User's Guide

Version 6.0.1

October 2026

ADABAS & NATURAL

This document applies to Version 6.0.1 and all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 2021-2026 Software GmbH, Darmstadt, Germany and/or its subsidiaries and/or its affiliates and/or their licensors.

The name Software AG and all Software GmbH product names are either trademarks or registered trademarks of Software GmbH and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

Detailed information on trademarks and patents owned by Software GmbH and/or its subsidiaries is located at https://softwareag.com/licenses.

Use of this software is subject to adherence to Software GmbH's licensing conditions and terms. These terms are part of the product documentation, located at https://softwareag.com/licenses and/or in the root installation directory of the licensed product(s).

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third-Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software GmbH Products / Copyright and Trademark Notices of Software GmbH Products". These documents are part of the product documentation, located at https://softwareag.com/licenses and/or in the root installation directory of the licensed product(s).

Use, reproduction, transfer, publication or disclosure is prohibited except as specifically provided for in your License Agreement with Software GmbH.

Document ID: NOW-ONOPUSERGUIDE-601-20261023

Table of Contents

Preface xv
1 About this Documentation 1
Document Conventions 2
Online Information and Support 2
Data Protection 3
I Using Entire Operations Web Client 5
2 Starting and Ending an Entire Operations Session 7
3 Entire Operations and Operating System User IDs 11
Entire Operations User IDs 12
Operating System User IDs 12
4 User Language 15
Options Menu 16
System Default and User Profile 16
Natural ULANG Parameter 16
5 Online Help and System Messages 17
Online Technical Information 18
Error Messages 20
6 Logging on and off an Operating System Server Node 23
Logon Function 24
Fields: Logon Node 25
Troubleshooting Logon Errors 26
Monitoring the Node Connection Status 27
Logoff Function 28
7 Elements of the User Interface 29
Title Bar 30
Welcome Message and Display of Connection Information 32
Workspaces 33
Activity History 36
8 Common and Global Functions 39
Cancelling Reading 40
Listing Objects 40
Refreshing Object Lists 40
Specifying Filter Criteria 41
Saving a List View as a Report File 42
Quick Access 42
Displaying Objects 45
Copying Objects 45
Deleting Objects 46
Using a Diagram to Monitor and Maintain a Network 46
Importing and Exporting Objects 47
Functions of the Metanode General 47
Monitor Status 48
Show Messages and Requests 49

9 Operating System Classes and Related Operating Systems 51
10 Date and Time Formats 53
Date Display and Input Options 54
Time Display and Input Options 55
II Using the Entire Operations Business Functionality 57
11 Using the Entire Operations Business Functionality 59
Structure of the Application and its Documentation 60
Required Knowledge 60
Authorization 61
Overview of Objects in Entire Operations 61
Overview of the Object-specific Maintenance Functions in Entire Operations 63
Three Steps to an Executable Job Network 64
Using Control and Monitoring Functions for Job Networks 65
Performing ad hoc Actions on Active Jobs in the Active Database 65
Creating Ancillary Objects for a Job Network or Job 66
Using Monitoring and Evaluation Functions 66
Using Sample Networks 66
III Entire Operations Utilities 67
12 Entire Operations Utilities 69
Introduction 70
Extract Log Data to Log Selection File 72
Delete Old Data in Log Selection File 73
Print Account Information from Entire Operations Log 74
Mass Change of Network/Symbol Table Version 74
Monitor or Task Wait Time Modification 75
Monitor Shutdown 76
Monitor Start 76
BS2000 Jobs 76
List or Delete TO-ACTIVATE Command Records 77
Check the Existence of Symbol Table Definitions 79
Mass Change of the Owner and Owner Deletion 79
Mass Change of Access Rights Granted to Networks 82
User ID Mass Update in Network and Job Definitions 84
Mass Update for User Access to Nodes 86
Mass Change of Node Numbers 87
Mass Logon Processing in Batch Mode 88
Bulk Execution of MACRO Commands 90
Data Migration to the current Entire Operations Version 91
IV Owner Maintenance 93
13 Owner Maintenance 95
Available Functions: Owner 96
Listing Owners 97
Owner at Logon 99
Linking Additional Owners 99

Owners Granted Access to Individual Networks 99
Adding and Removing an Owner 101
V Network Maintenance 103
14 Use of Job Networks 105
15 Subnetworks 107
16 Maintaining Job Network Definitions 109
Listing all Network Definitions 110
Available Functions: Network Master 112
Adding a Network Definition 114
Displaying a Network Definition 115
Modifying a Network Definition 115
Fields: Network Definition 116
OS Specials - Operating System and Environment Defaults 121
Specifying Recipients for Network Messages 124
Granting Definition: Authorizing Other Users or Owners to Access a Network 125
Long Description - Documenting Your Networks 127
Deleting a Job Network 128
17 Scheduling a Network 131
Viewing a Network Schedule Definition 132
Defining a Network Schedule 133
Processing of Time Frame Definitions 137
Defining and Deleting Time Frames for a Network 139
Defining Dates for Explicit Network Execution 141
Defining Multiple Network Activations 145
Viewing a Network Schedule Definition as a Calendar 148
Producing a Network Start Summary 150
Displaying Next Network Starts - Next Activations 150
Displaying the Network Execution History 155
Deleting a Scheduling Definition for a Single Network 156
18 Maintaining Job Network Versions 157
Versioning of Job Networks 158
Using Network Versions for Activations 159
Handling Network Version Usage Definitions 161
19 Copying Job Network Definitions 165
Fields: Paste Data Network Master 167
20 Viewing and Maintaining a Job Network Diagram 169
Explanations of Diagram Symbols 170
Maintenance Functions for Diagrams 174
Editing and Navigating in the Network Diagram 179
Examples of Diagrams 182
21 Activating a Job Network Manually 187
Fields: Network Activation 190
JCL Check only 191
22 Checking for a Loop in a Job Network 193

23 Applying Network Defaults to Jobs (Mass Update) 195
Columns: Application of Network Defaults to Jobs/Copy Defaults 198
Functions: Application of Network Defaults to Jobs/Copy Defaults 199
VI Job Maintenance 201
24 Use of Jobs 205
25 Maintaining Jobs 207
Listing Jobs 208
Selecting a Range of Job Definitions to be Listed 210
Available Functions: Job Master 210
Displaying and Modifying a Job Definition 211
Creating a Job Definition 218
Writing and Viewing Online Documentation for a Job 219
Deleting a Job Definition 221
Displaying Job Dependencies 221
Activating a Single Job Manually 222
Defining Extended Log Information for a Job 225
26 Defining Job Types and Job Execution Features 229
Available Job Types 230
Defining Job Type Specific Execution Features 232
Defining Operating System Specific Execution Features 235
27 Executing a Job as Dummy or Including Epilog Scripts 241
Job Execution as a Dummy Job 242
Job Execution Including an Epilog Script (UNIX and Windows) 242
28 Using a Dummy Job 243
Permanent Dummy Jobs 244
Temporary Dummy Jobs 244
Excluding a Job from Actual Execution 245
Supervision of Running Jobs 246
29 Defining a Subnetwork 247
Link to the Main Network 249
Subnetwork Activation and Execution 250
Restrictions for Subnetworks 251
Adding or Modifying a Subnetwork Definition 252
Listing Jobs of a Subnetwork 254
30 Defining and Managing JCL for a Job 255
Using Job Control in Entire Operations 256
Dynamic JCL Generation (JCL Location MAC) 257
Job Control for Jobs under BS2000 262
Job Control for Jobs under UNIX 262
Job Control for Jobs under Windows 262
Defining Master JCL for a Job 266
Displaying Master JCL 270
JCL Locations 270
Defining Operating System Dependent JCL Specifications 274
Handling JCL during Job Submission 275

