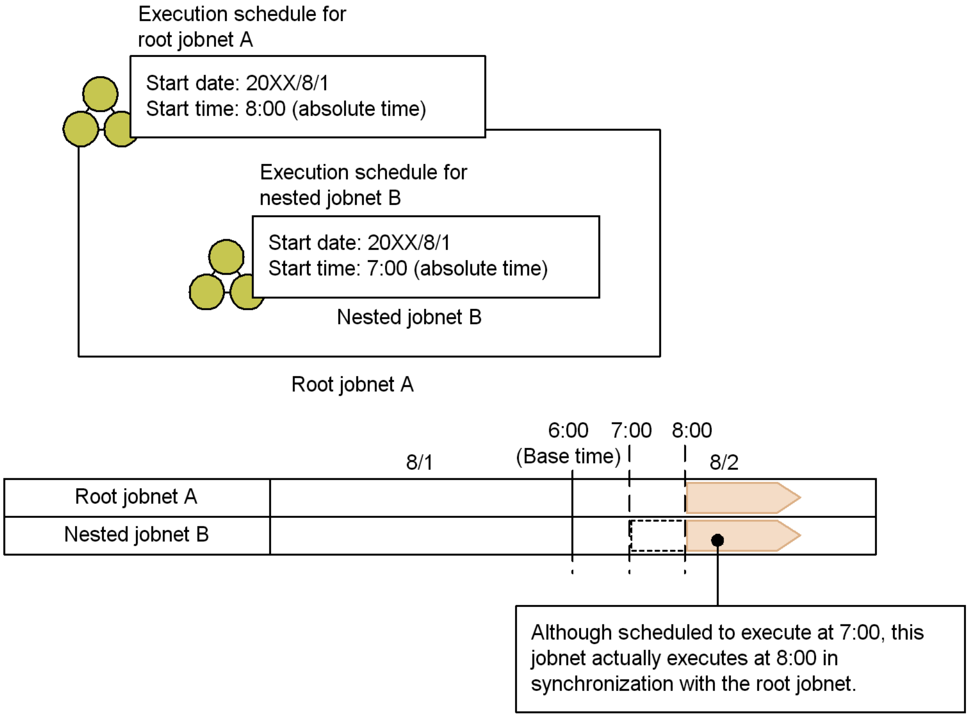
In this example, the schedule rule for nested jobnet B is not linked to schedule rule #2 of the root jobnet. This means that on August 1, when the root jobnet is executed according to schedule #2, the nested jobnet B is not executed.

Take note of the following when you define a schedule for a nested jobnet:

- A nested jobnet is not executed if there is no point at which the schedules of the upper-level jobnet and the nested jobnet overlap. If you copy a nested jobnet from another location, check the upper-level schedule before you define the schedule.

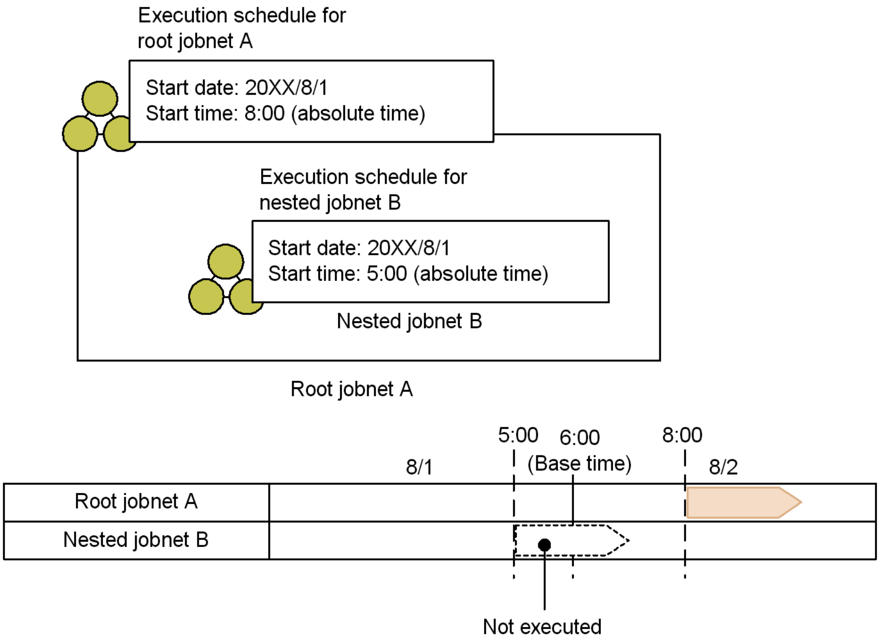
- For a nested jobnet, you can define a schedule that extends over two days. In such a case, specify an execution start time for the nested jobnet between 24:00 and 47:59. For example, assume that the execution start time of the root jobnet is 20XX/8/1 23:00, and you want the execution start time of the nested jobnet to be 2 a.m., on the next day. In this case, specify 20XX/8/1 26:00 as the execution start time. This gives the root jobnet and the nested jobnet the same execution start day. If you specify 20XX/8/2 2:00 as the execution start time for the nested jobnet, the nested jobnet will not execute since the execution start dates for the root jobnet and the nested jobnet are different.
- If the start time of the nested jobnet is earlier than that of the root jobnet, and the start time is earlier than the base time, the root jobnet and the nested jobnet will be scheduled to execute on different days. As a result, the nested jobnet fails to execute. Two examples are shown below, one where the nested jobnet executes and one where it does not.

Figure 3-7: When the nested jobnet start time precedes the root jobnet start time (example where the nested jobnet executes)


Execution schedule for root jobnet A
Start date: 20XX/8/1
Start time: 8:00 (absolute time)
Execution schedule for nested jobnet B
Start date: 20XX/8/1
Start time: 7:00 (absolute time)
Nested jobnet B
Root jobnet A
6:00
(Base time)
7:00
8:00
8/1
8/2
Root jobnet A
Nested jobnet B
Although scheduled to execute at 7:00, this jobnet actually executes at 8:00 in synchronization with the root jobnet.

In this example, the nested jobnet's start time (7:00) comes after the base time (6:00). This gives the root jobnet and the nested jobnet the same execution start day, so the nested jobnet will execute.

Figure 3-8: When the nested jobnet start time precedes the root jobnet start time (example where the nested jobnet does not execute)

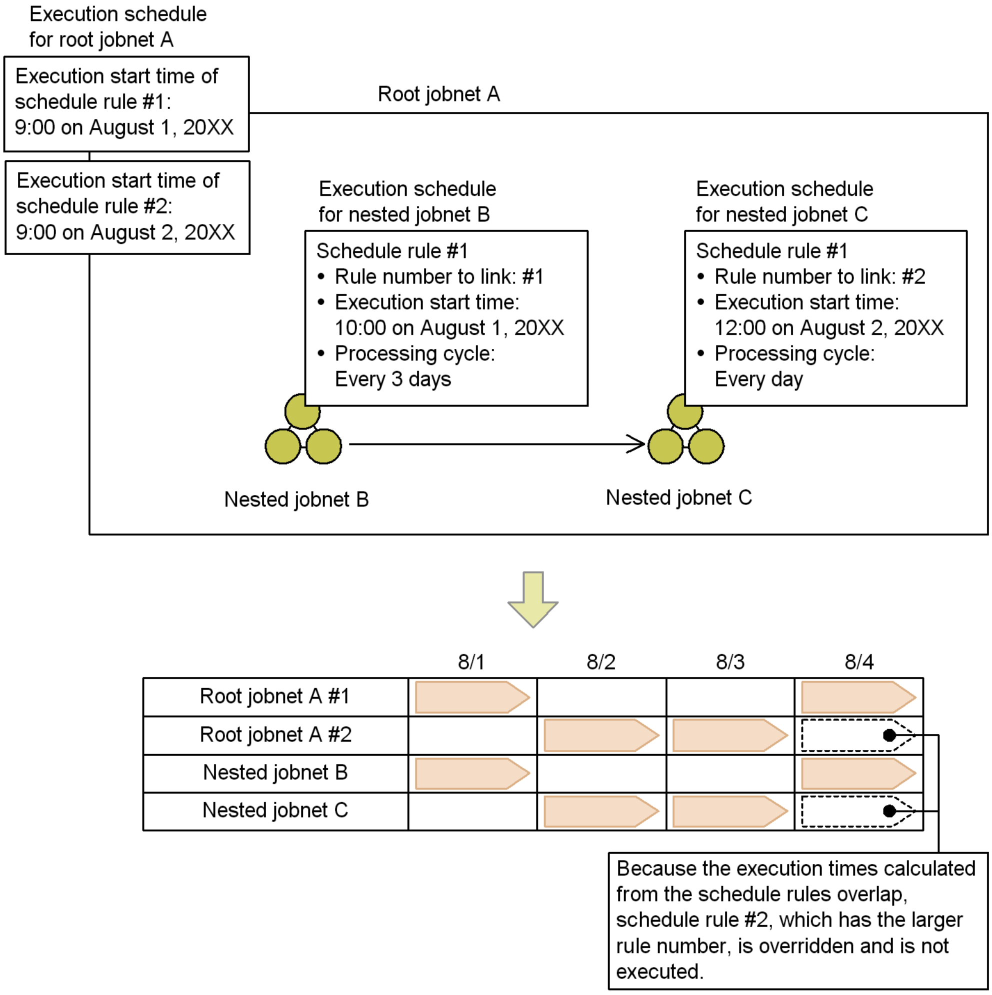

Execution schedule for root jobnet A
Start date: 20XX/8/1
Start time: 8:00 (absolute time)
Execution schedule for nested jobnet B
Start date: 20XX/8/1
Start time: 5:00 (absolute time)
Nested jobnet B
Root jobnet A
5:00
6:00
(Base time)
8:00
8/1
8/2
Root jobnet A
Nested jobnet B
Not executed

In this example, the nested jobnet's start time (5:00) precedes the base time (6:00). The nested jobnet will not execute because it has a different execution start date from the root jobnet.

- When multiple schedule rules have been defined for a root jobnet, schedules whose execution times overlap might be created, depending on the schedule rules. For the schedules whose execution times overlap, the schedule that has the smallest rule number overrides other schedules. If overridden schedule rules have been linked to nested jobnets, the schedules of the nested jobnets are also overridden.

  The following figure shows an example of scheduling when execution times calculated from schedule rules overlap.

Figure 3-9: Example of scheduling when execution times calculated from schedule rules overlap


Execution schedule for root jobnet A
Execution start time of schedule rule #1: 9:00 on August 1, 20XX
Execution start time of schedule rule #2: 9:00 on August 2, 20XX
Root jobnet A
Execution schedule for nested jobnet B
Schedule rule #1
• Rule number to link: #1
• Execution start time: 10:00 on August 1, 20XX
• Processing cycle: Every 3 days
Execution schedule for nested jobnet C
Schedule rule #1
• Rule number to link: #2
• Execution start time: 12:00 on August 2, 20XX
• Processing cycle: Every day
Nested jobnet B
Nested jobnet C
8/1
8/2
8/3
8/4
Root jobnet A #1
Root jobnet A #2
Nested jobnet B
Nested jobnet C
Because the execution times calculated from the schedule rules overlap, schedule rule #2, which has the larger rule number, is overridden and is not executed.

In this example, execution times of root jobnet A on August 4 calculated from schedule rules #1 and #2 overlap. In this case, schedule rule #1, which is the schedule rule with the smaller rule number, overrides schedule rule #2. Nested jobnet B linked to schedule rule #1 is executed, but nested jobnet C linked to overridden schedule rule #2 is not executed on August 4.

3.5.4 Executing the same jobnet several times a day (defining cycle jobs)

If you want to execute the same jobnet more than once and at regular intervals during the course of a day, define the jobnet using one of the following three methods:

- Method 1: Create a single jobnet, and then define multiple schedule rules.
- Method 2: Create a new jobnet for each start time.
- Method 3: Specify an interval control setting as a start condition for the jobnet.

(1) Methods for defining the jobnet

Each of these methods is described in the following paragraphs. Each description uses, as an example, the creation of a jobnet that executes 24 times a day in one-hour intervals, between 7 a.m. on the current day and 6 a.m. the following day.

Method 1: Create multiple schedule rules

1. Create a single jobnet, and define the jobs to be executed.
2. Create 24 individual schedule rules with staggered start times of 07:00, 08:00, 09:00 and so on.

When you register the jobnet for execution, the jobnet is executed every hour according to the start times specified in the schedule rules.

Method 2: Create a new jobnet for each start time

1. Create a single jobnet.
2. Create a nested jobnet, and define the jobs to be executed.
3. Copy the nested jobnet you created in step 2, and create a total of 24 identical nested jobnets.
4. Set a different start time between 07:00 and 06:00 for each nested jobnet.

When you register the jobnet for execution, each nested jobnet is executed according to its respective start time.

Method 3: Specify an interval control setting as a start condition for the jobnet

1. Create a single jobnet, and define the jobs to be executed.
2. Create a start condition, and attach an execution interval control job.
3. Set the **Waiting time** of the execution interval control job to 60 minutes.
4. Create a schedule rule, setting the execution start time to 6:00, and the valid range of the start condition to 24 times.

The execution interval control job is activated at the execution start time. It then waits until the specified waiting time has passed before executing the jobnet. When you define the jobnet, set the start time to a time that is earlier than the time when you first want the jobnet to execute, by a length of time equivalent to the waiting time.

When you register this jobnet for execution, the jobnet begins to monitor the start condition at 6:00. 60 minutes later, at 7:00, the start condition is met and the jobnet is executed for the first time. The jobnet is then executed every 60 minutes, according to the start condition.

(2) Advantages and disadvantages of each method

The following table explains the advantages and disadvantages of the three methods.

Table 3-4: Advantages and disadvantages of each method of executing the same jobnet periodically

Method	Advantages	Disadvantages
Method 1	• Defining jobnets is easy. • Jobnets have a simple configuration. • When you define a jobnet as a root jobnet, you can enable concurrent execution or schedule skip.	• Defining schedules is time consuming. • To change the schedule, you must change all the schedule rules.
Method 2	• You can apply this method to any type of jobnet. • You can check which instance of the jobnet is currently executing using the jobnet monitor. • You can rerun a job or cancel execution of a job from within the jobnet monitor.	• You need to create one nested jobnet each time you want to execute the jobnet. • You need to define a schedule for every nested jobnet. • To change a schedule rule, you must change the schedule rules for every nested jobnet. • Changing the process that is executed is time consuming.
Method 3	• Jobnets have a simple configuration.	• You can only use this method with a root jobnet.

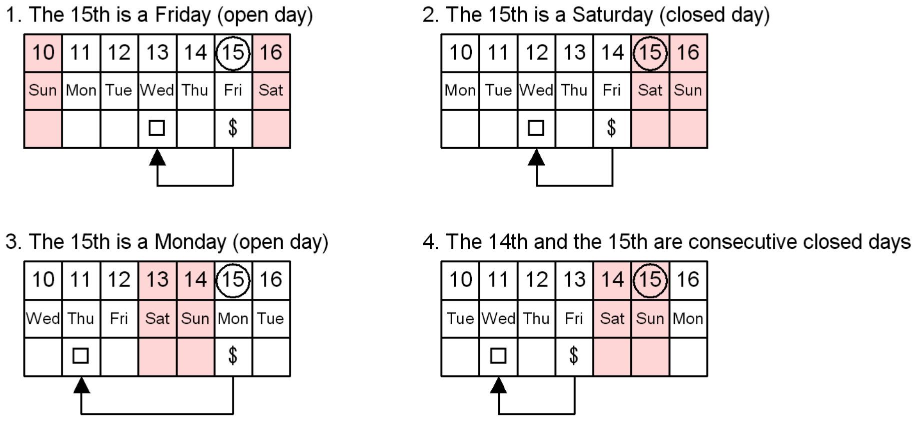
3.5.5 Shifting the scheduled execution date forward or back based on a calculated schedule (Schedule by days from start)

This subsection describes the settings required to calculate a date that acts as a base day from a calendar or schedule, and then execute a process on the previous day or the next day. You define these settings using the schedule by days from start function. For details about the schedule by days from start function, see *3.3.2 Defining a schedule* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.

For example, imagine you pay salaries on the 15th, and want to execute a jobnet that carries out the salary calculation process two open days before this date.

In this case, if the 15th is a closed day, then salaries are paid on the first open day before the 15th. The jobnet that carries out the salary calculation process needs to complete processing before the payment day arrives. For this example, the scheduled execution dates are as follows.

Figure 3-10: Examples of scheduled execution dates


1. The 15th is a Friday (open day)
10 11 12 13 14 15 16
Sun Mon Tue Wed Thu Fri Sat
□ $
2. The 15th is a Saturday (closed day)
10 11 12 13 14 15 16
Mon Tue Wed Thu Fri Sat Sun
□ $
3. The 15th is a Monday (open day)
10 11 12 13 14 15 16
Wed Thu Fri Sat Sun Mon Tue
□ $
4. The 14th and the 15th are consecutive closed days
10 11 12 13 14 15 16
Tue Wed Thu Fri Sat Sun Mon
□ $

Legend:
$: Salary payment date
□: Scheduled execution date for salary calculation process

Sometimes, the circumstances of the calendar for a particular month may cause the salary payment date itself, which acts as the base day, to move. Accordingly, you cannot accurately determine the execution date for the calculation process simply by using the substitute schedule for closed days. For example, assume that you set the scheduled execution date to the 13th, two days before the 15th (salary payment day). In this case, if both the 14th and 15th are closed days (case 4 in the above figure), then both the salary payment day and the salary calculation day will fall on the same day.

If you use schedule by days from start, you can calculate the salary payment date based on factors including the calendar settings. You then calculate the execution date for the calculation process based on the payment date. By following these steps, you can calculate an execution schedule that places the calculation process two open days prior to the actual salary payment date.

To set the schedule shown in the figure by using the schedule by days from start function, define the schedule rule as follows:

Definition

Execution start date: 15th (absolute day)

Processing cycle: 1 month

Substitute schedule in case of a closed day job: Execute on previous open day

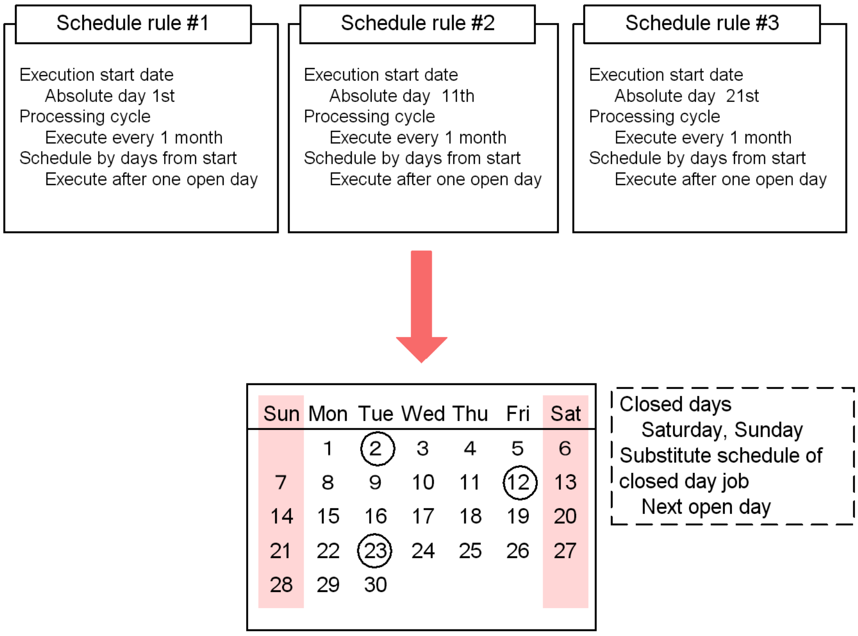
Schedule by days from start: Execute 2 open days before start date

When you register this definition for execution, first the start date (salary payment date) is calculated based on the settings you specified for the execution start date, the processing cycle, and the substitute schedule in case it falls on a closed day. Accordingly, the actual scheduled execution date is set to a date two open days before the start date.

3.5.6 Defining a schedule that has different behavior at different times of the month

This subsection describes how to define a schedule that causes a jobnet to execute in the first ten-day period, the second ten-day period, and the last ten-day period in a month. You can define this type of schedule by creating schedule rules that define execution start dates in the first ten-day period, second ten-day period and last ten-day period of the month. For example, to execute a process on the last open day of each ten-day period in the month, specify the 10th, the 20th and the final day of the month as the execution start dates. Use the schedule by days from start function if you want the execution dates to shift from these dates when required. The following example illustrates the use of schedule by days from start to execute a process once every ten-days.

Figure 3-11: Example of a schedule that executes a process every ten-days



This schedule is structured so that the process is executed on the open day that follows the first open day in each ten-day period in the month. The first day in the last ten-day period is the 21st. However, because the 21st is Sunday and therefore a closed day, the execution date shifts to the next business day, in this case the 22nd. The **Schedule by**

days from start setting then shifts the execution date back by one day. As a result, the scheduled execution date is calculated as the 23rd.

3.5.7 Defining a different calendar for each application

Sometimes a certain application or a particular department will have different open days. In such a case, you can define different calendars for different applications.

As an example, consider a case in which the Tokyo head office, the Osaka branch office, and the Nagoya branch office all execute a certain application, but run the application on different days. The respective application calendars are as follows.

- Tokyo head office: Execute daily
- Osaka branch office: Execute Monday through Saturday
- Nagoya branch office: Execute Monday through Friday

There are two ways of structuring the application.

- Group all jobnets with the same application calendar in job groups.
- Create the application within one job group, and refer to calendars of other job groups.

These methods are explained in detail below.

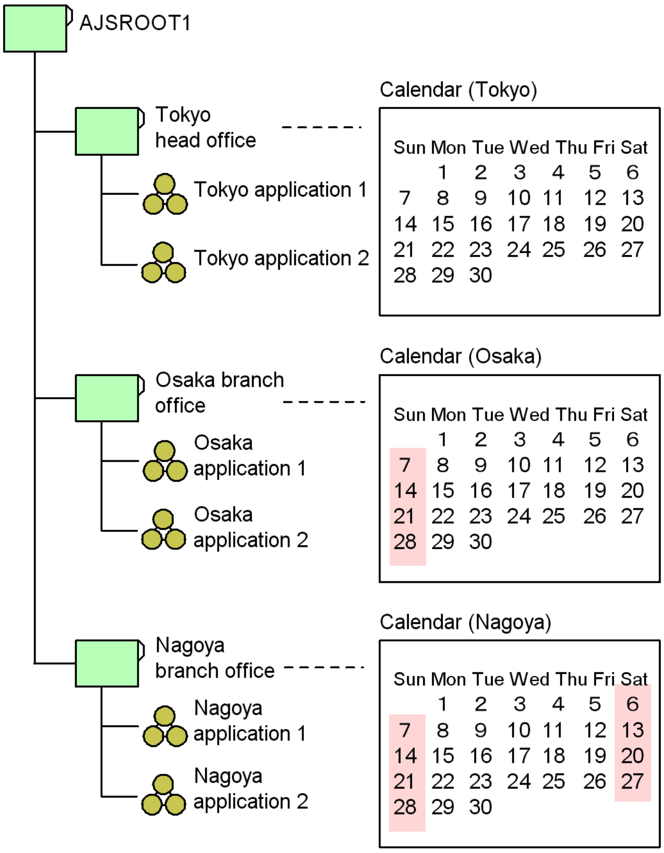
(1) Grouping jobnets with the same application calendar in job groups

This method is best used for applications that meet the following conditions:

- There are multiple application groups, but no dependent relationships between the application groups.
- There is a system administrator responsible for each application group.
- Multiple application groups are managed by one manager.

The following figure shows an example of the application configuration.

Figure 3-12: Grouping jobnets with the same application calendar in job groups



An overview of the method used to set the schedule is given below.

1. Create job groups for Tokyo, Osaka, and Nagoya under the scheduler service.
2. Define jobnets under each job group.
3. Define a calendar for each job group.
4. Register the jobnets for execution.

With this method, you can only display one job group at a time in the monitor window. To monitor jobnets in a single window, use the functions provided by JP1/AJS3 Console. The JP1/AJS3 Console functions enable you to display and monitor jobnets from different job groups in a single window.

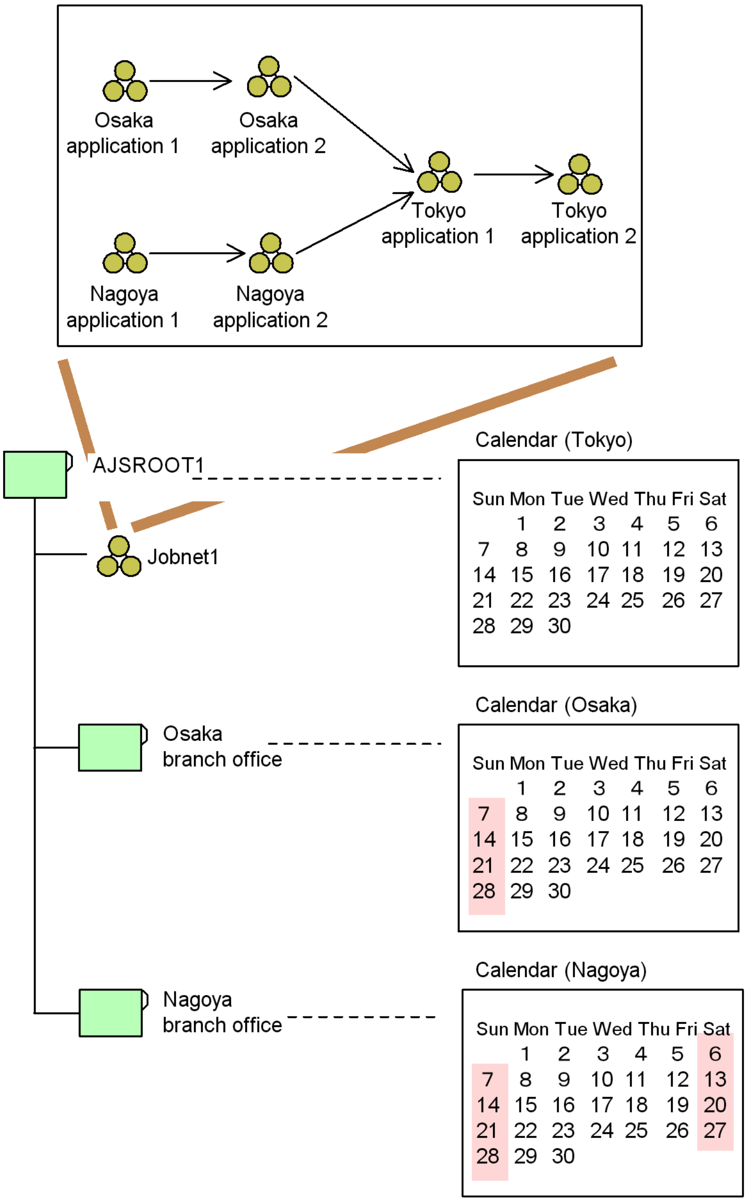
(2) Creating the application within one job group, and referencing calendars defined for other job groups

This method is best used for applications that meet the following conditions:

- There are multiple application groups, and dependent relationships between the application groups.
- One system administrator manages all applications.

The following figure shows an example of the application configuration.

Figure 3-13: Referring to a calendar of another job group



Calendar (Tokyo)

Sun	Mon	Tue	Wed	Thu	Fri	Sat
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

Calendar (Osaka)

Sun	Mon	Tue	Wed	Thu	Fri	Sat
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

Calendar (Nagoya)

Sun	Mon	Tue	Wed	Thu	Fri	Sat
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

An overview of the method used to define the schedule is given below.

1. Create a single jobnet (jobnet 1) under the scheduler service.
2. Define the applications to be executed at Tokyo, Osaka, and Nagoya as nested

jobnets within the root jobnet.

3. Specify the calendar of the Tokyo head office in the scheduler service.
4. Create job groups for Osaka and Nagoya.
5. Define calendars for the Osaka and Nagoya job groups.

   Define only calendars for these job groups; do not create jobnets for them.
6. Set the nested jobnets you want to execute at each branch to refer to the calendar defined for the respective job group.

   In the Schedule Settings dialog box, select **Refer to a calendar of another job group**. Set Osaka application 1 and Osaka application 2 to refer to the calendar of the Osaka branch job group. Set Nagoya application 1 and Nagoya application 2 to refer to the calendar of the Nagoya branch job group.
7. Register the root jobnet for execution.

With this method, you can monitor all of the jobnets in a single monitor window.

(3) Examples of calendar work tasks

This subsection presents some examples that show how calendars are applied.

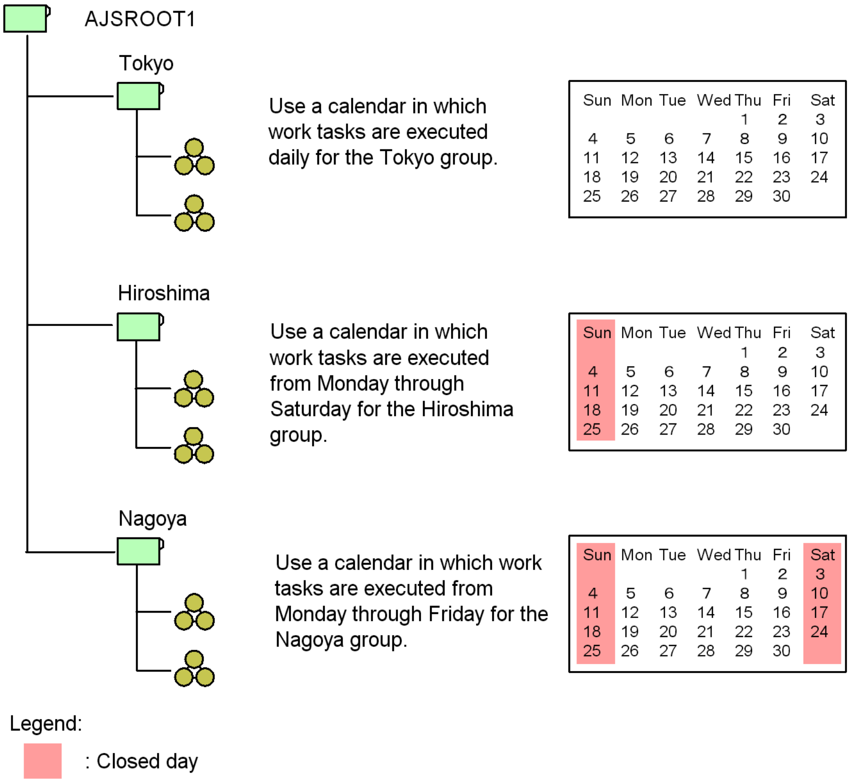
Example 1

In the first example we will explain how to set a calendar premised on the following conditions:

- There are multiple work task groups, but there is no interdependency among the information in the individual calendars.
- Each work task group is controlled by its own system administrator.
- The multiple work task groups are managed by a single work task manager host.

Based on these premises, we will create a hierarchy for the work task group and define calendars, as shown in the following figure.

Figure 3-14: Calendar work task example 1



Use a calendar in which work tasks are executed daily for the Tokyo group.

Sun	Mon	Tue	Wed	Thu	Fri	Sat
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

Use a calendar in which work tasks are executed from Monday through Saturday for the Hiroshima group.

Sun 🟥	Mon	Tue	Wed	Thu	Fri	Sat
				1	2	3
4 🟥	5	6	7	8	9	10
11 🟥	12	13	14	15	16	17
18 🟥	19	20	21	22	23	24
25 🟥	26	27	28	29	30	

Use a calendar in which work tasks are executed from Monday through Friday for the Nagoya group.

Sun 🟥	Mon	Tue	Wed	Thu	Fri	Sat 🟥
				1	2	3 🟥
4 🟥	5	6	7	8	9	10 🟥
11 🟥	12	13	14	15	16	17 🟥
18 🟥	19	20	21	22	23	24 🟥
25 🟥	26	27	28	29	30	

When working from the monitor screen with an example like this, you can only check the execution schedule and results of work tasks under a single work task group (for example the *Tokyo* group) at one time. In a situation where multiple system administrators are managing individual work task groups, this arrangement makes it easy for the responsible system administrators to check the schedules and results for the work tasks they are concerned with.

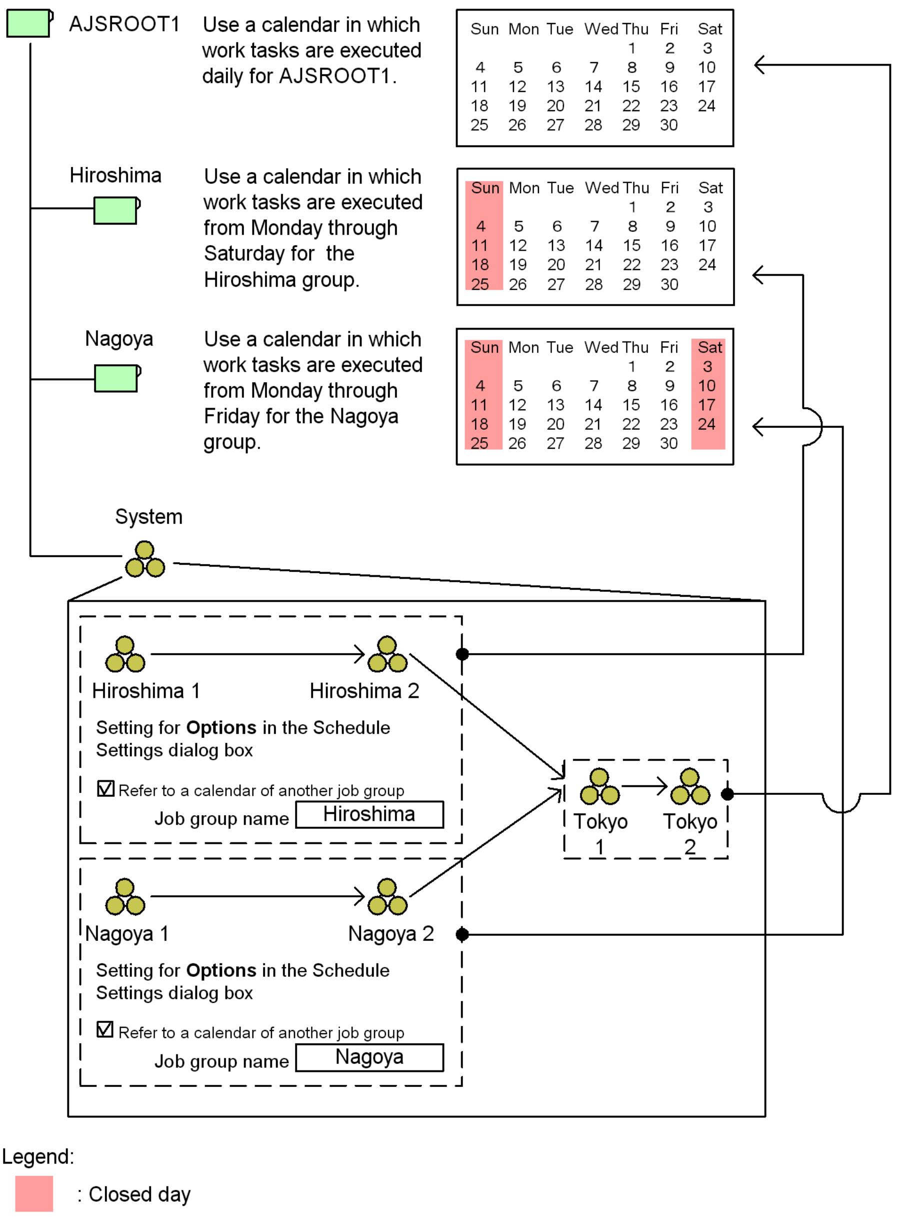
Example 2

We will explain the setting of a calendar premised on the following conditions.

- There are multiple work task groups and the information in their calendars is interdependent.
- All of the work tasks are managed by a single system administrator.

Taking these conditions as the premises, we will create a hierarchy for the work task group and define a calendar, as shown in the following figure.

Figure 3-15: Calendar work task example 2


AJSROOT1
Use a calendar in which work tasks are executed daily for AJSROOT1.
Hiroshima
Use a calendar in which work tasks are executed from Monday through Saturday for the Hiroshima group.
Nagoya
Use a calendar in which work tasks are executed from Monday through Friday for the Nagoya group.
Sun Mon Tue Wed Thu Fri Sat
System
Hiroshima 1
Hiroshima 2
Setting for Options in the Schedule Settings dialog box
☑ Refer to a calendar of another job group
Job group name Hiroshima
Nagoya 1
Nagoya 2
Setting for Options in the Schedule Settings dialog box
☑ Refer to a calendar of another job group
Job group name Nagoya
Tokyo 1
Tokyo 2
Legend:
: Closed day

When you make the settings in the figure above, the individual work tasks are executed as follows.

Tokyo work tasks

The work tasks reference a calendar for daily execution so they are executed every day.

Hiroshima work tasks

The work tasks reference a calendar in which only Sunday is a closed day, so they are executed from Monday through Saturday.

Nagoya work tasks

These work tasks reference a calendar in which Saturdays and Sundays are closed days, so they are executed from Monday through Friday.

In this example, if you check the group AJSROOT1 on the monitor screen, you can check the schedules and results of all the work tasks under this group. This makes it easy for a single system administrator to manage all the work tasks.

Chapter

4. Execution Registration Method Considerations

When a jobnet with a schedule rule is registered for execution, the jobnet is added to the schedule in JP1/AJS3.

This chapter describes the methods you can use to register jobnets for execution in JP1/AJS3.

4.1 Jobnet execution registration methods

In JP1/AJS3, you can use the following three methods to register jobnets for execution:

- Planned execution registration
- Fixed execution registration
- Immediate execution registration

For details about each mode, see *4.1.1 Methods of registering a jobnet for execution* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.

Take into account the characteristics of each method, and choose the one that best suits the way you use the system.

4.1.1 Planned execution registration

A jobnet registered for planned execution is scheduled and executed according to the calendar information and schedule rules set for the jobnet. The calculated schedule is treated as a dummy schedule (simulated execution schedule).

(1) Characteristics of planned execution registration

The characteristics of planned execution registration are as follows:

(a) Execution timing

The jobnet is scheduled and executed according to the calendar information and the schedule rule set for the jobnet. The schedule for the next execution is finalized when execution of the jobnet has ended.

(b) Temporarily rescheduling the jobnet

You can make a temporary change only to the next scheduled execution of a jobnet registered for planned execution. Further rescheduling is not possible because the schedule is simulated.

(c) Modifying the calendar information or schedule rules

If you change the calendar information or redefine the jobnet's schedule rule, JP1/AJS3 reschedules the jobnet based on the new information. This allows you to modify calendar information and schedule rule without canceling registration of the jobnet, such as when open or closed days need to be redefined for the next business year, or when you change a schedule rule.

(2) Recommended uses

We recommend use of planned execution registration in situations like the following:

- When changes to calendar information or schedule rule might be necessary in the future
- When you want to execute the jobnet a specific number of times per day in response to a trigger such as a file being updated or a specific action by the user (this requires a start condition to be set)
- With jobnets that never need to be temporarily rescheduled

4.1.2 Fixed execution registration

A jobnet registered for fixed execution is scheduled and executed according to the calendar information and the schedule rule set for the jobnet. The schedule based on the specified period or generation count is calculated and fixed as soon as the jobnet is registered for execution.

(1) Characteristics of fixed execution registration

The characteristics of fixed execution registration are as follows:

(a) Execution timing

The jobnet is scheduled and executed according to the calendar information and the schedule rule set for the jobnet.

(b) Temporarily rescheduling the jobnet

Temporary changes can be made to a fixed schedule scheduled for a specified period or times. You can also add execution dates.

(c) Modifying the calendar information or schedule rules

Changes to the calendar information or jobnet's schedule rule do not take effect until the jobnet has run for the specified period or number of generations. To apply changes made to calendar information or schedule rules to the schedule, you must cancel and then re-register the jobnet.

(2) Recommended uses

We recommend use of fixed execution registration in situations like the following:

- With jobnets that run for a set period or for a set times
- With jobnets that have a fixed schedule that may require a temporary change, addition, or prohibit execution

4.1.3 Immediate execution registration

A jobnet registered for immediate execution is executed once only, as soon as it is registered, regardless of the calendar information and the schedule rule set for the jobnet.

(1) Characteristics of immediate execution registration

The characteristics of immediate execution registration are as follows:

(a) Execution timing

The jobnet is executed immediately upon registration. Any schedule rules set for the jobnet are ignored.

(b) Temporarily rescheduling the jobnet

You cannot temporarily reschedule a jobnet registered for immediate execution, because the jobnet has no execution schedule.

(c) Modifying the calendar information or schedule rules

Not applicable because the calendar information and jobnet schedule rule are not referenced.

(2) Recommended uses

We recommend use of immediate execution registration in situations like the following:

- With jobnets that are started manually or by command
- With jobnets executed from within an application
- With jobnets whose execution is triggered by a command initiated when a file transfer application program (e.g. JP1/FTP) ends normally
- With jobnets whose execution is triggered by a JP1/IM automated action
- When running a consistency test on the processing flow for jobnets with defined schedules

Chapter

5. Monitoring Method Considerations

This chapter describes matters to consider when you use JP1/AJS - View and JP1/AJS3 Console to monitor work tasks.

5.1 Monitoring methods in JP1/AJS3 - View

This section describes the considerations necessary for using JP1/AJS3 - View to monitor work tasks.

For details about how to plan the JP1/AJS3 - View environment, see *4.5 Environment settings for JP1/AJS3 - View* in the *Job Management Partner 1/Automatic Job Management System 3 System Design (Configuration) Guide*.

(1) Planning which work tasks to monitor in the Summary Monitor window

By designating selected jobnets as monitoring targets in the JP1/AJS3 - View window (Summary Monitor window), you can monitor the execution status of the jobnets, including the execution progress and the statuses of subordinate jobs. You can also view a simulation of the jobnet's end time, calculated from the jobnet's execution times in the past. This can be useful when any delay in starting or ending the work task would have a major effect on the execution schedule for subsequent work tasks.

JP1/AJS3 - View also allows you to centrally monitor jobnets that are distributed over different jobnet hierarchies or are governed by different scheduler services.

(2) Jobnet delay monitoring considerations

JP1/AJS3 provides the following methods for detecting delays in jobnet execution.

(a) Monitoring by date and time specification

With this method, you specify a date and time, and JP1/AJS3 monitors whether a jobnet starts and ends at the scheduled time. JP1/AJS3 provides two methods of monitoring for delays by date and time specification: *Monitor start delay* and *Monitor end delay*.

Use these methods when a delay in starting or ending the jobnet will have a major effect on the execution schedule for subsequent jobnets.

Monitor start delay

> Set a date and time for judging when a start is delayed. If the jobnet starts within a certain time after the scheduled start time, it is deemed normal. If the jobnet has still not executed when the grace period elapses, some abnormality is assumed to have occurred.

Monitor end delay

> Set a time for judging when completion is delayed. If the jobnet ends within a certain time after the scheduled start time, it is deemed normal. If jobnet execution has still not ended when the grace period elapses, some abnormality is assumed to have occurred.

You can designate both a start delay and an end delay, or just one or the other.

(b) Monitoring by time-required-for-execution

You can also monitor for delays based on the time required for execution of the jobnet.

If the time set is exceeded, the color of the work task icon changes, and an event is issued to report the delay.

(3) End delay monitoring based on time-required-for-execution

You can also monitor for end delays based on the time-required-for-execution of a job. In this case, JP1/AJS3 monitors how long the job takes to execute on the manager host, which may differ from how long it takes on the agent host. This discrepancy can affect whether an end delay is detected.

Timeout monitoring, on the other hand, is based on how long the job is active on the agent host. For this reason, there is a time difference between timeout detection and the detection of an end delay.

The following are examples of situations where a job has a different time-required-for-execution on the manager host and agent host.

- The job is started on the agent host while the scheduler service is stopped.

  If a job begins executing on the agent host while the scheduler service is stopped (that is, the scheduler service stops before the manager host receives notification from the agent host that the job has started), the job does not enter *Now running* status until the scheduler service restarts. Here, the job will have a longer execution time on the agent host than on the manager host. For example, suppose the job's time-required-for-execution is 10 minutes, and it has been executing for 20 minutes on the agent host. In this case, if the job finishes five minutes after the scheduler service restarts, an end delay will not be detected.

- The job ends on the agent host while the scheduler service is stopped.

  If a job that was already running on the agent host ends while the scheduler service is stopped (that is, the scheduler service stops before the manager host receives notification from the agent host that the job has ended), the job does not acquire an end status until the scheduler service restarts. Here, the job will have a shorter execution time on the agent host than on the manager host. For example, suppose the job's time-required-for-execution is 10 minutes, and it ends on the agent host after five minutes. Because the job is considered to be running the entire time the scheduler service was stopped, an end delay will be detected if the scheduler service is stopped for longer than 10 minutes.

- The job starts and ends on the agent host while the scheduler service is stopped.

  If a job starts and finishes on the agent host while the scheduler service is stopped (that is, the scheduler service stops before the manager host receives notification from the agent host that the job has started and ended), the job enters *Now running* status then ends as soon as the scheduler service restarts. Here, the job execution time on the manager host will be zero minutes. For example, suppose the job's time-required-for-execution is 10 minutes, and it took 20 minutes to execute on the agent host. In this case, an end delay will not be detected because the job's execution time on the manager host is zero minutes.

Supplementary notes

- When an end delay is detected for a job, you can check the job's execution time on the agent host by using the `ajsshow` command. Execute the command with the format specifiers `%V` and `%Q` to acquire the execution start and end times of the job. For the command syntax, see *ajsshow* in *2. Commands* in the manual *Job Management Partner 1/Automatic Job Management System 3 Command Reference 1*.
- If you specify `yes` in the `JOBDELAYWARNMSG` environment setting parameter of the scheduler service, the message KAVS0249-W `The scheduler services stopped before execution of the job began.` is output to the integrated trace log when a job being monitored for an end delay starts executing while the scheduler service is stopped. You can identify the job name and execution ID from the content of the message.

  To verify an end delay based on the time-required-for-execution of the job on the agent host, use the `ajsshow` command to acquire the execution start and end times of the job identified in the message KAVS0249-W.

  For information about the environment setting parameter `JOBDELAYWARNMSG`, see *2.2 Setting up the scheduler service environment* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 2*.

5.2 Considerations when centrally monitoring work tasks in JP1/AJS3 Console

In JP1/AJS3 Console, you can centrally monitor, from a single screen, the status of work tasks that are managed by multiple managers. To use JP1/AJS3 Console to centrally monitor work tasks, determine in advance the work tasks to be monitored and the monitoring method. For details about the features of JP1/AJS3 Console, see *7. Monitoring Applications Using JP1/AJS3 Console* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.

The following table lists and explains the considerations involved when work tasks are centrally monitored in JP1/AJS3 Console.

Table 5-1: Considerations for centralized monitoring of work tasks

Item	Description
Monitored objects	Determine which work tasks (objects) are to be monitored.
Hierarchization (grouping) of monitored objects	Determine the hierarchy levels in which the objects will be monitored.
Monitoring method Monitoring interval	Determine how work tasks (objects) are to be monitored. • The monitoring method from **Prioritize current time**, **Prioritize all unit times**, **Prioritize current time schedules**, and **Prioritize all unit time schedules**. • Determine the interval at which to refresh the status of the monitored object. • When a hold attribute is set for a monitored work task, decide whether this status (**Hold plan**) is indicated with the corresponding display color.

(1) Setting monitored objects

Use the AJS3 unit monitored object icon to define the type of work task (object) to be monitored. When creating the AJS3 unit monitored object, specify information including the name of the work task manager host that will do the monitoring, and the name of the work task to be monitored. For details about defining the AJS3 unit monitor object, see *13.4.2 Defining an AJS3 unit monitored object* in the *Job Management Partner 1/Automatic Job Management System 3 Operator's Guide*.

We recommend that you place the work task manager host that will do the monitoring in the same user authentication bloc as JP1/AJS3 Console (JP1/AJS3 Console Manager). A user authentication bloc means a group of hosts that use the same authentication server as the JP1 user. For details, see the *Job Management Partner 1/Base User's Guide*. If the user authentication blocs are the same, no login operation is required to open the JP1/AJS3 - View window from the AJS3 unit monitored object. If the user authentication blocs are different, a login operation is required.

(2) Hierarchization of monitored objects

Use the *nested business scope* icon to define the type of hierarchy in which the object is monitored. Place work tasks that are related in terms of execution order or type (meaning) in the same group, and monitor them as a single work task. For details about defining nested business scopes, see *13.4.1 Defining business scope* in the *Job Management Partner 1/Automatic Job Management System 3 Operator's Guide*.

(3) Determining the monitoring method and monitoring interval

Define the way that you monitor a work task (object) in the Monitoring Properties dialog box, as follows:

- Choose one monitoring method from among **Prioritize current time**, **Prioritize all unit times**, **Prioritize current time schedules**, and **Prioritize all unit time schedules** (Monitoring method).
- Enter the interval at which the monitoring status is to be refreshed (Monitoring interval).
- If a hold attribute is set for the monitored object, decide whether it is to be shown in a color that indicates the *Hold plan* status (Monitoring method).
- For details about the monitoring properties, see *13.4.4 Defining monitoring properties* in the *Job Management Partner 1/Automatic Job Management System 3 Operator's Guide*.

Chapter

6. Access Permission Considerations

This chapter describes matters you must consider when setting the access permissions for jobnets and for associating JP1 users and OS users.

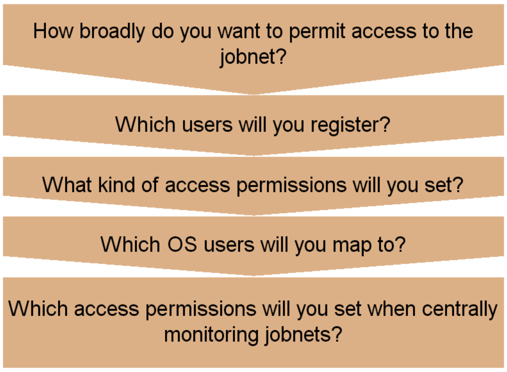
6.1 Steps for considering work task access permissions

This section explains the considerations involved when setting work task access permissions for individual users. By granting JP1/AJS3 users the permission to execute processing only, or the permission to view processing results only, you can prevent erroneous operations.

The figure below shows the steps when considering work task access permissions.

Use the user management functionality of JP1/Base (user authentication, user mapping), to set and manage work task access permissions. For details on the user management functionality, see the *Job Management Partner 1/Base User's Guide*.

Figure 6-1: Steps for considering work task access permissions


How broadly do you want to permit access to the jobnet?
Which users will you register?
What kind of access permissions will you set?
Which OS users will you map to?
Which access permissions will you set when centrally monitoring jobnets?

6.2 Ranges for setting access permissions

JP1/AJS3 manages access permissions, such as the permission to execute processing and the permission to change processing details, based on the unique user names of the JP1 users.

Use the user authentication functionality provided by JP1/Base to register and manage the JP1 users. For the purposes of the user authentication functionality, a JP1/Base host that manages the access permissions of the JP1 users is called an *authentication server*. Once you have specified a JP1/Base host among those on the network that is to be the authentication server, you can decide the range of hosts where that authentication server manages the access permissions (the user authentication bloc). You must set user authentication blocs when you introduce JP1/AJS3.

Consult the following table when considering how to set user authentication blocs.

Table 6-1: Number of authentication blocs and merits/demerits

Number of authentication blocs	Merits and demerits
Only one user authentication bloc is set in the system	The system administrator can manage all the JP1 users centrally, and not much time and effort are taken up with registration and changes.
Multiple authentication blocs are set in the system	The system administrator has to manage a number of JP1 users equivalent to the number of authentication blocs set. Some time and effort is expended managing JP1 user registration, changes, and so on, and login authentications are conducted at each of the authentication servers. However, since the individual authentication servers are independent, the resilience of the system as a whole is good.

You can set two authentication servers within a single user authentication bloc. The authentication server in normal use is called the *primary authentication server*, while another that serves as a backup and is used in the event of trouble is called the *secondary authentication server*. If you set only one authentication server in a user authentication bloc, there is the risk that if it is not possible to make contact with the authentication server for any reason - for example because the authentication server will not start or a communication fault has occurred - it will not be possible to execute jobs or remote commands, and work tasks will stop.

If you set two authentication servers, even if trouble occurs at the one used under normal circumstances (the primary authentication server), you can still execute jobs and remote commands at the backup authentication server (secondary authentication server). Even if there is only one authentication bloc in the system, setting two authentication servers will increase the resilience of the system. Use two authentication servers if you consider it necessary.

If you set primary and secondary authentication servers, make the settings at both the same by copying the JP1 users, JP1 resource group and other information from the primary authentication server to the secondary authentication server. If the settings are not the same, an authentication error will occur when you switch servers.

For details on the settings at the primary and secondary authentication servers, see the *Job Management Partner 1/Base User's Guide*.

For details on how to specify the authentication server when primary and secondary authentication servers are set, consult the following references.

- For a Windows host running JP1/AJS3 - Manager:

  See *3.1.1 Setting up JP1/Base* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1*.

- For a Windows host running JP1/AJS3 - Agent:

  See *3.2.1 Setting up JP1/Base* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1*.

- For a UNIX host running JP1/AJS3 - Manager:

  See *12.1.1 Setting up JP1/Base* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1*

- For a UNIX host running JP1/AJS3 - Agent:

  See *12.2.1 Setting up JP1/Base* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1*.

6.3 Setting registered users

When you have settled the range in which to manage user access permissions and the host that will manage access permissions, you must register the JP1 users at JP1/AJS3.

The following table gives an example of the registration of JP1 users.

Table 6-2: Example of registration of JP1 users

JP1 User name	Purpose
`jp1admin`	Used to manage the entire system. This user has administrator's permissions for JP1/AJS3 (JP1/AJS3 Console), JP1/IM, and JP1/Base set as defaults. The following permissions are set: jp1admin: *=JP1_AJS_Admin, JP1_JPQ_Admin, JP1_Console_Admin JP1_AJS_Admin : Administrator's permission for JP1/AJS3 JP1_JPQ_Admin : Administrator's permission for JP1/AJS3 job execution environment, queues, etc., and Administrator's permission for agent management information JP1_Console_Admin : Administrator's permission for JP1/IM Normally, do not define or execute work tasks with "jp1admin". We recommend that you create a JP1 user called `jp1mngr` to define and execute work tasks. As far as possible do not delete `jp1admin` or change any of its settings other than the password.
`jp1mngr`	Used to manage work tasks (user who is a work task administrator) and execute processing (user for executing jobs). Set for this user the permission to refresh, execute and monitor all work tasks. Also set the permission that enables execution of processing that - owing to the specifications of the OS or program - cannot be executed without an administrator's permission or superuser permission. The following is an example of a permission setting. jp1mngr: *=JP1_AJS_Manager, JP1_JPQ_Operator JP1_AJS_Manager : permission to edit and execute work tasks JP1_JPQ_Operator : permission to operate job execution environments, queues, etc., permission to operate jobs, and permission to operate agent management information
`jp1user`	Used to execute processing (user who executes jobs) To ensure efficient operation management, standardize the users who execute jobs at the work task manager host. The following is an example of a permission setting. jp1user: *=JP1_AJS_Manager, JP1_JPQ_Operator JP1_AJS_Manager : permission to edit and execute work tasks JP1_JPQ_Operator : permission to operate job execution environments, queues, etc., permission to operate jobs, and permission to operate agent management information
`jp1ope`	Used to monitor work tasks and to rerun them manually. This user is intended for operators, allowing them to monitor work tasks and perform manual operations. The following is an example of a permission setting. jp1ope: *=JP1_AJS_Operator JP1_AJS_Operator : permission to execute work tasks, permission to view work tasks

JP1 User name	Purpose
`jp1guest`	Used to monitor work tasks only. This is a work task monitoring user for operators. The following is an example of a permission setting. jp1guest: *=JP1_AJS_Guest JP1_AJS_Guest : permission to reference work tasks
`root`	Used to monitor JP1/AJS3 job execution environments, queues, etc. Note that when the `jpq`*xxxx* command is executed from the command prompt, the OS user that executes the command must be registered as a JP1 user. The JP1 user called *root* explained in this example means the root user of the OS (UNIX). (When using Windows, the OS user of the Administrators group must be registered as the JP1 user of the Administrators group.) The following is an example of a permission setting. root: *=JP1_JPQ_Admin JP1_JPQ_Admin : permission to manage job execution environments, queues, etc.

For details on the permissions that can be set for JP1 users, see *6.4(2) Determining JP1 permission levels*.

You can omit registration of JP1 users but you need to register such JP1 users to execute jobs.

If you omit the registration of JP1 users and set access permissions as described in *6.4 Setting access permissions*, JP1/AJS3 assumes that JP1 users having the same names as the OS users are registered.

6.4 Setting access permissions

When you have registered the required JP1 users, you must decide what access permissions (JP1 permission levels) to grant to these JP1 users in regard to which groups (JP1 resource groups).

(1) Determining the JP1 resource groups to be defined

A job network element (unit) is saved and managed as a member of a number of groups. These groups are called *JP1 resource groups*. You can define any name you like for a JP1 resource group. For example you can set definitions such as purchasingdep or personneldep for company departments (in this case the Purchasing Department and Personnel Department).

You need to define a JP1 resource group to a JP1 user. If you fail to do so, access control is not applied to that group. It is therefore inadvisable to operate JP1/AJS3 without specifying the JP1 users for which a JP1 resource group is defined.

Note that you can define multiple JP1 resource groups for a single JP1 user. For example, a single JP1 user can be defined both as a user who can execute and edit work tasks for the purposes of the General Affairs Department JP1 resource group, and also as a user that can only view work tasks for the purposes of the Sales Department JP1 resource group.

(2) Determining JP1 permission levels

A *JP1 permission level* determines the operations that can be done with respect to a work task defined with JP1/AJS3.

JP1 permission levels are set for JP1 resource groups. A JP1 permission level setting only becomes effective when a JP1 resource group is specified. For example, different JP1 resource groups can be granted different access permissions for defining and executing jobnets: one JP1 resource group could be granted JP1_AJS_Admin access permissions, while another could be granted JP1_AJS_Manager permissions. In this case, note that the JP1 permission level will not be valid when no JP1 resource group is specified or when a different JP1 resource group is specified.

There are three types of JP1 permission level:

- Access permission for defining and executing jobnets
- Access permissions for executing and working with commands in the execution environment of QUEUE jobs and submit jobs
- Access permissions for working with execution agents

The names of the permission levels and the operations possible with each are covered below. Refer to the details on operations given here when setting JP1 permission levels.

(a) Access permissions when defining and executing jobnets

There are the following five kinds of access permissions for defining and executing jobnets:

- JP1_AJS_Admin

  This is the administrator's permission. It confers ownership of the unit, the permission to operate resource groups, the permissions to define, execute, and edit jobnets, and so on.

- JP1_AJS_Manager

  Confers permissions including the permissions to define, execute, and edit jobnets.

- JP1_AJS_Editor

  Confers permissions including the permissions to define and edit jobnets.

- JP1_AJS_Operator

  Confers permissions including the permissions to execute and view jobnets.

- JP1_AJS_Guest

  Confers permissions including the permissions to view jobnets.

The following table lists the JP1 permission level names and details of their operation when defining and executing jobnets.

Table 6-3: JP1 permission level names and available operations when jobnets are defined and executed

Operation details	JP1_AJS _Admin	JP1_AJS _Manager	JP1_AJS _Editor	JP1_AJS _Operator	JP1_AJS _Guest
Changing the owner, JP1 resource group name, or type of job execution user of a unit whose owner permission resides with another user	P[#1]	--	--	--	--
Defining units	P	P	P	--	--
Changing the definition of units defined for jobnets	P	P[#2]	P[#2]	--	--
Changing jobnet definitions	P	P	P	--	--
Replicating and moving units, and changing their names[#3]	P	P	P	--	--
Deleting units[#4]	P	P	P	--	--

Operation details	JP1_AJS _Admin	JP1_AJS _Manager	JP1_AJS _Editor	JP1_AJS _Operator	JP1_AJS _Guest
Outputting units to standard output files	P	P	P	P	P
Outputting definitions of units to standard output files	P	P	P	P	P
Backing up units	P	P	P	P	P
Recovering units	P	P	P	--	--
Defining calendar information for a job group	P	P	P	--	--
Defining a jobnet execution schedule for a specific period	P	P	P	P	P
Registering a defined jobnet for execution	P	P	--	P	--
Canceling the registration of a jobnet for execution	P	P	--	P	--
Registering a jobnet for release	P	P	P#5	P#5	--
Canceling the release of a jobnet	P	P	P#5	P#5	--
Viewing the release information for a jobnet	P	P	P	P	P
Outputting information such as the job execution history, current status, and next planned execution to a standard output file	P	P	P	P	P
Temporarily changing a schedule defined in a jobnet	P	P	--	P	--
Temporarily changing the status of a job	P	P	--	P	--
Changing the status of a job	P	P	--	P	--
Suspending the execution of a jobnet	P	P	--	P	--
Rerunning a jobnet	P	P	--	P	--
Killing a job or jobnet	P	P	--	P	--
Exporting units	P	P	P	P	P
Importing units	P	P	P	--	--

Operation details	JP1_AJS _Admin	JP1_AJS _Manager	JP1_AJS _Editor	JP1_AJS _Operator	JP1_AJS _Guest
Exporting the execution registration status of a jobnet	P	P	P	P	P
Importing the execution registration status of a jobnet	P	P	--	P	--

Legend:

P: Possible

--: Not possible

#1

If you are the owner of a unit, you can perform these operations even if you have not been granted JP1_AJS_Admin permission. However, even if you are the owner of a unit, if no reference permission has been assigned to the JP1 resource group set for that unit, you will not be able to change the JP1 resource group name, the owner, or the type of user who executes jobs from JP1/AJS3 - View. If you want to change the JP1 resource group name, the owner, or the type of user who executes jobs from JP1/AJS3 - View, execute the `ajschange -g` *JP1-resource-group-name unit-name* command, and change to a JP1 resource group that has been granted permission to reference JP1 resource group names.

Note that if no owner has been set for a unit, all users can change the JP1 resource group names, owner, and type of user who executes jobs. When *owner user* is set as the type of user who executes the unit, if a user other than those indicated at (a) and (b) below changes the unit owner to another JP1 user, the registered user will be set as the type of user who executes the unit. In this case, the type of user who executes jobs is taken to be the user who registered the jobnet for execution. This is intended to prevent jobs being executed by users other than those indicated at (a) and (b) (users with any user permission).

(a) : Users granted administrator's permissions or superuser permissions

(b) : JP1 users granted JP1_AJS_Admin permissions with respect to the JP1 resource group set for the unit

#2

When the type of user who executes the unit is *owner user*, JP1 users with permissions other than JP1_AJS_Admin cannot execute change operations in units except those that they own. This is intended to prevent general users that have not been granted JP1_AJS_Admin permission from executing any job they like. If the type of user who executes the unit is the registered user, anyone who obtains the JP1 permission level that permits the operation can execute a change operation.

#3

The permission is required for the parent unit of the unit to be copied, moved, or renamed. For example, when you want to move the unit /AAA/BBB/CCC, the permission must be set for the JP1 resource group of the unit /AAA/BBB. Note that you need the following permissions for a unit that you want to copy, move, or rename.

To copy a unit, the JP1_AJS_Admin, JP1_AJS_Manager, JP1_AJS_Editor, JP1_AJS_Operator, or JP1_AJS_Guest permission must be set for the JP1 resource group of the unit (including subordinate units).

To move or rename a unit, the JP1_AJS_Admin, JP1_AJS_Manager, or JP1_AJS_Editor permission must be set for the JP1 resource group of the unit.

#4

The permission is also required for the parent unit of the unit to be deleted. For example, when you want to delete the unit /AAA/BBB/CCC, the permission must be set for the JP1 resource groups of the unit /AAA/BBB/CCC (including subordinate units) and the unit /AAA/BBB.

#5

JP1_AJS_Editor and JP1_AJS_Operator permissions are both required.

This is because registering a jobnet for release and canceling the release involves changing the jobnet definition and submitting the redefined jobnet for execution.

A user who performs operations on a unit must have permission to operate the JP1 resource group set for that unit. In addition, the JP1_AJS_Admin, JP1_AJS_Manager, JP1_AJS_Editor, JP1_AJS_Operator, or JP1_AJS_Guest permission must be set for the JP1 resource groups of the upper-level units.

JP1 users mapped to OS users with administrator's permissions or superuser permissions can execute all operations regardless of their JP1 permission level. If you execute a command that affects a job network element as a user with administrator's permissions or superuser permissions, you will be able to perform all operations regardless of the user permissions of the JP1 user set in the JP1_USERNAME environment variable.

Note that if no JP1 resource group has been set for a unit, all users can perform all JP1/AJS3 operations with respect to that unit.

Cautionary notes

- Access permission for the referenced JP1/AJS3 - Manager is granted for manager job groups and manager jobnets.
- While JP1/AJS3 - View is connected to JP1/AJS3 - Manager, access

permission information is stored in a JP1/AJS3 - Manager cache. For this reason, when access permissions are changed, the change might not be applied to the connected JP1/AJS3 - View. When you change access permissions, disconnect JP1/AJS3 - View from JP1/AJS3 - Manager, change the access permissions, and then connect JP1/AJS3 - View again.

(b) Access permissions for executing and working with commands in the execution environment of QUEUE jobs and submit jobs

There are three types of access permissions for executing and working with commands in the execution environment of QUEUE jobs and submit jobs.

- JP1_JPQ_Admin

  This is the administrator's permission. It confers the permission to set job execution environments, the permission to operate queues and job executing agents, and the permission to operate jobs queued by other users.

- JP1_JPQ_Operator

  This confers the permission to operate queues and agents, and the permission to operate jobs queued by other users.

- JP1_JPQ_User

  This confers the permission to register submit jobs and to operate jobs that you have queued yourself.

When setting access permissions for executing and working with commands in the execution environment of QUEUE jobs and submit jobs, assign these permission levels to the JP1 resource group `JP1_Queue`. Note that the entry `JP1_Queue` is case sensitive.

The following table lists the JP1 permission level names and details of their operation for executing and working with commands used in the execution environment of QUEUE jobs and submit jobs.

Table 6-4: JP1 permission level names and available operations for executing and working with commands in the execution environment of QUEUE and submit jobs

Operation details	JP1_JPQ_ Admin	JP1_JPQ_ Operator	JP1_JPQ_ User
Registering submit jobs	P	P	P
Canceling/killing job execution	P	P	O
Holding job execution/releasing a job from held status	P	P	O
Moving a job	P	P	O

Operation details	JP1_JPQ_ Admin	JP1_JPQ_ Operator	JP1_JPQ_ User
Outputting job information	P	P	O
Outputting ended job information	P	P	O
Deleting ended job information from the database	P	P	--
Opening a queue	P	P	--
Closing a queue	P	P	--
Adding a queue	P	--	--
Deleting a queue	P	--	--
Outputting queue information	P	P	P
Changing the definition of a queue	P	--	--
Connecting a queue to an agent	P	--	--
Disconnecting a queue from an agent	P	--	--
Changing the maximum number of concurrently executable jobs	P	--	--
Adding an agent	P	--	--
Deleting an agent	P	--	--
Outputting agent host information	P	--	--
Adding an execution-locked resource	P	--	--
Deleting an execution-locked resource	P	--	--
Outputting information about an execution-locked resource	P	P	P

Legend:

P: Possible.

O: Possible, but not for jobs executed by other users.

--: Not possible

Cautionary note

The execution and operation of commands used in the execution environment of QUEUE and submit jobs is subject to the access permissions defined for the manager that requested the processing.

(c) Access permissions for working with agent management information

There are three kinds of access permissions for working with agent management information:

- JP1_JPQ_Admin

  This is the administrator's permission. It is the permission required to add, edit, and delete execution agent and execution agent group definitions.

- JP1_JPQ_Operator

  This is the permission required to change the job transfer restriction status of execution agents and execution agent groups.

- JP1_JPQ_User

  This is the permission required to view statuses and definition information for execution agents and execution agent groups.

When setting access permissions for working with agent management information, assign these permission levels to the JP1 resource group JP1_Queue. Note that the entry JP1_Queue is case sensitive.

The following table lists the JP1 permission level names and operational details for working with agent management information.

Table 6-5: JP1 permission level names and available operations for working with agent management information

Operation details	JP1_JPQ_ Admin	JP1_JPQ_ Operator	JP1_JPQ_ User
Adding an execution agent	P	--	--
Adding an execution agent group	P	--	--
Deleting an execution agent	P	--	--
Deleting an execution agent group	P	--	--
Changing the execution host for an execution agent	P	--	--
Changing the number of concurrently executable jobs at an execution agent	P	--	--
Changing the description of an execution agent	P	--	--
Changing the description of an execution agent group	P	--	--
Adding an execution agent that connects to an execution agent group	P	--	--

Operation details	JP1_JPQ_ Admin	JP1_JPQ_ Operator	JP1_JPQ_ User
Changing the priority of an execution agent connected to an execution agent group	P	--	--
Removing an execution agent as a connection-destination of an execution agent group	P	--	--
Changing the job transfer restriction status of an execution agent	P	P	--
Changing the job transfer restriction status of an execution agent group	P	P	--
Displaying the status of an execution agent#	P	P	P
Displaying the status of an execution agent group#	P	P	P
Displaying the status of all execution agents and execution agent groups#	P	P	P
Displaying the names of all execution agents and execution agent groups#	P	P	P
Outputting the definition of an execution agent#	P	P	P
Outputting the definition of an execution agent group#	P	P	P
Outputting the definition of all execution agents and execution agent groups#	P	P	P

Legend:

P: Possible.

--: Not possible

#

Users with administrator's permissions or superuser permissions can perform all operations regardless of their JP1 permission levels.

(3) Unit owner permission

When a job or jobnet is defined, the user who defines it has the owner permission for that job or jobnet. Holding the owner permission means that you can change JP1 resource group names, the owner, and the type of user who executes jobs regardless of the JP1 permission level. You should therefore take care regarding the following points.

Cautionary notes

- If no owner is set for a unit, all users are able to change JP1 resource group names, owners, and the type of user who executes jobs.
- When *owner user* is set as the type of user who executes the unit, if a user other than those indicated at (a) and (b) below changes the unit owner to another JP1 user, the registered user will be set as the type of user who executes the unit. In this case, the type of user who executes jobs is taken to be the user that registered the jobnet for execution. This is intended to prevent jobs being executed by users other than those indicated at (a) and (b) (users with any user permission).

  (a): Users granted administrator's permissions or superuser permissions

  (b): JP1 users granted JP1_AJS_Admin permissions with respect to the JP1 resource group set for the unit
- When *job owner* is set as the type of user who executes the job, if a user other than those indicated below changes the definition of the job, a permission error occurs:

  - Owner of the job

  - JP1 user mapped to an OS user granted administrator's permissions or superuser permissions

  - General users granted JP1_AJS_Admin permissions
- If you are not granted reference permission to a JP1 resource group set for a unit even though you are the owner of that unit, you cannot change the JP1 resource group name, owner, or type of user who executes jobs from JP1/AJS3 - View. If you want to change the JP1 resource group name, owner, or type of user who executes jobs, execute the `ajschange -g` *JP1-resource-group-name unit-name* command and then change to a JP1 resource group granted the permission to reference the JP1 resource group name.

6.5 Mapping users

You need to establish the correspondence between the JP1 users at the JP1/AJS3 work task manager and the OS users at the host that executes the processing.

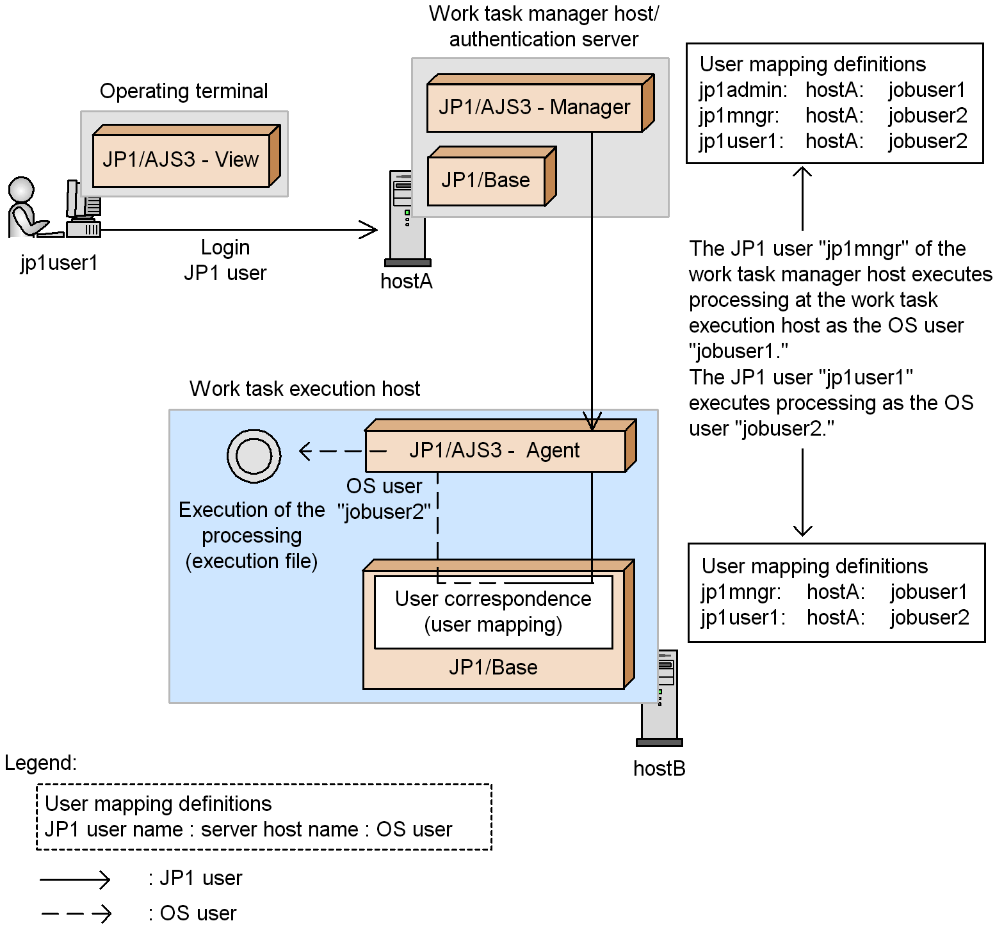
When JP1/AJS3 executes processing, OS resources such as executable files are accessed in accordance with the permissions of the OS user that corresponds to the JP1 user. This means that you must specify the OS user corresponding to the JP1 user at the host that executes the processing. This is called *user mapping* and utilizes the JP1/Base user mapping function.

User mapping is also necessary when you log in from JP1/AJS3 - View. You must set the user mapping before using JP1/AJS3 - View.

Note that JP1/AJS3 event jobs (in Windows) do not rely on JP1 users at execution. They rely on the account permissions of JP1/AJS3 Service.

The figure below gives an overview of processing execution using user mapping.

Figure 6-2: Overview of processing execution using user mapping


Work task manager host/
authentication server
User mapping definitions
jp1admin: hostA: jobuser1
jp1mngr: hostA: jobuser2
jp1user1: hostA: jobuser2
Operating terminal
JP1/AJS3 - View
JP1/AJS3 - Manager
JP1/Base
jp1user1
Login
JP1 user
hostA
The JP1 user "jp1mngr" of the work task manager host executes processing at the work task execution host as the OS user "jobuser1."
The JP1 user "jp1user1" executes processing as the OS user "jobuser2."
Work task execution host
JP1/AJS3 - Agent
OS user
"jobuser2"
Execution of the processing (execution file)
User correspondence (user mapping)
JP1/Base
User mapping definitions
jp1mngr: hostA: jobuser1
jp1user1: hostA: jobuser2
hostB
Legend:
User mapping definitions
JP1 user name : server host name : OS user
: JP1 user
: OS user

In the figure above, the following mapping is performed on the agent host:

- jp1mngr: jobuser1
- jp1user1: jobuser2

For the OS user jobuser1, set a user with administrator's permissions or superuser permissions. These permissions are used when they are required by the program specifications; e.g. for rebooting.

For the OS user jobuser2, set permissions for the executed processing (OS user account, file access permissions, etc.) so that the processing does not end abnormally. Remember that standardizing the OS user name (job-executing user) at all agent hosts makes administration easier.