Symbol Replacement in JCL 277
Pregenerating Active JCL 277
31 Editing Master JCL and Natural Sources 279
General Editing Considerations 280
UNIX and Windows: Handling of Tab Characters (H'09') within JCL Lines 282
Starting an Edit Session 283
Handling Macro Sources for Dynamic JCL Generation 287
Usage of Text Objects in JCL 294
Locking of Natural Sources 296
Avoiding Exceptions while Editing JCL Location (NAT, MAC) 297
32 Defining and Managing Job Conditions 299
Use of Input and Output Conditions 300
Maintaining Input Conditions for a Job 302
Adding and Modifying a Master Input Condition 305
Accessing another Network Defined for an Input Condition 310
Input Condition References 310
Global Conditions 313
Input Condition with User Exit 314
Input Condition: Multiple Suffixes 317
Input Condition: File Existence 318
Input Condition: Mailbox 321
Input Condition: Symbol Value 322
Input Condition: BS2000 User Switch 325
Input Condition: BS2000 Job Variable 326
Listing Jobs Linked to an Input Condition 328
Deleting an Input Condition Definition 330
33 Handling Prerequisite Resources for a Job 331
Use of Resources and Resource Allocation 332
Listing Prerequisite Resources Defined for a Job 336
Viewing the Usage of a Prerequisite Resource 337
Displaying, Modifying and Adding a Prerequisite Resource Definition 338
Columns and Fields: Prerequisite Resource Definitions 339
Deleting a Prerequisite Resource Definition 340
34 Defining and Managing End-of-Job (EOJ) Checking and Actions 341
End-of-Job (EOJ) Checking and Actions - General Information 342
Maintaining End-of-Job Events and Actions 346
Adding and Modifying an Event Definition 349
Deleting an Event Definition 356
Operating System Dependent Defaults for Event Checking 356
Creating and Viewing Online Documentation for Events 358
Examples of Event Definitions 360
Handling End-of-Job Actions 368
End-of-Job Actions after Execution as a Temporary Dummy Job 372
Defining Output Condition Actions 373

Defining Actions for Symbol Value Modification 377
Defining Actions for Modifying Job Variable Values 380
Defining Action User Exits 382
Defining Other Actions: Deactivate Job Automatically 384
Defining Network and Job Activation Actions 385
Defining Recovery Actions 387
Defining SYSOUT Actions 392
Defining Notification Messages 393
Defining Release Actions for Kept Resources 404
35 Scheduling a Job 407
Defining Scheduling Parameters for a Job 408
Maintaining Schedule Dependencies for a Job 412
Sending a Late Message to One or More Users 417
36 Passing Files to Entire Output Management 419
Basic Requirements for File Transfers to Entire Output Management 420
Listing Files Defined for Entire Output Management 421
Adding and Modifying File Definitions 423
Deleting File Definitions 427
Handing Over SYSOUT and Files to Entire Output Management 428
37 Viewing Job/Network Accounting Information 433
Displaying Accounting Data 434
Fields: Range Specification for Accounting Data 436
VII Active Job Network and Active Job Maintenance 437
38 Meaning and Use of Active Job Networks and Active Jobs in Entire Operations 439
39 Activation of Networks or Jobs 441
Terminology 442
Manual Activation 443
Automatic (Scheduled) Activation 444
Start of Job Activation 445
Run Number 445
Cleanup of the Active Database 446
Cleanup of the Active Database in Batch Mode 447
40 Prerequisite Check before Job Submission 451
Order of Prerequisite Checking 452
Passive Wait 453
Prerequisite Check according to the Round-Robin Procedure 455
41 Messages in Active Jobs Lists 457
42 Maintaining Active Job Networks 461
Available Functions: Network Active 462
Listing Active Job Networks 462
Deactivating Active Job Networks 464
Viewing the Execution History of an Active Network 465
Maintaining Active Runs 466
43 Maintaining Active Jobs 471

Available Functions: Job Active 472
Listing Active Jobs 474
Listing Jobs of an Active Subnetwork 478
Adding a New Job to the Active Network 479
Modifying the Latest Start Time for an Active Job 481
Choosing the Job ID of a Job for Logging 482
Cancelling, Holding and Releasing Active Jobs 483
Resubmitting Active Jobs 485
Deactivating a Job in an Active Network 488
Reactivating an Active Job 490
Displaying and Modifying an Active Job Definition 491
Modifying EOJ Checking and Actions 493
Viewing Long Descriptions of Active Jobs 494
Displaying Prerequisites for Active Jobs: Waiting for 495
Viewing and Modifying Resources Used by Active Jobs 496
Browsing Active Job SYSOUT 498
44 Maintaining Active Job Conditions 507
Available Functions: Condition Active 508
Listing Active Conditions 509
Viewing an Active Condition and Changing Its Status 511
Adding an Active Condition 511
Using Global Active Conditions 512
Displaying Conditions for an Active Job 516
Viewing Job Usage of an Active Condition 516
45 Maintaining Active JCL (Job Control Language) 519
Browsing Active JCL 520
Editing Active JCL 521
Release Edit Lock 522
Viewing the Extended Log for Active JCL Changes 523
Exchanging Active JCL 524
Regenerating Active JCL 524
VIII Schedule Maintenance 527
46 General Scheduling Considerations 529
Use of Schedules 530
Possible Schedule Definitions 530
Schedule Extraction Times 532
Manual and Automatic Activations on the Same Day 532
Multiple Network Activations 532
Influence of Deactivations on Schedules 532
Imported Schedules 533
Schedule Dependencies across the Turn of the Year 533
Using Calendars 533
47 Maintaining a Schedule Master 535
Listing Schedules 536
Displaying, Adding or Modifying a Schedule 537

Deleting a Schedule 548
IX Calendar Maintenance 551
48 Calendar Maintenance 553
General Rules and Restrictions 554
Available Functions: Calendar 555
Listing Calendars 556
Displaying, Modifying or Adding a Calendar Definition 557
Defining Workdays and Holidays 559
Where Used - Listing Schedules Using a Calendar 560
Deleting a Calendar Definition 561
X Mailboxes 563
49 Mailboxes 565
Concept of Single or Multiple Mailbox Users 566
Listing and Maintaining Mailboxes 567
Viewing Mailbox Messages 567
Handling Messages and Requests 570
XI Symbol Table and Symbol Maintenance 573
50 Purpose and Use of Symbol Tables and Symbols 575
Symbol Tables 576
Symbols 577
Handling Active Symbol Tables and Active Symbols 579
Subnetworks and Recovery Jobs 580
User Exits for User-Specific Symbol Maintenance Tasks 580
51 Maintaining and Finding Symbol Tables 581
Available Functions: Symbol Table 582
Listing Symbol Table Masters Defined for an Owner 584
Listing Active Symbol Tables 585
Listing Usable Symbol Tables 585
Listing Jobs and Networks that Use a Symbol Table 587
Determination and Activation of Necessary Symbol Tables 590
Displaying and Modifying a Symbol Table 591
Adding a Symbol Table Master 594
Versioning of Symbol Tables 595
Maintaining the Usage of Symbol Table Versions 599
Saving Symbol Tables as Files 603
Deleting a Symbol Table Master 604
52 Defining Symbols and Symbol Values 605
Available Functions: Symbol 606
Listing Symbols of a Symbol Table 607
Displaying a Symbol 607
Modifying a Symbol 608
Adding a Symbol 612
Defining Multiple Symbol Values 613
Specifying a Range Check for Numeric Symbol Values 615
Reserved Symbols 616

Predefined Symbols 620
Symbols in Node Definitions 627
Validating Symbol Values with a User Exit 628
Global Symbol Modification Exit 629
Symbol Setting triggered by the SYSOUT of a Job 629
Deleting a Symbol 630
53 Symbol Replacement 631
Symbol Escape Characters 632
Symbol Prompting during Network or Job Activation 635
Displaying Descriptions of Prompted Symbols 638
Repeating Symbol Prompting 639
Specifying User Exits for Symbol Modification 639
Rules and Restrictions for Symbol Replacement 642
54 Functions for Symbol Replacement 649
Results Returned by a Symbol Function 650
Function !D or ?D - Date Calculation and Date Formatting 651
Function !E or ?E - Date from Date Calculation 658
Functions !MV or ?MV and !MM or ?MM - Access to Multiple-Value Symbols 659
Function !TIMN or ?TIMN - Constant Time Values 661
XII Log Information 665
55 Log Information 667
Displaying Logged Information - Browse Log Function 668
Displaying Extended Log Information 671
Monitoring Entire Operations Activities 672
XIII Reporting 675
56 Reporting 677
Report Types 678
Viewing the Report Status List 681
Generating or Regenerating Online Reports 682
Viewing Report Properties and Deleting a Report 683
Fields and Columns: Reporting 684
Determination Date for Report Data 688
Report Output Options 688
Using Reports with Bar Charts 689
Using Reports with Monitor Task Time Tables 692
User Restrictions for Reports 694
Retention Period for Reports from Entire Operations GUI Client 695
Examples of Reports 695
Generating Batch Reports 709
XIV Cross-References 711
57 Cross-References 713
Types of Cross-Reference Reports 714
Generating Cross-Reference Reports Online 715
Fields and Columns: Cross-References 722

Examples of Cross-Reference Reports 727
Generating Cross-Reference Reports in Batch 730
XV API Routines 731
58 Purpose and Use of Entire Operations APIs 733
Features Provided by Entire Operations APIs 734
Locating and Implementing an API 734
API Usage Rules and Restrictions 735
Testing Available API Routines 736
API-specific Parameter Definitions 737
59 Available Entire Operations API Routines 739
EORUCB1N: Check Use of BS2000 User IDs 740
NOPFB2-N: Generate SYSOUT File Names for BS2000 740
NOPMLA1N: Start and Stop the Monitor Activity Log 743
NOPU--1N: Read Network Available to a Specific User 744
NOPU--2N: Return all Usable Symbol Tables for a Network 745
NOPU--3N: Get Correlation ID for an Activated Network 746
NOPU--4N: Store New Event in Entire Operations System File 746
NOPUAC5N: Activate Job Networks or Jobs 747
NOPUAS1N: Retrieve Numbers of Active Jobs in Defined Status Ranges 752
NOPUCN3N: Access Entire Operations Conditions 753
NOPUCS1N: Access Calendars and Schedules 754
NOPUJI4N: Import Operating System Jobs into the Active Queue 758
NOPUJS2N: Job Schedule Inquiry and Modification 760
NOPULW9N: Write Messages to System Automation Tools Log 761
NOPUMI1N: Set/Reset Text Milestones in Master and Active Jobs 763
NOPUMT3N: Expand Message Texts 764
NOPUNI1N: Invalidate Entire System Server Node Table Entries 766
NOPUNX1N: Entire System Server Calls to Access UNIX and Windows Files 766
NOPURE2N: Handle Resource Allocations 780
NOPURS1N: Access Entire Operations Resource Masters 784
NOPUSN2N: Inquire Calling Job or Called Network for Subnetworks 785
NOPUSP3N: Display Long Texts for Symbol Prompting 787
NOPUST3N: Inquire Network and Job Status, Symbol Table 788
NOPUSY7N: Access Entire Operations Symbols 792
NOPUVI2N: Obtain Entire Operations Version Information 797
NOPUXD1N: Maintain End-of-Job User Exits for a Network 798
NOPUXI0N: Add Input Condition to an Activated Job 800
XVI User Exits 801
60 User Exits 803
What User Exits Can Do 804
Global User Exits 805
Front-end User Exits 805
Common User Exit Parameter Data Area NOPXPL-A 805

Generation of Dynamic JCL and SYSOUT File Names (BS2000 only) 812
User Exits for Resource Master Determination 812
User Exits for Setting Input Conditions 812
User Exits for End-of-Job Checking and Actions 813
User Exits for Symbol Functions 815
User Exits for Validation Checks of Symbol Values 818
Reserved Symbols for UNIX and Windows Environment Variables 822

Preface

This documentation contains all information relevant to the user of the Entire Operations system.

Before reading this document, you should be familiar with the *Concepts and Facilities* documentation, which defines and briefly describes the system entities and facilities referred to in this document.

Fundamentals of Usage	
Using Entire Operations Web Client	Provides details on using the application's access and presentation functionality (web-baseduser interface, help system, etc.).
Using the Entire Operations Business Functionality	Brief introduction to the structure and use of the application's business logic, i.e. the operation planning and control functions provided by Entire Operations to define, update and execute job networks, jobs, and ancillary objects.
Functions of the Metanode General	Describes common functions for metanode General.
Entire Operations Utilities	Describes Entire Operations utilities.
Object-specific Maintenance and Control Functions	
Owner Maintenance	Explains how to maintain and assign owners.
Network Maintenance	Explains how to define and maintain job networks.
Job Maintenance	Explains the job maintenance facility and how to define different job types and job dependencies, create and use JCL, create online documentation for jobs, define input conditions and specify resources. Explains how to define and manage End-of-Job event checking and trigger actions after a job has terminated.
Active Job Network and Active Job Maintenance	Provides information on the maintenance functions you can perform on active jobs and networks.
Maintenance Functions for Ancillary Objects	
Schedule Maintenance	Describes how to define schedule objects, and how to schedule a job network and a job.
Calendar Maintenance	Explains the calendar maintenance facility that can be used to create calendars which are the basis of schedules for jobs and job networks.
Mailboxes	Describes how to use mailboxes.
Symbol Table and Symbol Maintenance	Provides information on symbol tables, symbol maintenance and symbol replacement functions.
Monitoring and Evaluation Functions	
Log Information	Describes how to display logged information and monitor Entire Operations activities.
Reporting	Explains the Entire Operations reporting facility, which makes a number of different reports available: information at the job and event level,

Fundamentals of Usage	
	information on job network definitions and schedules and overviews of schedules and planned activations.
Cross-References	Describes functions that provide information about the use of objects in Entire Operations.
Interface Programming	
API Routines	Provides information on Application Programming Interfaces (APIs) supplied to access Entire Operations from other applications.
User Exits	Explains how to work with user exits.

1 About this Documentation

- Document Conventions 2
- Online Information and Support 2
- Data Protection 3

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
`Monospace font`	Identifies service names and locations in the format *`folder.subfolder.service`*, APIs, Java classes, methods, properties.
`Italic`	Identifies: Variables for which you must supply values specific to your own situation or environment. New terms the first time they occur in the text. References to other documentation sources.
`Monospace font`	Identifies: Text you must type in. Messages displayed by the system. Program code.
{ }	Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols.
|	Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the | symbol.
[ ]	Indicates one or more options. Type only the information inside the square brackets. Do not type the [ ] symbols.
...	Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...).

Online Information and Support

Product Documentation

You can find the product documentation on our documentation website at **https://documentation.softwareag.com**.

Product Training

You can find helpful product training material on our Learning Portal at **https://learn.softwareag.com**.

Tech Community

You can collaborate with Software GmbH experts on our Tech Community website at **https://techcommunity.softwareag.com**. From here you can, for example:

- Browse through our vast knowledge base.
- Ask questions and find answers in our discussion forums.
- Get the latest Software GmbH news and announcements.
- Explore our communities.
- Go to our public GitHub and Docker repositories at **https://github.com/softwareag** and **https://hub.docker.com/publishers/softwareag** and discover additional Software GmbH resources.

Product Support

Support for Software GmbH products is provided to licensed customers via our Empower Portal at **https://empower.softwareag.com**. Many services on this portal require that you have an account. If you do not yet have one, you can request it at **https://empower.softwareag.com/register**. Once you have an account, you can, for example:

- Download products, updates and fixes.
- Search the Knowledge Center for technical information and tips.
- Subscribe to early warnings and critical alerts.
- Open and update support incidents.
- Add product feature requests.