The way that the user names and user mapping used when operating jobs and jobnets are decided differs according to the command used. Cases where units (jobs and jobnets) are operated with an `ajs`*xxxx* command and JP1/AJS3 - View, where a job in the job execution environment is operated and executed with a `jpq`*xxxx* command, and agent management information is operated with commands are shown below. Approach mapping by referring to the rules described below.

Note that since commands that operate event jobs do not rely on the JP1 permissions level, they do not use a JP1 user name.

(1) JP1 user names when a job network element is operated with JP1/AJS3 - View and commands

When you operate on a job network element from JP1/AJS3 - View, the JP1 user name used to check the permissions is the one used to log in to JP1/AJS3 - View. When you operate on a job network element with an `ajs`*xxxx* command, the JP1 user name is decided in accordance with the following rules:

- When the environment variable `JP1_USERNAME` is set

  When the environment variable `JP1_USERNAME` is set, the setting made for it is taken as the JP1 user name. You must ensure that the OS user name at command execution and the setting for the environment variable `JP1_USERNAME` are mapped by user mapping, except when the OS user when the command is executed is a member of the Administrators group or a superuser, in which case mapping is not necessary.

  The user mapping also differs depending on whether the environment variable `JP1_HOSTNAME` is set.

  When the environment variable `JP1_HOSTNAME` is set

  The user mapping defined at the logical host set for the environment variable `JP1_HOSTNAME` is used.

  When the environment variable `JP1_HOSTNAME` is not set

  The user mapping defined at the physical host is used.

- When the environment variable `JP1_USERNAME` is not set

  When the environment variable `JP1_USERNAME` is not set, the OS user name is taken as the JP1 user name. When a job is executed the user mapping is checked, so a JP1 user with the same name as the OS user must be registered.

If a JP1 resource group name is specified in the attributes of the jobs and jobnets operated, JP1/AJS3 checks with the authentication server about access permissions. If the environment variable `JP1_HOSTNAME` is set, the logical server defined in the logical host in the setting is used, and if the environment variable `JP1_HOSTNAME` is not set, the authentication server defined in the physical host is used. However, if the OS user when the command is executed is a member of the Administrators group or a superuser, the authentication server is not asked about access permissions.

Next, we explain how to remotely execute a command for operating units. For details about the commands that can be remotely executed, see *1.1 Command syntax* in the manual *Job Management Partner 1/Automatic Job Management System 3 Command Reference 1*.

The following settings are required on the hosts that remotely execute commands:

- When environment variable `JP1_USERNAME` is set

  If environment variable `JP1_USERNAME` is set when a command is remotely executed, the value set for `JP1_USERNAME` is used as the JP1 user name when the command is executed on the target host. One of the OS user names on the command execution destination host and the value set for environment variable `JP1_USERNAME` must be mapped by user mapping on the command execution host.

  The type of user mapping to be performed differs depending on whether a logical host name or a physical host name is specified as the command execution destination host.

  When a logical host name is specified for the command execution destination host:

  > The user mapping defined for the specified logical host is used.

  When a physical host name is specified for the command execution destination host:

  > The user mapping defined for the specified physical host is used.

- When environment variable `JP1_USERNAME` is not set

  If environment variable `JP1_USERNAME` is not set, JP1/AJS3 treats the OS user name of the command execution source host as the JP1 user name.

If a JP1 resource group name is specified in the attributes of the job or jobnet to be operated, JP1/AJS3 checks with the authentication server about access permissions. If you specify a logical host name for the command execution destination host, the authentication server defined in the logical host is used. If you specify a physical host name for the command execution destination host, the authentication server defined in the physical host is used. Set the JP1 permission level required for using the command. However, if the mapped primary user is a member of the Administrators group or a superuser, the authentication server is not asked about access permissions.

(2) JP1 user names when a job in the job execution environment is executed and operated with commands

When you use a `jpq`*xxxx* command to perform operations on a job in the job execution environment, or you perform operations on the job execution environment itself, the permissions are checked based on the JP1 user name with the same name as the OS user who executes the command.

You should therefore register the OS user that executes the command as the JP1 user. Then, map the registered JP1 user and OS user. Note that regardless of the setting of the environment variable `JP1_USERNAME`, you must register the OS user that executes the command as the JP1 user and set the user mapping.

Before executing a job, and this includes cases where you start the job from JP1/AJS3 - View, you also have to register the OS user that executes the job as a JP1 user in the authentication server.

For details on how to register JP1 users and how to set JP1 permissions levels, see *3.1.1 Setting up JP1/Base* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1* (for Windows hosts) or see *12.1.1 Setting up JP1/Base* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1* (for UNIX hosts).

In addition, for details on the permission levels required to use the various commands, see *1.5 Commands* in the manual *Job Management Partner 1/Automatic Job Management System 3 Command Reference 1.*

(3) JP1 user names when agent management information is operated on with commands

When you use a command to perform operations on agent management information, the JP1 user name is decided in accordance with the following rules:

- When the environment variable `JP1_USERNAME` is set

  The user name set in the environment variable `JP1_USERNAME` is used as the JP1 user name. Note that you must ensure that the user name set in the environment variable `JP1_USERNAME` and the OS user at command execution are mapped in the JP1/Base user mapping definition. User mapping is unnecessary if the OS user at command execution is a member of the Administrators group or a superuser.

  The user mapping also differs depending on whether the environment variable `JP1_HOSTNAME` is set.

  If the environment variable `JP1_HOSTNAME` is set

    The user mapping defined on the logical host set for the environment variable `JP1_HOSTNAME` is used.

  If the environment variable `JP1_HOSTNAME` is not set

    The user mapping defined on the physical host is used.

- When the environment variable `JP1_USERNAME` is not set

  Because the OS user name is used as the JP1 user name, a JP1 user with the same name as the OS user must be registered.

When you attempt to perform an operation on agent management information, JP1/AJS3 queries the authentication server about access permissions. If you specify a logical host as the target host for the agent management information, the authentication server defined on the logical host is used. If you specify a physical host as the target host, the authentication server defined on the physical host is used. Note, however, that when you use the `ajsagtshow` and `ajsagtprint` commands as a member of the Administrators group or a superuser, user mapping is unnecessary and the authentication server is not queried about access permissions.

6.6 User management when monitoring work tasks centrally

This subsection explains how users are managed when you monitor work tasks centrally using JP1/AJS3 Console. To monitor work tasks centrally using JP1/AJS3 Console, you must make the following settings using the user management functionality (user authentication, user mapping) of JP1/Base.

- Settings required for login
- Settings required for status monitoring

For details on the user management functionality of JP1/Base, see the section on setting the user management functionality in the *Job Management Partner 1/Base User's Guide*.

The conditions required for login and the conditions required for status monitoring are indicated below. Set users and permissions that satisfy these conditions.

(1) Conditions required for login

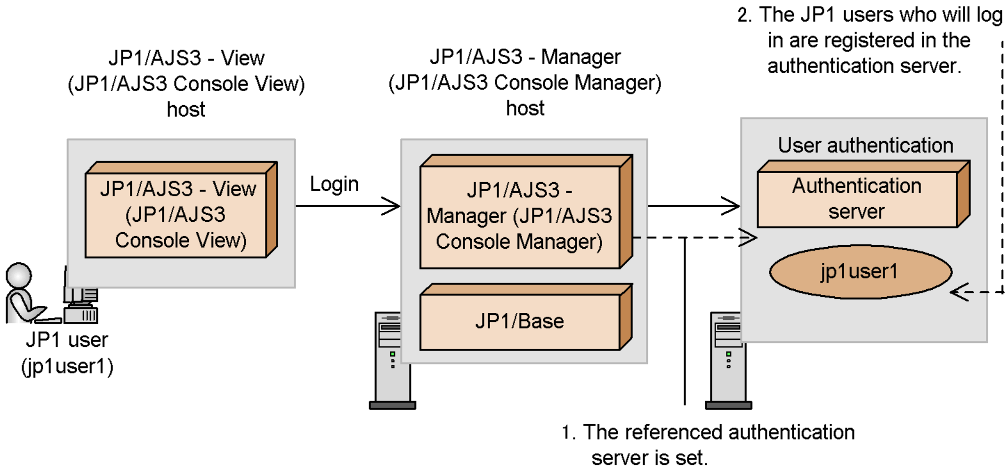
JP1/AJS3 Console uses the JP1/Base user authentication functionality when you log in to the JP1/AJS3 Console Manager from JP1/AJS3 Console View.

JP1/AJS3 Console retains a Main Scope window, to permit the JP1 users to monitor work tasks, at the JP1/AJS3 Console Manager hosts at the connection-destinations, and manages the definitions (monitoring objects) for each JP1 user. In order to prevent access by inadmissible users, there is a login process and users receive authentication.

The JP1 users that log in are authenticated by the authentication server that the JP1/AJS3 Console Manager host at the login destination references.

The following figure shows the conditions required to log in to a JP1/AJS3 Console Manager host from JP1/AJS3 Console View.

Figure 6-3: Conditions required for login


2. The JP1 users who will log in are registered in the authentication server.
JP1/AJS3 - View (JP1/AJS3 Console View) host
JP1/AJS3 - Manager (JP1/AJS3 Console Manager) host
User authentication
JP1/AJS3 - View (JP1/AJS3 Console View)
Login
JP1/AJS3 - Manager (JP1/AJS3 Console Manager)
Authentication server
jp1user1
JP1/Base
JP1 user (jp1user1)
1. The referenced authentication server is set.

Note:

Entries in parentheses are component names.

1. The authentication server that the JP1/AJS3 - Manager (JP1/AJS3 Console Manager) references must be set at the JP1/AJS3 Console Manager host.
2. The JP1 users who will log in must be registered at the authentication server that the JP1/AJS3 - Manager (JP1/AJS3 Console Manager) references.

Since the definition details in the Main Scope window are saved individually for each JP1 user, no JP1 permission level setting is necessary.

(2) Conditions required to monitor work task statuses

JP1/AJS3 Console utilizes the user mapping functionality to monitor work tasks in JP1/AJS3.

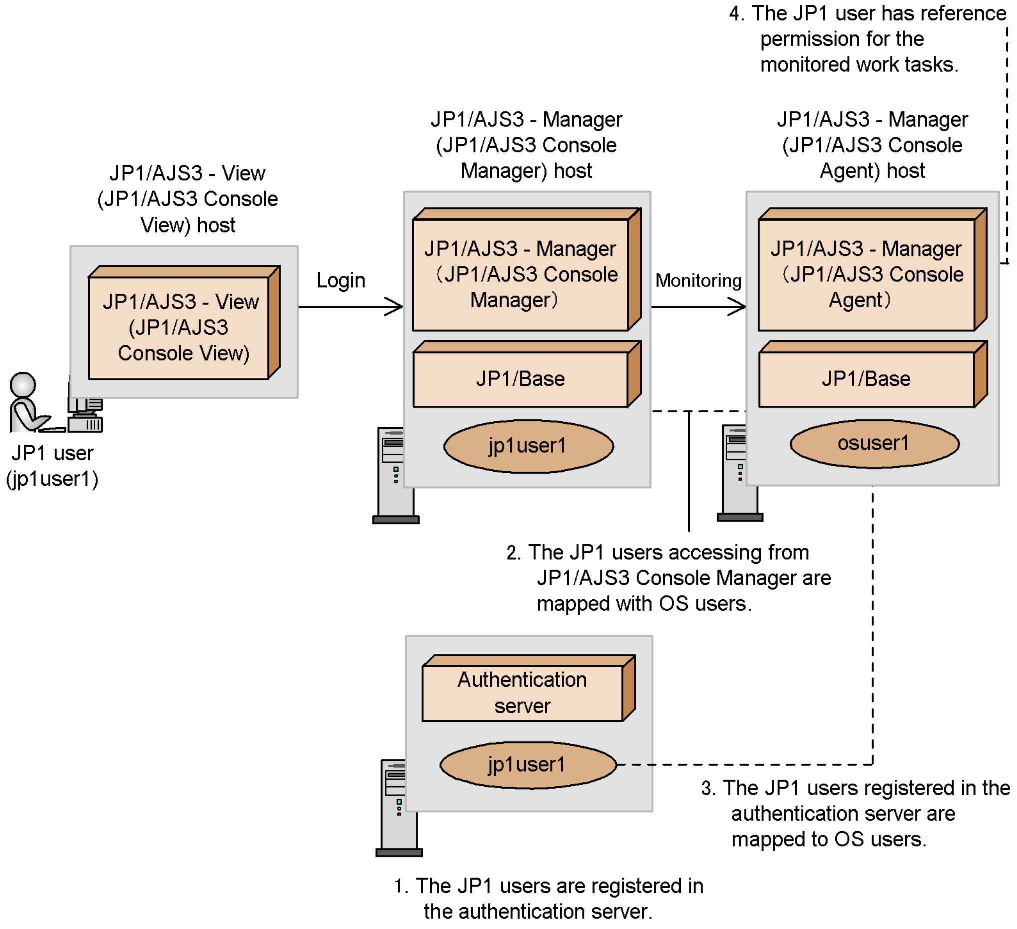
Problems may occur if all the JP1 users who log on to JP1/AJS3 Console Manager are allowed to reference and operate the work tasks on JP1/AJS3 Console Agent (JP1/AJS3 - Manager) hosts. You should therefore map OS users onto the JP1 users that are to monitor work tasks at the JP1/AJS3 Console Agent hosts, so that the work task access permissions of the JP1 users that log in at the JP1/AJS3 Console Manager can be managed.

Note that these access permissions conform to the JP1 permission levels set on the authentication server referenced by the JP1/AJS3 Console Agent hosts. Since it is the OS users who actually monitor work tasks, the OS users must also be mapped to the JP1/AJS3 Console Agent hosts.

The following figure shows the conditions required to monitor work tasks in JP1/

AJS3.

Figure 6-4: Conditions required for status monitoring


4. The JP1 user has reference permission for the monitored work tasks.
JP1/AJS3 - View (JP1/AJS3 Console View) host
JP1/AJS3 - View (JP1/AJS3 Console View)
JP1 user (jp1user1)
Login
JP1/AJS3 - Manager (JP1/AJS3 Console Manager) host
JP1/AJS3 - Manager (JP1/AJS3 Console Manager)
JP1/Base
jp1user1
Monitoring
JP1/AJS3 - Manager (JP1/AJS3 Console Agent) host
JP1/AJS3 - Manager (JP1/AJS3 Console Agent)
JP1/Base
osuser1
2. The JP1 users accessing from JP1/AJS3 Console Manager are mapped with OS users.
Authentication server
jp1user1
3. The JP1 users registered in the authentication server are mapped to OS users.
1. The JP1 users are registered in the authentication server.

Note:

Entries in parentheses are component names.

1. You must register the JP1 users who will monitor work tasks on JP1/AJS3 Console Manager at the authentication server that the JP1/AJS3 Console Agent host references.
2. At the JP1/AJS3 Console Agent hosts, you must map OS users onto the JP1 users who are to monitor work tasks (specify the JP1/AJS3 Console Manager host name as the server host name).

3. When making setting 2 above at the JP1/AJS3 Console Agent hosts regarding JP1 users that will monitor work tasks, you must map the OS user specified as the primary OS user (specify the JP1/AJS3 Console Agent (local host) host name).
4. The JP1 users that monitor work tasks must have permission to reference the work tasks to be monitored (JP1_AJS_Guest permission or higher).

Supplementary notes

- In the user mapping setting in 2 above, if you set * (asterisk) as the server host name, the setting in 3 will not be required.
- To start up JP1.AJS3 - View from JP1/AJS3 Console View and operate work tasks, or to operate monitored work tasks directly from JP1/AJS3 Console View, you need operation permissions for the work tasks (JP1_AJS_Operator permission or higher).
- In the setting in 2 above, if the OS user specified as the primary OS user has administrator's permissions (Windows) or superuser permissions (UNIX), no restrictions apply to the JP1 permission level settings.
- You can monitor work tasks even if the user authentication blocs of the JP1/AJS3 Console Manager host and the JP1/AJS3 Console Agent host are different. However, if they are in the same user authentication bloc, you can log in without displaying the Login screen when starting JP1/AJS3 - View from JP1/AJS3 Console View. For details, see *13.5 Monitoring application* in the *Job Management Partner 1/Automatic Job Management System 3 Operator's Guide*.
- JP1/AJS3 Console can monitor work tasks even if you do not register JP1 users for monitoring work tasks on the authentication server referenced by the JP1/AJS3 Console Agent host, as mentioned setting in 1 above. In this case, the following restrictions apply:

  (a) If the mapped OS user setting in 2 above is a member of the Administrators group (for Windows) or a superuser (for UNIX):

  - To start JP1/AJS3 - View from JP1/AJS3 Console View, login must be done on a started JP1/AJS3 - View.

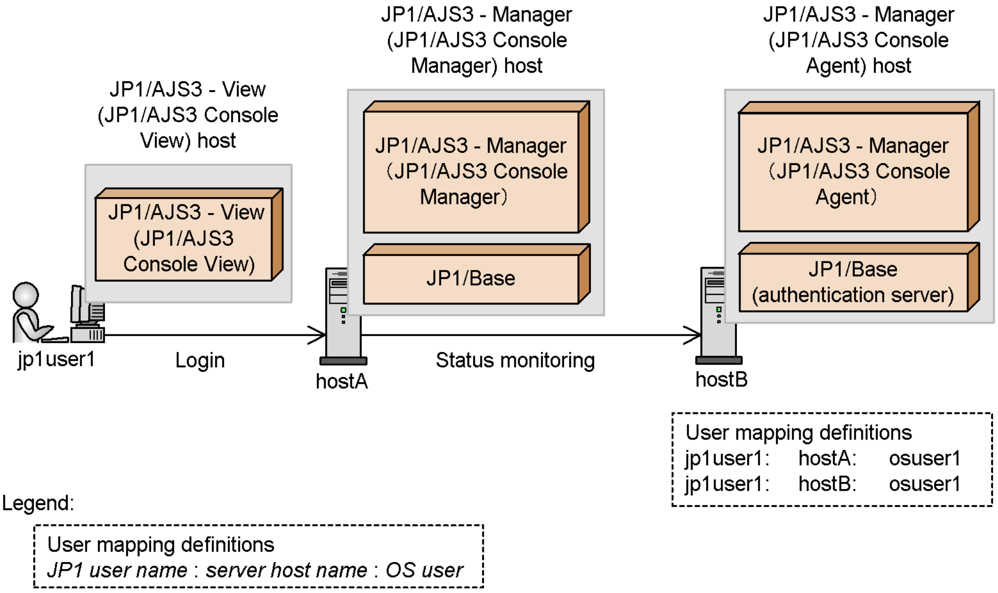
  (b) If the mapped OS user in condition 2 is a general user (users other than those in (a)):

  - To start JP1/AJS3 - View from JP1/AJS3 Console View, login must be done on a started JP1/AJS3 - View.
  - Monitoring is not available if a resource group is specified for the jobnet to be monitored or a higher-level unit.

(3) Example user mapping definition

The following figure shows an example of defining user mapping.

Figure 6-5: Example of user mapping definition



Note:
Entries in parentheses are component names.

From JP1/AJS3 Console View, a JP1 user (jp1user1) logs in to the JP1/AJS3 Console Manager host (hostA). The JP1/AJS3 Console Manager host (hostA) directs the JP1/AJS3 Console Agent host (hostB) to monitor work task statuses for the JP1 user (jp1user1). The JP1/AJS3 Console Agent host (hostB) monitors the status of work tasks based on the permission level of the OS user (osuser1) mapped onto the JP1 user (jp1user1) from the JP1/AJS3 Console Manager host, and the JP1 access permission of the JP1 user (jp1user1).

Chapter
7. Cautionary Notes on Designing Work Tasks

This chapter provides cautionary notes on designing work tasks.

7.1 Notes on the number of jobnets registered for execution

Without taking into account available disk and memory resources and processing performance, the maximum number of jobnets that can be registered for execution in theory is 2,147,483,647. As long as four or five thousand jobnets are registered in JP1/AJS3, there must be no functional problem. However, when executing jobnets and jobs in JP1/AJS3, consider the following points from a performance viewpoint.

(1) Notes on defining thousands of jobnets in one hierarchy level

If 5,000 jobnets are defined in one hierarchy level, system performance might suffer. Take the following measures if necessary. These measures assist not only execution performance, but also GUI-based monitoring and command performance.

- Start multiple JP1/AJS3 scheduler services.
- When there are a large number of jobnets that contain only a few units, aggregate the units into as few jobnets as possible.
- Arrange job groups into hierarchies.

  Do not group more than 500 jobnets, including nested jobnets, in one job group.
- Consider the following points about the jobnets you register:
  - Provide two or three hierarchy levels under a root jobnet.
  - Define no more than 50 to 80 nested jobnets and jobs in one jobnet.

(2) Other notes

- If you specify a *number of logs to keep*, *number-of-logs-to-keep* x *number-of-registered-jobnets* is used as the number of registered jobnets for disk capacity management and resource management purposes. Therefore, be careful about the disk capacity and other resource management items.
- Although a large number of jobnets can be registered, this does not mean that a large number of jobs can be executed concurrently. When you design a work task, be careful about system resources and processing performance. For example, when designing a work task that uses a large number of jobs, design the work task so that there is no time period where a large number of jobs are required to run concurrently.
- Design work tasks with a view to distributing tasks to more than one time period and CPU.
- When many jobnets are defined, consider jobnet registration methods (all or divided), registration time, time for start-up, and performance of commands for batch definition, operation, and display.

- If you register a large number of jobnets for execution, release, or a cancellation of release within a short period, and you specify a long fixed schedule period for fixed execution registration, the system load might temporarily spike, delaying execution of jobnet generations created when start conditions are satisfied.

  If you want to register many jobnets for execution, release, or a cancellation of release, reduce the number of jobnets you register at one time. If you want to register a jobnet for fixed execution, shorten the fixed schedule period to reduce the load on the system.

- Defining a large number of jobnets or jobs in one jobnet, or using too many relation settings, could cause a deterioration in operability and performance of JP1/AJS3 - View. To maintain operability and performance, the jobnet should have two or three hierarchy levels, and each jobnet should have about 50 to 80 units.

7.2 Relationship between number of logs to keep and performance

The *number of logs to keep* is the number of generations to be saved as the execution result of a jobnet. You can set a number of logs to keep for a root jobnet. When you set a number of logs to keep, you can view the execution result of the set generations (times) in the Daily Schedule window or Monthly Schedule window, or by using the `ajsshow` command. You can set a value from 1 to 99 as the number of logs to keep, which can be expanded to 999 in the environment settings of the JP1/AJS3 - Manager scheduler service.

However, as the number of logs to keep increases, the number of records calculated by *number-of-logs-to-keep* x *number-of-registered-units* also increases, in proportion to the scale of the jobnet. This has a significant effect on operations that access the database. For this reason, carefully consider the effect on system performance when you determine a value to set as the number of logs to keep. This setting affects the following main functionality (processing):

- Cancellation of registration for execution
- Display of the Monthly Schedule window and Daily Schedule window of JP1/AJS3 - View
- Creation of generations when a start condition is satisfied
- Execution of `ajsshow` and other commands
- Job execution

For the above processing, there is a tendency for the number of records and the processing time to increase in proportion to each other. From an unloaded state, it will take about one or two minutes to cancel the registration for processing of 1,000,000 records, and 8 to 10 minutes to display the Monthly Schedule window (although this might vary depending on the system performance). This processing will take even longer if the system is experiencing heavy loads such as high CPU utilization or frequent disk access. In contrast, it will take only a few seconds to cancel the registration for processing of 100,000 records, and one or two minutes to display the Monthly Schedule window.

If you use the recommended configuration of two or three jobnet hierarchy levels with each jobnet containing about 50 to 80 units, the processing speed will be acceptable for operation even when the number of logs to keep is set to the maximum of 999.

If you want to increase the number of logs to keep, try to design jobnets on a small scale. If you want to create a large-scale jobnet, decrease the number of logs to keep. Keep this balance in mind when you design a jobnet.

Under most circumstances, design jobnets on a small scale and set a small number of logs to keep, bearing in mind that processing might be performed in parallel. For example, a job might be running during cancellation of registration while its status is being monitored in the Monthly Schedule window.

The processing time for cancellation of registration mentioned above is the time that applies when the registration cancellation mode is asynchronous, as described in *6.1.6 Changing the mode in which unregistration or generation management deletes the generations of a jobnet* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1* (for Windows), or *14.1.6 Changing the mode in which unregistration or generation management deletes the generations of a jobnet* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1* (for UNIX). If the registration cancellation mode is synchronous, the processing will take dozens of minutes. We recommend that you use asynchronous mode.

If you have upgraded from version 8 or earlier, in the case of a jobnet for which a start condition is set, monitored generations (generations that enter *Now monitoring* status) and execution generations (generations produced when a start condition is satisfied) are managed separately by default. For this reason, setting a large number of logs to keep will have an even greater effect for such jobnets. In this case, the number of records is: *number-of-registered-units* x *number-of-logs-to-keep* x *number-of-times-start-condition-is-satisfied-per-schedule*. If start conditions are satisfied many times per schedule in your system, review the operation to ensure that unnecessary logs are deleted periodically (by canceling registration after a specific time has elapsed or on a specific date). For points to note when increasing the number of logs to keep for jobnets for which start conditions are set after upgrading from version 8 or earlier, see *4.2.3(5) Notes on setting a large number of logs to keep for a jobnet with a start condition (upgrade installation from version 8 or earlier)* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.

7.3 Notes on using PC jobs

This section gives notes on using PC jobs.

Read this section in conjunction with *2.7.2 Troubleshooting problems related to standard jobs, action jobs, and custom jobs* in the manual *Job Management Partner 1/ Automatic Job Management System 3 Troubleshooting*, which provides information about what might cause a PC job to fail to start or end abnormally, and cautionary notes about using PC jobs.

■ Avoiding a system resource shortage (in Windows)

When the agent host is a Windows host, if you execute more than a specific number of jobs concurrently, you might encounter a shortage of a system resource (desktop heap) and an error might occur depending on the system environment. In this case, consider taking the following action:

- Stop sharing the desktop heap with other applications

  Change the JP1/AJS3 service account to a user account. By setting the JP1/AJS3 service account to a user account that differs from the accounts of other services and the user account of the logon user, you can use the system without sharing the desktop heap.

- Decrease usage of the desktop heap

  You can decrease consumption of the desktop heap by setting the account of the OS user frequently used for job execution to the same account as the JP1/AJS3 service account.

■ Using space characters in an application file name (in Windows)

When the agent host is a Windows host, if an application file name with an extension associated with a file type contains a space character, check whether the file type is registered correctly in Windows in the File Types page in Explorer, and enclose the application file name with double quotation marks (`"`).

■ Preventing jobs from entering Failed-to-start status (in Windows)

When the agent host is a Windows host, and a job enters *Failed to start* status, make sure that the user who started the JP1/AJS3 service has the permissions listed below. When using queueless jobs, make sure that the following permissions have been set for the account of the queueless agent service:

- For the executable file of the job: Read and execute permissions
- For the environment variable file for the job: Read permission
- For the standard input file for the job: Read permission

- For the standard output file for the job: Read and write permissions
- For the standard error output file for the job: Read and write permissions
- For the transfer source file for the job: Read permission

■ Do not include a command that shuts down the OS in a PC job

Do not execute a PC job that includes a command for shutting down the operating system. Because Windows does not wait for JP1/AJS3 to stop before it shuts down, there might be damage to the JP1/AJS3 data files if Windows shuts down while JP1/AJS3 is running. If you want to shut down the OS from a job, use a Local Power Control job in conjunction with the JP1/Power Monitor.

To shut down the system manually, use the power control command (`aompwcon` command) of JP1/Power Monitor. If you want to shut down the system manually but do not have JP1/Power Monitor installed, stop the JP1/AJS3 service manually before shutting down the OS.

■ Use file names of 254 bytes or less in jobs (in Windows)

A job might enter *Failed to start* or *Ended abnormally* status if any of the following file names specified in the job is 255 bytes or longer:

- The executable file of the job
- The environment variable file for the job
- The standard input file for the job
- The standard output file for the job
- The standard error output file for the job
- The end judgment file for the job
- The transfer source file for the job
- The transfer destination file for the job

This problem also applies to queueless jobs. Make sure that the above file names have no more than 254 characters.

■ When the executable file name specifies a command or program incompatible with data input from a standard input file

To execute a job for which a command or program incompatible with data input from a standard input file is specified in the **File name** field of the Define Details dialog box, you specify `CON` in the **Standard input**, **Standard output**, and **Standard error** fields. In this case, JP1/AJS3 will not read data from standard input or acquire any data output to standard output and standard error output, and the job's execution results do not appear in the Execution Result Details dialog box of JP1/AJS3 - View. The `timeout` command is one example of a command that does not support data input from the standard input file.

7.4 Notes on using Unix jobs

This section gives notes on using Unix jobs.

Read this section in conjunction with *2.7.2 Troubleshooting problems related to standard jobs, action jobs, and custom jobs* in the manual *Job Management Partner 1/ Automatic Job Management System 3 Troubleshooting*, which provides information about what might cause a Unix job to fail to start or end abnormally, and cautionary notes about using Unix jobs.

■ End codes that can be set by jobs executed in UNIX

Jobs executed in UNIX can set end code values from 0 to 255. When a job fails to start, or when acquisition of standard output data or standard error output data fails, the end code is set to `-1`.

■ Preventing jobs from entering Failed-to-start status (in UNIX)

When the agent host is a UNIX host, and a job enters *Failed to start* status, make sure that the user who started the JP1/AJS3 service has write and read permissions for the following files and storage directories:

- Standard output file for the job
- Standard error output file for the job
- Work directory at agent process execution
- Work path specified in the job's detailed definition#1
- Home directory of the OS user who executed the job#1
- Log files used for job execution control#2

#1

The work path specified in the detailed definition of the job is the current directory at the time the job is executed. If you do not specify a work path, the home directory of the OS user is assumed. If no home directory is defined for the OS user, root (`/`) is assumed.

#2

For details about the log files used in job execution control, see *1.2.4 List of log files and directories* in the manual *Job Management Partner 1/Automatic Job Management System 3 Troubleshooting*.

■ Resource limits when Unix jobs are executed

In JP1/AJS3 for UNIX, the resource limits in effect at JP1/AJS3 startup apply when a job is executed. To apply resource limits, set them for the root user who starts JP1/AJS3. However, if you specify a limit in the job to be executed, the value specified in the job takes effect. For details, see *2.3 Setting up the job execution environment* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 2*.

The following is an example of changing file size limits:

1. In the login profile for the root user (usually [`/.profile`] (`$HOME/.profile`)), include the setting below:

   For *fsize*, specify the required file size. If you do not want to impose a limit, specify `unlimited`.

   ```
   ulimit -f fsize
   ```

2. Log in as the root user.
3. From the root account, start the JP1/AJS3 service.

   The *fsize* value takes effect.

Cautionary note

> In AIX, the resource limits you define in the resource settings file for the OS (`/etc/security/limits`) apply only to processes started by users using the `login` command over a telnet connection or similar. Because jobs started by JP1/AJS3 take the form of processes started by a service, the settings in the file do not apply.

■ Precautions applying when the JP1/AJS3 service starts automatically

In a system where the JP1/AJS3 service starts automatically, the login profile of the root user is not loaded. For this reason, even if you change the resource limits in the login profile of the root user, different limits might apply when you log in manually and start the JP1/AJS3 service. In this case, set resource limits in the environment setting parameters of the job execution environment. For details about these parameters, see *2.3 Setting up the job execution environment* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 2*.

You can also write resource limits into the automatic start script (`/etc/opt/jp1ajs2/jajs_start`) of JP1/AJS3. If you do so, test your settings thoroughly before using the system.

Note that jobs might be executed under a group ID that differs from the group ID set for the root user at login. For details, see *5.3.12 Group ID for job execution (UNIX* *only)* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.

■ Explicitly declaring the umask value for a job process in the profile of the job execution user or the executing shell

Job processes started by the JP1/AJS3 service adopt the umask value of the shell that started the JP1/AJS3 service, unless a umask value is explicitly declared.

If you specify the value of umask in a profile of the job-execution user such as `/etc/profile` and `$HOME/.profile`, the value in the profile overwrites the value of `umask` in the shell that started the JP1/AJS3 service.

You must explicitly specify the value of umask for the job process in a profile of the job execution user or the shell.

To change the value of umask for the standard output file and the standard error output file that are specified in a job definition, see *5.3.7 Value of umask set for the standard output file and the standard error output file (UNIX only)* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.

■ Using a value other than /bin:/usr/bin as the PATH environment variable

When you start a job from JP1/AJS3, JP1/AJS3 explicitly specifies `/bin:/usr/bin` for the `PATH` environment variable. If you want to specify a different value, define it in the command or script file that you specify when defining the job, or in the local login script.

■ Executing a user program that requires a terminal as a job

In the UNIX version of JP1/AJS3, a user program that requires a terminal might not operate correctly if executed as a job (the job might end abnormally).

■ Changing OS user information for the login shell or other feature when using queueless jobs

If OS user information for the login shell or other feature is changed when you use a queueless job, use either of the following methods to clear the cache:

- Execute the `ajsqlalter` command with the `-r` option specified.
- Restart the queueless agent service.

7.5 Notes on using a recovery job or recovery jobnet

The following provides precautions for defining a recovery job or a recovery jobnet.

Hereafter, the term *recovery job* also includes *recovery jobnet*.

- Even if a recovery job ends normally, the jobnet containing the recovery job is treated as having ended abnormally.
- Only a recovery job can be defined as the succeeding job to another recovery job. If you define a normal job as the succeeding job to a recovery job, the normal job is not executed.
- A job defined as the succeeding job of a recovery job is not executed when the jobnet executes normally. It is executed only after the preceding recovery job is executed due to an abnormal end, and the recovery job ends normally.
- A warning issued by a recovery job or a recovery jobnet or the abnormal status of a recovery job or a recovery jobnet does not affect the status of upper-level jobnets. However, if a recovery jobnet starts or ends later than expected, the statuses of upper-level jobnets are set as delayed.
- A recovery job or recovery jobnet is executed when the preceding job ends with one of the following abnormal statuses:
    - *Ended abnormally*
    - *Invalid exe. seq.*
    - *Interrupted*
    - *Killed*
    - *Failed to start*

  However, when the JP1/AJS3 service is restarted in warm-start mode, or if the root jobnet is interrupted or killed, the recovery job or recovery jobnet will not be executed even when the preceding job ends with one of these abnormal statuses.

7.6 Notes on using event jobs

This section gives notes on using event jobs.

- Event jobs do not support operations that use execution agent groups. If an event job without a specified execution agent is included in a root jobnet or nested jobnet for which an execution agent group is specified, JP1/AJS3 will attempt to use the agent group specified for the jobnet as the execution agent for the event job. If an execution agent with the same name as the agent group exists, the event job will be executed by that execution agent. If there is no execution agent with the same name as the agent group, an error occurs and the following message is output to the integrated trace log: `KAVT0403-E The specified agent is not defined in the job execution environment. (host=`*exec-agent*`,` *maintenance-information*`)`. Therefore, if you want to specify an execution agent group for a root jobnet or nested jobnet, make sure that an execution agent is explicitly specified for the event job in the jobnet.
- You cannot specify the name of an execution agent group in the **Exec-agent** field of the detailed definition of an event job.
- JP1/AJS3 cannot correctly detect events issued by the JP1/AJS3 program itself, including JP1 events, events logged to the Windows event log or syslog, or events recorded in HNTRLib and Scheduler log files. For details about the events that can be monitored by an event job, see *7.6.9 Monitoring events or messages issued by JP1/AJS3*.
- For an event job waiting for a start condition to be satisfied, if JP1/AJS3 - Manager stops during start condition monitoring, you can resume event monitoring after JP1/AJS3 - Manager restarts. If multiple events were being monitored for a start condition, you can hold the reception information of the events that satisfy the condition even after JP1/AJS3 - Manager restarts.
- When an event job is defined at the beginning of a jobnet, the jobnet is executed after the condition is satisfied, in the same way as a jobnet with a start condition. A jobnet with a start condition stays in *Wait for start condition* status while event reception is being monitored. A jobnet with an event job defined at the beginning is kept in *Now running* status while event reception is being monitored. When you define an event job at the beginning of a jobnet, we recommend that you use the event job to wait for events you expect to occur.
- When you monitor multiple events, you can improve performance by using regular expressions to create an event job that monitors all of them in a batch.

  For example, if you use the Receive JP1 Event job to monitor JP1 events in the form of messages KAVS0272-E and KAVS00273-E with event ID 00004131, group the event jobs by specifying only the event ID or by specifying `00004131` for the event ID and KAVS for the message. You can identify the contents of events by using passing information.

- A timeout period specified for an event job is counted at the execution agent. If the execution agent goes down during monitoring due to power failure or other problem, and then restarts and resumes monitoring events, the execution agent starts counting the timeout period again from the moment it restarts. This is called *monitoring of the timeout period according to relative time*. You can check whether counting of the timeout period has been restarted and the time at which the count was restarted in the execution result details for the event job.

  If you want monitoring to time out based at an absolute time from job registration, regardless of the status of the execution agent, define a start condition for the monitored event so that a JP1 event is sent to the manager when the start condition is satisfied. Then use a Receive JP1 Event job on the manager to specify the timeout time and start monitoring. This is called *monitoring of the timeout period according to absolute time*.

- In Windows, JP1/AJS3 event jobs do not depend on the settings of the JP1 users that run them. Rather, they depend on the account permissions for the JP1/AJS3 service.

  The following permissions must be granted to the accounts associated with the JP1/AJS3 service:

  - Write permission for monitoring target files and folders when a Monitoring Files job is used. The monitoring target files must not be read-only.
  - Write permission for message queues during MQSeries linkage
  - Write permission for message queues during TP1/Message Queue linkage

  If a required permission is not granted, the following will occur:

  - Event jobs (Monitoring Files jobs and Receive MQ Message jobs) end abnormally.
  - Events do not occur.

- JP1/AJS3 event jobs are independent of the permissions set for JP1 users defined in the JP1/Base environment settings and in the respective jobs.

- Sometimes there is a time lag between the execution of an event job and the actual time when event monitoring begins in the execution agent. The system detects only events that occur after event monitoring begins. Therefore, you must provide some leeway when executing an event job, to allow the system to be ready and monitoring when a target event occurs.

- When you execute event jobs (including those with a start condition), definition data for the executed event jobs and information about events that satisfy monitoring conditions is transmitted between processes such as the event/action control manager and the event/action control agent. If transmission fails due to some problem such as a temporary network error or because the remote process is busy, the information is saved to a file and transmission is attempted again after a predetermined interval. In JP1/AJS3, this is referred to as an *unreported information file*. At successful re-transmission, the information is deleted.