Data Protection

Software AG products provide functionality with respect to processing of personal data according to the EU General Data Protection Regulation (GDPR). Where applicable, appropriate steps are documented in the respective administration documentation.

I Using Entire Operations Web Client

This chapter primarily describes how to access an Entire Operations online session using the Entire Operations Web Client and how to use the user interface and basic technical functions provided within the Entire Operations application infrastructure.

Starting and Ending an Entire Operations Session
Entire Operations and Operating System User IDs
User Language
Online Help and System Messages
Logging on and off an Operating System Server Node
Elements of the User Interface
Operating System Classes and Related Operating Systems
Date and Time Formats

Related Topic

Using the Operations Business Functionality

2 Starting and Ending an Entire Operations Session

To start an Entire Operations session:

1 Start your web browser and enter the following into the address bar:

```
https://war-file-name/now_tomcat/NOPWEB_Login
```

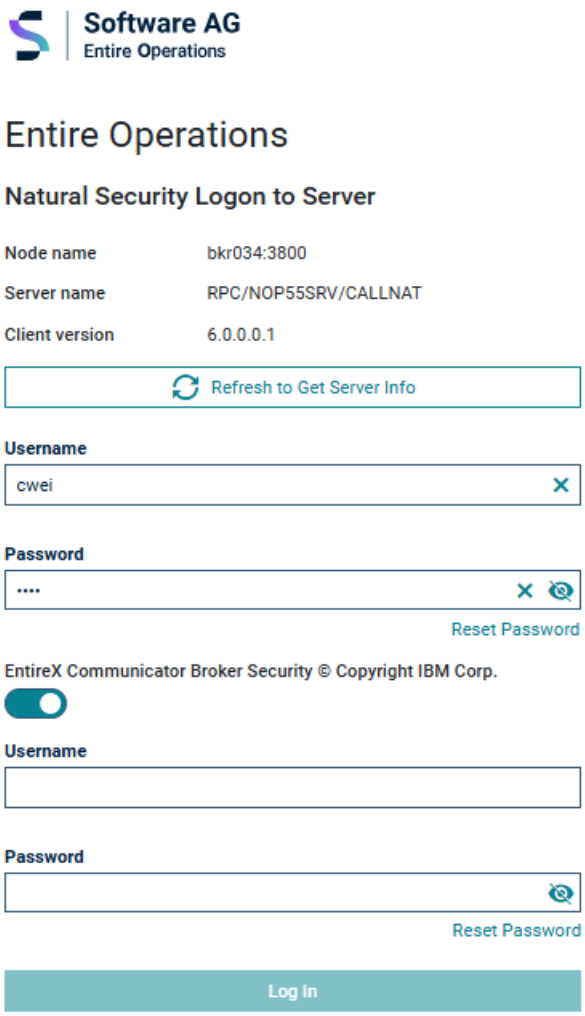
where *war-file-name* is an arbitrary name that was set during the installation.

The website with the Entire Operations login dialog is displayed.

In the box at the top right of the page, the current user language is displayed. By default, this is the language that is set in the browser, for example **English**.

2 To switch to German, click the arrow down symbol (▾) and choose **German**.

The Login screen is displayed in the user language you have chosen.



3 Enter your login details:

- **Username**
  Mandatory. User ID for Natural Security.
- **Password**
  Mandatory. Password for Natural Security.
- **EntireX Communicator Broker Security © Copyright IBM Corp.**
  Optional: **User ID** for EntireX Broker Security.

  Activate the button ( ) to enter your credentials in **Username** and **Password**.

Notes on submitting the password:

- By default, the password is not displayed. By clicking the visibility off symbol () the password is displayed. By clicking again, the display of the password is being suppressed again.
- If necessary, choose **X** to change your password or **Reset Password**.

4 Choose **Log In** to confirm your input.

The **home page** appears. The **user language** is English.

For further information, please refer to **home page** in chapter *Elements of the User Interface* .

Terminating an Entire Operations session

To terminate an Entire Operations session, choose the user symbol ( *user-id*) in the right site of the title bar. Note that *user-id* is a placeholder for the **user ID** of the current Entire Operations user.

Choose **Logout**.

After the logout, the **Login screen** is displayed again.

3 Entire Operations and Operating System User IDs

- Entire Operations User IDs 12
- Operating System User IDs 12

Entire Operations User IDs

In Entire Operations, a user ID can be used to enter the system. Entire Operations user IDs should, but need not be defined to the host TP monitor.

Multiple users can log on to Entire Operations with the same user ID and password at the same time. For reasons of data security and in order to trace data modifications, however, each user usually has a personal user ID and password.

Entire Operations user IDs are relevant to the following:

- **Entire Operations User Profiles**
  Each Entire Operations user ID can have individual access rights to Entire Operations functionality and Entire Operations objects. For details, see *User Definitions and Profile Settings* in the *Administration* documentation.
- **Mailboxes**
  A user ID can be associated with up to ten mailboxes through which the user is notified of any pending logical conditions linked to those mailboxes (see the section ***Working with Mailboxes***).
- **Logging**
  Entire Operations logs all activities and events occurring within the system, including user activities.

A user ID always has a link to at least one owner (see the section ***Using Owners***).

Operating System User IDs

This document covers the following topics:

- Working with Entire System Server Nodes
- Logon to an Operating System User ID
- Operating System User ID, Group, Domain

- Default User ID Determination

Working with Entire System Server Nodes

If you want to work with operating system objects (e.g. editing JCL), you must perform Entire System server logons to the nodes you want to work with. After such a logon, you have access the access rights of the operating system user ID you specified. See the section ***Logging on and off an Operating System Server Node***.

For **Entire Operations networks** and **jobs**, you must define operating system user IDs specifically as JCL user IDs and submit (execution) user IDs. See also ***Defining Operating System Dependent JCL Specifications*** in the section *Job Maintenance*.

Logon to an Operating System User ID

If you want to work with an operating system object and if you are not logged to the defined Entire System Server node, the node logon screen is presented automatically in many cases.

You may also perform an explicit node logon by using the `LOGON NODE` direct command.

You can use the direct command `STATUS NODES` to view your current node logon status.

Operating System User ID, Group, Domain

In network and job definitions, it is possible to specify

- a JCL user ID for the **JCL node**;
- a submit (execution) user ID for the **execution node**.

Additionally, it is possible to specify a group (UNIX), respective a domain (Windows).

If no group is defined for a UNIX node, the user ID's default group will be in effect.

If no domain is specified for a Windows node, the user ID is treated as a local user. If you enter the node's host name in the domain (group) field, the user ID is treated as local user too.

Default User ID Determination

- Determination Rules
- Search Hierarchy for Submit User IDs

- Symbol Replacement

Determination Rules

If no operating system user ID definition is made for **JCL node** or **execution node** locally, Entire Operations determines an operating system user ID, depending on

- The global submit security user type;
- The node-specific submit security user type.

For detailed information, see the relevant sections in the *Administration* documentation.

Search Hierarchy for Submit User IDs

If an operating system user ID other than the user ID of the Entire Operations Monitor (**Submit Security User Type** = `M`/User ID of the Entire Operations Monitor) is to be used, a search hierarchy for the operating system user ID is in effect. The fields **Monitor User ID** and **Submit Security User Type** are described in *Fields: Monitor Defaults - General* in the *Administration* documentation.

The search order is:

1. The job's (JCL or submit) user ID;
2. The network's (JCL or submit) user ID;
3. The node's default user ID (mainframe, UNIX and Windows).

Symbol Replacement

This applies to the network master definition, job master definition and job active definition.

Symbol replacement is possible in the fields:

- **JCL User ID**
- **JCL Group**
- **Submit User ID**
- **Submit Group**
- **SYSOUT User ID** (BS2000 only)

If the **activation escape character** is used, the replacement is performed at activation time. This is required for JCL User ID and JCL Group. If the **submission escape character** is used, the replacement is performed before job submission. Symbol replacement errors in one of these fields are treated as permanent errors.

4 User Language

- Options Menu 16
- System Default and User Profile 16
- Natural ULANG Parameter 16

In Entire Operations, the languages English and German are available. The specified language controls the display in the following locations:

- The nodes of the tree view and the context menu functions in the **object workspace**;
- The field and columns in the open windows in the **content pane** and in the **result list**.



Note: Entire Operations log messages are saved independent of the language. You can view them in English or German.

This section describes the locations where you can change the user language depending on your authorizations:

Options Menu

From the **Options** > **Language** menu of the main application window, select **English** or **German**.

The language settings are kept for future Entire Operations sessions.

System Default and User Profile

As an administrator, you can specify the language in the following locations:

- On the **Character interface settings** page of the Entre Operations Defaults window.
- In a user profile on the **Main** page of the **Maintenance/Create new User** window: see *Viewing, Adding and Modifying a User*.

  See the relevant sections in the *Administration* documentation.

Natural ULANG Parameter

The Natural `ULANG` profile parameter controls the language used by the Entire Operations Monitor such as the SYSOUT of the Monitor tasks and the output.

You can specify `ULANG` dynamically at the start of a Natural session or, if authorized, statically in the Natural NATPARM parameter file.

`ULANG` is described in the *Parameter Reference* of the *Natural* documentation.

5 Online Help and System Messages

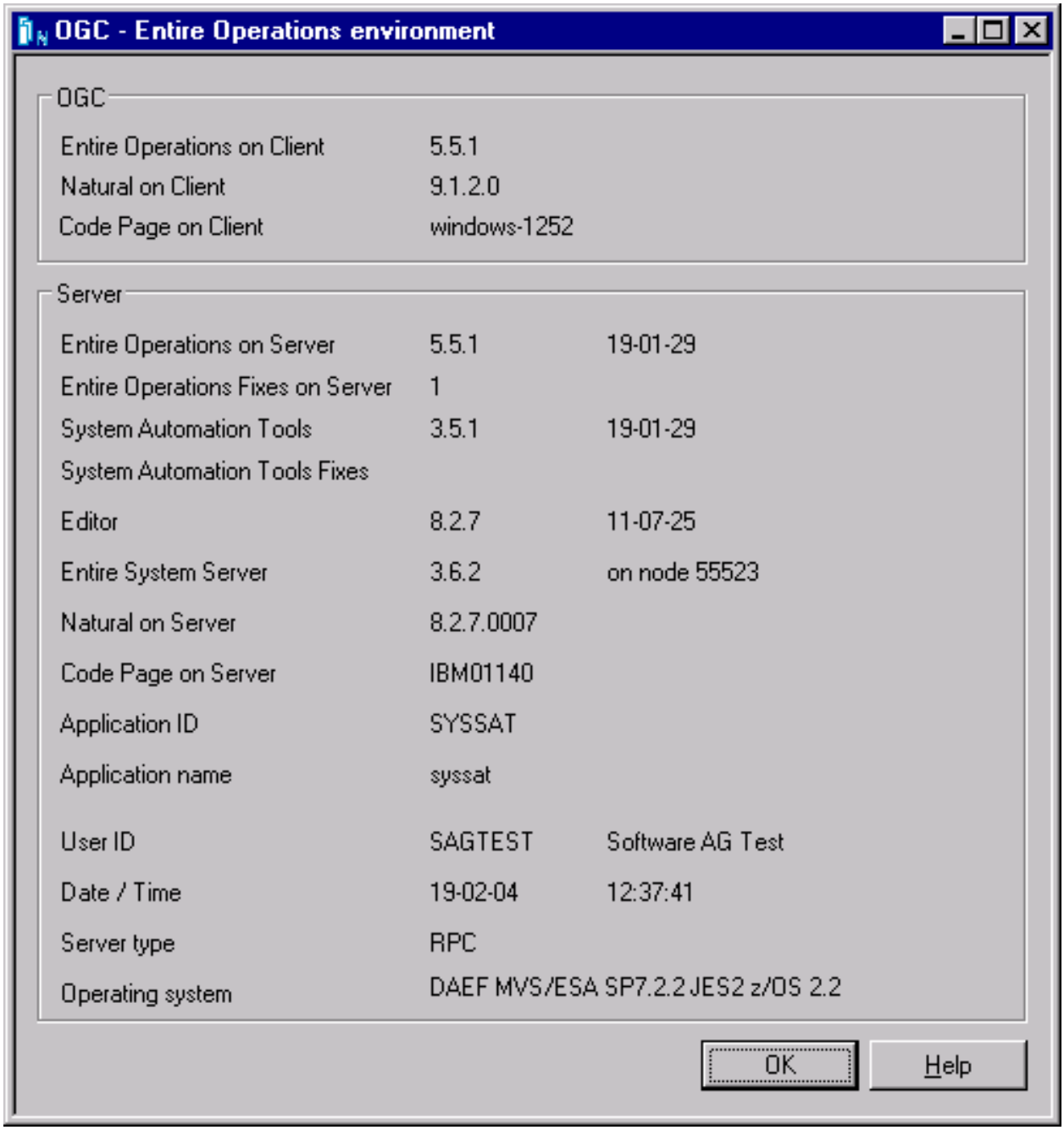
- Online Technical Information 18
- Error Messages 20

You can access the *Entire Operations Web Client* documentation online to obtain general usage information or help on a specific function.

The *Entire Operations Web Client* documentation (including PDF books) is also available at Software GmbH's Empower web site at ***https://empower.softwareag.com/*** (Empower login required).

Online Technical Information

You can use the `TECH` direct command to display technical information on your current Entire Operations session as shown in the following example:



For further information, see *TECH* in the *Direct Commands* documentation.

Error Messages

- Help for System Messages
- Finding System Messages

Help for System Messages

You can obtain additional information on the system messages that can appear in the Entire Operations version installed at your site.

To display the long text for a short system message

- Enter the `HELP` direct command in the **Command** input field for the message (here: `EOR0024 - Invalid Date Format`) received:

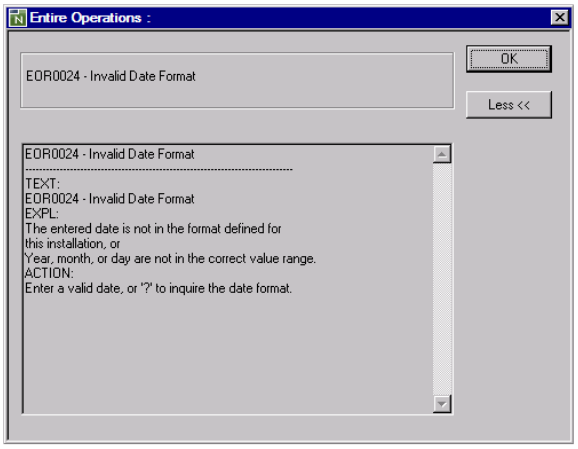
  For example:

  ```
  HELP EOR0024
  ```

  (`EOR`*nnnn* for Entire Operations messages or `NAT`*nnnn* for Natural messages)

  Press ENTER.

  A help dialog opens with additional explanations and (if relevant) advice on how to correct an error:

  For further information on the `HELP` direct command, see the relevant section in the *Direct Commands* documentation.

Related Information:

- ***Results Window*** and ***Status Bar***

Finding System Messages

Entire Operations Web Client displays status or error messages at the following locations:

placeholder

6 Logging on and off an Operating System Server Node

- Logon Function 24
- Fields: Logon Node 25
- Troubleshooting Logon Errors 26
- Monitoring the Node Connection Status 27
- Logoff Function 28

This section provides instructions for logging on or off an Entire System Server node.