- When using JP1/AJS3, in the API settings file for the JP1/Base event service, set `keep-alive` for the communication type of the `server` parameter. Setting `close` as the communication type can result in operational problems, including JP1/AJS3 being unable to issue JP1 events at startup, causing message KAVT1040-E to be output to the integrated trace log. This in turn renders Receive JP1 Event jobs, Monitoring Log Files jobs, Monitoring Event Log jobs, and Send JP1 Event jobs unable to detect events, and causes Send JP1 Event jobs to end abnormally. For details about the API settings file and how to set parameters, see the *Job Management Partner 1/Base User's Guide*.
- In a manager/agent configuration, if the event/action control manager and the event/action control agent are unable to communicate due to a network error or other problem, inconsistencies in event job monitoring (including event jobs defined as start conditions) can occur if you perform any of the operations listed below. In this case, the event job will continue monitoring on the agent despite having ended on the manager.
  - Killing an event job
  - Killing a jobnet that has a start condition
  - Changing the status of an event job to *Ended*

  These operations can result in problems such as failure to restart the event job where the inconsistency occurred, and delays to processing of other normal event jobs.

  For this reason, if you perform one of the above operations in a system affected by a network error or similar problem, execute the `jpomanjobshow` command on the manager host, and the `jpoagtjobshow` command on the agent host. Compare the results of the commands to see whether any event jobs that have ended on the manager are still monitoring on the agent.

  If you discover an event job that is in *Now monitoring* status only on the agent host, restart the JP1/AJS3 service on the agent host, and then terminate the event job that is still monitoring on the agent.

7.6.1 Notes on the Receive JP1 Event job

The following provides precautions (items you must know in advance) for using the Receive JP1 Event job.

JP1 events are events that are managed by JP1/Base and issued when events occur in

JP1 programs. JP1 events contain information about various event levels, ranging from errors and warnings to reports and messages. You can execute a different succeeding job for each event level, or execute the succeeding job only when a specific message is received. You can use regular expressions to extract parts of messages or detailed information from within JP1 events and pass the information to succeeding jobs.

Examples of jobnets that use the Receive JP1 Event job are as follows:

- Execute the succeeding job when an error occurs in a JP1 program or a warning is reported.
- Execute the succeeding job at the completion of all processing in situations where processing is executed by two or more JP1 programs.
- Execute the Send JP1 Event job when a jobnet executed by JP1/AJS3 - Manager in another host ends, receive the sent JP1 event in another jobnet, and execute the succeeding job.
- Use the JP1/Base event conversion facility to execute the succeeding job when a non-JP1 application is terminated.

  For details about the event conversion facility, see the *Job Management Partner 1/Base User's Guide*.

(1) Cautionary note

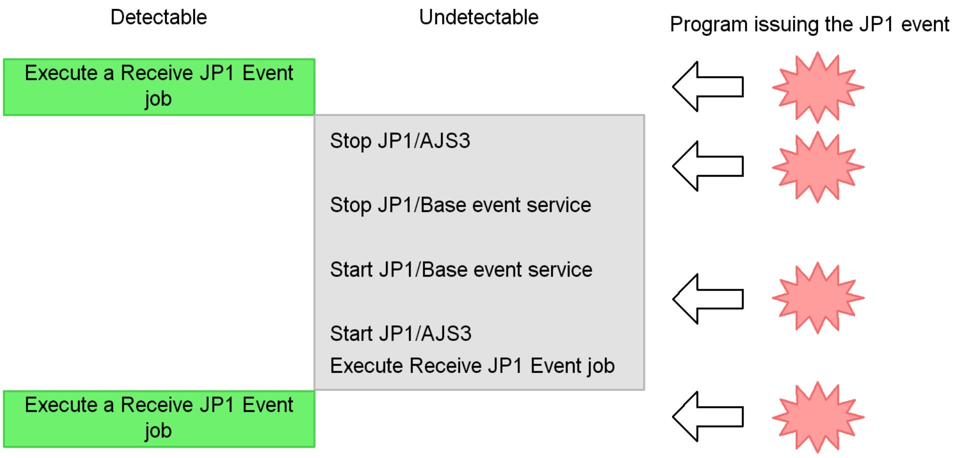
The Receive JP1 Event job can only monitor events that occur after it begins monitoring. Therefore, JP1/AJS3 does not detect the following JP1 events:

- JP1 events that occur while JP1/AJS3 is inactive
- JP1 events that occur between the time when JP1/AJS3 starts and the job begins monitoring

As the JP1 events to be monitored by the Receive JP1 Event job, choose JP1 events that will be issued after the JP1/AJS3 event jobs enter monitoring status.

The following figure shows the timing under which detection cannot take place for a Receive JP1 Event job.

Figure 7-1: Cases where a Receive JP1 Event job cannot detect events


Detectable
Undetectable
Program issuing the JP1 event
Execute a Receive JP1 Event job
Stop JP1/AJS3
Stop JP1/Base event service
Start JP1/Base event service
Start JP1/AJS3
Execute Receive JP1 Event job
Execute a Receive JP1 Event job

(2) Options available with the Receive JP1 Event job

With the Receive JP1 Event job, you can set an option that governs whether to monitor reception of JP1 events that occur before the job starts (and before monitoring of JP1 events starts). This is called the *Find events prior to exec.* option (the option for finding events before starting event monitoring). This option takes effect when you specify an event ID and a search range that dictates how many minutes before the start of the job are to be included in the scope of the Receive JP1 Event job. You can specify a value from 1 to 720 (in minutes). The reference time for this option is based on the local time of the host on which the Receive JP1 Event job is executed.

If you do not use this option, the job only monitors reception of JP1 events that occur after event monitoring starts.

The following cautionary notes apply when this option is used:

- The greater the search range of the Find events prior to exec. option (the number of minutes prior to the Receive JP1 Event job starting that are included in its scope), the longer it takes to search for events that occurred prior to executing the Receive JP1 Event job. The search will also take longer if a large number of JP1 events occurred in the specified search range. We recommend that you specify the smallest possible value (at most 10 minutes) as the search range.
- Take care when using this option with the Receive JP1 Event job in situations where many JP1 events could be found. In this case, CPU usage might increase and delay execution of other jobs, regardless of restrictions to execution counts such as limits to the number of start conditions satisfied per schedule.

  We recommend that you use the Find events prior to exec. option only when you want to find a few specific events. Do not use it when many events will be found or when you want to use constantly occurring events as conditions. If it is likely that many events will be found, you should fine-tune the conditions of the Receive JP1 Event job to reduce the number of target JP1 events or limit the search range.

- We recommend that you use the Find events prior to exec. option from within a start condition job. You can still use this option from within other kinds of event jobs, but if the Find events prior to exec. option is specified in these jobs, the same event might satisfy a condition multiple times. Keep the usage method in mind when using a Receive JP1 Event job as an event job.

  The following shows an example when the same JP1 event satisfies a monitored condition multiple times.

  Example

  Suppose that the Receive JP1 Event job is registered as an event job with the following conditions in the jobnet recv.

  - Start condition:

    The jobnet recv is set to be started by an Interval Control job at 9:00 and 9:10.

  - Search range of Find events prior to exec. option:

    30 (minutes)

  - Event ID:

    `111`

  - Issuance conditions of the JP1 event (event ID: `111`):

    8:20 and 8:50

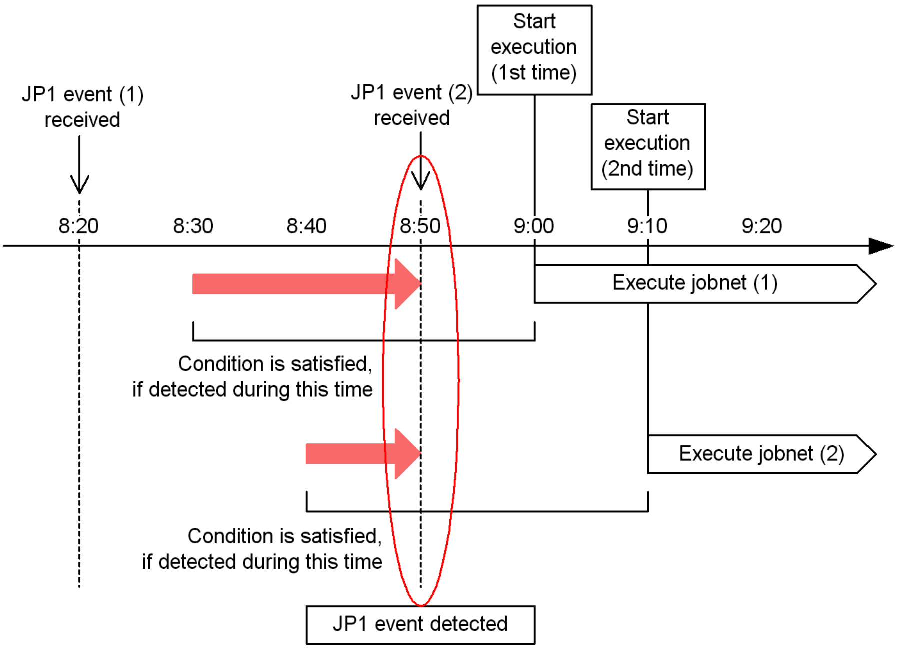
  Operation

  1. The Interval Control job starts the jobnet recv at 9:00.
  2. The Receive JP1 Event job searches for JP1 events that occurred from 8:30 according to the conditions of the Find events prior to exec. option.
  3. The Interval Control job starts the jobnet recv at 9:10.
  4. The Receive JP1 Event job searches for JP1 events that occurred from 8:40 according to the conditions of the Find events prior to exec. option.

  Result

  If the only JP1 event that satisfied the search condition occurred at 8:50, the Receive JP1 Event jobs started at 9:00 and 9:10 both detect the same JP1 event that occurred at 8:50.

Figure 7-2: Example when the same JP1 event is detected more than once


Start execution (1st time)
JP1 event (1) received
JP1 event (2) received
Start execution (2nd time)
8:20
8:30
8:40
8:50
9:00
9:10
9:20
Execute jobnet (1)
Condition is satisfied, if detected during this time
Execute jobnet (2)
Condition is satisfied, if detected during this time
JP1 event detected

(3) Notes on defining the Receive JP1 Event job

The following are cautionary notes on defining the Receive JP1 Event job:

- Start the JP1/Base event service before executing a Receive JP1 Event job, and make sure that the API setting of the JP1/Base event service is `keep-alive`. If the JP1/Base event service is not started, the Receive JP1 Event job waits until the event service has started before executing.
- The Receive JP1 Event job cannot receive JP1 events that occur before it enters *Now running* status. Make sure that the JP1 events you want to receive occur after the Receive JP1 Event job enters this status. Alternatively, you can use the Find events prior to exec. option to find events that occurred prior to the job entering *Now running* status.
- A JP1 event contains messages and detailed information. If you want to monitor this information for a specific character string, you can specify the desired character string by using a *regular expression*. Windows now supports the XPG4 extended regular expressions used in JP1/Base 07-00 and later versions. Because the Receive JP1 Event job, Monitoring Event Log job, and Monitoring Log Files job operate based on the settings of JP1/Base, the type of regular expressions you can use depends on how JP1/Base is configured.

  For details about how to set the available regular expressions, see the explanation about extending the default regular expressions in the *Job Management Partner 1/Base User's Guide*. For details about the regular expressions available in UNIX, see your UNIX documentation. If you overuse the expression `.*`, which matches any character or characters, it could take a long time to compare the expression against the JP1 event. For long messages, use `.*` only where necessary. Executing multiple Receive JP1 Event jobs that use the expression `.*` might exponentially increase the time it takes to compare monitoring conditions for each job against the JP1 event, greatly delaying event detection. For this reason, we recommend that you use the JP1/Base event transfer function and filter conditions on a separate host to reduce the number of JP1 events before monitoring, or add monitoring conditions to reduce the frequency with which `.*` is used in comparisons. Note that in UNIX you can reduce the time it takes to compare the expression against the JP1 event if you use the expression `[^ ]*` (matches repeated characters other than spaces) instead of `.*`.

- In the start condition for a Receive JP1 Event job, you must define one or more items. If no item is defined, the Receive JP1 Event job terminates each time a JP1 event occurs on the host that performs event reception monitoring. Because the monitoring condition is satisfied even for a JP1 event issued by JP1/AJS3, the jobnet starts each time a job is executed.

- When the JP1/Base event service on a Windows host receives JP1 events issued by the event service of JP1/SES Version 5 or JP1/AJS Version 5 or earlier, or by the JP1/SES protocol, originating IP addresses are not set in the JP1 events. As a result, in Windows, even when you specify an IP address for the event source in the monitoring conditions of a Receive JP1 Event job, the type of event described above will not satisfy the monitoring conditions.

- Event details of a JP1 event are monitored only when the details in the JP1 event are in text format. If the details contain binary data, the details in the JP1 event are ignored and do not satisfy the monitoring condition. If the JP1 events to be monitored contain binary data, do not specify event details as a monitoring condition.

- The name of the OS user who issued the JP1 event is set in the JP1 event as the name of the event issuer. If you specify a JP1 user name as the event issuer name in the monitoring condition of a Receive JP1 Event job, the Receive JP1 Event job cannot correctly monitor the reception of JP1 events. User names of event issuers specified in the monitoring condition of a Receive JP1 Event job are case sensitive. Specify the correct user names of event issuers in JP1 events.

- Host names of event issuers specified as the monitoring condition of a Receive JP1 Event job are case sensitive when JP1 event reception monitoring is performed. Specify the correct host names in JP1 events (issuing event server names).

- Partial-string matching is used for the items you specify using a regular expression. If you want to use whole-string matching, you must explicitly specify the whole string by using a regular expression. Windows now supports the XPG4 extended regular expressions used in JP1/Base 07-00 or later. Because the Receive JP1 Event job, Monitoring Event Log job, and Monitoring Log Files job operate based on the settings of JP1/Base, the type of regular expressions you can use depends on how JP1/Base is configured. For details about how to set the available regular expressions, see the explanation of extending the default regular expressions in the *Job Management Partner 1/Base User's Guide*. For details about the regular expressions available in UNIX, see the documentation for UNIX.

- The JP1 event information monitored by the Receive JP1 Event job conforms to the specifications of the JP1/Base event service. For details about the JP1 events you can monitor with the Receive JP1 Event job, see the *Job Management Partner 1/Base User's Guide* and the manual for the product that issued the JP1 event.
- In Windows, for the `users` parameter in the event server settings file (`conf`) of the JP1/Base event service, set the user name of the account from which the JP1/AJS3 service is started. By default, this parameter is set to allow anyone to receive JP1 events.

7.6.2 Notes on the Monitoring Files job

The following provides precautions (items you should know in advance) for using the Monitoring Files job.

Examples of jobnets that use the Monitoring Files job are as follows:

- Monitor the file write time and execute the succeeding job when the file is updated.
- Monitor the files output by applications and the files transferred from other hosts and execute the succeeding job when the files have been created.

The following paragraphs describe the events that are monitored by the Monitoring Files job, how to specify file names, and the options of the Monitoring Files job.

(1) Events monitored by the Monitoring Files job

The following table lists the events that can be monitored.

Table 7-1: Events monitored by the Monitoring Files job

Monitored event	Details
Create#1, #2	Monitors whether a file with the specified name has been created.
Delete#3	Monitors whether a file with the specified name has been deleted.

Monitored event	Details
Change size#2, #4	Monitors whether the size of a file with the specified name has been changed.
Final time write#2, #4	Monitors whether a file with the specified name has been updated. The start condition is satisfied when the update time changes.

#1

If a file with the specified name already exists when monitoring starts, the condition is satisfied when the old file is deleted and a new file is created.

You can use the start monitoring option to specify whether the condition is satisfied if a file with the specified name already exists when monitoring starts. For details about the start monitoring option, see *(3) Options of the Monitoring Files job*.

#2

If the monitoring condition is satisfied, the system performs a *close check* to make sure the monitored file is not being accessed by any process other than the Monitoring files job.

If another process is accessing the file, the condition is judged to be unsatisfied, and another close check is performed when the next monitoring interval occurs. If the file is not being accessed by a process other than the Monitoring files job, the condition is judged to have been satisfied. The close check prevents the job from assuming that the condition is satisfied before the transmission (for example, copying) of a monitored file has been completed.

#3

If a file with the specified name does not exist when monitoring starts, the condition is satisfied when a new file with the specified name is created and then deleted.

#4

If the file with the specified name does not exist when monitoring starts, the condition is satisfied when a new file with the specified name is created and the size of the file or the last write time is changed. Simply creating a file does not satisfy the condition.

You can specify multiple conditions simultaneously. For example, specify **Delete** and **Final time write** if you want to execute the succeeding job when the file is deleted or updated. However, you cannot specify **Change size** and **Final time write** simultaneously.

The following figures show the basic operation of the Monitoring Files job.

(a) Example with "Create" specified as a monitoring condition

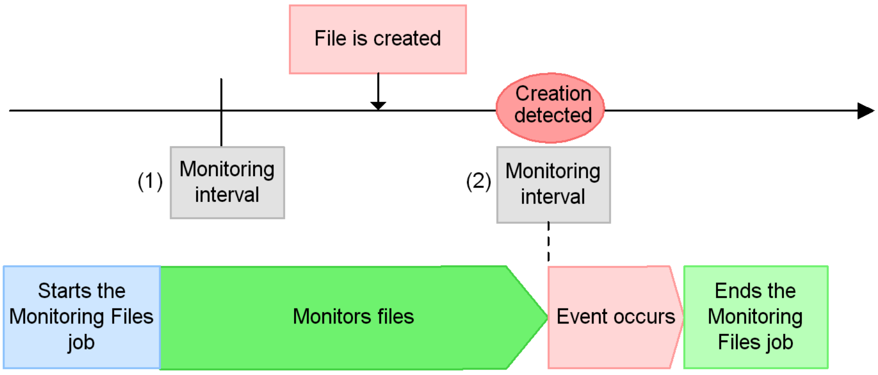
The following figure shows the basic operation of the Monitoring Files job when **Create** is specified as a monitoring condition.

The term *File* in the figure refers to the files being monitored by the Monitoring Files job.

■ When the monitoring target file does not exist at the start of the job

The Monitoring Files job behaves as follows when the monitoring target file does not exist when the job starts.

Figure 7-3: When the monitoring target file does not exist at the start of the job

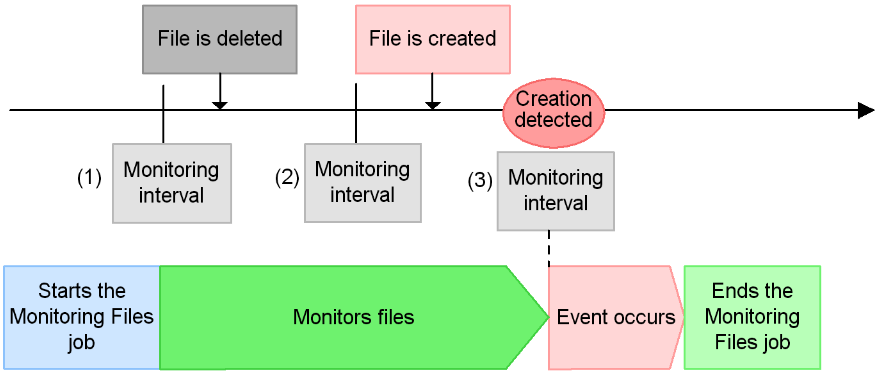


(1): A check on monitoring target files is conducted per monitoring interval. At this time, it is recognized that no monitoring target file exists.
(2): Creation of a monitoring target file is detected, and a file monitoring event occurs.

■ When the monitoring target file exists at the start of the job

The Monitoring Files job behaves as follows when the monitoring target file already exists when the job starts.

Figure 7-4: When the monitoring target file exists at the start of the job



(1): A check on monitoring target files is conducted per monitoring interval. At this time, it is recognized that a monitoring target file exists.
(2): It is recognized that a monitoring target file is deleted.
(3): Creation of a monitoring target file is detected, and a file monitoring event occurs.

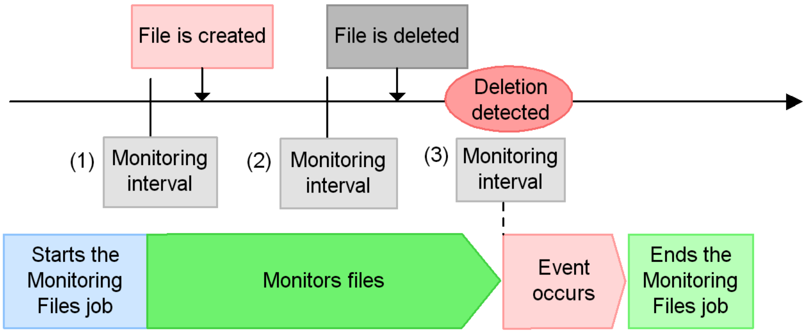
(b) Example with "Delete", "Change size", or "Final time write" specified as a monitoring condition

The following figure shows the basic operation of the Monitoring Files job when **Delete**, **Change size**, or **Final time write** is specified as a monitoring condition. This particular example uses **Delete** as the start condition.

■ When the monitoring target file does not exist at the start of the job

The Monitoring Files job behaves as follows when the monitoring target file does not exist when the job starts.

Figure 7-5: When the monitoring target file does not exist at the start of the job



(1): A check on monitoring target files is conducted per monitoring interval. At this time, it is recognized that a monitoring target file exists.
(2): It is recognized that a monitoring target file is created.
(3): Creation of a monitoring target file is detected, and a file monitoring event occurs.

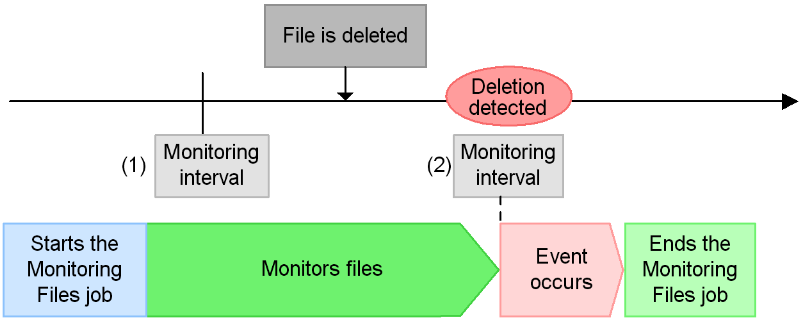
Note:

If the monitoring condition is **Change Size** or **Final time write**, read "File is deleted" above as "Change file size" or "Update file rewriting (Change last entry time)" .

■ When the monitoring target file exists at the start of the job

The Monitoring Files job behaves as follows when the monitoring target file already exists when the job starts.

Figure 7-6: When the monitoring target file exists at the start of the job



(1): A check on monitoring target files is conducted per monitoring interval. At this time, it is recognized that no monitoring target file exists.

(2): Deletion of the monitoring target file is detected, and a file monitoring event occurs.

Note:

If the monitoring condition is **Change Size** or **Final time write**, read "File is deleted" above as "Change file size" or "Update file rewriting (Change last entry time)" .

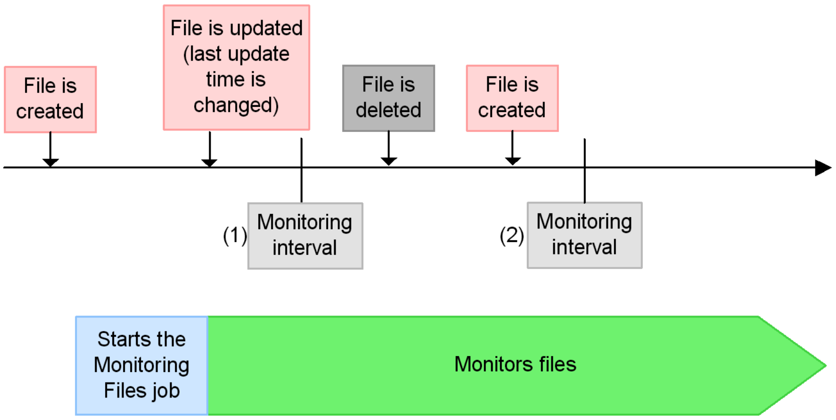
(c) Example when multiple events occur during a monitoring interval (when the monitoring condition is "Create")

The following figure shows the basic operation of a Monitoring Files job that uses the **Create** start condition, in a situation where a file is updated multiple times in one monitoring interval.

■ Example when file creation is not detected

The Monitoring Files job behaves as follows when file creation is not detected.

Figure 7-7: Example when file creation is not detected



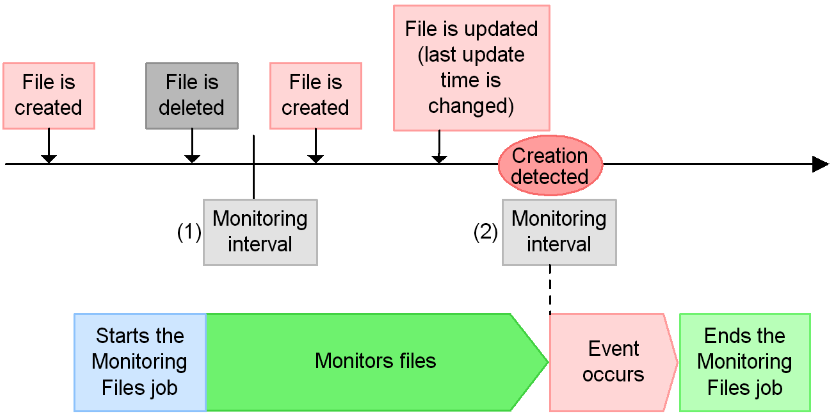
(1): The monitoring target file is created before starting the Monitoring Files job, so that it does not detect the creation of the monitoring target file at this point. However, it is recognized that the last entry time is changed.

(2): Although the monitoring target file is created after the file was deleted, the Monitoring Files job compares it to the status of (1). Therefore it does not detect that the monitoring target file is created, and no file monitoring event occurs.

■ Example when file creation is detected

The Monitoring Files job behaves as follows when file creation is detected.

Figure 7-8: Example when file creation is detected



(1): It is recognized that the monitoring target file created before starting the Monitoring Files job is deleted.

(2): Although the last update time of the monitoring target file is changed after the file was created, the Monitoring Files job compares it to the status of (1). Therefore it detects that the monitoring target file is created, then the file monitoring event occurs.

(2) Specifying file names

To specify a file name, you can use an absolute path or wildcard characters consisting of an absolute path and a wildcard (*). The following table lists examples of using wildcards to specify file names.

Table 7-2: Examples of using wildcards to specify the names of files to be monitored

Example	Files specified	Examples of monitored files
`/jp1/*`	All the files in `/jp1` are monitored, except files whose name begins with a period (.).	`/jp1/aaa` `/jp1/aaa.sh`
`/jp1/.*`	All the files in `/jp1` whose name begins with a period (.) are monitored.	`/jp1/.aaa`

`/jp1/aaa*`	All the files in `/jp1` whose name begins with the character string `aaa` are monitored.	`/jp1/aaa`
		`/jp1/aaabbb`
		`/jp1/aaa.sh`

`/jp1/*aaa`	All the files in `/jp1` whose name ends with the character string `aaa` are monitored.	`/jp1/aaa`
		`/jp1/bbbaaa`
		`/jp1/bbb.aaa`

Example	Files specified	Examples of monitored files
/jp1/aaa*bbb	All the files in /jp1 whose name begins with the character string aaa and ends with the character string bbb are monitored.	/jp1/aaabbb /jp1/aaacccbbb /jp1/aaa.bbb
/jp1/*aaa*	All the files in /jp1 whose name contains the character string aaa are monitored.	/jp1/aaa /jp1/bbbaaa /jp1/bbbaaaccc /jp1/bbbaaa.sh

Note

The file name examples in the above table are for UNIX. When you specify a file name for Windows, it appears in the format c:\jp1\*. An error occurs at execution of the Monitoring Files job in the following cases:

- A wildcard is specified in a directory name.
- A relative path is specified.
- The specified file name already exists as a directory (for example, when you specify file name /jp1/aaa, but directory /jp1/aaa already exists).

(3) Options of the Monitoring Files job

You can specify the following two options for the Monitoring Files job:

- Start monitoring option
- Monitoring Files job status passing option

Details of each option are given below.

(a) Start monitoring option

You can use the *start monitoring option* to specify whether the monitoring condition of the Monitoring Files job is satisfied if the target file already exists when the job starts executing.

The start monitoring option takes effect when you select **Create** as the monitoring condition. If you do not specify this option, the monitoring condition is not satisfied even if the target file exists.

The following examples illustrate how the Monitoring Files job behaves depending on the setting of the start monitoring option.

■ Operation when the start monitoring option is disabled

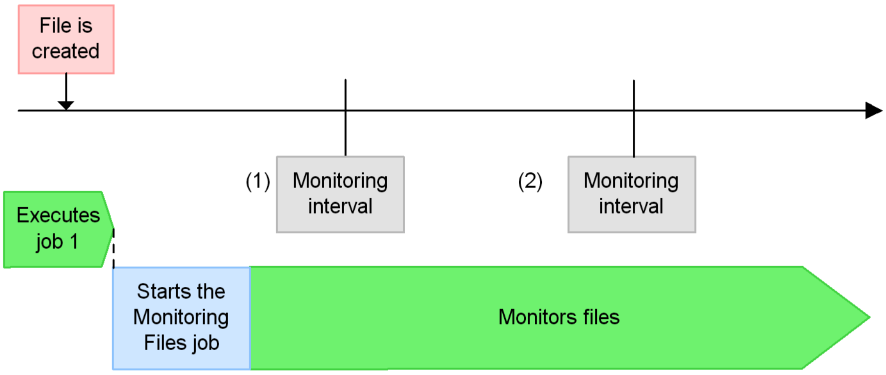
The figure below shows the operation of the Monitoring Files job when the start monitoring option is disabled.

The term *File* in the figure refers to the files being monitored by the Monitoring Files job.

If the monitoring target file is created before the Monitoring Files job is executed

The Monitoring Files job behaves as follows when the monitoring target file is created before the job is executed.

Figure 7-9: Operation when the monitoring target file is created before the Monitoring Files job executes

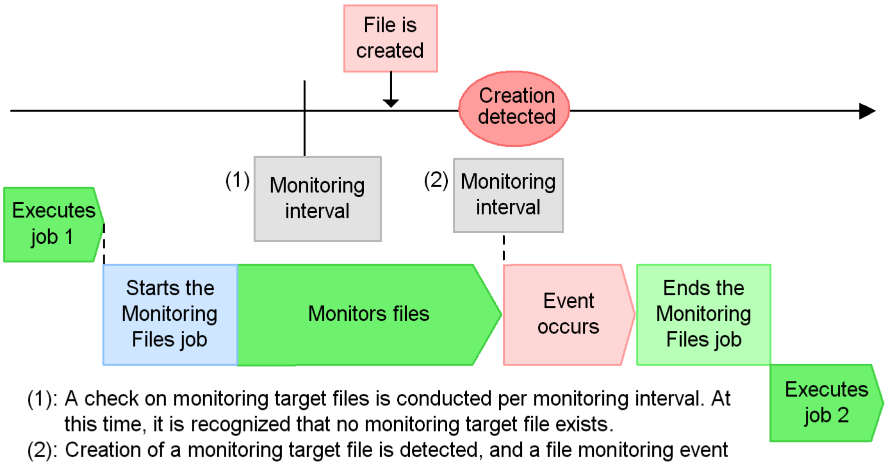


(1): A check on monitoring target files is conducted per monitoring interval. At this time, it is recognized that a monitoring target file exists. Therefore, no file monitoring event occurs.
(2): A check on monitoring target files is conducted per monitoring interval. At this time, it is recognized that a monitoring target file exists. Therefore, no file monitoring event occurs. The file is deleted and monitoring remains until the file is recreated.

If the monitoring target file is created after the Monitoring Files job is executed

The Monitoring Files job behaves as follows when the monitoring target file is created after the job is executed.

Figure 7-10: Operation when the monitoring target file is created after the Monitoring Files job executes



(1): A check on monitoring target files is conducted per monitoring interval. At this time, it is recognized that no monitoring target file exists.
(2): Creation of a monitoring target file is detected, and a file monitoring event occurs.

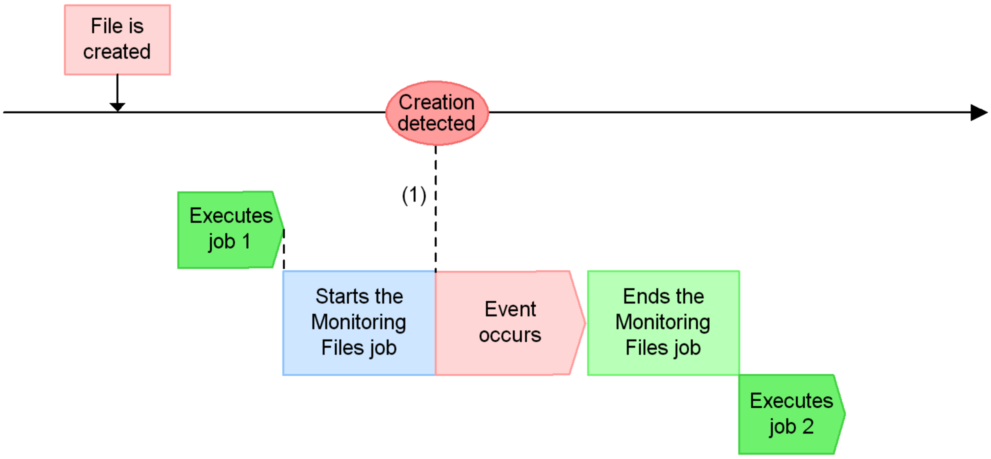
■ Operation when the start monitoring option is enabled

The figure below shows the operation of the Monitoring Files job when the start monitoring option is enabled.

If the monitoring target file is created before the Monitoring Files job is executed

The Monitoring Files job behaves as follows when the monitoring target file is created before the job is executed.

Figure 7-11: Operation when the monitoring target file is created before the Monitoring Files job executes

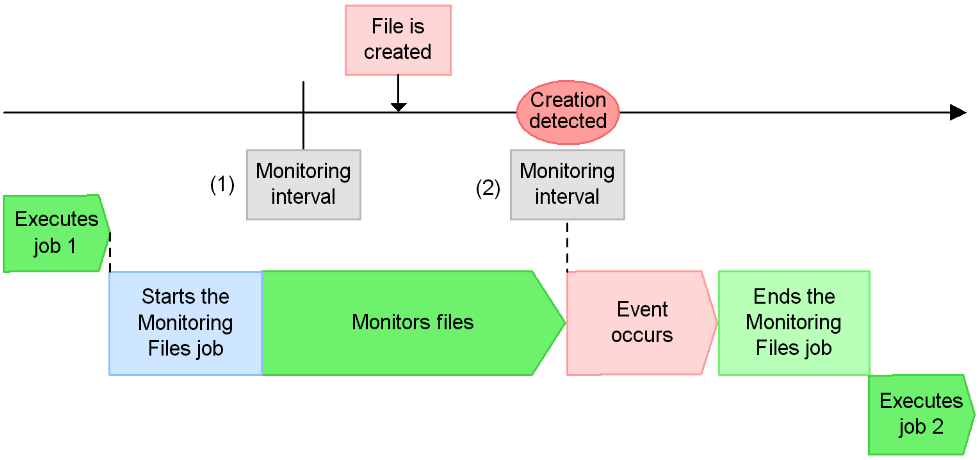


(1): At the start of the Monitoring Files job, a monitoring target file exists. Therefore, the file monitoring event occurs.

If the monitoring target file is created after the Monitoring Files job is executed

The Monitoring Files job behaves as follows when the monitoring target file is created after the job is executed.

Figure 7-12: Operation when the monitoring target file is created after the Monitoring Files job executes



(1): At the start of the Monitoring Files job, no monitoring target file exists. Therefore, check monitoring file per monitoring interval. At this point, it is recognized that no monitoring target file exists.

(2): Creation of a monitoring target file is detected, and a file monitoring event occurs.

See the following for details about how the Monitoring Files job behaves depending on the specification of the start monitoring option.

When **Establish for existing files** is specified

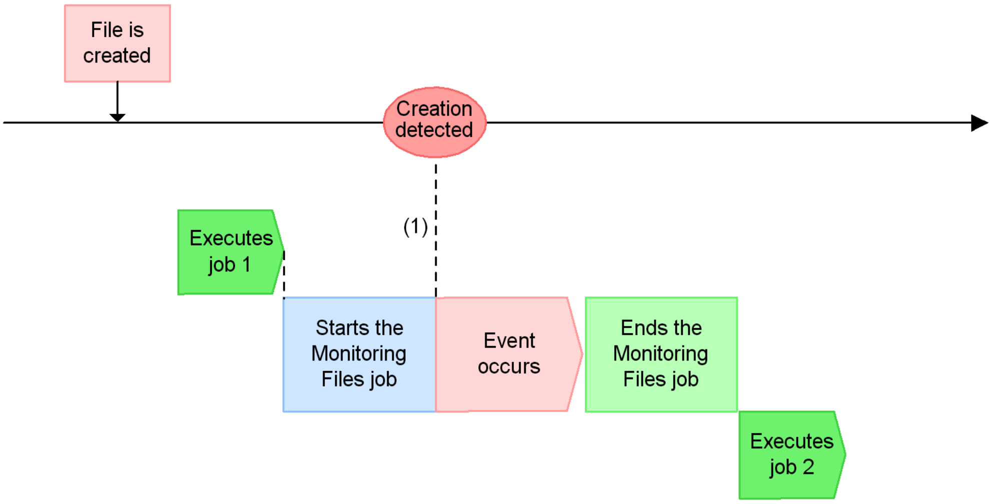
When the Monitoring Files job is executed with **Create** specified as the monitoring condition, if the target file already exists, the monitoring condition is satisfied and the Monitoring Files job terminates normally. A close check ensures that if the file is in use when the Monitoring Files job is executed, the job remains in *Now monitoring* status until the target file is no longer being used.

When the start monitoring option is specified, the monitoring condition is satisfied differently depending on the type of the Monitoring Files job, as follows:

Monitoring Files job in a jobnet with a monitoring target file name specified

If the monitoring target file exists when the Monitoring Files job is executed, the monitoring condition is satisfied by the start monitoring option. The event occurs immediately, and the Monitoring Files job terminates normally.