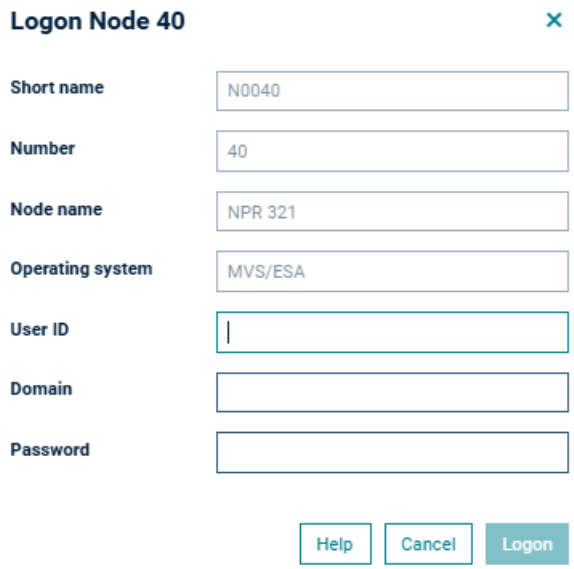
Logon Function

To log on a node

1 In the **object workspace**, choose **Node** ( ).

2 From the submenu, select a node instance and choose **Logon** ( )from the context menu.

A **Logon Node** dialog such as the following opens:



In the input fields, enter the required information. All fields are described in *Fields: Logon Node*.

3 Choose **OK**.

Fields: Logon Node

The **Logon Node** window contains the following fields:

Field	Description
Short name	Short node name.
Number	Node number.
Node name	Logical (long) node name.
Operating system	Operating system on node.
User ID	Valid user ID. If you logon to a mainframe node, input text is converted to uppercase letters. If you logon to a UNIX or Windows node, input text is not converted to uppercase letters. Therefore, make sure an exact spelling in mixed case. See also *Field Defaults*.
Domain	Input field for Windows or UNIX nodes: Enter a Windows domain or UNIX group, respectively. ■ Windows: If this field is left blank, the logon will be carried out without domain. Otherwise, this field must contain a valid domain. The default user ID and domain can be used (see the *Administration* documentation) for combinations of Entire Operations users and operating system servers (nodes). ■ UNIX: If this field is left blank, the user's default UNIX group (from `/etc/passwd`) is used. Otherwise, this field must contain one of the UNIX groups, which is visible in the output of the UNIX groups command.
Password	Input field for the password of the user specified in the **User ID** field. **Logon to BS2000 nodes:** The password can have up to 32 characters. Input text is converted to uppercase letters. **Logon to z/OS nodes:** This field can be used for both a password (up to 8 characters) and a password phrase (9 to 32 characters). A password is converted to uppercase letters depending on the setting of the password mode for the node (see the *Administration* documentation). A password phrase is always treated as a case-sensitive text string.

Field	Description
	Note: You can resize the window to enter a long text string. **Password restrictions:** For user passwords used for nodes in ESM products, some restrictions apply. ■ German Umlauts must not be used. ■ The characters | (pipe), [,], {, }, \ (backslash), ~ (tilde) must not be used (code points 91 - 93, 123 - 126).

Field Defaults

The default user ID - and perhaps also the group or domain - for logging on to a node can be defined for any combination of Entire Operations users and nodes (see *User Definitions and Profile Settings* in the *Administration* documentation). In this case, these values are displayed as defaults during the logon process.

They serve to simplify the logon process. If you want to logon with another user ID and/or group or domain, then these fields may simply be overwritten.

Troubleshooting Logon Errors

If a logon to a node fails, the error message returned by Entire Operations usually indicates that the data entered collides with the Entire Operations user security definitions. For example:

```
EOR0254 - Invalid User ID or Password.
```

The Entire Operations log file can contain additional error information returned from the underlying security system of an operating system, such as RACF for z/OS. For example:

```
Logon failed, Node 146 User ID SAG1

... IRR013I VERIFICATION FAILED. INVALID PASSWORD GIVEN.
```

Monitoring the Node Connection Status

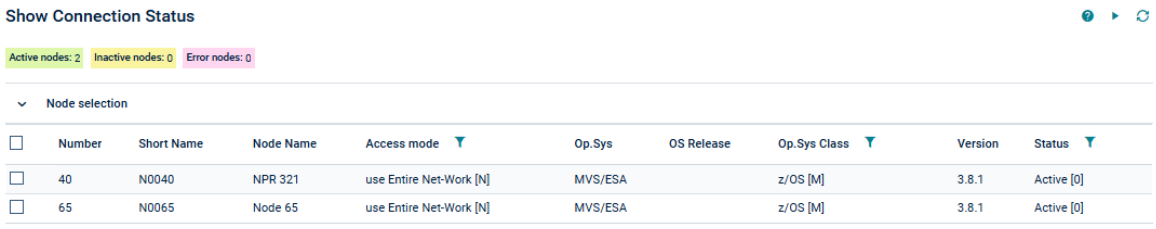
You can view a list with the current status of nodes defined in your environment. In addition, you can add a node to the list of nodes to be monitored.

To monitor the node connection status

1 In the **object workspace**, choose **Node** ().

2 From the submenu, select a node instance and choose **Add to Connection Status** ()from the context menu.

A **Connection Status** window such as the following opens:



Refer to *Columns and Selection Options: Node Connection Status*.

3 Choose **Clear Entry** to remove the selected node status entry from the list or choose **Clear All** to remove all node status entries.

Note: By default, a node connection status is transient and can change when you terminate an Entire Operations session. You can save the status of all node connections for a future session by selecting the **Node Connection Status** option in the session profile described in the section *User Maintenance* in the *Administration* documentation.

For further information, see also the API `NOPUST3N` described under *Inquire Network and Job Status, Symbol Table* in the section *API Routines*.

Columns and Selection Options: Node Connection Status

The columns, selection options and output fields contained in the **Connection Status window** are described in the following table.

Column/Section	Description
Operating system class	In this section, check or uncheck single or multiple operating systems (BS2000, z/OS, Windows, and UNIX) defined for the nodes or select **All** for all operating systems. **Unknown** selects all nodes for which no operating system has been defined.
Access mode	In this section, check or uncheck single or multiple access modes used by the nodes (network for mainframe nodes, EntireX Broker for UNIX and Windows nodes and local node) or select **All** for all access modes.
Status	Node status of last Entire System Server call, for example, active or not active. In this section, check or uncheck single or multiple status values or select **All** for all status values. **Other** selects all nodes with a status other than active or not active.
Number	Node number.
Short Name	Short node name.
Node Name?	Logical (long) node name of the server.
AM	Access mode.
Op.Sys.	Operating system on node.
OS Release	Operating system release.
Op.Sys Class	Operating system class.
Version	Entire System Server version.
Statistics	The output fields in this section show the numbers of active and inactive nodes and the nodes where errors occurred.