Figure 7-13: Monitoring Files job in a jobnet with a monitoring target file name specified

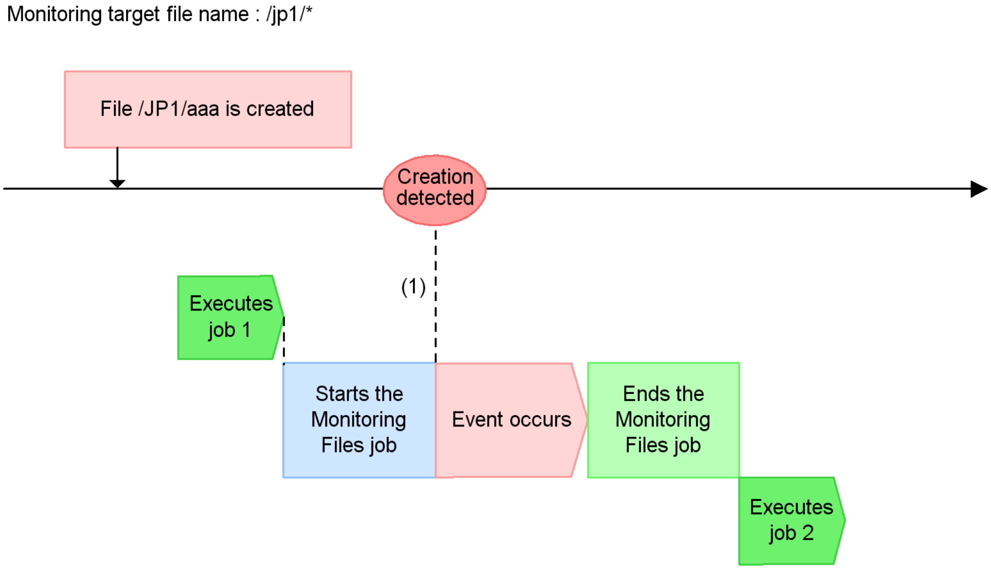


(1): At the start of the Monitoring Files job, a monitoring target file exists. Therefore, the file monitoring event occurs.

Monitoring Files job in a jobnet with a monitoring target file name specified using a wildcard (*):

If at least one file exists in the monitoring target directory when the Monitoring Files job is executed, the monitoring condition is satisfied by the start monitoring option. The event occurs immediately, and the Monitoring Files job terminates normally.

Figure 7-14: Monitoring Files job in a jobnet with a monitoring target file name specified using a wildcard (*)


Monitoring target file name : /jp1/*
File /JP1/aaa is created
Creation detected
(1)
Executes job 1
Starts the Monitoring Files job
Event occurs
Ends the Monitoring Files job
Executes job 2

(1): At the start of the Monitoring Files job, a file exists in the /JP1 directory. Therefore, the file monitoring event relating to the file /jp1/aaa occurs.

Supplementary note:

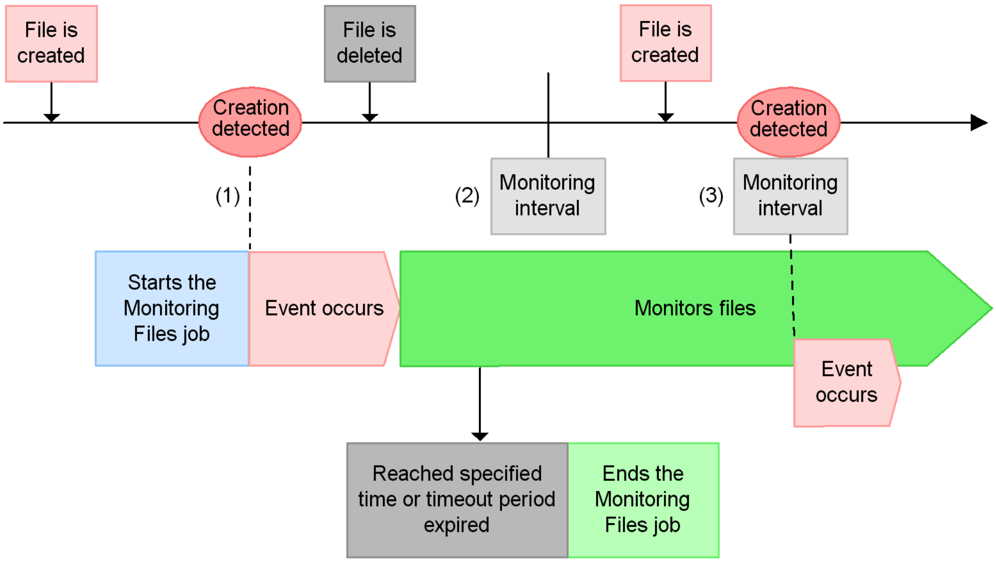
For details about wildcard usage, see *(2) Specifying file names*.

Monitoring Files job in a start condition with a monitoring target file name specified

If the monitoring target file exists at execution of a Monitoring Files job that serves as a start condition, the monitoring condition is satisfied by the start monitoring option, and the event occurs immediately.

The start monitoring option is disabled once an event occurs. After the event occurs, the Monitoring Files job continues monitoring in the *Now monitoring* status. However, after the option is disabled, a condition is satisfied when a new file is created.

Figure 7-15: Monitoring Files job in a start condition with a monitoring target file name specified



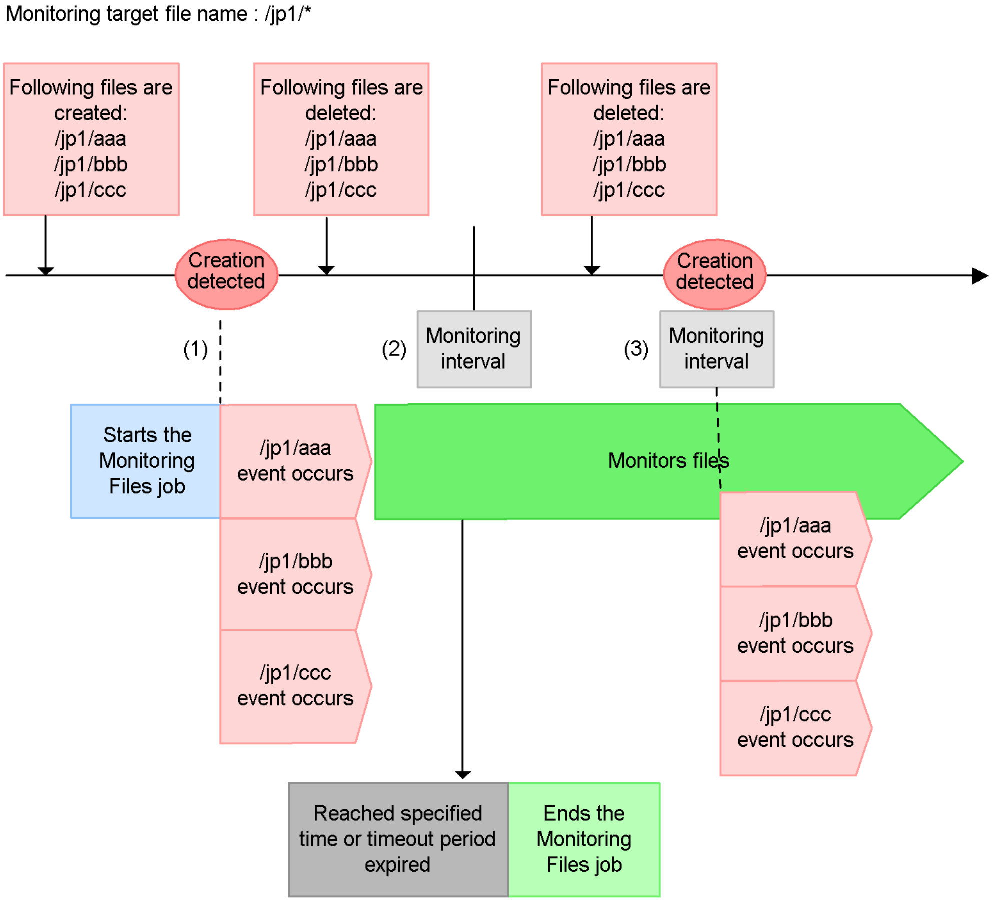
(1): At the start of the Monitoring Files job, the monitoring target file exists. Therefore, the condition for the event occurrence is satisfied and the file monitoring event occurs.
(2): It is recognized that the monitoring target file is deleted.
(3): Creation of a monitoring target file is detected, and a file monitoring event occurs.

Monitoring Files job in a start condition with a monitoring target file name specified by a wildcard (*)

While the Monitoring Files job in the start condition is executed, a monitoring condition is satisfied for any file in the monitoring target directory by the start monitoring option, and an event occurs immediately.

The start monitoring option is disabled after events occur for each of the files. After events occur, the Monitoring Files job continues monitoring in the *Now monitoring* status. However, after the option is disabled, a condition is satisfied when a new file is created.

Figure 7-16: Monitoring Files job in a start condition with a monitoring target file name specified by wildcard (*)


Monitoring target file name : /jp1/*
Following files are created: /jp1/aaa /jp1/bbb /jp1/ccc
Following files are deleted: /jp1/aaa /jp1/bbb /jp1/ccc
Following files are deleted: /jp1/aaa /jp1/bbb /jp1/ccc
Creation detected
Creation detected
(1)
(2)
Monitoring interval
(3)
Monitoring interval
Starts the Monitoring Files job
/jp1/aaa event occurs
/jp1/bbb event occurs
/jp1/ccc event occurs
Monitors files
/jp1/aaa event occurs
/jp1/bbb event occurs
/jp1/ccc event occurs
Reached specified time or timeout period expired
Ends the Monitoring Files job

(1): At the start of the Monitoring Files job, the monitoring target file exists. Therefore, the condition for the event occurrence is satisfied and the file monitoring events occur. All events, /jp1/aaa, /jp1/bbb, and /jp1/ccc, that meet the condition of *(wild card) occur.
(2): It is recognized that the monitoring target file is deleted.
(3): Creation of a monitoring target file is detected, and a file monitoring event occurs. All events, /jp1/aaa, /jp1/bbb, and /jp1/ccc, that meet the condition of *(wild card) occur.

Supplementary note:

For details about wildcard usage, see *(2) Specifying file names*.

When **Establish at file creation** is specified (default)

When a Monitoring Files job with **Create** specified as the monitoring condition is executed, the monitoring condition is not satisfied even if the monitoring target file exists when the job enters *Now running* status. The Monitoring Files job continues monitoring.

When JP1/AJS3 is used in a cluster system, any Monitoring Files job with the start monitoring option specified will be re-executed after failover of the JP1/AJS3 service. If the Monitoring Files job status passing option is enabled, the job retains the same status as before the failover. If this option is disabled, the start monitoring option applies once again.

For details about how to specify the start monitoring option of the Monitoring Files job, see *15.4.16 Detailed Definition - [Monitoring Files] dialog box* in the *Job Management Partner 1/Automatic Job Management System 3 Operator's Guide* or *4.2.10 File monitoring job definition* in the manual *Job Management Partner 1/ Automatic Job Management System 3 Command Reference 2*.

(b) Monitoring Files job status passing option

You can save the information about the file monitoring performed by the Monitoring Files job as needed, and pass the status to a later instance of the job. If you enable the status passing option, a status passing information file is created for each Monitoring Files job.

For example, suppose that the JP1/AJS3 service stops in a cluster system while a Monitoring Files job is running. If the same Monitoring Files job is executed after the JP1/AJS3 service restarts, it inherits the previous monitoring status. A Monitoring Files job can inherit previous information even in a non-cluster system.

The monitoring status can be passed only if the Monitoring Files job continues running. Whether the monitoring status is passed depends on whether the Monitoring Files job runs continuously, or terminates.

The following table shows the conditions for passing the monitoring status.

Table 7-3: Conditions for passing the monitoring status

Type of Monitoring Files job	Restart of JP1/AJS3 service	Failover followed by a stop	Failover due to system going down
Monitoring Files job in a jobnet	Not passed because the job terminates#.	Not passed because the job terminates#.	Passed.
Monitoring Files job in a start condition	Passed.	Passed.	Passed.

#

If a failover that requires the JP1/AJS3 service on the execution agent to be either restarted or stopped occurs, *Table 7-3* applies even if the option to continue execution of active event jobs is enabled. This is because the file monitoring jobs are stopped during the failover. As a result, no events can be detected from the time the JP1/AJS service is stopped on the execution agent on which the file monitoring jobs are running until the service is restarted and event monitoring is resumed.

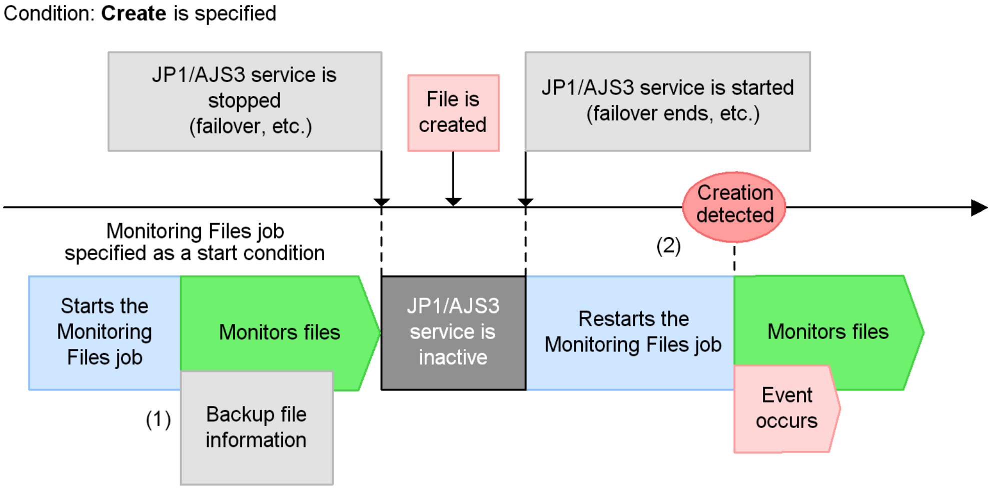
At this time, the messages KAVT2031-E and KAVT2034-W are output to the integrated trace log on the execution agent. If the option to continue execution of active event jobs is enabled, ignore these messages and continue operation. For details about the option to continue execution of active event jobs, see *9.2.1 Continuing the execution of event jobs if the JP1/AJS3 service stops* in the *Job Management Partner 1/Automatic Job Management System 3 Administration Guide*.

The status passing information file is deleted in the following cases:

- The status passing information file created for each Monitoring Files job is deleted when the Monitoring Files job terminates.
- If you disable status passing after status information has been passed, all the status passing information files are deleted.
- If you start the JP1/AJS3 service with the cold option, all the status passing information files are deleted.

The following figure shows an example of operation when the status passing option of the Monitoring Files job is enabled.

Figure 7-17: Example of operation when the status passing option of the Monitoring Files job is enabled


Condition: **Create** is specified
JP1/AJS3 service is stopped (failover, etc.)
File is created
JP1/AJS3 service is started (failover ends, etc.)
Creation detected
Monitoring Files job specified as a start condition
(2)
Starts the Monitoring Files job
Monitors files
JP1/AJS3 service is inactive
Restarts the Monitoring Files job
Monitors files
Event occurs
(1)
Backup file information

(1): When the status passing option is enabled, the information of the monitoring target file held by the file monitoring job is backed up into the passing information storage file.

(2): When JP1/AJS3 services start and execute the Monitoring Files job before the services end, the status passing information storage file is read. (For the condition to pass the status, see *Table 7-3 Conditions for passing the monitoring status*. Therefore, it detects that the monitoring target file is created, then the file monitoring events occur.

The functionality for passing the status of the Monitoring Files job is disabled by default. To enable the functionality, perform the setting procedure for all the hosts and nodes that execute the Monitoring Files job. In a cluster system, both the primary node and the secondary node require the setting. For details about the setting procedure, see *6.3.3 Setting the status passing option for the file monitoring job* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1* (for Windows hosts) or *14.3.3 Setting the status passing option for the file monitoring job* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1* (for UNIX hosts).

Note that if you change the setting for monitoring files that exceed 2 gigabytes (large file support) from *yes* to *no*, and a job inherits status passing information for a file larger than 2 gigabytes, the message KAVT2038-E is output to the integrated trace log and to the execution result details, and the job ends abnormally.

(4) Notes on defining the Monitoring Files job

Note the following when you define the Monitoring Files job:

- When you execute a Monitoring Files job with a monitoring interval of 1 to 9 seconds, succeeding jobs are not executed immediately (in 1 to 9 seconds) when a file update (event) occurs. When you execute many Monitoring Files jobs, it could take some time before succeeding jobs are executed. In such a case, set a monitoring interval that provides sufficient leeway.

To calculate the file monitoring interval, use the expression below. This expression determines the minimum interval required when you use full names to specify the files to be monitored by the Monitoring Files job. We recommend that you set a value greater than the calculated value for real-world use. Note that the lower limit of the monitoring interval differs depending on the hardware you are using.

Expression for calculating the monitoring interval of the Monitoring Files job (lower limit)

> (Monitoring interval) = 0.9 x *number-of-events-per-unit-of-time* (seconds)# / *unit-of-time* (seconds) + 0.02 x *number-of-Monitoring-Files-jobs*

#

> The *number of events per unit of time* (seconds) indicates the number of updates that need to be detected in the monitoring target file by the Monitoring Files job within the specified time (seconds).

For example, if 100 Monitoring Files jobs are executed and 50 file updates occur every 60 seconds, the monitoring interval is calculated as follows:

Monitoring interval = 0.9 x 50/60 + 0.02 x 100 = 2.75 (about 3 seconds)

In this case, we recommend that you set 3 seconds or greater as the monitoring interval.

- When you use wildcard characters containing a wildcard (`*`) to specify the name of the monitoring target file in the Monitoring Files job, and the job is executed in a UNIX environment, the OS has to run a shell to acquire the file name. Therefore, if you execute too many Monitoring Files jobs at once, the CPU usage in the file monitoring process increases.

  If you want to execute a large number of Monitoring Files jobs, use full names to specify monitoring target files, or set a larger monitoring interval if you use wildcard characters.

- The Monitoring Files job can monitor files that do not exceed 2 gigabytes (2,147,483,647 bytes). To monitor larger files, you must set up the environment accordingly. For details about how to perform the settings, see *2.9 Setting up JP1/AJS3 shared information* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 2* (for Windows hosts), or *14.3.12 Enabling monitoring of a large file* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1* (for UNIX hosts).

Note the following when you perform the setting to monitor files larger than 2 gigabytes after upgrading to JP1/AJS3 08-00 or later from an earlier version:

  - When a user program such as a Unix job references or uses the status passing information attribute FLSIZE from a Monitoring Files job, a value that exceeds 2 gigabytes might be passed to the user program. This might cause an overflow in the process that handles file sizes in the user program, and the program will need to be amended or other remedies found.

- If the file being monitored by the Monitoring Files job is also the subject of another event job or log file monitoring by the JP1/Base event service functionality, the target file is considered to be open and the monitoring condition is not satisfied. Therefore, do not set the files as the monitoring target of the Monitoring Files job if they are the target of another event job or subject to log file monitoring by the JP1/Base event service.

- Do not specify as monitoring targets the syslog, device special files, and other system files of the OS. The Monitoring Files job might be unable to identify each of the many unspecified processes accessing the files, processes that access the file frequently, and processes that use the file. Also, do not specify a RAW file as a monitoring target because the Monitoring Files job cannot identify a process that is updating or using such a file.

- When you specify a monitoring target file name in the Monitoring Files job, operation is not guaranteed if you specify a network drive or a file that can be referenced using UNC when you are using Windows, or you specify a file that can be referenced using an NFS mount if you are using UNIX. This is because a network disconnection or other problem might cause access to the file to become unstable, and the Monitoring Files job cannot determine whether the target file is being used in another system. If you want to monitor the files of another system, install JP1/AJS3 - Agent, and monitor them as local files.

- The Monitoring Files job cannot correctly monitor a file that is frequently opened and closed to append data. If such a file is closed when the job performs monitoring, the job might assume that file updating has finished and an event might occur. If you want to monitor such a file, do not use the Monitoring Files job directly. Instead, prepare another file that indicates when file updating has finished, and use the job to monitor the file you prepared.

- Note the following when you are using names containing wildcards (*) to specify the names of files to be monitored:

  - Monitoring Files events occurring within a given monitoring interval cannot be used to monitor files based on the order in which they were created, updated, or deleted. The *event order option* prevents the order of Monitoring Files events from being altered by factors affecting the communication status. It does not change how the Monitoring Files job behaves.

- Note the following regarding satisfying the monitoring condition for the Monitoring Files job:
    - When you specify **Delete** as a monitoring condition and the monitoring target file does not exist when monitoring starts, the monitoring condition is satisfied after the file is created and then deleted.
    - When **Change size** or **Final time write**, is specified as a monitoring condition and the monitoring target file does not exist when monitoring starts, the monitoring condition is satisfied after the file is created, and then the file size is changed or the last write time is changed.

      Change size and **Final time write** monitoring conditions are satisfied after a write is actually made to the target disk. Simply editing a file by using a text editor does not satisfy the monitoring condition.
- If you select the state Do not inherit for the Monitoring Files job status passing option, the information kept by the Monitoring Files job will be lost if the JP1/AJS3 service stops while the Monitoring Files job is active. When you restart the JP1/AJS3 service and begin file monitoring again, the previous file monitoring information will not be inherited.
- The Monitoring Files job checks files at a specified monitoring interval. When an asterisk (`*`) is specified, the job checks all the monitored files. If a file update that matches the monitoring condition takes place more than once for one of the monitored files between monitoring times, the Monitoring Files job detects only the last update. Also, if a single file update is detected by a large number of jobs, the next monitoring time might be delayed because events will be issued for all the jobs. If a file update that matches the monitoring condition takes place more than once before the delayed monitoring time arrives, the Monitoring Files job detects only the last update.

  If you specify a short monitoring interval and there are a large number of jobs to monitor, monitoring might take longer than the monitoring interval.
- When the Monitoring Files job detects that the status of a monitoring target file has changed, the job performs a close check to see if any process has the monitoring target file open. If the Monitoring Files job determines that the file is not open, the job issues an event. If the monitoring target file is open, the job does not issue an event, and enters *Now monitoring* status again. The Monitoring Files job, after the monitoring interval, checks if any process has the monitoring target file open again. In short, as long as the monitoring target file is open, the Monitoring Files job does not issue an event even if it detects that the status of the monitoring target file has changed.
- If you select the state Inherit for the Monitoring Files job status passing option, and execute a saved JP1/AJS3 unit or use the environment under the installation directory for a backup copy of JP1/AJS3, you need to have cold-started the JP1/

AJS3 service. If you execute a Monitoring Files job of the backup JP1/AJS3 without cold-starting the JP1/AJS3 service, monitoring starts from the monitoring status in effect before the unit was saved.

- If an information file or folder is affected by some error that prevents files that store passing statuses from being created or written to, the message KAVT2034-W is output to the integrated trace log, and job execution continues. In this case, the file for inheriting the monitoring status is not created.
- If you select the state Inherit for the Monitoring Files job status passing option, execute a Monitoring Files job in a start condition, and then kill the JP1/AJS3 service, the status of the Monitoring Files job will be passed. However, if the JP1/AJS3 service is subjected to planned termination rather than killed, the status of the Monitoring Files job will not be passed.
- The following directories might contain a file that could be read or written by the file monitoring process at any time while the process is running. Therefore, do not specify a file in the following directories as a monitoring target:

  When running on the physical host

  In Windows Server 2008:

  `%ALLUSERSPROFILE%\HITACHI\JP1\JP1_DEFAULT\JP1AJS2\jp1ajs2\sys` (the default location of `%ALLUSERSPROFILE%` is *system-drive*`\ProgramData`).

  In Windows Server 2003:

  JP1/AJS3-installation-folder`\jp1ajs2\sys`

  In UNIX:

  `/var/opt/jp1ajs2/sys`

  When running on a logical host

  In Windows:

  shared-folder`\jp1ajs2\sys`

  In UNIX:

  shared-directory`/jp1ajs2/sys`
- A Monitoring Files job executed under Windows might detect updates to files other than the expected target files when both of the following conditions are met:

  1. The monitoring target files are specified using a 3-character extension, with a wildcard for the rest of the file name.

     Example:

     Monitoring target files specified as `C:\Temp\*.txt`.

2. The monitoring target folder contains files with extensions that are four characters or longer, and the first three characters match the extension as specified in 1 above.

   Example:

   Suppose that the monitoring target files are specified as `*.txt` (to match the beginning of a string by using a wildcard), and the target folder contains the three files `aaa.txt`, `aaa.txta`, and `aaa.txtab`. When any one of these files is updated, the Monitoring Files job will assume that the monitoring conditions are satisfied. The table below shows examples of how different monitoring target files are selected according to how you specify the file extension.

Table 7-4: Detection of monitoring target files by file extension

Monitoring target file name	Updated files			
	aaa.tx	**aaa.txt**	**aaa.txta**	**aaa.txtab**
`C:\Temp\*.tx`	Y	N	N	N
`C:\Temp\*.txt`	N	Y	Y	Y
`C:\Temp\*.txta`	N	N	Y	N
`C:\Temp\*.txtab`	N	N	N	Y

Legend:

Y: An event is issued when this file is updated.

N: No event is issued when this file is updated.

- When you use a file transfer tool to perform an operation on a monitoring target file such as updating the file, the Monitoring Files job might assume that the **Delete** or **Create** monitoring condition is satisfied, even if the file was only overwritten. That is, if the tool performs file deletion processing internally, and the deletion timing happens to coincide with the monitoring interval, the Monitoring Files job assumes that the file has been deleted. To avoid such problems, we recommend that you use the **Create** and **Final time write** monitoring conditions even if the monitoring target files are always overwritten in your particular setup.
- If you change the Monitoring Files job status passing option from the state Do not inherit to Inherit during monitoring by a Monitoring Files job specified as a start condition, and then restart JP1/AJS3, the messages KAVT2031-E and KAVT2034-W are output to the integrated trace log. This indicates that the status of the job before the restart could not be inherited because the Monitoring Files

job status passing option for the job was set to the state Do not inherit before you restarted JP1/AJS3. For this reason, do not change the option from the state Do not inherit to Inherit while a Monitoring Files job is active, as you will be unable to inherit the status from before JP1/AJS3 restarted.

- When you use a Monitoring Files job in Windows and specify `*.*` as the name of the monitoring target file, the job will also detect files without an extension.

  For example, if you specify `C:\temp\*.*` as the monitoring target file, an event will be issued when any of the following files under `C:\temp` is updated:.

  - `abc`
  - `abc.txt`
  - `abc.txt.txt`

- You cannot specify a symbolic link as a monitoring target file. However, provided that the file itself is not a symbolic link, you can include a symbolically linked directory in the name of the target file.

- Each time the monitoring interval arrives, a Monitoring Files job checks whether the status of the monitoring target file has changed. When a change is detected, a close check is performed regardless of the monitoring condition (**Create**, **Change size**, or **Final time write**) specified in the Monitoring Files job. The check process attempts to open the monitoring target file in write mode and, if successfully opened, the file is immediately closed and a determination is made that no other process has the file open. No other program can access the monitoring target file while it is open for the purposes of a close check. For this reason, when you create a user program that operates a monitored file from within an event job or a jobnet with a start condition, take measures to deal with potential file access problems by incorporating retry processing into the user program.

- The Windows file system uses the following two file name types:

  - Long file names specified arbitrarily by users
  - Corresponding short file names generated automatically by Windows (8.3 filenames)

  The Monitoring Files job in JP1/AJS3 monitors both file name types, and therefore the Monitoring Files job will detect an event for a file whose short file name matches the monitoring target file. In this case, an event might appear to have been detected for the wrong file.

  If the Monitoring Files job appears to have detected an event for the wrong file, run the following command at the command prompt to output the short file names of files in the folder that contains the monitoring target file. Check the command output to see whether the file name specified as the monitoring target file matches the short file name of another file in the folder.

```
dir full-path-of-folder-containing-monitoring-target-file /x
```

To prevent the Monitoring Files job from identifying a short file name as its monitoring target, specify the monitoring target file name as follows:

- When using wildcard characters to specify the monitoring target file, make sure that the base file name is at least 8 characters.

  The following table shows examples of defining the monitoring target file name to include and exclude short file names as monitoring targets.

Table 7-5: Examples of defining the name of the monitoring target file to include and exclude short file names as monitoring targets

Monitoring target file group		**Wildcard characters specified as**
Long file name	**Short file name**	**monitoring target file name**
`C:\temp\ABCDEFGH001.log`	`C:\temp\ABCDEF~1.log`	To exclude short file names as monitoring targets: `C:\temp\ABCDEFGH*.log` The base file name `ABCDEFGH` is 8 or more characters.
`C:\temp\ABCDEFGH002.log`	`C:\temp\ABCDEF~2.log`	To include short file names as monitoring targets: `C:\temp\ABCDEF*.log` The base file name `ABCDEF` is fewer than 8 characters.
`C:\temp\ABCDEFGH003.log`	`C:\temp\ABCDEF~3.log`	
`C:\temp\ABCDXXXXXXX.log`	`C:\temp\ABCDEF~4.log`	

- When using a full name to specify the monitoring target file, do not specify a file name in short file name format.

  Here, a file name in short file name format refers to a long file name in the 8.3 format used by short file names. If you want to monitor files whose long file name conforms to short file name format, make sure that all files in the same path conform to this format.

7.6.3 Notes on the Receive Mail job

The following provides precautions (items you should know in advance) for using the Receive Mail job.

Examples of jobnets that use the Receive Mail job are as follows:

- Execute the succeeding job only when an email is received from the system administrator.
- Execute the succeeding job only when the received email is titled Error.

In email reception monitoring, a condition is satisfied when an email is received. For details about email reception, see *2.5 Defining email reception monitoring job* in the *Job Management Partner 1/Automatic Job Management System 3 Linkage Guide*.

Cautionary note (UNIX only)

To change from an environment linked with a mail system to one that is not, change the parameter `ExecMode` under the `[JP1_DEFAULT#\JP1AOMAGENT\mail_link]` key from `U` to `N` by using the `jajs_config` command, and then restart JP1/AJS3.

#

If JP1/AJS3 is running on a logical host, this will be the logical host name.

7.6.4 Notes on the Monitoring Log Files job

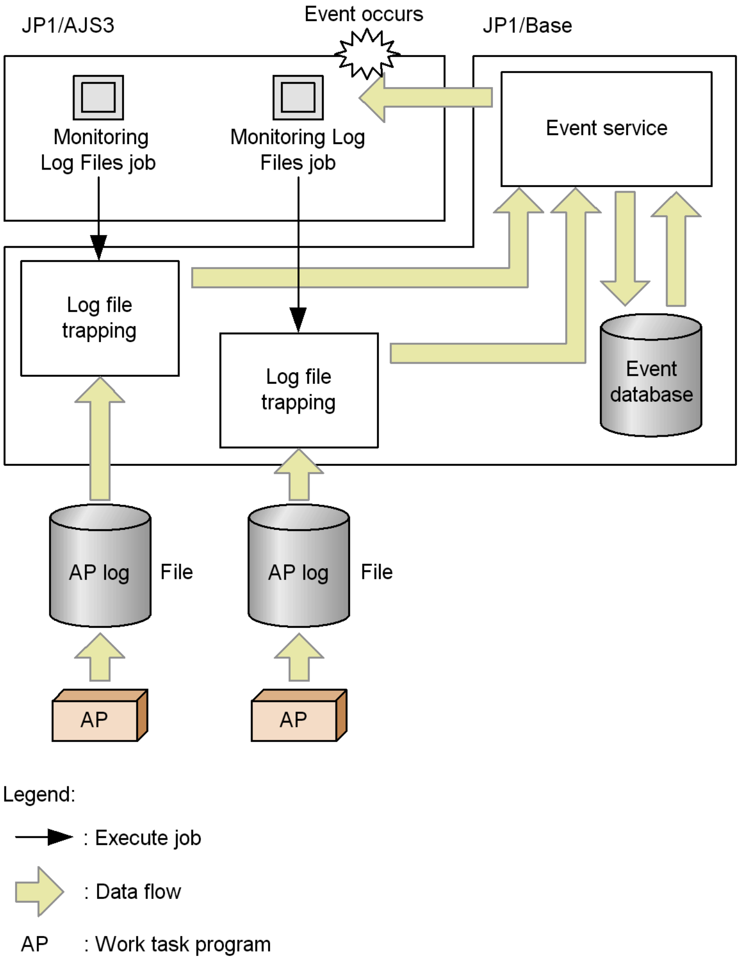
The following provides precautions (items you should know in advance) for using the Monitoring log files job.

Note:

- The Monitoring Log Files job is executed using the log file trap functionality of JP1/Base. Before you execute the Monitoring Log Files job, start the log file trap management service of JP1/Base and the JP1/Base event service. If the log file trap management service of JP1/Base and the JP1/Base event service are not running, the Monitoring Log Files job waits until the services have started before executing. The operating conditions of the Monitoring Log Files job and the files and data that can be monitored depend on the log file trap functionality of JP1/Base. For details about the log file trap functionality, see the *Job Management Partner 1/Base User's Guide*. To execute a Monitoring Log Files job, you must allocate memory, disk space, and system resources for JP1/Base log file trapping. This is required for every Monitoring Log Files job. To estimate the JP1/Base resource requirements, see the *Job Management Partner 1/Base User's Guide*.
- Do not monitor log files that might be mounted or unmounted while a job is executed. If you monitor such files, monitoring might not operate normally, or the system might incorrectly assume that a new log file was created and read the file from the beginning.

The following figure gives an overview of the operation of the Monitoring Log Files job.

Figure 7-18: Overview of the operation of the Monitoring Log Files job



Examples of jobnets that use the Monitoring Log Files job are as follows:

- Execute the succeeding job when log data containing a specific character string is written to the log file.
- Monitor multiple log files and execute the succeeding job when log data containing a specific character string is written to one of the log files.

When you define Monitoring Log Files job, specify the name of the log file to be monitored and the character string to be monitored. You can use a regular expression to specify a character string.

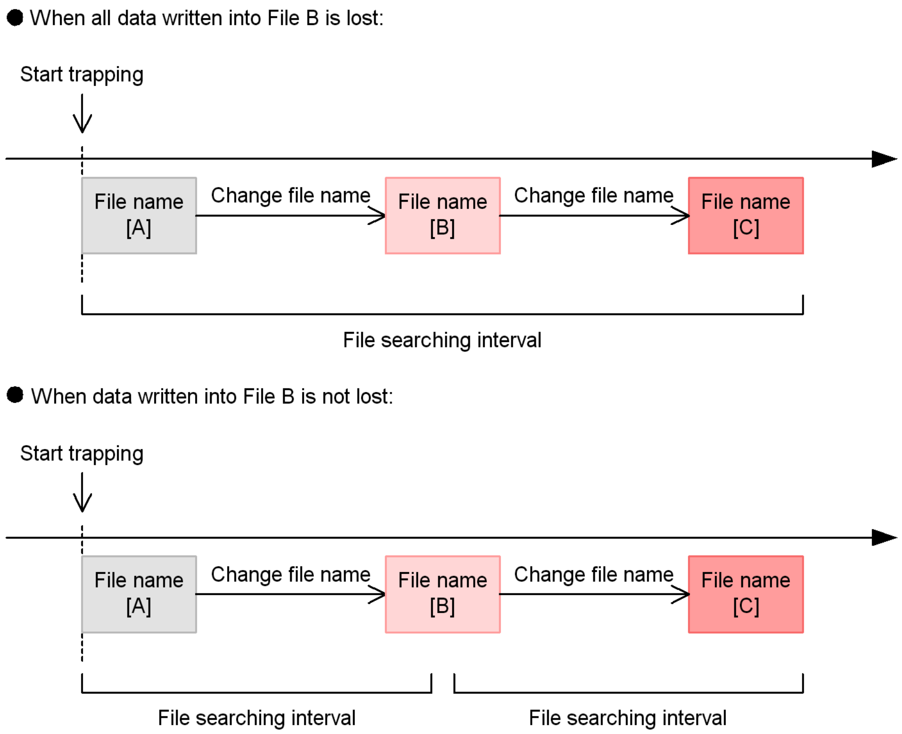
The event job ends and the condition is satisfied when the specified log data is written to the log file and the log data is acquired from the log file.

Cautionary notes

- When **SEQ2** is selected as the output format for log files, if a file is renamed more than once during a file search interval, the acquired messages will be lost. For this reason, take care when specifying the file search interval.

  If you want to edit a file renamed during monitoring with **SEQ2** set, edit it after a new file is created and the specified file search interval elapses.

Figure 7-19: Examples when acquired messages are lost and not lost with SEQ2 selected


● When all data written into File B is lost:

● When data written into File B is not lost:

- If you want to select **SEQ2** as the output format for log files, use one of the environments below that is appropriate for the OS of the host specified as the execution host.

  Windows:

  - JP1/AJS 08-00 or later

- JP1/Base 08-10 or later

UNIX:

- JP1/AJS 07-00 or later

- JP1/Base 07-00 or later

If you execute a job in any other environment, the job is placed in the *Ended abnormally* status.

- If you use version 06-00 to 06-71 of JP1/AJS2 - View to view the detailed definition of a job whose file output format is **SEQ2**, the file output format appears blank. At this time, if you click **OK** in the Detailed Definition dialog box, all the items defined as **SEQ2** are re-defined as **SEQ**.
- If you specify multiple Monitoring Log Files jobs that reference the same log file (including the syslog), the load on the system is increased by the increased number of accesses to the log file. In this case, we recommend that you use the log file trap service of JP1/Base to define the operation of a single log file trap. Then use the JP1/Base JP1 event conversion facility to define a log file update as a JP1 event, and use the Receive JP1 Event job to monitor the created JP1 event.
- Do not attempt to monitor log files on a Windows network drive or UNIX NFS mount. JP1/AJS3 cannot monitor these files in the event of a network disconnection, and cannot determine whether such files are in use on another system.
- Do not attempt to monitor the following files (if you do, monitoring might not work properly):

  - Special files and device files

  - Log files containing records with binary data except at the end character of one line

  - Files whose exact names are not known beforehand
- If you are monitoring the scheduler log and execute `ajsalter -c COPY` during monitoring, monitoring will no longer work properly.
- If the target log file is not found when log file monitoring starts, the job continues to search for the target log file until it is found. Therefore, log data that is written to the log file after monitoring starts but before the file is found could fall outside the scope of the Monitoring Log Files job.
- When you execute a Monitoring Log Files job, you can specify an option that abnormally ends the job if the target file is not found. In this case, if the file is not found when monitoring starts, the Monitoring Log Files job ends without searching for the target file.