Logoff Function

To log off a node

1 In the **object workspace**, choose **Node** ( ).

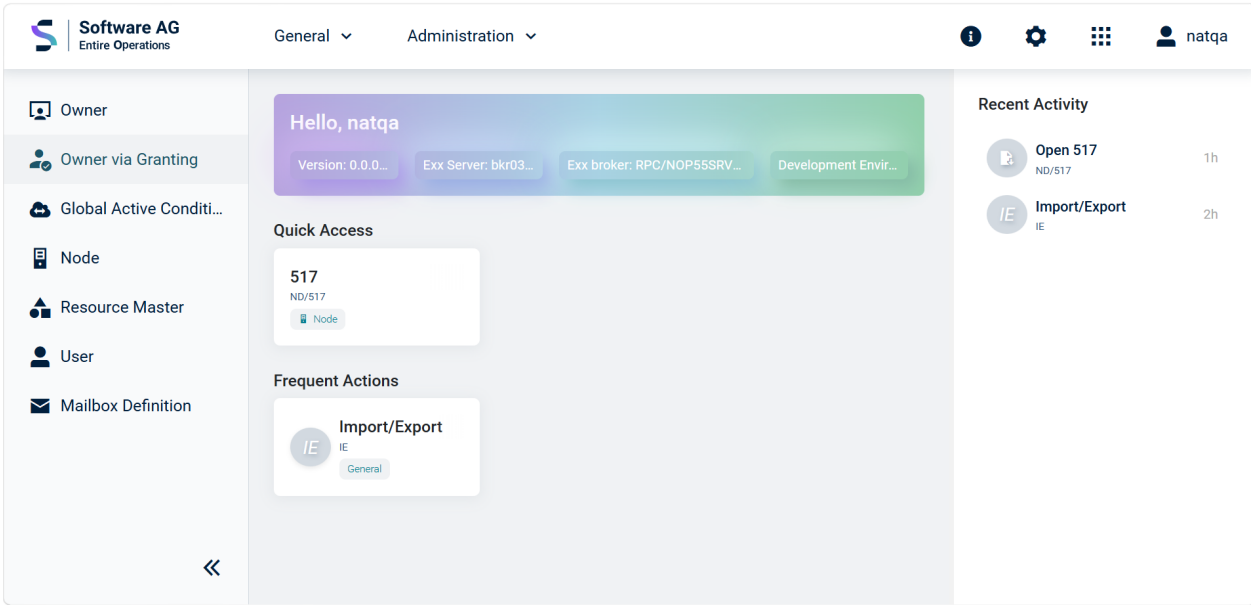
2 From the submenu, select a node instance and choose **Logoff** ( )from the context menu.

The previously selected node is now logged off.

7 Elements of the User Interface

- Title Bar 30
- Welcome Message and Display of Connection Information 32
- Workspaces 33
- Activity History 36

After you log in to the Entire Operations Web Client, the **Entire Operations home page** is displayed:



This chapter describes the elements of an Entire Operations home page and covers the following topics:

Title Bar

This section describes the *Title bar* which provides quick access to functions and settings of Entire Operations. The following topics are covered:

- Product name
- General
- Administration
- License Information
- Log Functions
- Quick Access
- Current User
- Language

- Tooltips

Product name

Name of the product. To get further system information, such as on version information of clients or servers, refer to ***Show system information***.

General

This menu contains various general functions

For futher information, refer to ***Functions of the Metanode General***.

Administration

This menu contains functions to be used by system administrators only. For detailed information, refer to *Accessing Administration Functions* in the *Administration* documentation.

License Information

Tab `placeholder` provides license information.

Log Functions

Tab `placeholder` provides information on the system log and various functions.

Quick Access

`placeholder`

Current User

Next to symbol the current user name is shown.

If you click the symbol, the following menu is displayed:

Function	Explanation
Sign out	The current user is logged off. The tab **Entire Operations log in** appears in which you can enter credentials. You can log in again with different credentials and a user language. or: You can terminate Entire Operations by choosing **Close**.

Language

The user language can be selected during Starting an Entire Operations session.

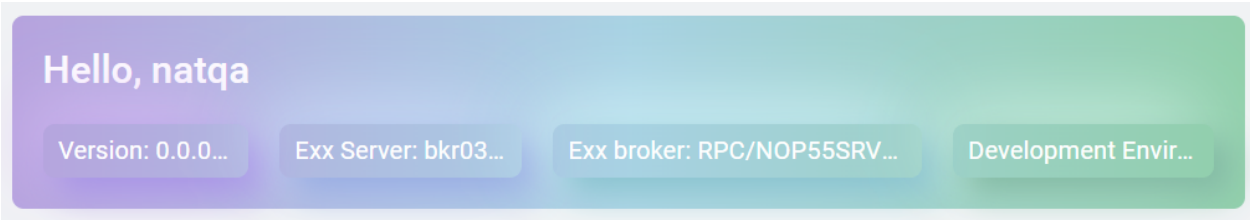
To change the user language, terminate your current session and change the language settings as described in *Starting and Ending an Entire Operations Session*.

Tooltips

A tooltip is a pop-up window with a short message that is available for a specific element such as a title, a button or a symbol. By hovering over an element, the message specific to that element is displayed.

Welcome Message and Display of Connection Information

Below the title bar a welcome message is displayed: **Hello,** *user name*, where "*user name*" equals the current user's user ID. Below, you will find technical information on the current connection:

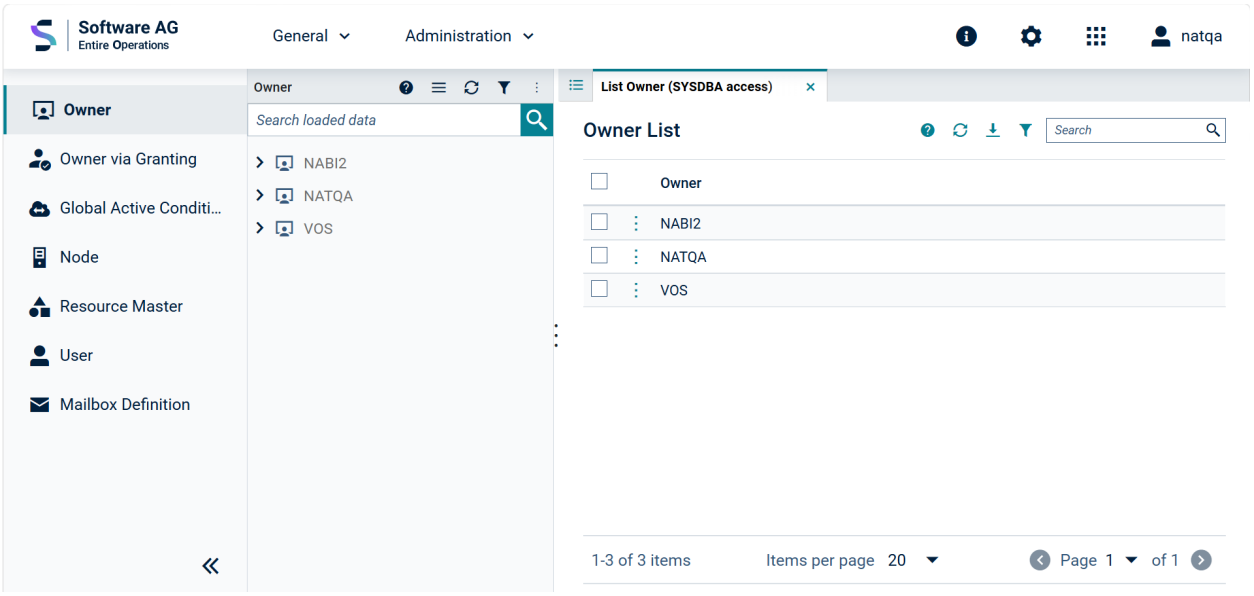


Version	Version number of the Entire Operations Web Client software being used.
Exx Server	EntireX Server being used.
Exx Broker	EntireX Broker being used.
Development Environment	Environment being used such as development, test, production.

This area is not needed as you continue using the interface and will be completely covered by the *Workspaces* used.