- An item specified using a regular expression matches the condition if part of the specified character string matches. To require a full match, use a regular expression that explicitly specifies the full name. For information about the use of regular expressions in Windows, see the *Job Management Partner 1/ Base User's Guide*. For information about regular expressions in UNIX, see your UNIX documentation.
- You might want to execute the Monitoring Log Files job as a start condition for a job that uses an OR condition to trap multiple data items. In this case, if log data written to the log file contains the specified data items, the condition is satisfied multiple times for the same log entry. You can avoid this problem by using the AND, OR, and NOT operators as shown in the following example, so that the second and subsequent items are combined with all the prior items.

Example:

To monitor `Error`, `Warning`, `Information`, and `Notice`, specify:

```
lftpd="Error";
lftpd="Warning":!"Error";
lftpd="Information":!"Error":!"Warning";
lftpd="Notice":!"Error":!"Warning":!"Information";
```

- When you use a relative path to specify a log file, the current directory is assumed as follows.

  In Windows:

  system-folder\`system32`

  In UNIX:

  Directory where the `jajs_spmd` command is executed
- In some cases, when several log file monitoring jobs are executed concurrently in Windows, a JP1/Base error message might be output and the jobs may end abnormally. If this occurs, refer to the *Job Management Partner 1/Base User's Guide* and take the appropriate corrective action for the JP1/Base error message.

7.6.5 Notes on the Monitoring Event Log job

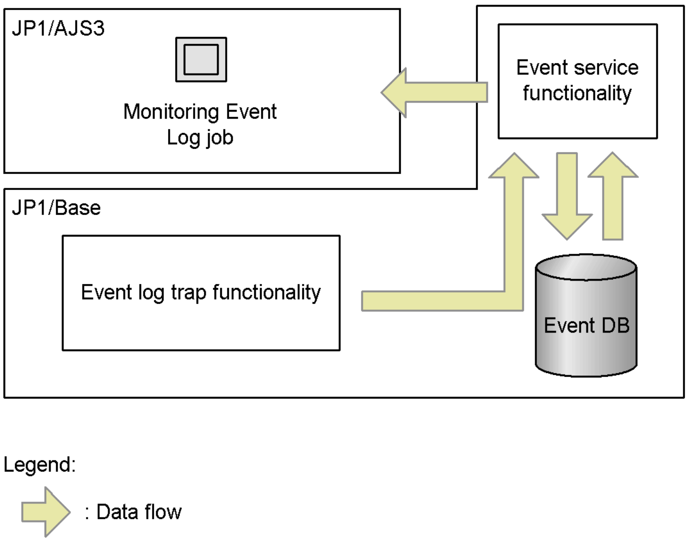
The following provides precautions (items you should know in advance) for using the Monitoring Event Log job.

Note:

> Windows Monitoring Event Log jobs are executed using the JP1/Base event log trapping function. Before executing such a job, start the JP1/Base event log trap service and the JP1/Base event service. If these services are not running, the job remains in *Now monitoring* or *Now running* status until the services start, and does not terminate even when an event occurs in the Windows event log. Also, which Windows events can be monitored depends on the JP1/Base event log trapping function. For details about this function, see the *Job Management Partner 1/Base User's Guide*.

The following figure gives an overview of the operation of the Monitoring Event Log job.

Figure 7-20: Overview of Monitoring Event Log job



Examples of jobnets that use the Monitoring Event Log job are as follows:

- Execute a jobnet when a Windows event reporting the start of an application is output to the Windows event log.
- Execute a jobnet when a Windows event reporting a successful authentication by the security system is output to the Windows event log.

When you define a Monitoring Event Log job, specify the type of log to be monitored, and the type of event. For the type of log to monitor, select system logs, security logs, application logs, DNS Server logs, Directory service logs, or File Replication Service logs. For the type of event, select information, warning, error, failed audit, and successful audit.

The event job ends and the condition is satisfied when the specified Windows event is output and data is acquired from the Windows event log.

Cautionary notes

- When a Windows event cannot be placed in a category, "Others" appears in the Windows event viewer. However, if the event is converted to a JP1 event by using the JP1/Base event log trapping function, the category of the event will be "None". For this reason, when you define a Monitoring Event Log job, specify "None" rather than "Others" for **Category**. If you specify the string "Others", the monitoring condition will not be satisfied.
- In its default state, the JP1/Base event log trapping function monitors **Error** and **Warning** events only. To monitor **Information**, **Failure audit**, and **Success audit** events, add the definitions for the events to be monitored to the operation definition file for JP1/Base event log trapping.
- The monitoring condition is satisfied when the monitoring condition specified in **Category** of the Monitoring Event Log job completely matches the category of the Windows event. The maximum size of a monitoring condition is 255 bytes. However, the Windows event log might contain entries longer than 255 bytes. In this case, the system compares the character string specified in **Category** of the Monitoring Event Log job against the first 255 bytes of the category information of the Windows event, and the condition is satisfied if they match.
- An item specified using a regular expression matches the condition if part of the specified character string matches. To require a full match, use a regular expression that explicitly specifies the full name. For information about the use of regular expressions in Windows, see the *Job Management Partner 1/Base User's Guide*.
- When you specify a character string that includes a linefeed character in the Description definition item of the Monitoring Event Log job, the condition is satisfied if the character string before the linefeed character matches the monitoring condition, regardless of the character string following the linefeed character. To allow descriptions that contain linefeed characters to be monitored successfully, use regular expressions for the linefeed code, for example, use `\n` if the linefeed code is `\n` and use `.*\n` if the linefeed code is `\r\n`. For information about the use of regular expressions in Windows, see the *Job Management Partner 1/Base User's Guide*.

Example when the following character strings are monitored:

```
Starting TEST1...
Starting TEST2...
```

- If the linefeed character after `Starting TEST1` is `\n`, specify:

```
Starting TEST1\nStarting TEST2
```

- If the linefeed character after `Starting TEST1` is `\r\n`, specify:

```
Starting TEST1.*\nStarting TEST2
```

7.6.6 Notes on the Interval Control job

The following provides precautions (items you should know in advance) for using the Interval Control job.

Example of jobnets that use the Interval Control job are as follows:

- Register a jobnet for execution, and execute the succeeding job 30 minutes later.

  For details about how to register a jobnet for execution, see *4. Executing an Application* in the manual *Job Management Partner 1/Automatic Job Management System 3 Overview*.

As the wait time for the Interval Control job, you can specify a value from 0 to 1,440 (in minutes).

Cautionary note

> The wait time you specify in the Interval Control job is the wait time for the interval control process, not the time from the start to end of the job execution. Depending on such factors as the status of the network at the time, the wait time might not exactly match the time you specify.

7.6.7 Notes on defining passing information

Note the following when defining passing information:

- For details about the characters you can use with JP1/AJS3, see *2.4.3 Language type of the system* in the *Job Management Partner 1/Automatic Job Management System 3 System Design (Configuration) Guide*. Characters that are not supported by JP1/AJS3 cannot be used even in a stand-alone configuration. Make sure that passing information does not include such characters.
- If multiple event jobs are related to a single succeeding job, the succeeding job can inherit the information received by all the event jobs. However, if multiple event jobs define the same macro variables, the older received information will be overwritten.
- If the same macro variable name is defined more than once in the passing information of a single event job, the information that is defined first is passed.

Example for the Receive JP1 Event job:

Suppose that the following two macro variables are defined in passing information:

- `?AJS2111?:EVID` (specifies that macro variable `?AJS2111?` inherit an event ID)
- `?AJS2111?:EVMSG` (specifies that macro variable `?AJS2111?` inherit message information)

In this case, `?AJS2111?` inherits an event ID.

- Do not pass data that contains an escape sequence to the command line. If you pass data that contains a space character, unexpected behavior might result. To prevent this, when you define a macro variable, enclose it in double quotation marks (`"`).
- When you specify a macro variable in the command line of the succeeding job, the succeeding job cannot correctly inherit passing information if the information contains space characters or single quotation marks (`'`). Note that macro variables are executed on the agent host that executes the succeeding job of the event job by replacing the macro variable in the command line with its value. When defining passing information, define only information that can be treated as command arguments at execution.
- If there is no information to be inherited from the event job or if the event job is not executed, the macro variables defined in the succeeding job will not inherit any information. In this case, if you define `?AJS2111?` as the macro variable name when you execute the job, the character string `?AJS2111?` will be passed.
- When you pass the information received in an event job to the parameters of a standard job or an action job, if the data to be passed contains a double quotation mark ("), you must add a backslash (\) before the double quotation mark. If you do not add a \ before the double quotation mark, the double quotation mark will be ignored when it is passed to a standard job or an action job.

  To prevent this, enable the option for handling data containing double quotation marks (as is) in passing information.

  For details about how to set this option, see *6.3.4 Passing event data containing double quotation marks* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1* (for Windows systems) or *14.3.4 Passing event data containing double quotation marks* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1* (for UNIX systems).
- When passing information is used in a command line of a job, even if the passing information contains a double quotation mark (`"`), the information is passed to the succeeding job without being converted. Therefore, the job might not be executed

correctly depending on the command line restrictions of the OS in question. If you want to use passing information that contains a special character, do not use it directly in a command line. Such information must be passed to an environment variable.

- If you define a macro variable in an event in a start condition, information is passed to the entire jobnet that starts when the start condition is established.
- Make sure that the macro variable names and passing information together do not exceed 4,096 bytes. In particular, if you use the AND condition to define a start condition, the system merges the macro variable names and passing information for all of the event jobs in the start condition. For this reason, take care that the macro variable names and passing information do not exceed 4,096 bytes in total.

7.6.8 Notes on restarting the JP1/AJS3 service while event jobs are running

If you perform any of the operations listed below while event jobs (including those with a start condition) are running, communication and correlation processing take place between the schedule control and event/action control functions to ensure consistency in job statuses between the functions. If a large number of event jobs are in *Now running* status, the amount of data to be processed places a considerable load on the system.

Operations that result in high system load

- Stopping the scheduler service, and then performing a warm or hot start.
- Stopping the JP1/AJS3 service on the manager host, and then performing a warm or hot start.
- Stopping and then restarting the JP1/AJS3 service on the agent host.
- Executing the `jajs_maintain` command.

Use the following workarounds to avoid placing a high load on the system:

Workarounds

- Before you perform any of the above operations, forcibly terminate any jobnets that use event jobs. Register them for execution again when the operation is complete.
- Before you perform any of the above operations, forcibly terminate all event jobs. Re-execute them when the operation is complete.
- Do not run any jobs until 30 minutes to an hour has elapsed after performing the operation.
- Do not trigger any monitored events until 30 minutes to an hour has elapsed after performing the operation.

The following problems can occur when a high load is placed on the system.

Problems resulting from a high system load

1. Event jobs (including those with a start condition) registered for execution immediately after you performed one of the above operations take a long time to enter *Now running* status.#1
2. When you forcibly terminate an event job in *Now running* status, or a jobnet with a start condition in *Now monitoring* status, it takes a long time to terminate.#1
3. When you change the status of an event job in *Now running* status, the change takes a long time to take effect.#1
4. It takes a long time for an event to be detected when a monitoring condition is satisfied.#1
5. Event jobs (including those with a start condition) registered for execution immediately after you performed one of the above operations remain in *Queuing* status.#2
6. You attempt to forcibly terminate an event job in *Now running* status, or a jobnet with a start condition in *Now monitoring* status, but it does not terminate.#2
7. A monitoring condition is satisfied but no event is detected.#2

#1

Problems 1 through 4 can occur when you perform an operation that results in a high system load while the event/action control manager is near its resource limits. For details about the resource limits that apply to event/action control, see *B(7) Limits for the event/action control* in the *Job Management Partner 1/Automatic Job Management System 3 System Design (Configuration) Guide*.

#2

Problems 5 through 7 can occur when you perform an operation that results in a high system load after the event/action control manager has exceeded its resource limits. For details about the resource limits that apply to event/action control, see *B(7) Limits for the event/action control* in the *Job Management Partner 1/Automatic Job Management System 3 System Design (Configuration) Guide*.

If you perform an operation that results in a high system load when a large number of event jobs are running, a large volume of communication takes place between the scheduler control and event/action control functions. This results in an increased number of unreported items of information that are generated and managed in the event of a communication error. JP1/AJS3 imposes a limit on the number of unreported item that can be kept, to prevent the high system load that results from processing this data from monopolizing system resources and delaying job execution and event detection. When the number of unreported items reaches the limit, the information is deleted starting with the oldest item. Any of problems 5 through 7 can occur as a result, depending on the content of the deleted data.

The upper limit of the number of unreported items of information that can be kept is not disclosed to JP1/AJS3 users. Instead, the limit for event/action control is calculated from the number of unreported items generated by an operation that results in a high system load. When using JP1/AJS3, be careful not to exceed this limit.

If any of the problems mentioned above occurs, take the following remedial action:

Recovery procedure

For problems 1 to 4

Wait until the processing finishes. This can take 30 minutes to an hour, depending on the number of jobs affected.

For problem 5

Forcibly terminate the event job or jobnet in question, and then re-register it for execution.

For problem 6

For an event job, change the status of the job and terminate it. For a jobnet with a start condition, forcibly terminate the jobnet again.

For problem 7

Forcibly terminate the event job or jobnet in question, and then re-register it for execution. Then, generate the event again.

7.6.9 Monitoring events or messages issued by JP1/AJS3

When monitoring JP1/AJS3 messages, monitor the message ID and the first part of the message, rather than the entire message text. If you monitor the entire message text, JP1/AJS3 could fail to detect the target message if it has extra information added at the end.

You might want to set a JP1/AJS3 event job to monitor JP1 events or Windows events issued by JP1/AJS3, or to monitor the messages output to log files. However, such an event job might be unable to detect its monitoring targets as events depending on the status of JP1/AJS3, or might be executed repeatedly depending on the monitoring condition settings. For example, the following can occur:

- Like a normal job, an event job is executed as a job in a jobnet. Monitoring of events does not commence until both JP1/AJS3 and the jobnet have started. Because the jobnet-initiating event is issued after the first job starts, an event job cannot monitor a jobnet-initiating event of JP1/AJS3 which is executing the event job.
- If a problem occurs in JP1/AJS3 and the jobnet in which that event job is defined ends abnormally, the event job itself also ends abnormally. If this situation occurs, events cannot be monitored.
- An event job also issues JP1 events or Windows events, or outputs messages to log files, as part of its job processing. Because the event job monitors the JP1 events, Windows events, and messages that the event job itself outputs to the log files, monitoring continues in an infinite loop.

When you want to execute actions by using events from JP1/AJS3 or messages in log files as triggers, you can avoid these problems by using use the automatic action functionality of JP1/IM. Alternatively, you could use a JP1/AJS3 manager that is different from the one executing the event job. The preventive measures when you want to execute commands by using events from JP1/AJS3 or messages in log files as triggers are described below for two cases: when there are multiple JP1/AJS3 managers (including a configuration where multiple instances of JP1/AJS3 are concurrently running on multiple logical hosts) and when there is only one JP1/AJS3 manager (only one instance of JP1/AJS3). The following also gives notes on monitoring events and messages.

Note that the description in this section applies to monitoring targets such as JP1 events issued by JP1/AJS3 itself, Windows events, and messages output to log files or the syslog. The description does not apply when the Send JP1 Event job is used or when JP1 events are sent as a user job.

(1) When there are multiple JP1/AJS3 managers (including a configuration where multiple instances of JP1/AJS3 are concurrently running on multiple logical hosts)

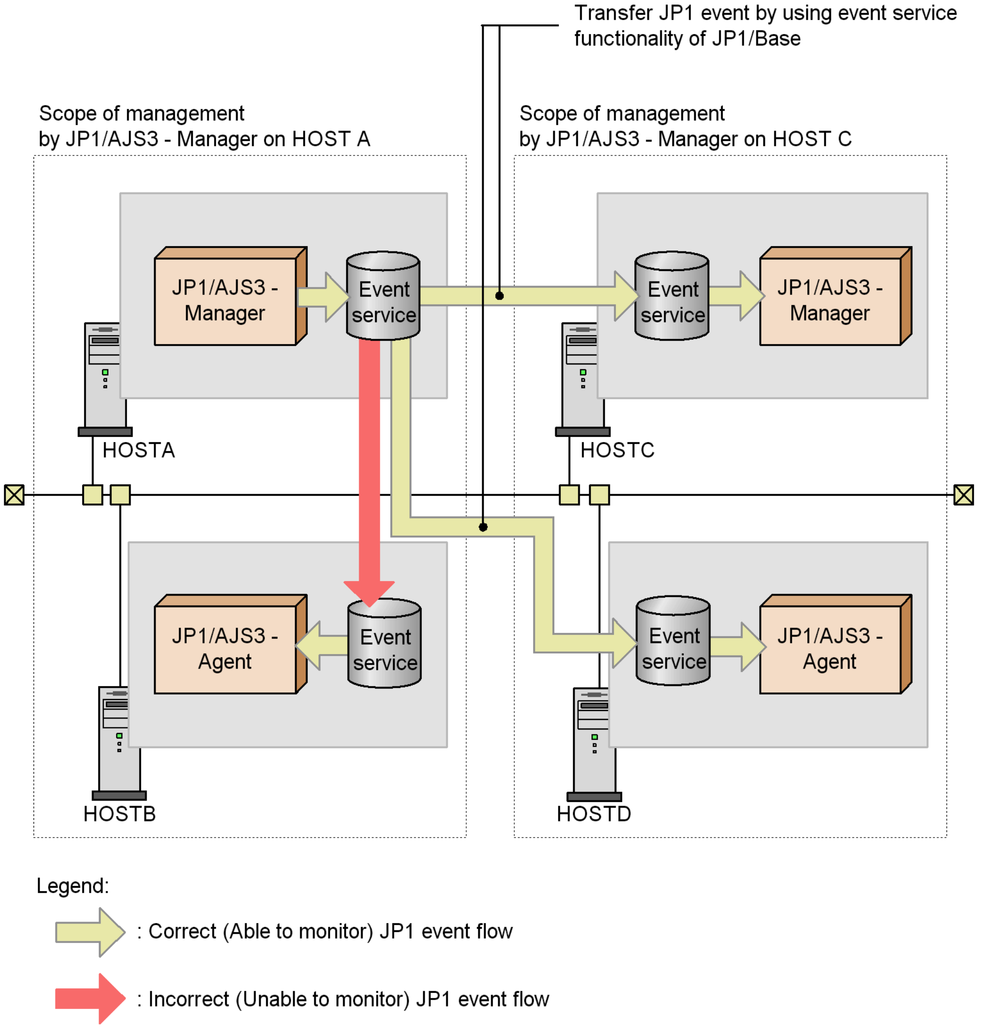
The following describes the measures to take to monitor events issued by JP1/AJS3 on another logical host or messages in log files when there are multiple JP1/AJS3 managers (or when multiple instances of JP1/AJS3 are concurrently running on multiple logical hosts). In this case, JP1/IM is not needed.

(a) Monitoring JP1 events issued by JP1/AJS3 on another logical host, or when there are multiple JP1/AJS3 managers

Under normal circumstances, you cannot monitor JP1 events issued by an instance of JP1/AJS3 running on another host (logical host). However, you can monitor such events if you transfer them using the event service functionality of JP1/Base. To do this, transfer JP1 events to an event service on another host (logical host) that is not managed by the JP1/AJS3 that issued the JP1 events. You cannot monitor the events if you transfer them to the event service running on the host managed by the JP1/AJS3 that registered the JP1 events.

The following figure shows an example flow of JP1 events.

Figure 7-21: Flow of JP1 events


Transfer JP1 event by using event service functionality of JP1/Base
Scope of management by JP1/AJS3 - Manager on HOST A
Scope of management by JP1/AJS3 - Manager on HOST C
JP1/AJS3 - Manager
Event service
Event service
JP1/AJS3 - Manager
HOSTA
HOSTC
JP1/AJS3 - Agent
Event service
Event service
JP1/AJS3 - Agent
HOSTB
HOSTD
Legend:
: Correct (Able to monitor) JP1 event flow
: Incorrect (Unable to monitor) JP1 event flow

In the above example, if you want to monitor the JP1 events issued by JP1/AJS3 on HOSTA, transfer them to the event service on HOSTC or HOSTD. You cannot monitor the events if you transfer them to the event service on HOSTB.

(b) Monitoring Windows events issued by JP1/AJS3 on another logical host

Windows events can only be monitored on logical hosts that use the JP1/Base event log trapping function.

When you want to monitor Windows events issued by JP1/AJS3, set up a separate logical host and monitor the events from a different manager. In this case, you cannot use the Monitoring Event Log job from the monitored JP1/AJS3.

(c) Monitoring log file messages output by JP1/AJS3 on another logical host

Reference the monitored log files from JP1/AJS3 on another logical host.

(2) When there is only one JP1/AJS3 manager (only one instance of JP1/AJS3)

The following describes the measures to take to monitor events issued by JP1/AJS3 or messages in log files when there is only one JP1/AJS3 manager. In this case, JP1/IM is needed.

(a) Monitoring JP1 events issued by JP1/AJS3

Use the automatic action functionality of JP1/IM.

(b) Monitoring Windows events issued by JP1/AJS3

Use the Windows event trapping functionality of JP1/Base and the automatic action functionality of JP1/IM.

(c) Monitoring log file messages output by JP1/AJS3

Use the log file trapping functionality of JP1/Base and the automatic action functionality of JP1/IM.

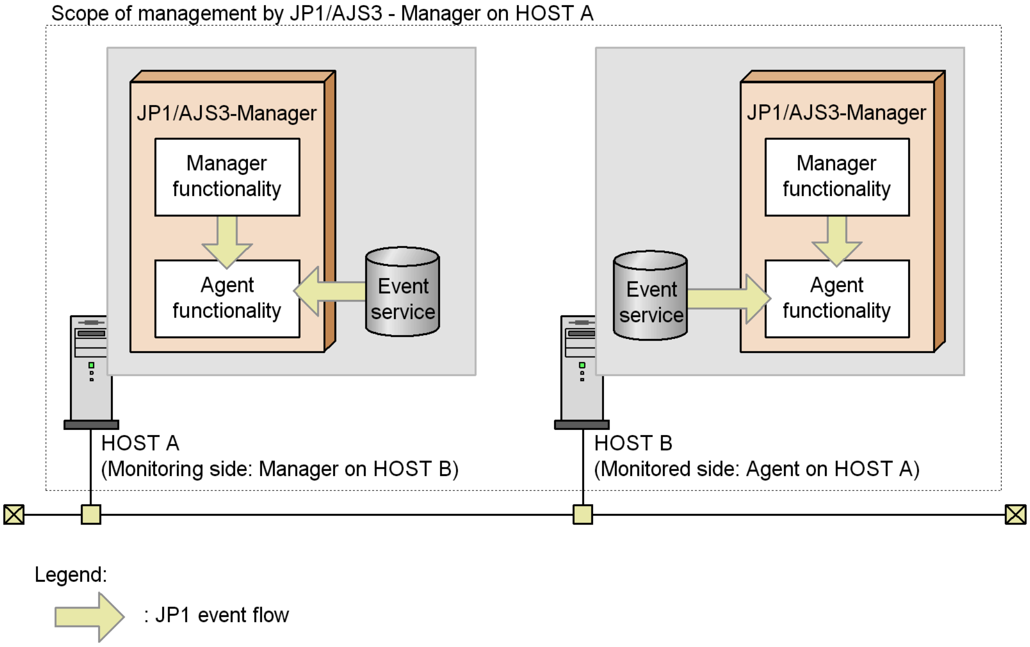
(3) Notes on monitoring events and messages

The following provides precautions for monitoring events or messages.

(a) Restrictions when using an event job to monitor a JP1/AJS3 - Manager host from another JP1/AJS3 - Manager host

The following describes the restrictions that apply when HOSTB is monitored from HOSTA by using an event job as shown in the figure below. In this case, JP1/AJS - Manager host (HOSTA) is set as the manager of HOSTB (monitoring host), and another JP1/AJS3 -Manager host (HOSTB) is set as the agent of HOSTA (monitored host).

Figure 7-22: Monitoring a JP1/AJS3 - Manager host from another JP1/AJS3 - Manager host by using an event job (1)



Restrictions

- When JP1/AJS3 is not running on HOSTB, you cannot use an event job to monitor HOSTB.

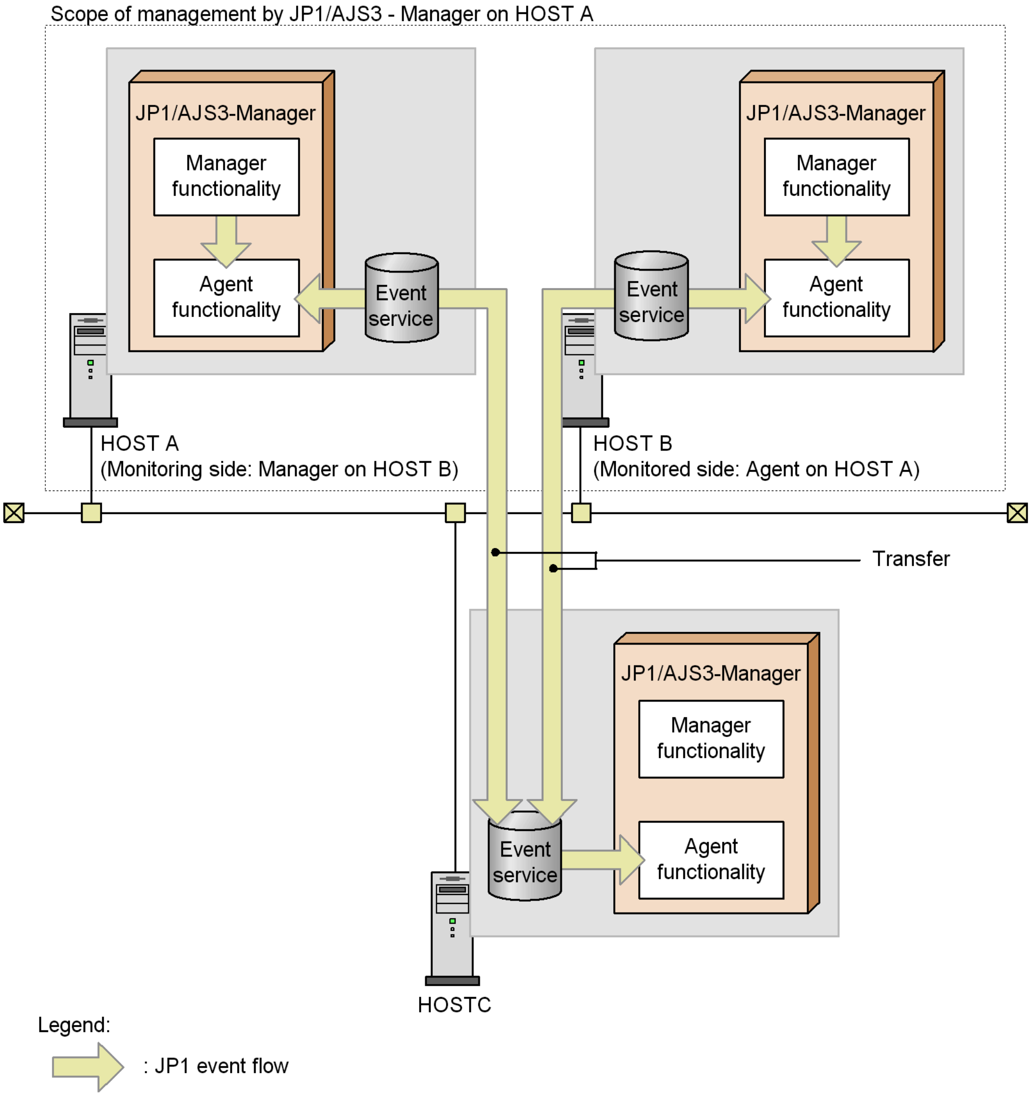
  The manager provides the agent functionality. Therefore, the agent of HOSTB monitored by HOSTA is started or stopped in tandem with the manager of HOSTB starting or stopping.

- If JP1/AJS3 on HOSTB terminates abnormally, it cannot provide information to HOSTA (you cannot monitor HOSTB from HOSTA).

  The manager provides the agent functionality. Therefore, the agent of HOSTB monitored by HOSTA stops in tandem with the manager of HOSTB ending abnormally.

To monitor a host in this configuration, transfer events to a JP1/AJS3 - Manager host that is neither the monitoring JP1/AJS3 - Manager host nor the monitored host.

Figure 7-23: Monitoring a JP1/AJS3 - Manager host from another JP1/AJS3 - Manager host by using an event job (2)

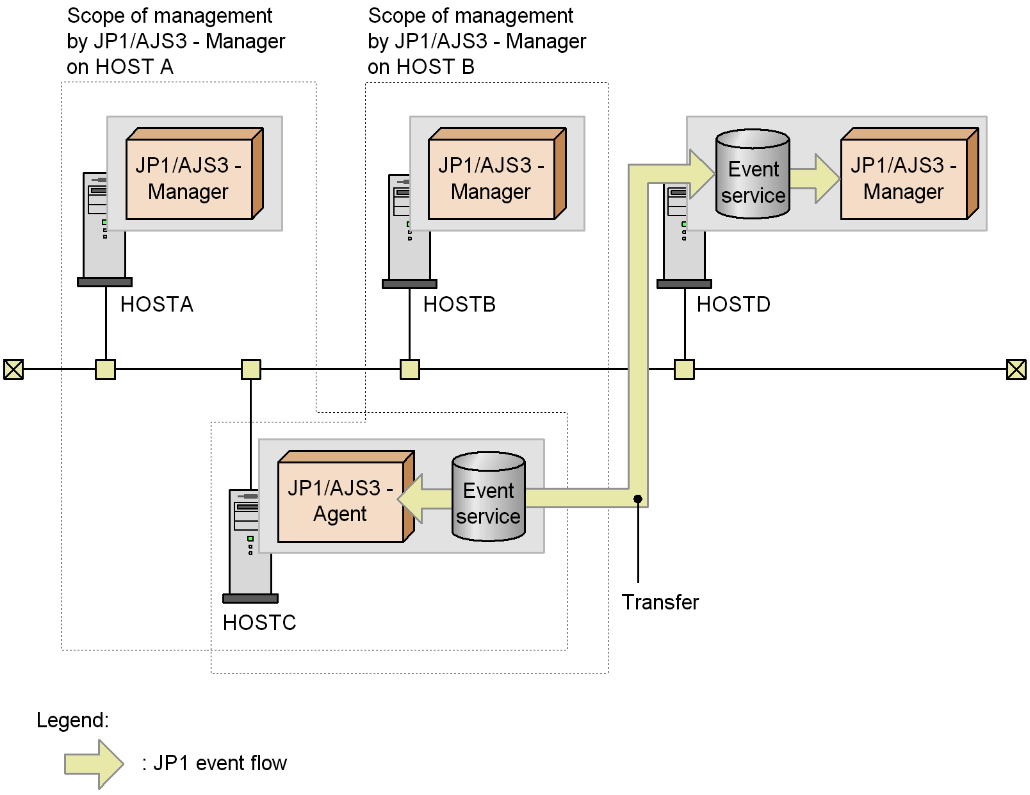


(b) Monitoring events and messages when multiple JP1/AJS3 - Manager hosts use the same JP1/AJS3 - Agent host as an agent

When the same JP1/AJS3 - Agent host is configured as an agent for multiple JP1/AJS3 - Manager hosts, JP1/AJS3 - Agent monitors the same log files (JP1/AJS3 log files to which the instance of JP1/AJS3 on the Agent host outputs information) for events and messages. Therefore, you cannot monitor the log files correctly even from a Manager on a different host.

If you want to monitor events and messages in this configuration, transfer the events and messages to a different JP1/AJS3 - Manager host from the one managing the JP1/AJS3 - Agent.

Figure 7-24: Monitoring events and messages when multiple JP1/AJS3 - Manager hosts use the same JP1/AJS3 - Agent host as an agent



7.7 Notes on using action jobs

The following provides precautions (items you should know in advance) for using action jobs.

Cautionary notes

- You cannot specify a **User name** in an action job. For details, see the cautionary notes for each type of action job.
- Action jobs are affected by the concurrently executable job limit that JP1/AJS3 imposes. If you attempt to execute more jobs on the agent host than the limit allows, the jobs are placed in *Waiting to execute* status. For details about the maximum number of concurrently executable jobs, see *2.5.4 Maximum number of concurrently executable jobs* in the *Job Management Partner 1/Automatic Job Management System 3 System Design (Configuration) Guide*.

7.7.1 Notes on the Send JP1 Event job

The following provides precautions (items you should know in advance) for using the Send JP1 Event job. When defining the JP1 event you want to send, you specify the event level, event ID, and event destination host. After sending a JP1 event, you can check whether the JP1 event has reached the event destination host. JP1/Base must be installed in the destination host.

Note:

Before executing the Send JP1 Event job, you must start the JP1/Base event service on the event-originating host and event destination hosts.

Examples of jobnets that use the Send JP1 Event job are as follows:

- Define a jobnet so that a JP1 event is sent if a specific job ends abnormally. In another jobnet, define the start condition so that the jobnet starts if the jobnet receives the JP1 event that is sent from the other jobnet. With these settings, the second jobnet starts when a specific job ends abnormally.
- Define a jobnet to send a JP1 event to the server on which JP1/IM - Manager is installed when a job ends, and check the sent JP1 event by using JP1/IM - View.

Cautionary notes

- The arrival confirmation facility for JP1 events merely allows you to wait until you can confirm that the event has arrived at the destination event server. The feature does not perform retries even if transmission fails. Therefore, when you execute a Send JP1 Event job while the event service is inactive at the destination, the job ends immediately regardless of whether **Check interval** or **Check count** is specified.

- If you do not check whether the JP1 event you sent has arrived at the destination, an error does not occur even if you specify the following event servers as event destination host names:

  - An event server that is not defined as the JP1/Base event server

  - An inactive event server

  - An event server that cannot receive JP1 events due to a network error or other problem
- The event server of the local host cannot acquire a sent JP1 event that specifies a destination host name other than the local host.
- When you send a JP1 event to the event server of another host with a destination host name specified, the JP1 event is not subject to retry processing according to the `forward-limit` setting in the event server settings file (`conf`) of JP1/Base.
- When you specify a macro variable as an event destination host name, message, or extended attribute, a double quotation mark (`"`) in the passing information might result in the information being passed incorrectly or cause an error. Specify macro variables only if you are sure that they will not cause any problems with the passing information.
- If you do not specify a destination host name, JP1 events are sent to the event server of the host that executes the Send JP1 Event job.
- You cannot specify a **User name** in an action job. The sender of the JP1 event will be the primary OS user mapped to the **User who registered** or **User who owns** of the Send JP1 event job.
- When a logical host name that does not exist in the local host is specified in the `JP1_HOSTNAME` environment variable, even if the execution host of the Send JP1 Event job is a logical host, the name of the host that sends the event will be the name of the physical host.
- A Send JP1 Event job uses the JP1/Base event server functionality. For this reason, if you want to set up a transmission path for a JP1 event that traverses multiple LAN environments, you must make the settings on the JP1/Base side (in the `conf` file). For details about how to do so, see the *Job Management Partner 1/Base User's Guide*.
- In the API settings file for the JP1/Base event service, set `keep-alive` for the communication type of the `server` parameter. Setting `close` as the communication type can result in operational problems, including JP1/AJS3 being unable to issue JP1 events at startup, or Send JP1 Event jobs ending abnormally with message KAVT1040-E output to the integrated trace log. For details about the API settings file and how to set parameters, see the *Job Management Partner 1/Base User's Guide*.

7.7.2 Notes on the Send Mail job

The following provides precautions (items you should know in advance) for using the Send Mail job. This job sends email messages when JP1/AJS3 is linked to a mail system. This action job can send email to both Windows and UNIX hosts.

Examples of jobnets that use the Send Mail job are as follows:

- The jobnet sends an email if a job ends with a warning.
- The jobnet sends an email when all the jobs in the jobnet end.

For details about the functionality that can be implemented by the Send Mail job, see *2.6 Defining email sending job* in the *Job Management Partner 1/Automatic Job Management System 3 Linkage Guide*.

Cautionary note

> You cannot specify a **User name** in an action job. When you execute a Send Mail job on a UNIX host, log in as a JP1 user mapped to the primary OS user who will be the mail sender and then register the job for execution.

7.7.3 Notes on the OpenView Status Report job

The following provides precautions (items you should know in advance) for using the OpenView Status Report job. This job reports designated statuses to HP NNM. For details about linkage with HP NNM, see *A. Monitoring Jobnets Using HP NNM* in the *Job Management Partner 1/Automatic Job Management System 3 Linkage Guide*.

Cautionary note

> You must have superuser privileges to execute an OpenView Status Report job on a UNIX host. Because you cannot specify a **User name** in an action job, log in as a JP1 user that mapped to the root user as the primary OS user, and then register the job for execution.

7.7.4 Notes on the Local Power Control job and Remote Power Control job

The following provides precautions (items you should know in advance) for using the Local Power Control job and Remote Power Control job.

- The Local Power Control job links with JP1/Power Monitor and shuts down the manager host or agent host.
- The Remote Power Control job links with JP1/Power Monitor and starts and shuts down the agent host of JP1/Power Monitor on the network. This job can control hosts in the manager/agent configuration of JP1/Power Monitor. It does not depend on the manager/agent configuration of JP1/AJS3.
- The Local Power Control job or Remote Power Control job must be executed by

a superuser or a member of the Administrators group.

- When the Remote Power Control job is executed, the job sends a request to JP1/ Power Monitor on the agent (job-executing host) to perform remote power control. JP1/Power Monitor then starts remote power control. The agent's next power-on time is determined by the time zone setting of the agent.
- The host that executes the Remote Power Control job must be set as the manager host for the remote power linkage function of JP1/Power Monitor.

Supplementary note

If you use the option to continue execution of active event jobs, the event job will remain in *Now running* status on the manager host if JP1/AJS3 terminates on the agent host. This rules out planned termination and other modes of operation that depend on waiting for jobs to terminate on the manager host. If you must use a Local Power Control job or Remote Power Control job in an environment where this option is enabled, you can work around this problem by waiting for active event jobs to end or forcibly terminating them, or waiting for the scheduling function to end the event jobs before performing power control.

(1) Example of a jobnet that uses the Local Power Control job

Example jobnets defined with the Local Power Control job are as follows:

- When the preceding job ends, the jobnet shuts down the manager host or agent host specified as the execution host.

Note that executing a Local Power Control job from JP1/AJS3 is the same as using the JP1/Power Monitor calendar to perform a planned stop.

For details about the functionality that can be executed by the Local Power Control job, see the *Job Management Partner 1/Power Monitor Description, User's Guide and Reference*.

Supplementary note

To execute a Local Power Control job, you must be a member of the Administrators group (on a Windows host) or have superuser permission (on a UNIX host). Because you cannot specify a **User name** in an action job, to execute a Local Power Control job, log in as a JP1 user that mapped to the root user as the primary OS user, and then register the job for execution.

(2) Example of a jobnet that uses the Remote Power Control job

The following shows examples of jobnets that use the Remote Power Control job:

- The jobnet starts the agent host (agent host of JP1/Power Monitor) before executing a job on the host. When the job ends, the host is shut down.
- If the preceding job ends abnormally, the jobnet restarts the host (agent host of JP1/Power Monitor) that executed the job, and executes the job again.

For details about the functionality that can be executed by the Remote Power Control job, see the *Job Management Partner 1/Power Monitor Description, User's Guide and Reference*.

Cautionary note

To execute a Remote Power Control job, you must be a member of the Administrators group (on a Windows host) or have superuser permission (on a UNIX host). Because you cannot specify a **User name** in an action job, to execute a Remote Power Control job, log in as a JP1 user that mapped to the root user as the primary OS user, and then register the job for execution.

(3) Handling a jobnet for which a start condition is satisfied more than once

When a jobnet's start condition is satisfied more than once within the valid range of the start condition, some subordinate jobnets might be placed in the *Skipped so not executed* status. Execution of the jobnets placed in this status is skipped. This occurs when the following conditions coexist:

- A jobnet with a start condition for which **Disable** is set for **Concurrent exe.** is registered for execution.
- During execution of a generation with a satisfied start condition, the start condition is satisfied again. The latter generation (in the *Wait for start cond.* status) is waiting for the previous generation to terminate.
- The timeout period set for the root jobnet has expired.

When the above three conditions are met, the message KAVS0277-I is output, and the generation waiting for the end of the previous generation is placed in the *Skipped so not executed* status and not executed.

To avoid this situation, take one of the following measures:

- Enable concurrent execution of jobnets.

  To perform this setting from JP1/AJS3 - View, open the Define Details - [Jobnet] dialog box. On the **Definition** page, set **Concurrent exec.** to **Enable**.

  To perform this setting by command, in the unit definition file, set the `mp` jobnet definition parameter to `y`.

- Set an unlimited timeout period for the jobnet.

  To perform this setting from JP1/AJS3 - View, open the Define Details - [Jobnet] dialog box. On the **Definition** page, set **Time-out period** to **Unlimited**.

  To perform this setting by command, in the unit definition file, set the `cd` jobnet definition parameter to `un`.

7.8 Notes on defining jobs

This section gives cautionary notes on jobnet definition.

7.8.1 Notes on the standard output file and standard error output file

The following cautionary notes relate to the standard output file and standard error output file.

(1) When defining the standard output file and standard error output file

Note the following when defining the standard output files and standard error output file output at job execution (of PC and Unix jobs including queueless jobs):

1. By default, jobs executed from jobnets acquire the content of the standard error output to serve as their execution result. Their execution results are not acquired from the standard output. You can view the contents of the standard error output in the Execution Result Details dialog box of the JP1/AJS3 - View window. This window also displays the error messages output to the standard error output by JP1/AJS3.

2. To output the contents of the standard output and standard error output to files of your choosing, specify the file names in the **Standard output** and **Standard error** fields when defining the job.

3. To also have the contents of the standard output displayed in the Execution Result Details dialog box of the JP1/AJS3 - View window, specify the same file name in the **Standard output** and **Standard error** fields when defining the job.

4. If you specify the same file name in the **Standard output** and **Standard error** fields, use the same **Append** setting for both files. If you specify the option to create a new file for one file and the option to add data to an existing file for the other, a parameter error occurs and the job fails to register for execution.

5. If you specify the same name as the script file name or the execution file name as the environment variable file name, the standard input file name, the standard output file name, or the standard error output file name in the Define Details - [*unit-name*] window, the job might terminate abnormally. Do not use the same file name as the script file or the execution file for any of these files.

6. If you specify the same standard output file name or standard error output file name for jobs that are executed concurrently, the older output results in the standard output or the standard error output will be overwritten by the new output results. Also, when multiple jobs that use the standard output or standard error output run concurrently, the older output results will be overwritten by the new output results.

   Specify a different standard output file name or standard error output file name for each of the jobs to be executed concurrently.

7. If you specify an inaccessible network file name as the standard output file name or the standard error output file name, the job might fail to start or end abnormally. Specify a correct network file name.

8. To specify the name of a network file as the standard output file name or standard error output file name, you must have the following permission for the network file:

   In Windows

   > The account for the JP1/AJS3 service (or the JP1/AJS3 Queueless Agent service for queueless jobs) must have Create, Read, and Modify permissions for the file.

   In UNIX

   > The account specified as the OS user who executes the job must have Create, Read, and Modify permissions for the file.

Supplementary note

> By default, submit jobs executed using the `jpqjobsub` command do not save data to standard output or standard error output. Specify the desired file name in an option of the `jpqjobsub` command.

(2) When outputting large quantities of data to the standard output or standard error output file

In JP1/AJS3, when a job ends (meaning a standard job, action job, or custom job, but not queueless jobs), the standard output file and standard error output files are transferred from the agent host to the manager host. If you execute a job that outputs large quantities of data (several megabytes or more) to the standard output or standard error output file, transferring and analyzing the data can place a high demand on system resources. The additional CPU and memory resources such operations require can delay job execution, and even slow down the entire system. This problem affects the manager host and the agent host.

If you specify the option to append data, output data accumulates with each executed job. This markedly increases the size of the transferred file. In this situation as well, transferring and analyzing the data can place a high demand on system resources, which can cause jobs to terminate abnormally or delay the transfer of data from the agent host to the manager host. In this case, we recommend that you disable the append option. Alternatively, you can periodically back up and delete the standard output and standard error output data files.

In JP1/AJS3, you can set an upper limit on the sizes of the standard output file and standard error output file so that excess data is discarded or a warning message is output when the upper limit has been reached. In this way, you can control the file sizes so that the processing of only a few jobs will have little effect on overall system performance. For details about how to set a maximum file size for the standard output or standard error output file, see *6.2.7 Placing restrictions on file reception* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1* (for Windows systems) or *14.2.7 Placing restrictions on file reception* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1* (for UNIX systems).

If you have not imposed a limit on the sizes of the standard output file and standard error output file, and a job is not operating correctly, take appropriate action as follows:

- When a large amount of data is output to the standard output

  For a PC job or Unix job, the standard output file is transferred only when the file name is explicitly specified in the job definition. If a large amount of data is transferred as the standard output file, the job might end abnormally. If the job ends abnormally, do not specify anything as the standard output file name, and redirect the standard output from a batch file or shell script within the job.

  If you do not specify a standard output file name in the job definition, the standard output file is not transferred to the manager host.

- When a large amount of data is output to the standard error output

  Specify a NULL device as the standard error output file name in the job definition. If the job is to be executed in UNIX, specify `/dev/null`; if the job is to be executed in Windows, specify `NUL`. If a standard error output file name is explicitly specified in the job definition, use a batch file or shell script to redirect the standard error output.

  When you specify these settings, the standard error output file is not transferred to the manager host, and you will be unable to view the contents of the file in the Execution Result Details dialog box of JP1/AJS3 - View.

Supplementary notes

1. If the problems described above occur, a file named `A_JPQ*`*_job-number* might remain in the following folder (by default):

   In Windows Server 2008:

   `%ALLUSERSPROFILE%\HITACHI\JP1\JP1_DEFAULT\JP1AJS2\tmp`

   (the default location of `%ALLUSERSPROFILE%` is *system-drive*`\ProgramData`).

   In Windows Server 2003:

   JP1/AJS3-installation-folder`\tmp`

In UNIX: `/var/opt/jp1ajs2/tmp`

You can safely delete files whose file name contains the job number of a job that ended abnormally.

In a cluster environment, the location of these folders might have been changed. If so, check for the temporary files in the new folders.

2. When you use submit jobs that output large amounts of data to the standard output or standard error output, the amount of data in the temporary files created in the work directory of the job execution environment of the manager host (`M_JPQSTDE_`*job-number* or `M_JPQSTDE_`*job-number*) also increases. Usually, when the setting determining how many days to retain job information is exceeded for a job, temporary files for that job are deleted when job information deletion processing is performed. However, if the disk becomes full due to the temporary files, you can delete them manually after the job terminates. If the disk often becomes full, consider reducing the number of days job information is retained, or redirecting the standard output or standard error output from a script file in the job.

   Note that once you delete a temporary file, you can no longer use the `jpqjobget` command to view the standard output and standard error output files.

The following tables show examples of specifying redirection as **Parameter** in the detailed definition of a job.

Table 7-6: Examples of specifying redirection as Parameter in the detailed definition of a job (in Windows)

Job definition	Items to be specified	Specification
The names of the standard output file and standard error output file are different, and the executable file is an `exe` file.	Executable file: `test.exe` Parameters: None Standard output file: `out.txt` Standard error output file: `err.txt`	Executable file: `cmd.exe` Parameters: `/C test.exe >out.txt 2>err.txt` Standard output file: None Standard error output file: None
The names of the standard output file and standard error output file are different, and the executable file is an `bat` file.	Executable file: `test.bat` Parameters: None Standard output file: `out.txt` Standard error output file: `err.txt`	Executable file: `test.bat` Parameters: `>out.txt 2>err.txt` Standard output file: None Standard error output file: None

Job definition	Items to be specified	Specification
The names of the standard output file and standard error output file are the same.	Executable file: `test.bat` Parameters: None Standard output file: `out.txt` Standard error output file: `out.txt`	Executable file: `test.bat` Parameters: `>out.txt 2>&1` Standard output file: None Standard error output file: None

Table 7-7: Examples of specifying redirection as Parameter in the detailed definition of a job (in UNIX)

Job definition	Items to be specified	Specification
The names of the standard output file and standard error output file are different.	Script file: `test.sh` Parameters: None Standard output file: `out.txt` Standard error output file: `err.txt`	Script file: `test.sh` Parameters: `>out.txt 2>err.txt` Standard output file: None Standard error output file: None
The names of the standard output file and standard error output file are the same.	Script file: `test.sh` Parameters: None Standard output file: `out.txt` Standard error output file: `out.txt`	Script file: `test.sh` Parameters: `>out.txt 2>&1` Standard output file: None Standard error output file: None

(3) Other notes

1. You can specify only text files as the output destination for standard output and standard error output data.
2. Make sure that machine-dependent characters are not output in the standard output and standard error output data. JP1/AJS3 - View cannot display these characters correctly.
3. Standard output data and standard error output data is saved to temporary files on the agent host, which are transferred to the manager host when the job has terminated. These files are created with the following names under the work directories in the job execution environment on both the manager and agent hosts.

   Manager host (submit jobs only)

   `M_JPQSTDE`*job-number*

   `M_JPQSTDO`*job-number*

   Agent host

   `A_JPQSTDE_*_`*job-number*

   `A_JPQSTDO_*_`*job-number*

   The following describes when the files are deleted from each host.

   Manager host:

In a standard configuration: Temporary files for QUEUE jobs and submit jobs are deleted when job information deletion processing is performed. For other jobs, the temporary files are deleted when the job ends.

In a compatible ISAM configuration: Temporary files are deleted when job information deletion processing is performed.

Agent host:

The temporary files are deleted automatically when the job ends.

For queueless jobs, only the standard error output data is stored in a temporary file on the agent host. The data in this file is transferred to the manager host when the queueless job terminates. The file is deleted automatically after the queueless job terminates or when the queueless agent service starts.

4. When you display the Execution Result Details dialog box of JP1/AJS3 - View after executing a job for which a standard error output file name is specified and **Append** is selected, the displayed information differs depending on the destination service of the job.

   - When **Standard** is specified as the **Exec. Service**:

     The content of the file specified in **Standard error** is displayed.

   - When **Queueless Agent** is specified as the **Exec. Service**:

     The content output to standard error output at job execution is displayed.

7.8.2 Notes on the execution priority of jobs

JP1/AJS3 sets the execution priority for jobs (PC jobs including queueless jobs, Unix jobs, QUEUE jobs executed in JP1/AJS3, action jobs, and custom jobs).

You can set **None**, **1**, **2**, **3**, **4**, or **5** as the execution priority. If no execution priority is specified for a job, JP1/AJS3 sets **None**, which is equivalent to execution priority **1**. When the execution priority is not specified for a job executed from JP1/AJS3, the job is assigned a low execution priority by default. See the following notes to determine whether you need to change the job execution priority for your system environment and operating conditions.

The following subsections describe the values that JP1/AJS3 sets for jobs according to the execution priority of jobs in Windows and UNIX.

(1) Job execution priorities and the priority values that JP1/AJS3 assigns to jobs (in UNIX)

In UNIX, JP1/AJS3 assigns a nice value that corresponds to the execution priority of the job. The base value from which a job's execution priority is determined differs depending on which of **Standard** or **Queueless Agent** is specified as the execution service.

(a) When Standard is specified as the execution service

When **Standard** is specified as the execution service for the job, the nice value of a job executed by JP1/AJS3 is based on the nice value# of the JP1/AJS3 service when it starts (when you executed the `jajs_spmd` command). A job's nice value is assigned by adding to or subtracting from this base value, according to the execution priority specified for the job.

#

> The nice value at startup of the JP1/AJS3 service is the default value assigned by the OS when it started the process. The maximum and minimum nice values that can be assigned depend on your operating system. For details about the default nice value assigned by the OS, and the maximum and minimum assignable values, see the documentation for your operating system and the UNIX reference documentation.

The following table shows an example of the correspondence between job execution priorities specified in JP1/AJS3 and the nice values assigned to jobs. In this example, the operating system is HP-UX, **Standard** is specified as the execution service for the job, and the default nice value to the JP1/AJS3 service at startup is `20`.

Table 7-8: Job execution priorities and the nice values assigned to jobs (when Standard is specified as the execution service)

Execution priority	nice value assigned to the job	When the nice value of JP1/AJS3 service is 20
1	nice value + 20	39
2	nice value + 10	30
3	nice value	20
4	nice value - 10	10
5	nice value - 20	0

If no execution priority is specified for a job, **1** is set as the execution priority. In this case, a nice value of `39` is assigned to the job. This value is determined by adding `20` to the nice value of the JP1/AJS3 service (`20` by default). System restrictions in this case mean that nice values are capped at `39`, and cannot be lower than `0`.

(b) When Queueless Agent is specified as the execution service

When **Queueless Agent** is specified as the execution service, the job's nice value is not based on that of the JP1/AJS3 service. In this case, JP1/AJS3 sets `39`, `30`, `20`, `10`, or `0` as the nice value according to the execution priority, where `0` corresponds to the highest priority. To change a job's nice value, specify the job execution priority that corresponds to the nice value that you want to assign.

The following table shows the correspondence between job execution priorities specified in JP1/AJS3 and the nice values assigned to jobs, when **Queueless Agent** is specified as the execution service.

Table 7-9: Job execution priorities and the nice values assigned to jobs (when Queueless Agent is specified as the execution service)

Execution priority	nice value assigned to job
1	39
2	30
3	20
4	10
5	0

(2) Job execution priorities and the values that JP1/AJS3 assigns to jobs (in Windows)

In Windows, JP1/AJS3 sets one of three job execution priorities. The following table shows the correspondence between job execution priorities and the base priorities of Windows. You can use Task Manager to check the base priority.

Table 7-10: Job execution priorities and Windows base priorities

Job execution priority	Base priority assigned to job
1	Low
2	
3	Normal
4	High
5	

The following describes the priority class that JP1/AJS3 sets based on the job execution priority when it starts the job process.

- If you set **1** or **2** as the execution priority value for a job, the job is executed when the system is idle (the Windows `IDLE_PRIORITY_CLASS` is assigned). This priority is lower than the priority of interactive processing.
- If you set **3** as the execution priority value for a job, the job is executed as a general process (the Windows `NORMAL_PRIORITY_CLASS` is assigned). This priority is the same as the priority of interactive processing.
- If you set **4** or **5** as the execution priority value for a job, the job is executed before

the threads of the processes assigned the above priority classes (the Windows `HIGH_PRIORITY_CLASS` is assigned). This priority is higher than the priority of interactive processing.

(3) Changing the job execution priority

The execution priority of a process started directly from a service or console and not through JP1/AJS3 depends on the operating system. In Windows, the process is assigned the *Normal* base priority, which is equivalent to the priority of interactive processing. In UNIX, the process is assigned a nice value of `20`. If you want jobs executed from JP1/AJS3 to have the same execution priority as these processes, you must set the job execution priority to `3`. Use either of the following methods to change a job's execution priority.

In these examples, the job execution priority is set to **3**:

1. In the Define Details - [*unit-name*] dialog box for the job in question, on the **Definition** page, specify **3** in **Priority**.
2. In the Define Details - [*unit-name*] dialog box for the job in question, on the **Definition** page, specify **None** in **Priority**. Then, in the Define Details - [jobnet] dialog box for the upper-level jobnet, such as the root jobnet, on the **Definition** page, specify **3** in **Priority**.

Cautionary note

In UNIX, if a user without superuser permission tries to execute a job with a job execution priority of **4** or **5**, a permission error occurs.

Also, if you assign jobs a higher execution priority than other non-job processes, some jobs might monopolize system resources and slow down the entire JP1/AJS3 system. This is the opposite situation to that described in *2.1.4(b) Execution priority*. Carefully consider the system environment and operating conditions before you assign jobs a higher priority than other processes.

7.8.3 Checking the return code of a job

You can check the return code of a job in the Monitor Details - [*unit-name*] dialog box of JP1/AJS3 - View, or from the value of `EXITCODE` in the job information output by the `jpqjobget` command. Under normal circumstances, `0` is set as the return code when a job terminates normally. If the job terminates abnormally, the return code of the job process is set. However, in the following cases, the return code is set by JP1/AJS3.

Table 7-11: Return codes set by JP1/AJS3 and associated conditions

Condition	Return code set by JP1/AJS3
If one of the following statuses occurs in a QUEUE job for which JP1/AJS3 is specified as a host, a PC job, a Unix job, an action job, or a custom job: • Failed to start • Killed (including jobs killed due to their timeout period elapsing) • Ended abnormally#	`-1`
If an unexpected JP1/AJS3 error occurs during execution of a Unix job.	`255`

#

If a problem occurs during post-processing by JP1/AJS3 after a job process has terminated (for example, if the result file could not be transferred to the manager host), the return code of the job process is overwritten with `-1` by JP1/AJS3 and the job process terminates abnormally.

Cautionary note

For a PC job, the started job process itself can return the return code `-1`. For a Unix job, if a job returns a negative value, the return code is assumed to be the returned value plus 256. For example, if a job process ends with -1, the return code is 256 - 1 = 255. In a user application, if you need to determine whether the return code was set by the job process or JP1/AJS3, do not use a return code of -1 or 255 in your user applications.

For details about the return codes set while an event job or action job is being executed (from the start to end of the job process), see *A. Return Values from Event Jobs and Action Jobs*.

In some cases the return code of a job is set by the OS, rather than by a user application or JP1/AJS3. The following table gives examples of typical return codes. Because upgrading your OS might change these return codes, check the technical information for your OS.

Table 7-12: Examples of return codes set by the OS

OS	Return code	Cause	Action to take
Windows	259 or -1	The process might have failed to open the results file. In this case, one of the following occurs: • Message KAVU3284-W is output: `A system call error occurred in the internal process (`*logical-host-name*`) (reason module:` *reason-module*`, reason code 1: 0x2013000a, system call name: CreateFile, reason code 2:` *reason-code-2*`)` • The following message is output to the standard error output: `The process cannot access the file. The file is being used by another process.`	As the standard output file or standard error output file when you register a job, do not specify a file opened from within a program executed as a job or opened by redirection from a batch file. However, where the file is opened from within the program by a function call, you can work around the problem by opening the file with a setting that permits shared read and write modes.
	128	There might be a desktop heap shortage. If so, one of the following messages is output: • *XXXX.XXX* `- application error: application not initialized correctly`; or • `Failed to initialize` *XXXX.XXX*	To reduce use of the desktop heap, use the same user account as the JP1/AJS3 service to execute jobs.
UNIX	Signal number + 128	A program started from a job process might have received a signal.#	Take one of the following actions: • Set up the program started from a job process so that it does not receive signals. • Set up a signal handler or similar for the program to provide additional processing that sets signal-dependent return codes.

#

If the job process itself receives a signal, the job is forcibly terminated with the return code `-1`.

Chapter

8. Definition Pre-Check

JP1/AJS3 allows you to check for errors in job definitions.

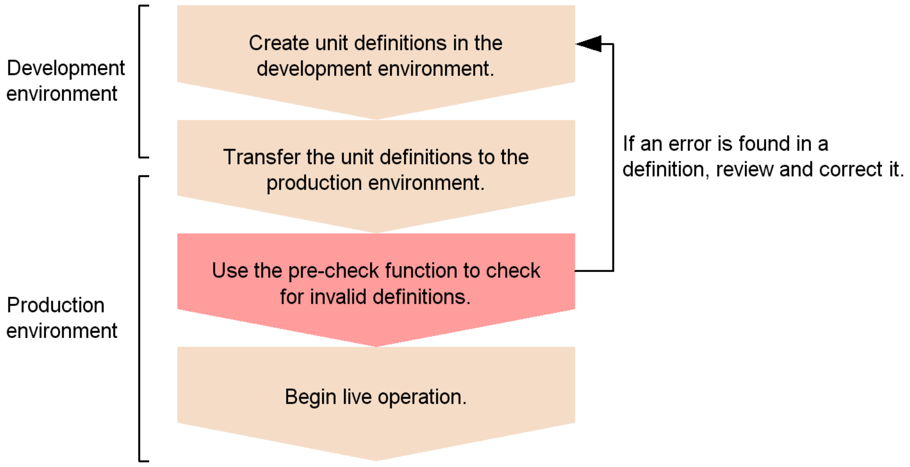
This chapter describes the procedure for the definition pre-check performed before live JP1/AJS3 operation starts. The chapter also describes check items and includes cautionary notes.

8.1 Checking definitions before live JP1/AJS3 operation

JP1/AJS3 allows you to check whether job definitions are valid before you deploy them, to prevent failures in a production environment.

The following figure shows the sequence of operations involved in a definition pre-check, up to the point when you begin live operation.

Figure 8-1: Flow of system design and definition pre-check up to commencing operation


Development environment
Create unit definitions in the development environment.
Transfer the unit definitions to the production environment.
Production environment
Use the pre-check function to check for invalid definitions.
Begin live operation.
If an error is found in a definition, review and correct it.

Use the `ajschkdef` command to run a definition pre-check. When you execute the `ajschkdef` command, the check results are output to the pre-check result file. You can see the results of the pre-check by viewing the file. The pre-check result file is output to the following location by default:

In Windows Server 2008:

`%ALLUSERSPROFILE%\HITACHI\JP1\JP1_DEFAULT\JP1AJS2\log\ajscheckfile.txt`

(The default location of `%ALLUSERSPROFILE%` is *system-drive*`\ProgramData`.)

In Windows Server 2003:

JP1/AJS3 - Manager-installation-folder`\log\ajscheckfile.txt`

In UNIX:

`/var/opt/jp1ajs2/log/ajscheckfile.txt`

For details on the `ajschkdef` command, see *ajschkdef* in *2. Commands* in the manual *Job Management Partner 1/Automatic Job Management System 3 Command*

Reference 1.

Before you run a definition pre-check, you must set up the definition pre-check function. For details on how to set up the function and its environment, see *6.5.1 Setting up the JP1/AJS3 definition pre-check function* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1* (for Windows systems) or *14.5.1 Setting up the JP1/AJS3 definition pre-check function* in the *Job Management Partner 1/Automatic Job Management System 3 Configuration Guide 1* (for UNIX systems).

(1) Checked items

The following table lists the items checked by the definition pre-check function.

Table 8-1: Items checked by the definition pre-check function

Category	Description		
Job execution order	• Checks whether the execution order creates a loop. • Checks whether a subordinate job is connected to a judgment job by a condition. • Checks whether relations other than conditions are defined for subordinate jobs.		
Detailed definition of a jobnet	Jobnet connector	Connection range	• Checks whether the connection range matches that specified for the connection-destination jobnet
		Connection host	• Checks whether the specified host is accessible.
		Connection service	• Checks whether the specified scheduler service exists.

Category	Description		
		Connect destination	Performs the following: • Checks whether a connection-destination jobnet is specified. • Checks whether the specified unit is a root jobnet or a planning group. • Checks whether the specified unit exists. • Checks whether the specified unit is subject to execution order control. • Checks whether the specified unit specifies a different host. • Checks whether the specified unit specifies a different scheduler service. • Checks whether the specified unit specifies a different jobnet connector.
Empty job definition	For a Unix job, checks whether a **Script file name** or **Command statement** is specified. For a PC job, checks whether a **File name** is specified. For a Receive JP1 event job, checks whether a **Event ID** is specified.		
Execution agent name	• Unix job • PC job • Action job	When **Standard** is specified as the **Exec. service**#1	Performs the following: • Checks whether the agent host is registered with the manager. • Checks whether the manager can resolve the execution host name (agent host name) set for the execution agent. • Checks whether the agent can resolve the host name of the manager.

Category	Description		
		When **Queueless Agent** is specified as the **Exec. service**	Performs the following: • Checks whether the manager can resolve the host name of the agent. • Checks whether the agent can resolve the host name of the manager.
	Event job		Performs the following: • Checks whether the agent host is registered with the manager. • Checks whether the manager can resolve the execution host name (agent host name) set for the execution agent. • Checks whether the agent can resolve the host name of the manager.
	• Jobnet[#1] • Custom job[#1]		Performs the following: • Checks whether the execution agent is registered with the manager. • Checks whether the manager can resolve the execution host name (agent host name) set for the execution agent. • Checks whether the agent can resolve the host name of the manager.
User mapping	Checks whether user mapping can be performed correctly on the agent. This check uses the same method for JP1 users, execution source hosts, and OS users as when normal jobs are executed. Note that JP1/AJS3 does not check whether the OS user mapped to a JP1 user actually exists.		
Detailed definition of a job	Unix job	Script file name	1. Checks whether the file exists on the Agent.[#2, #3] 2. Checks the access permission for the file.

Category	Description		
		Environment file	1. Checks whether the file exists on the Agent.#4 2. Checks the access permission for the file.
		Working path	1. Checks whether the path exists on the Agent.#2 2. Checks the access permission for the path.
		Standard input	1. Checks whether the file exists on the Agent.#4 2. Checks the access permission for the file.
		Standard output	1. Checks whether the directory containing the file exists on the Agent#4 2. If the file exists, checks the access permission for the file.
		Standard error	1. Checks whether the directory containing the file exists on the Agent#4 2. If the file exists, checks the access permission for the file.
		User name	Checks the user by user mapping.
		File to transfer	1. Checks whether the file exists on the Manager#2. 2. Checks the access permission for the file.
		Destination file	1. Checks whether the file exists on the Agent.#2 2. Checks the access permission for the file.
	PC job	File name	Checks whether the file exists on the Agent.#2, #3

Category	Description		
		Environment file	Checks whether the file exists on the Agent.[#4]
		Working path	Checks whether the path exists on the Agent.[#2]
		Standard input	Checks whether the file exists on the Agent.[#4]
		Standard output	Checks whether the directory containing the file exists on the Agent.[#4]
		Standard error	Checks whether the directory containing the file exists on the Agent.[#4]
		User name	Checks the user by user mapping.
		File to transfer	Checks whether the file exists on the Manager.[#2]
		Destination file	Checks whether the file exists on the Agent.[#2]
	Receive JP1 Event job	Event ID	Checks whether the event ID confirms to the proper format.
		Find events prior to execution	Checks whether the pre-search time is within the specified range.
	Monitoring File job	Monitoring file	Checks the format of the file name.
	Receive Mail job	Platform	Checks whether the OS matches that of the execution destination host.
	Common to action jobs	Platform	Checks whether the OS matches that of the execution destination host.

Category	Description
Permission for executing files[#5]	Checks whether the OS user who executes the job has execution permission for the target file. This check uses the same method as when normal jobs are executed. However, for queueless jobs executed in UNIX, execution permissions are not checked when the `root` user executes the job (in this case, no execution permission is required). Note that JP1/AJS3 checks execution-file permissions only for the primary group of the OS user who executes the job.

#1

This item is not checked if the name of an execution agent group is specified.

#2

This item is not checked if a relative path or a UNC path is specified.

#3

For items that can take macro variables, there is no way to determine the actual data that will be used at execution. Therefore, the values resulting from macro variables are not checked.

#4

Relative paths are not checked when an invalid work path is detected.

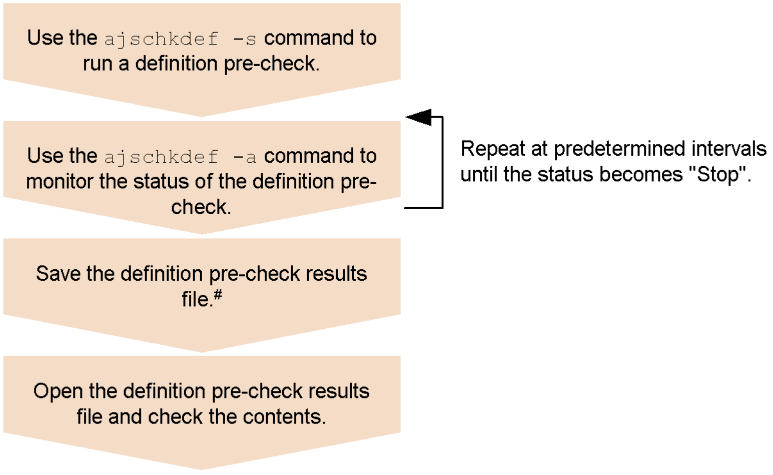
#5

PC jobs (on a Windows host) are not checked.

(2) Pre-check sequence

The following figure shows the flow of a definition pre-check.

Figure 8-2: Flow of the definition pre-check process


Use the ajschkdef -s command to run a definition pre-check.
Use the ajschkdef -a command to monitor the status of the definition pre-check.
Repeat at predetermined intervals until the status becomes "Stop".
Save the definition pre-check results file.#
Open the definition pre-check results file and check the contents.

#: Because the file may be overwritten by a later request, save the file under a different name.

(3) Cautionary notes

Note the following when performing a definition pre-check:

1. Do not run a definition pre-check while the system is being used to perform business tasks. Doing so can lead to the following problems:
   - Access contention for the scheduler database reduces job processing performance.
   - The system load might temporarily spike, causing errors during job execution processing.
2. To run a definition pre-check, the JP1/AJS3 service must be started on the manager.
3. You cannot perform multiple definition pre-checks simultaneously.
4. If you also specify parameters in the **Script file name** field for a Unix job, the definition pre-check detects an error.
5. Information passed from events is not checked.
6. Macro variables are not checked.
7. Environment variables are not checked.
8. Job runtime variables are not checked.

9. In UNIX, JP1/AJS3 Check Manager service does not use the port number for the `jp1ajs2chkman` service specified in the `services` file.

10. JP1/AJS3 checks the access permissions for a script file in a different way from that used during actual operation. Therefore, the definition pre-check may detect an error for a job that would end normally.

11. Messages KAVS3400 to KAVS3431 are output only to the integrated trace log or the standard output.

12. In Windows, when you specify a folder as an environment variable file, a standard input file, a standard output file, or a standard error output file, and execute the queueless agent service, the job ends normally but the pre-check detects an error.

13. In UNIX, if you specify a directory as an environment variable file and execute the job by using the queueless agent service, the job ends normally but the pre-check detects an error.

14. When you specify the `-A` option in the `ajschkdef` command, information about the OS user who executes the job is required in order to check the permissions for the executable file. Therefore, user mapping must also be checked. However, in a Windows system, the OS does not check the account details of the user who is specified in **User name** in the Define Details - [PC Job] dialog box.

15. When both of the following conditions exist, temporary files are created before the definition pre-check:

    - In the definition of a Unix job, a non-existent file is specified for the **Standard output**, **Standard error**, or **Destination file**.
    - A definition pre-check is executed for this job with the `-D` option specified.

    The temporary files are created in the location specified for the **Standard output**, **Standard error**, or **Destination file** of jobs that meet these conditions. As a result, the update time for the parent directory of the specified file might change. The temporary files are deleted when the check is finished.

16. The JP1/AJS3 Check Manager service performs communication processing according to the communication settings of the JP1/Base physical host. Therefore, to use definition pre-checking in a multi-LAN environment, the appropriate communication settings (in the `jp1hosts` and `conf` files) must be specified on the JP1/Base physical host. For details about how to specify the communication settings for JP1/Base, see the *Job Management Partner 1/Base User's Guide*.

(4) Notes on definition pre-checks in a cluster system

Note the following when performing a definition pre-check in a cluster system:

1. In a cluster system, when you perform a definition pre-check for units in a logical host, run the check on the primary node of the logical host. You cannot check unit

definitions on the standby node.

2. If a failover occurs while you are checking units on the primary node of the logical host in a cluster system, the checking process is interrupted. Pre-checks are not resumed after a failover to the standby node.

3. If failover of an agent host in a cluster system environment occurs after the agent host is requested to perform pre-checks, pre-checks are retried according to the values of the environment setting parameters. However, if JP1/AJS3 cannot connect to the previous agent host, the message KAVS3410-I `The connection with the check agent service (`*agent-name*`) was closed. :` *maintenance-information* is output and pre-checks are performed on another agent host.

Appendixes

A. Return Values from Event Jobs and Action Jobs

The following table lists the return values from event jobs and action jobs.

Table A-1: Return values from event jobs and action jobs

Return value	Description	Job status after execution or time-out
0	The job ended normally, or the system always assumes that the job ends normally.	Ended normally
1	The job ended normally with a matching message.	Ended normally
2	The job ended normally with detailed matching information.	Ended normally
3	The job ended normally without a matching message.	Ended normally
4	The job ended normally without detailed matching information.	Ended normally
5	Event-inherited information has exceeded the limit.	Ended normally
6	An error has occurred during event-inherited information creation.	Ended normally
7	The job ended normally after a time-out.	Ended normally
12	The job ended with a warning after a time-out.	Ended with warning
16	Condition is not satisfied (incomplete scheduled task).	Bypassed
17	Monitoring time-out has occurred.	Killed
18	The job was killed.	Killed
20	The system always assumes that the job ends abnormally.	Ended abnormally
21	The job ended abnormally with a matching message.	Ended abnormally
22	The job ended abnormally with detailed matching information.	Ended abnormally
23	The job ended abnormally without a matching message.	Ended abnormally
24	The job ended abnormally without detailed matching information.	Ended abnormally
25	The job ended abnormally after a time-out.	Ended abnormally
30	The job ended because the event action agent has stopped.	Ended abnormally
50	A communication error has occurred between the manager and the agent.	Ended abnormally or failed to start
55	Resources are insufficient. Alternatively, the specified file cannot be accessed.	Ended abnormally

Return value	Description	Job status after execution or time-out
56	Insufficient memory	Ended abnormally
64	The file cannot be read (I/O error).	Ended abnormally
90	Incorrect parameter	Ended abnormally
91	Required item was not defined.	Ended abnormally or failed to start
92	An incorrect macro variable was specified.	Ended abnormally
93	An incorrect platform was specified.	Ended abnormally
250	The mail system or message queue system is not available.	Failed to start
253	An option error occurred in a prerequisite program.	Ended abnormally
254	An error occurred in a prerequisite program.	Ended abnormally
255	System error	Ended abnormally or failed to start

Note

When an operation (such as killing a jobnet) is performed for all start conditions, or an error occurs for all start conditions, the return value of each event for all start conditions is set to `0`. In this case, if an error occurs for a specific event (such as for a definition mistake), the return value is set as shown in the table.

Table A-2: Return values from JP1 event sending jobs

Return value	Description
0	The job ended normally.
1	Argument error
100	A transfer has failed. In version JP1/AJS2 07-00 or later, the arrival of an event was not confirmed. This return value corresponds to return value 3 (transfer failure) for `jevsendd` command in an earlier version.
120	Processing continues (if the arrival of an event has not been confirmed within the maximum waiting time). In version JP1/AJS2 07-00 or later, the arrival of an event was not confirmed. This return value corresponds to return value 2 (processing continuing) for `jevsendd` command in an earlier version.
150	Insufficient memory

Return value	Description
255	Another error

Note

If a return value other than 0 is returned, determine the cause of the problem from the job execution result messages.

Table A-3: Return values from email sending jobs (in Windows)

Return value	Description
0	The job ended normally.
90	Parameter error (error in a definition item)
91	Parameter error (error in a definition)
92	Parameter error (required item undefined)
94	Parameter error (excessive number of bytes for a definition)
95	A text file, attached file, or list of attached files was not found.
99	An error indicating that the JP1/AJS mail monitoring process does not start
100	Environment settings reading error
101	Environment settings undefined error
102	JP1/AJS installation failure
103	Environment settings information is corrupted.
120	Communication with the JP1/AJS mail monitoring process or mail monitoring service has failed.
121	Sending mail information has failed.
150	Insufficient memory
151	Insufficient resources (for files)
152	Insufficient resources (for processes)
180	Acquiring the end status of a mail sending thread has failed.
182	Sending mail has failed.
255	System error

Table A-4: Return values from email sending jobs (in UNIX)

Return value	Description
0	The job ended normally.
90	Parameter error (error in a definition item)
95	Text file name is not found.
96	No mail address specified error
97	Error in a mail address specified
98	Reading the text file has failed.
126	The email transmission feature is not available.
130	Creating a temporary file to send mail has failed.
131	An incoming signal has interrupted mail sending.
150	Insufficient memory
254	The system command used for mail sending has no privilege for execution.
255	System error

Table A-5: Return values from message-queue message sending jobs (in Windows)

Return value	Description
0	The job ended normally.
1	Connecting to the specified queue has partially failed.
3	A message has been sent to the dead letter queue.
90	Parameter error (error in a definition item)
91	Parameter error (error in a definition)
92	Parameter error (required item undefined)
94	Parameter error (excessive number of bytes for a definition)
95	The specified file is not found.
98	The JP1/AJS2 message queue monitoring process is not available.
99	An error indicating that the JP1/AJS2 message queue monitoring process does not start

Return value	Description
100	Environment settings reading error
101	Environment settings undefined error
102	JP1/AJS2 installation failure
103	The type of the message queuing system used in environment settings is incorrect.
120	Communication with SUP of TP1/LiNK provided by JP1/AJS2 has failed.
121	Connecting to the specified queue has failed.
123	Error in accessing the message queue
150	Insufficient resources (for processes)
151	Insufficient resources (for files)
152	Insufficient resources (for threads)
153	Creating a mutex has failed.
154	`jposupwth` (the MQ monitoring process for linkage with TP1/Message Queue) has stopped.
155	Starting `jposupwth` (the MQ monitoring process for linkage with TP1/Message Queue) has failed.
156	Creating an event has failed.
157	Creating a file has failed.
158	Insufficient memory
159	Sending a message to the dead letter queue has failed.
180	Acquiring the end status of a message sending thread has failed.
181	Acquiring the file size has failed.
182	Reading a file has failed.
183	Closing a file has failed.
250	Another event occurred at startup of the MQSeries service.
255	System error

Table A-6: Return values from message-queue message sending jobs (in UNIX)

Return value	Description
0	The job ended normally.
2	Error with warning during message sending
90	Parameter error (error in a definition item)
91	Parameter error (error in a definition)
95	The specified file is not found.
103	The type of the message queuing system used in environment settings is incorrect.
120	Communication with SUP of TP1/Server Base provided by JP1/AJS2 has failed.
123	Error in accessing the message queue
126	Message queue transmission processing is not available.
150	Insufficient resources (for processes)
155	Stating the MQ monitoring process for linkage with TP1/Message Queue has failed.
156	Creating an event has failed.
157	Creating a file has failed.
158	Insufficient memory
255	System error

Table A-7: Return values from MSMQ message sending jobs

Return value	Description
0	The job ended normally.
90	Parameter error (error in a definition item)
91	Parameter error (error in an operand)
92	Parameter error (error in a definition)
93	Parameter error (required item undefined)
94	Message sending error
95	Fatal error The MSMQ linkage functionality is not available.

Table A-8: Return values from OpenView Status Report job

Return value	Description
0	The job ended normally.
16	Parameter error (error in a definition item)
17	Parameter error (error in an operand)
18	Parameter error (error in a definition)
19	Parameter error (required item undefined)
30	SNMP service not installed (For Windows only)
31	SNMP service not started (For Windows only)
32	SNMP service community name not specified
33	SNMP service trap sending destination not set
34	Message sending error

Table A-9: Return values from local power control jobs

Return value	Description
0	The job ended normally.
Other than 0	Local power supply control has failed.

Table A-10: Return values from remote power control jobs

Return value	Description
0	The job ended normally.
Other than 0	Remote power supply control has failed.

Table A-11: Return values from action jobs (special)#1

Return value	Description
128	The resource is insufficient.#2
-1	Startup failure, forced end, ended abnormally#3

#1

These return values might be set to values other than the return values of job processes of action jobs. Actions jobs include the following:

- JP1 event sending job
- Email sending job
- Message-queue message sending job
- MSMQ message sending job
- OpenView Status Report job
- Local power control job
- Remote power control job

#2

This return value is used only in Windows systems. For details, see *7.3 Notes on using PC jobs*. Because the OS sets this return value, the return value may change depending on the OS version.

#3

For details, see *7.8.3 Checking the return code of a job*.

B. Version Revisions

This appendix lists the changes in each version of the JP1/AJS series programs.

B.1 Revisions in 09-00

The following lists the revisions in 09-00 for each program.

(1) JP1/AJS3 - Manager

- The standard database of JP1/AJS3 is now an embedded database.
- Functions related to an embedded database have been changed as follows:
    - The sizes of the large-scale, medium-scale, and small-scale database models have been changed.
    - The database area auto-increment function and the system log auto-increment function have been added.
    - The system log is no longer used.
    - The functions of the commands used to control an embedded database have been enhanced.
- The ISAM database is now used only for QUEUE jobs and submit jobs.
- An agent management function has been added for specifying a logical execution agent name as the destination host for a job or jobnet. Previously, users could only specify execution hosts by their real names.
- Jobs that are in the *Now queuing* status when the service is stopped are now returned to the *Wait for prev. to end* status when the service restarts (in hot-start mode), before being resubmitted.
- A jobnet release function has been added for replacing the definition of a jobnet that is registered for execution with another definition.
- The job execution control manager process (jpqman) and event/action control manager process (jpomanager) can now be started on a scheduler service basis.
- A scheduler log file can now be output for an individual scheduler service or host.
- The following functions have been enhanced:
    - The method by which the number of logs to keep is managed
    - The process by which monitored generations of jobnets with start conditions are established
    - The process by which execution generations when a start condition is established are held

- A format specification has been added to the `ajsshow` command for outputting the standard output file name.
- The Manager Environment Settings dialog box is no longer provided. Instead, you can use the `jajs_config` command to set up the manager environment.
- A function has been added to support end delay monitoring based on how long a job takes to execute.
- The jobnet connector functionality has been enhanced to enable control of the execution order of root jobnets managed by different scheduler services.
- The definition pre-check has been enhanced so that if an invalid execution order is found in the units of the jobnet being checked, the names of the units are output to the check results file.
- The file permission check performed at execution of a Unix job has been enhanced to include checks of the access control list and secondary group settings as well as file permissions.
- A function has been added that enables event jobs to continue executing even if the JP1/AJS3 service stops on the execution host.
- A function has been added for exporting and importing the registration statuses of jobnets as registered execution-schedule information.
- Linkage with message queues on UNIX hosts (TP1/LiNK, TP1/Message Queue, MQSeries) is no longer supported.
- Windows Server 2008 has been added as platforms supported by JP1/AJS3 - Manager.
- A unit called a jobnet connector which controls the execution order of root jobnets has been added.
- An option has been added to output a detailed history of user operations, such as changes to jobnet definitions, to the scheduler log.
- The `ajslogprint` command for extracting log entries from the scheduler log has been added.

(2) JP1/AJS3 - Agent

- The Agent Environment Settings dialog box is no longer provided. Instead, you can use the `jajs_config` command to set up the agent environment.
- Linkage with a message queue system is no longer supported.
- The file permission check performed at execution of a Unix job has been enhanced to include checks of the access control list and secondary group settings as well as file permissions.
- Linkage with message queues on UNIX hosts (TP1/LiNK, TP1/Message Queue,

MQSeries) is no longer supported.

- Windows Server has been added as platforms supported by JP1/AJS3 - Agent.

(3) JP1/AJS3 - View

- An agent management function has been added for specifying a logical execution agent name as the destination host for a job or jobnet. Previously, users could only specify execution hosts by their real names.
- A jobnet release function has been added for replacing the definition of a jobnet that is registered for execution with another definition.
- Function menus have been added to the JP1/AJS3 - View window to facilitate task-oriented operation.
- The JP1/AJS3 - View window (Summary Monitor window) has been added. In this window, you can view the progress of jobnets and other information.
- JP1/AJS3 - View can now be started in the following modes:
    - Normal mode

      In this mode, the JP1/AJS3 - View window is equipped with function menus.
    - Monitoring mode

      A mode dedicated to monitoring jobs and jobnets. Only the JP1/AJS3 - View window (Summary Monitor window) is displayed.
    - Compatible mode

      JP1/AJS3 - View operates in the same way as JP1/AJS2 - View version 8 or earlier.
- A Detailed Information area has been added to the JP1/AJS3 - View window (Main window), which displays detailed information about a unit.
- The concurrent execution setting of monitored generations and the holding behavior of execution generations (produced when a start condition is satisfied) can now be selected in the detailed definition of a start condition.
- A list filter function has been added for filtering the information in a list.
- A function has been added for saving list information in CSV format.
- You can now click a button in the Daily Schedule window and Monthly Schedule window to move between days and months.
- A list area has been added to the Jobnet Editor window and Jobnet Monitor window. This area displays the jobs defined in the jobnet.
- A Search window has been added, in which you can set detailed search conditions and perform operations on units listed in the search results.

- You can now use a mouse wheel to scroll inside JP1/AJS3 - View.
- A function has been added that allows you to select whether **Type** in list areas are grouped by type or displayed in detailed format.
- A function has been added for prohibiting changes to specific definition items in the Define Details dialog box.
- A function has been added for removing icons you no longer use from the icon list area in the Jobnet Editor window.
- Windows 7 has been added as a supported OS (JP1/AJS3 - View 09-00-05 or later).
- A function has been added to support end delay monitoring based on how long a job takes to execute.
- The jobnet connector functionality has been enhanced to enable control of the execution order of root jobnets managed by different scheduler services.
- An option has been added to the Filter Settings dialog box so that jobnets with hold plans can be treated as jobnets in *Being held* status for filtering purposes in the Daily Schedule window and Monthly Schedule window.
- The ability to define, operate, and monitor jobnet connectors which control the execution order of root jobnets has been added.
- A function that displays the preceding and succeeding jobs of a given job or jobnet in bold has been added.
- Support for Windows Vista has been added.

B.2 Revisions in 08-00

The following lists the revisions in 08-00 for each program.

(1) JP1/AJS2 - Manager

- The recommended values for the environment settings are now set during installation and setup.
- A Monitoring Files job can now monitor files larger than 2 gigabytes (large files).
- The `ajsstatus` command can now output the connection status of JP1/AJS2 - View.
- The following commands used to control an embedded database have been added:
    - `ajsembdbaddarea` command (expands a database area in an embedded database)
    - `ajsembdbaddlog` command (expands a log area in an embedded database)
    - `ajsembdbcancel` command (cancels execution of a command

manipulating an embedded database)

- ajsembdboplog command (manipulates embedded database logs)
- ajsembdbreclaim command (maintains an embedded database)
- ajsembdbrorg command (unloads and reloads an embedded database)
- ajsembdbrstr command (backs up and restores an embedded database)
- ajsembdbstart command (starts an embedded database)
- ajsembdbstatus command (monitors an embedded database)
- ajsembdbstop command (stops an embedded database)
- ajsembdbunset command (removes the setup of an embedded database)

With support of the ajsembdbreclaim command, the time required to reclaim free pages has been reduced.

- JP1/Performance Management - Agent Option for JP1/AJS2 can now be linked with JP1/AJS2 to analyze the operating status.
- The jajs_start command and the jajs_start.cluster command can now check whether a process has already been started when JP1/AJS2 is started. (UNIX only)

(2) JP1/AJS2 - Agent

- The recommended values for the environment settings are now set during installation and setup.
- A Monitoring Files job can now monitor files larger than 2 gigabytes (large files).

(3) JP1/AJS2 - View

- Icons have been changed.

B.3 Revisions in 07-50

(1) JP1/AJS2 - Manager

- Macro variables can now be used during registration for execution to specify information to be passed.
- Judgment jobs can now perform variable judgment.
- A function has been added that suppresses jobnet executions that follow an abnormally terminated jobnet and that will be started when their start conditions are satisfied.
- A definition pre-check function has been added for conducting a final check before starting production in the production environment after the unit definitions are migrated from the development environment.

- The jpomanevreset command has been added for deleting data accumulated in the event action manager if a large amount of unprocessed data accumulated in the event action manager has caused delay. To identify the start conditions and agents that have caused this problem, the jpomanevshow command has also been added for displaying information about agents that frequently send data to the manager and the start conditions.
- A function that alleviates consumption of the Desktop heap has been added. (Windows only)
- A function has been added for specifying the maximum wait time for the scheduler service to connect to a database.
- Messages that were output to only the integrated trace log can now be output to syslog also. (UNIX only)
- The following functions have been added to the data collection tool:
  - Specifying a logical host name
  - Filtering the data to be collected
  - Adding types of data that can be collected
- Descriptions of messages have been improved.
- An urgent command has been added that can be executed if an error occurs.
- A function has been added that places limits on, for example, the size of files that can be received, to prevent a part of job processing from affecting the entire system operation.
- A function has been added that performs a synchronized write when updating event job information or the wait information file.
- The monitoring interval for linkage with MQ Series can now be specified in seconds.
- If a TCP/IP connection error occurs, the retry interval and count can now be changed.
- The policy to determine the agent hosts to which a job will be dispatched can now be specified.
- All the detailed processes of the event action function can now be stopped to terminate the agent process for the event action function if any of the detailed processes have terminated upon receiving a signal.
- Microsoft(R) Visual C++ .NET Version 2003 is now supported as a compiler for the provided code functions.
- The ajsshow command can now display the hold attribute of a jobnet or job even when the jobnet or job has already terminated.

(2) JP1/AJS2 - Agent

- A definition pre-check function has been added for conducting a final check before starting production in the production environment after the unit definitions are migrated from the development environment.
- The following functions have been added to the data collection tool:
    - Specifying a logical host name
    - Filtering the data to be collected
    - Adding types of data that can be collected
- Descriptions of messages have been improved.
- The monitoring interval for linkage with MQ Series can now be specified in seconds.
- All the detailed processes of the event action function can now be stopped to terminate the agent process for the event action function if any of the detailed processes have terminated upon receiving a signal.
- A function has been added that performs a synchronized write when updating event job information or the wait information file.

(3) JP1/AJS2 - View

- Macro variables can now be used during registration for execution to specify information to be passed.
- Judgment jobs can now perform variable judgment.
- A function has been added that suppresses the jobnet executions that follow an abnormally terminated jobnet and that will be started when their start conditions are satisfied.
- The **Add**, **Change Time**, **Execute Immediately**, and **Release Change** options have been added to the JP1/AJS2 - View window.
- The **Paste (Extension)** menu command has been added for copying units and relationship lines at the same time.
- Relationship lines can now be drawn from multiple units to a single job network element.
- When opening the Jobnet Monitor window of JP1/AJS2 - View from JP1/AJS2 Console View, if there is already an activated JP1/AJS2 - View, the window can now be opened in JP1/AJS2 - View.
- The following functions have been added to the data collection tool:
    - Specifying a logical host name

  - Filtering the data to be collected
  - Adding types of data that can be collected
- Descriptions of messages have been improved.
- The maximum log file size for JP1/AJS2 - View has been increased.
- The maximum log file size for JP1/AJS2 Console View has been increased.
- In JP1/AJS2 - View, log information that previously was output many times in small units can now be output at one time.
- In JP1/AJS2 Console View, log information that previously was output many times in small units can now be output at one time.
- In the Windows version of JP1/AJS2 - View, **Help** has been added to the **Start** menu.

B.4 Revisions in 07-00

Version 07-00 features the addition of a new program, JP1/AJS2 - Advanced Manager, to enable the use of an embedded database (HiRDB) in a JP1/AJS2 scheduler database.

This section explains the changes in each version from 06-71 to 07-00, on a program-by-program basis.

(1) About JP1/AJS2 - Manager

- A function was provided to temporarily compress JP1/AJS2 and reconfigure the ISAM database (scheduler database and job execution environment database) without stopping active applications.
- ISAM databases can now be reconfigured in parallel.
- The number of scheduler services that can be added has been changed from 9 to 20.
- An option was added for outputting the execution timings of reference commands, such as `ajsshow` and the history of service processing requests from operation commands, as the operation log to the scheduler log.
- The number of logs to keep for a jobnet has been changed from 99 to 999.
- For a cold start of JP1/AJS2, the job execution environment database is deleted so that the startup time of JP1/AJS2 becomes shorter.
- A function is now supported for validating the user profile information in the environment setup for job execution control.
- By setting the number of days that job information is held to 0 days, jobs that terminate abnormally can now be handled by changing the save time.
- The JP1/AJS2 job information deletion can now be suppressed.

- Any event job can now be used in a DNS environment (host name in the FQDN format).
- Event job reception information can now be inherited as macro variables as the parameters of standard jobs and action jobs without having to pay attention to double quotation marks in the inherited information.
- The extended regular expression supported by JP1/Base can now be used in Receive event job monitoring jobs, Monitoring log files jobs, and Monitoring event log jobs according to the JP1/Base settings.
- A function to execute queueless jobs is now supported.

(2) About JP1/AJS2 - Agent

- Event job reception information can now be inherited as macro variables of the parameters of standard jobs and action jobs without being aware of double quotation marks in the inherited information.
- A function for executing queueless jobs was supported.
- When JP1/AJS2 - Agent starts, it no longer accesses the authentication server (07-00-/C or later).

(3) About JP1/AJS2 - View

- A user profile can now be used to set the JP1/AJS2 - View environment.
- A line feed character can now be inserted at any point in a unit name displayed in the map area of the Jobnet Editor and Jobnet Monitor windows.
- The default values in the dialog box can now be changed.
- Display items (columns) in the following locations can now be selected.
    - List area in the JP1/AJS2 - View window
    - Execution result list in the Daily Schedule window
    - Execution result list in the Monthly Schedule window

C. Changes in 3020-3-S04-04(E)

The following table list the changes in this manual (3020-3-S04-04(E)).

Table C-1: Changes in 3020-3-S04-04(E)

No.	Location	Changes
1	All	Windows 7 has been added as an OS supported by JP1/AJS3 - View.
2	*6.5(3)*	A description of the JP1 user name used for command operations on agent management information has been added.
3	*7.7.1*	A note on setting up a JP1/Base event service environment has been added to the notes on the Send JP1 Event job.

D. Glossary

abnormal end

A jobnet ends abnormally if one of the processes defined in the jobnet fails to execute properly. The jobnet is interrupted at that point and subsequent processes are not executed.

A job ends abnormally if it fails to execute properly. The process is interrupted at that point.

The embedded database system ends abnormally when an error causes its status to change from active to stopped or paused, without any intervention by the user. For details, see *D. How the Embedded Database Operates* in the manual *Job Management Partner 1/Automatic Job Management System 3 Troubleshooting*.

abnormal threshold

A value that is compared with a job's return code to evaluate whether the job ended normally or abnormally.

action job

A job that sends email, or sends events reporting the system status to JP1/ IM or the HP NNM.

agent host

A host that executes jobs on request from a manager host. JP1/AJS3 - Agent must be installed on the agent host, or since JP1/AJS3 - Manager also provides JP1/AJS3 - Agent functionality, JP1/AJS3 - Manager might be installed on the agent host.

The agent host executes the job on receipt of a job request from the manager host. At completion of the job, the agent host receives the execution result (return value) of the executable file and forwards it to the manager host.

AJS3 unit monitored object

An object for monitoring the status of root jobnets in JP1/AJS3. By defining the monitoring conditions in this object, you can then switch to monitoring mode and monitor the root jobnets.

AJSPATH

An environment variable for defining the paths used by JP1/AJS3. When this environment variable is defined, you do not need to specify the full path when specifying a jobnet name in a command.

backup box

A directory or a folder for storing backup files.

backup file

A file containing the units defined in JP1/AJS3.

base day

A date specified as the starting day of the month in the calendar information.

base time

The time that marks when a day ends and the next day begins in a JP1/AJS3 system. For example, if 8:00 a.m. is set as the base time, the previous day is regarded as lasting until 7:59 a.m.

calendar information

Information about open days and closed days for jobnet execution. You can define calendar information separately for each job group. The calendar information specifies the days on which jobnets in the job group can and cannot be executed. (When the processing cycle falls on a closed day, the jobnet can be executed on another day if a substitute schedule is defined.) For open days, you can specify the base day, base month, and base time.

closed day

A day on which jobnets are not executed. However, if **Execute without shift** is specified, the jobnet will be executed on that closed day.

cluster system

A system configured as multiple linked server systems, designed to continue operation even if one system fails. If a failure occurs in the server currently executing applications (primary node), the other standby server (secondary node) takes over and continues processing the applications. Therefore, a cluster system is also referred to as a *node switching system*.

The term *cluster system* can also mean load balancing based on parallel processing. In this manual, however, *cluster system* refers only to node-switching functionality for preventing interruption of application processing.

common user profile

A file containing the environment settings for JP1/AJS3 - View, accessible to all JP1 users. The system administrator saves the common user profile in JP1/AJS3 - Manager. JP1 users can download this file, enabling the same JP1/AJS3 - View environment to be set for all JP1 users.

A common user profile is useful when a large number of JP1 users will be using JP1/AJS3 - View in the same environment.

compatible ISAM configuration

A system configuration in which JP1/AJS3 information is managed exclusively by the

ISAM database.

This configuration is offered to help users migrate from JP1/AJS2 version 8 or earlier. It can restrict to the same degree as in previous versions, the use of resources such as hard disk and memory. However, from version 9 only a subset of the new features offered is provided.

correlation ID

Information for identifying sent and received messages. The correlation ID is received in the character code set specified by the sender.

custom job

A predefined job for executing a task with a specific purpose. JP1/AJS3 provides standard custom jobs such as file transfer and job requests to a mainframe. In addition, you can register your own frequently used jobs as custom jobs. When registering a custom job, you can represent it by creating an icon with a special shape and design, and you can create a dialog box for entering job information.

To use a custom job, the requisite program for the job must be installed.

Daily Schedule window

A window that displays each day's execution schedules, execution status, and execution results.

default queue

A queue created in an agent host for executing jobs. You must always create a default queue.

When you submit a job for execution, if you specify an agent host name as the destination, the job will be submitted to the default queue of the specified agent host.

dependent job

A job executed when the judgment result of a judgment job is true.

dependent jobnet

A jobnet executed when the judgment result of a judgment job is true.

embedded database

The standard database of JP1/AJS3. An embedded database offers high reliability, and is well suited to large-scale systems that handle large quantities of information.

embedded database administrator (database administrator)

A user authorized to assign and cancel various permissions for an embedded database (a user with DBA permissions).

Database administrators are managed within an embedded database.

embedded database operation commands

A generic term for commands whose name begins with `ajsembdb`.

embedded database service

A service that provides the environment for using the embedded database in Windows. This service must be started before you can use the embedded database. The name of the embedded database service is `JP1/AJS3 Database` *setup-identifier*.

embedded database system administrator

The owner of an embedded database practical directory and embedded database file system areas (data area and system area). The embedded database system administrator can execute commands for an embedded database.

The OS manages embedded database system administrators.

end with warning

A status indicating that a jobnet finished, but some of the processes defined in the jobnet were executed incorrectly. The jobnet continues to the end without interruption.

This ending method is used when an error is not so serious as to terminate the jobnet.

environment setting parameter

A parameter for defining the information required to operate JP1/AJS3, written in an environment settings file. With these parameters, you can specify the directory in which information about JP1/AJS3 units is stored, whether to output syslog messages, and other such preferences.

environment settings file

A file containing the settings required to operate JP1/AJS3, such as the scheduler service environment and job execution environment.

event

A specific event, such as email reception or file update, that occurred in the system. Events can be used to start a job or jobnet, and can be monitored using an event job.

event job

A job that monitors specific events occurring in the system. When an event job is initiated, it starts monitoring for file updates, incoming messages, or other specified events.

execution agent

The logical name of an agent host that executes jobs or jobnets. Based on the agent information defined in the manager host, the manager maps the execution agent specified in the job or jobnet to the physical host name of the agent host, and distributes the job or jobnet accordingly.

execution agent group

A group of execution agents configured to realize load distribution. The manager distributes jobs among the execution agents according to their assigned priorities.

execution ID

A number assigned to an execution schedule of the uppermost jobnet.

execution-locked resource

A means of preventing multiple jobs from executing at the same time, by specifying the same resource name (execution-locked resource name) for each job.

fixed execution registration

A method of registering a jobnet so that it starts and runs at a predetermined date and time calculated by the system from schedule definitions.

fixed schedule

A schedule set by absolute times when a jobnet is registered for fixed execution.

HP NNM

A suite of integrated network management tools from Hewlett-Packard Co. for managing network configuration, performance, and failures.

immediate execution registration

A method for starting and processing a jobnet immediately after registering it for execution.

ISAM database

The database that manages the execution environment for QUEUE jobs and submit jobs. Data is indexed using the Indexed Sequential Access Method (ISAM) and is managed in the database. The ISAM database is provided as standard with JP1/Base.

job

A group of commands, shell scripts, or Windows executable files.

job execution environment

A job execution environment consists of a JP1/AJS3 manager and agents.

The job execution environment for the manager is used to manage the definition information for execution agents (such as the maximum number of concurrently executable jobs and job transfer restriction status), job distribution method, and job execution results.

The job execution environment for the agent is used mainly to manage how a job is executed.

These job execution environments are managed by using a database and environment setting parameters.

When QUEUE jobs and submitted jobs are used, the ISAM database and environment setting parameters are used as the job execution environment for the QUEUE jobs and submitted jobs.

Note that queueless jobs are managed in the queueless job execution environment.

job group

A folder for classifying and managing jobnets.

job network element

The generic term for these elements is *unit*.

jobnet

A set of jobs associated in execution order. When a jobnet is executed, the jobs in the jobnet are automatically executed in their predetermined order.

jobnet connector

A unit for controlling the execution order of root jobnets. A jobnet connector establishes connections between root jobnets and controls their execution order by having connected generations wait for their counterparts to start or finish.

Jobnet Editor window

A window in which you can create new jobnets or edit existing jobnets.

Jobnet Monitor window

A window that displays the execution status or detailed execution results of jobnets or jobs. You can manipulate jobnets or jobs in this window.

JP1 event

Event information that is reported to JP1/Base when an event occurs in the system. JP1 events are reported to other systems via JP1/Base.

JP1 permission level

A name that indicates the operations that a JP1 user is allowed to perform on management targets (resources) defined in JP1/AJS3, including applications and events. Use JP1/Base to define JP1 permission levels.

JP1 resource group

A name given to a specific JP1/AJS3 unit for controlling access by JP1 users to that unit.

JP1 user

A user designation for using JP1/AJS3 or JP1/IM - Manager. Each JP1 user is registered in the authentication server, which controls the user's access to management targets (resources).

JP1/AJS3 - Definition Assistant

This program allows you to register a large amount of JP1/AJS3 definition information edited using an Excel template into a manager host, or to retrieve JP1/AJS3 definition information from a manager host to an Excel template. The Excel templates provided by JP1/AJS3 - Definition Assistant are called *definition management templates*. With a definition management template in the spreadsheet format, you can enter or edit definition information efficiently by using automatic filling, automatic filtering, and other Excel functionalities.

JP1/AJS3 Console Agent

A JP1/AJS3 component that regularly monitors the status of objects (root jobnets) on the local host, specified in JP1/AJS3 Console Manager. Any change in status is notified to JP1/AJS3 Console Manager.

JP1/AJS3 Console Manager

A JP1/AJS3 component that stores definitions about monitored objects defined in JP1/AJS3 Console View, and gets status information about monitored objects by issuing requests to JP1/AJS3 Console Agent.

JP1/AJS3 Console View

A JP1/AJS3 component that allows you to define objects to be monitored, using a graphical user interface. The definitions are stored in JP1/AJS3 Console Manager. Using JP1/AJS3 Console View, you can view and monitor the status of target objects notified by JP1/AJS3 Console Agent to JP1/AJS3 Console Manager. You need to log in to JP1/AJS3 Console Manager before using JP1/AJS3 Console View.

JP1/AJS3 for Enterprise Applications

A program that allows you to control jobs in an R/3 system from another system. You can submit, delete, and monitor R/3 jobs.

R/3 jobs can be executed automatically from JP1/AJS3 if you register them as custom jobs for JP1/AJS3 for Enterprise Applications when you define a JP1/AJS3 jobnet.

JP1/AJS3 for Enterprise Applications is the successor to JP1/Application Manager for R/3.

JP1/AJS2 for Oracle E-Business Suite

A program that allows you to access Oracle E-Business Suite from another system and to request concurrent execution of applications.

Requests for concurrent execution can be issued from JP1/AJS3 if you register the requests as custom jobs for JP1/AJS2 for Oracle E-Business Suite when you define a JP1/AJS3 jobnet.

Using JP1/AJS3's schedule definition facility, you can specify the processing cycles and the execution dates of concurrent requests.

JP1/AJS2 for Oracle E-Business Suite is the successor to JP1/Application Manager for Oracle E-Business Suite.

JP1/Base

A program that provides the event service function. JP1/Base allows you to control the order in which services start, and it lets you send and receive JP1 events. JP1/Base is a prerequisite program for JP1/IM and JP1/AJS3. When JP1/IM is deployed in a system with JP1/AJS3, JP1/Base provides functionality for restricting operations by JP1 users.

JP1/FTP

A program for performing file transfer tasks efficiently, including file transfer/ reception linked to application execution, scheduled file transfer, and automated program execution following file reception. JP1/FTP supports monitoring of transfer status, enhancing file transfer reliability.

JP1/IM

A program for centrally monitoring a distributed system. Using the windows in JP1/ IM - View, the system administrator can monitor JP1 events, which provide information about job execution status or problems in the distributed system.

JP1/NQSEXEC

A program for executing routine batch processing on a distributed system and for running batch jobs efficiently.

JP1/OJE for Midrange Computer

A program for submitting batch jobs to AS/400 from a Windows or UNIX host, or for submitting batch jobs from AS/400 to a Windows or UNIX host.

JP1/OJE for VOS3

A program that links with JP1/AJS3 for executing and monitoring batch jobs between a Windows or UNIX system and a mainframe (VOS3).

JP1/Script

A program for creating and executing scripts (batch files) that control jobs on Windows. Job operation can be automated by linking JP1/Script with JP1/AJS3.

JP1/Software Distribution

A general term for a system that distributes software and manages clients using batch operations over a network.

By linking with JP1/AJS3 using the JP1/Software Distribution command interface, the user can automate software distribution and other tasks.

judgment job

A job that executes a dependent job or jobnet if the judgment result of a specified condition is true.

judgment value

A value for evaluating whether a job ended normally or abnormally.

kill

To forcibly terminate a unit being executed.

When the root jobnet is killed, all the jobs being executed are killed and the jobnets are terminated.

list file

A file containing a list of extracts from sent and received mail.

logical host

A logical server that provides the JP1 execution environment for running a cluster system. If a failure occurs on the primary node, the logical host is switched to the secondary node.

Each logical host has a unique IP address. At failover, the secondary node inherits the IP address. Thus, if the physical server fails, clients can access the secondary node using the same IP address. To the clients, it appears that one server is operating continuously.

macro variable

A variable defined for a succeeding job for referencing information received in an event. By defining a macro variable name in an event job, you can pass the event information to a succeeding job or jobnet.

Specify macro variables in the form `?AJS2`*xxxxxxxxx*`?:`*name-of-information-to-pass*.

mail filtering application

A program or a shell script that converts email formats.

A mail filtering application is required to convert the character set when exchanging email in formats other than RFC822.

mail receipt parameter file

A file containing the mail receipt monitoring parameters defined by the user. The file extension is `.prm`. This file is created automatically when the user defines a Receive Email Event job.

mail send parameter file

A file containing the mail send parameters defined by the user. The file extension is `.prm`. This file is created automatically when the user defines a Send Email Action job.

manager host

A host that manages jobnet definitions and schedule information in a database, and requests agent hosts to execute jobs. You must install JP1/AJS3 - Manager on the manager host.

The manager host creates jobnet execution schedules from the defined schedule information. At jobnet run time, the manager host starts the executable files defined as jobs, forwards the job definitions to an agent host, and requests the agent host to execute the jobs. When execution completes, the execution result is received by the agent host and the database is updated. Based on the updated information, the manager host executes a succeeding job or schedules the next execution of the jobnet.

manager job group

A job group for monitoring JP1/AJS3 - Manager applications from another JP1/AJS3 - Manager.

manager jobnet

A jobnet for monitoring JP1/AJS3 - Manager applications from another JP1/AJS3 - Manager.

MAPI (Messaging Application Programming Interface)

The standard messaging API for Windows.

max. shiftable days

A set number of days within which to shift the next scheduled execution date when the recalculated date falls on a closed day.

maximum number of concurrently executable jobs

The maximum number of jobs that can be executed concurrently.

message ID

One item in an MQSeries message descriptor. Message IDs are stored in the character set specified by the sender. They can be used as storage locations to help identify messages.

MIME (Multipurpose Internet Mail Extensions)

An extended SMTP function used for sending and receiving non-ASCII data.

MIME specifies various procedures, such as how data is to be transmitted between email systems, and the format of control messages for email transfer.

Monthly Schedule window

A window that displays each month's execution schedules and execution results.

nested jobnet

A jobnet defined within another jobnet.

node switching system

See *cluster system*.

normal end

A normal end of a jobnet occurs when all the processes defined in the jobnet have executed correctly and the jobnet has completed.

A normal end of a job occurs when the job has executed correctly.

open day

A day when jobnets run.

physical host

An environment unique to each of the servers (nodes) in a cluster system. When a secondary node takes over from the primary node, the environment of the physical host remains unchanged and is not inherited by the other server.

planned execution registration

A method of registering a jobnet so that it starts and executes according to schedule definitions.

planning group

A unit for switching execution among multiple root jobnets in a planned manner. Directly under a planning group, you can create a number of root jobnets, each defined differently and with differing execution schedules. This enables the root jobnets to be executed automatically in turn, according to the set schedules.

preceding job

A job executed immediately before another job or jobnet.

preceding jobnet

A jobnet executed immediately before another job or jobnet.

processing cycle

The interval between one execution start date and the next execution start date of a jobnet. By defining a processing cycle, you can execute a jobnet at regular intervals.

queue

An area for temporarily keeping jobs registered for execution. Jobs are submitted to the queue in order of registration, and are sequentially transferred for execution to the agent connected to that queue.

The queue controls the number of jobs that the agent executes concurrently, thereby preventing any degradation in performance caused by a large number of jobs being executed at the same time.

queueless job

A job transferred directly from the manager to an agent host for execution, without using a queue. Queueless jobs simplify processing because they are not managed in a queue by the job execution control. As a result, they offer better performance than ordinary queued jobs, allowing more jobs to be executed within a given period of time. However, job execution control functions such as execution agent names and execution agent groups are not available with queueless jobs.

You can define PC jobs and Unix jobs in a jobnet as queueless jobs by specifying **Queueless Agent** as the execution service.

Unless otherwise indicated, the descriptions in this manual apply to jobs for which **Standard** is specified as the execution service.

queueless job execution environment

A queueless job execution environment consists of execution environments for the JP1/AJS3 manager (scheduler service and queueless file transfer service) and queueless agents (queueless agent services). The execution of queueless jobs is managed by using the environment setting parameters for the job execution environment.

Note that the job execution environment must be set up by using the `ajsqlsetup` command before environment setting parameters are set.

queuing job

A job submitted directly to a queue and waiting to be executed.

recovery job

A job to be executed when a job or jobnet ends abnormally.

recovery jobnet

A jobnet to be executed when a job or jobnet ends abnormally.

schedule by days from start

A schedule defined for recalculating the next scheduled execution date, using as the base day the next scheduled execution date determined from the execution start time, processing cycle, and substitute schedule for closed days.

schedule information file

A text file containing schedule information parameters, entered by command when setting fixed execution registration for a jobnet.

schedule rule

Jobnet information such as execution start time and processing cycle. Up to 144 schedule rules can be defined for a single jobnet.

scheduler service

A service that manages the schedules for jobnet execution, and executes processes according to those schedules. Each scheduler service manages all the units in the root job group whose name matches the scheduler service name.

Multiple scheduler services can be activated in a single manager. This allows root job groups to be managed individually. For example, if you start a separate scheduler service for each application, each scheduler service can run its specific application (jobnet and jobs) in parallel, independently of the other scheduler services.

shift days

A set number of days within which to determine a substitute date when the next execution date falls on a closed day.

shutdown status

A situation in which a jobnet fails to start or end due to an error, and the execution status or the next scheduled execution cannot be verified. If this happens, you must cancel and then re-register the jobnet for execution.

SMTP (Simple Mail Transfer Protocol)

A protocol, generally used in UNIX networks, for transferring ASCII data by TCP/IP between heterogeneous systems.

standard configuration

A system configuration in which JP1/AJS3 information is managed by the embedded database.

Unless otherwise indicated, the descriptions in this manual relate to a system in a standard configuration.

Note that the ISAM database is still used to store some information related to QUEUE jobs and submit jobs.

start condition

A definition of the conditions under which a jobnet starts when the jobnet is driven by a specific event.

subject

A character string written in the subject line of an email message. Non-ASCII characters are supported in JP1/AJS3, but might not be supported in connected email systems.

submit

To request the system to execute a job.

submitted job

A standard job registered using the `jpqjobsub` command.

substitute schedule

A means of executing a jobnet on a different day when the next execution date, determined from the jobnet schedule, falls on a closed day.

succeeding job

A job executed immediately after another job or jobnet.

succeeding jobnet

A jobnet executed immediately after another job or jobnet.

suspend

To suppress the execution of the root jobnet and lower units.

When you change a definition under a root jobnet that has been registered for execution, you should suspend the root jobnet to prevent erroneous operation such as the execution control processing running with the old definition. By suspending the root jobnet, the redefinition processing can be synchronized with the execution control processing.

threshold

A value for evaluating the termination status of a job. You can define an abnormal threshold and a warning threshold for each job.

timeout period

A time interval after which an executed job is forcibly terminated if there was no response from the job or if it failed to complete during the specified period.

TP1/Server Base

Software for distributing transaction processing and server processing in an open system. JP1/AJS2 uses TP1/Server Base transaction processing.

unit

A generic term for any job network element.

unit definition parameter file

A text file containing unit definition parameters, entered by command when defining the units.

unit ID

A unique number allocated to a unit.

warning threshold

A value for evaluating whether a job ended with a warning.

Windows Messaging

A facility that provides an interface for sending and receiving email. Using Windows Messaging, you can manage, access, and share a variety of information such as data received from an online service.

Index

Reader's Comment Form

We would appreciate your comments and suggestions on this manual. We will use these comments to improve our manuals. When you send a comment or suggestion, please include the manual name and manual number. You can send your comments by any of the following methods:

- Send email to your local Hitachi representative.
- Send email to the following address:
  WWW-mk@itg.hitachi.co.jp
- If you do not have access to email, please fill out the following information and submit this form to your Hitachi representative:

Manual name:	
Manual number:	
Your name:	
Company or organization:	
Street address:	
Comment:	

(For Hitachi use)