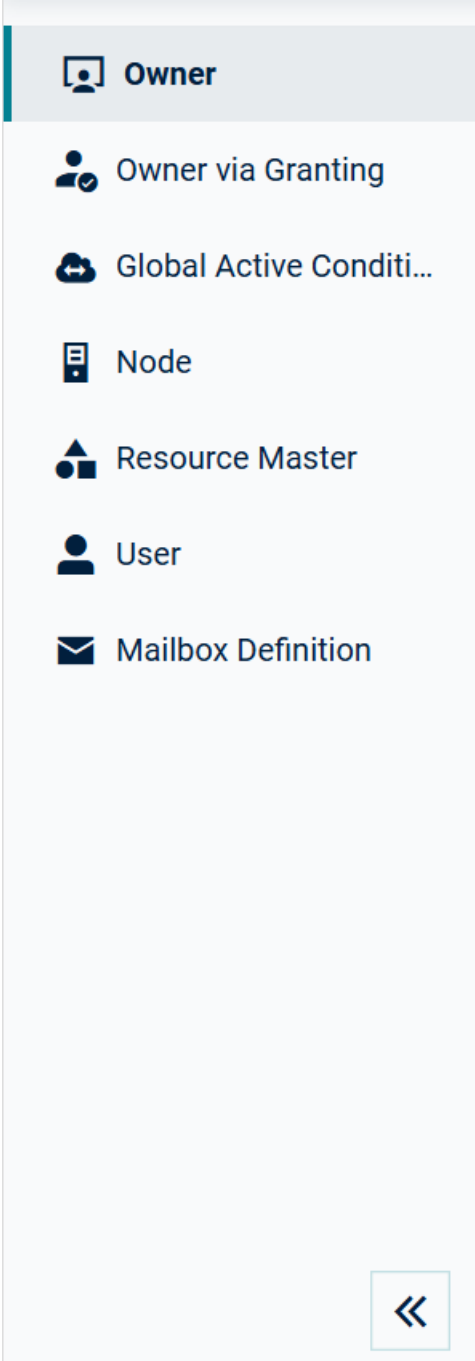
Workspaces

The home page contains the **object workspace** on the left, featuring a tree structure menu for selecting one of the application's business objects to be maintained. The area on the right site is covered by the **content pane**. It is used to display the object's content retrieved by a processing function. A typical processing function enables actions such as listing all objects of a type, creating a new object, displaying, modifying, or deleting a single object.



Object Workspace: Main menu and tree structure menu for managing objects

The tile on the left side of the **Object Workspace** contains the **Main menu** for selecting the application's key business object types:



To save space, you can collapse the text column containing the object names in the main menu.

To show or hide the text column for the entries in the main menu:

1 Click the backward arrow symbol («).

The column containing the text entries will be collapsed.

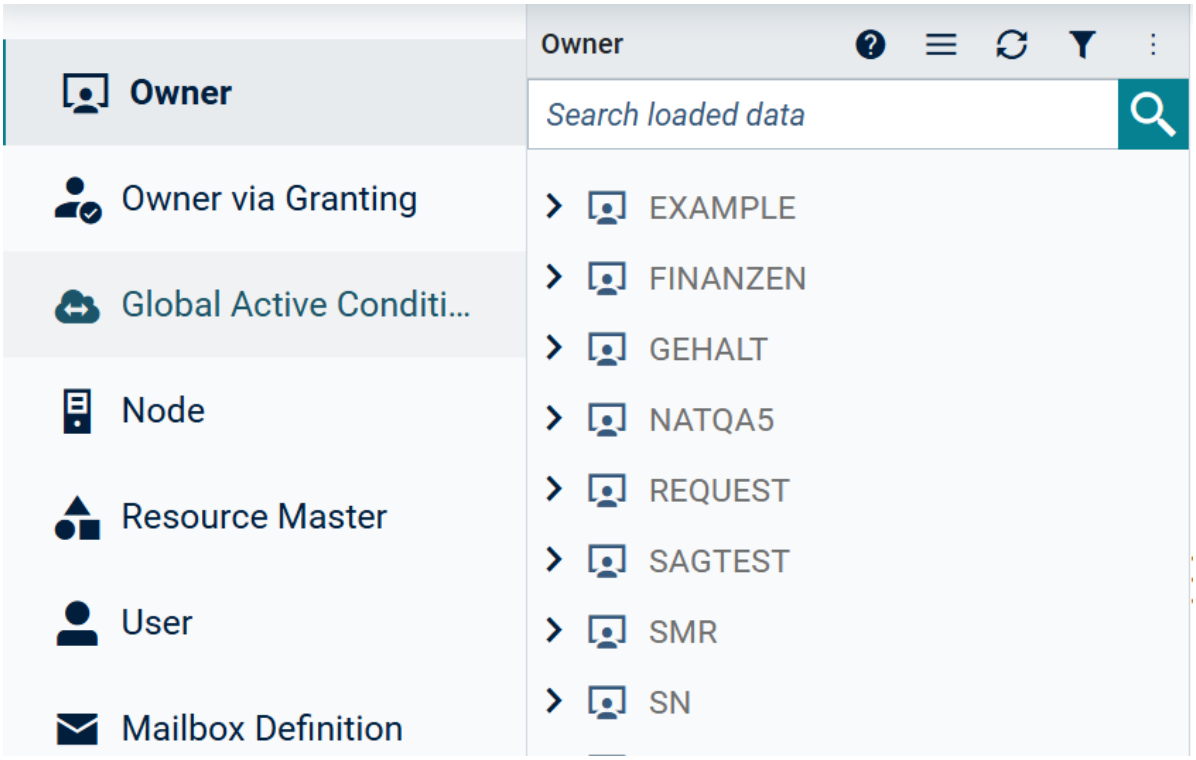
To select an object, you can then choose one of the remaining symbols. Note that the meaning of a symbol is explained by a tooltip, for example for the owner symbol ( ). See also *Tooltips*.

2 Click the forward arrow symbol (») and the column containing the text entries will be displayed again.

To use the tree menu to maintain an instance of an object:

- Choose an object in the main menu, for example **Owner**.

  A list of the existing instances for the "Owner" object is displayed in the column next:

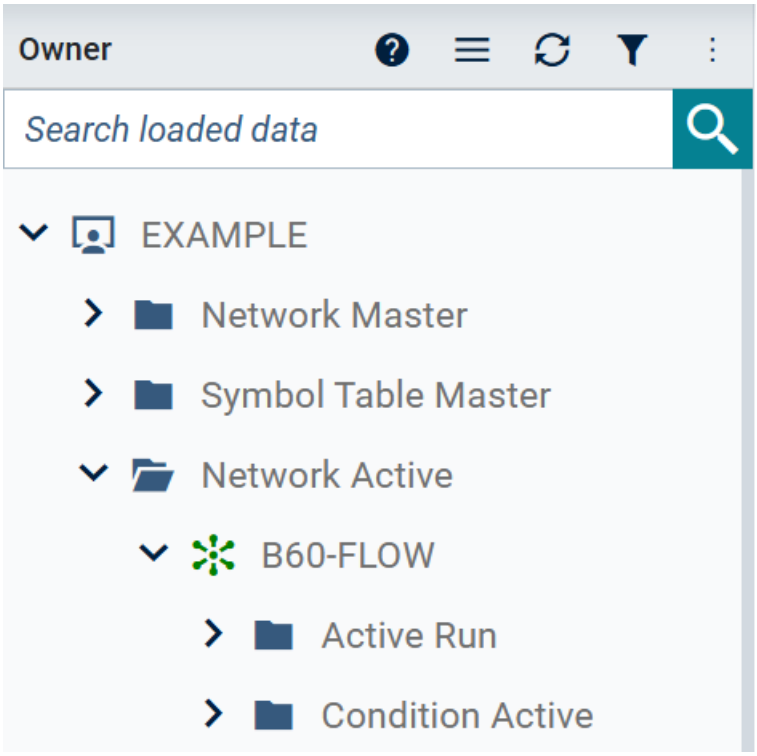


To expand an entry in the directory tree:

1 Click the arrow symbol (›) preceding an entry, for example the owner **EXAMPLE**.

The column below displays a list of the definitions available for this owner (Network, Symbol Tables, Active Networks, Schedule, Calendar).

2 To give an example on how to expand the list with additional entries, click the arrow symbol (›) preceding **Active Networks** and then repeat this procedure for **B60-FLOW**:



You can adjust the width of the expanded column to adjust the length of the menu items.

To move a separating line:

- To change the size of the area, move the mouse pointer over the separating line between the two areas until the pointer changes shape and ⟷ pointing in opposite directions appear.

To collapse a directory entry:

- Click ˅ in front of the relevant entry.

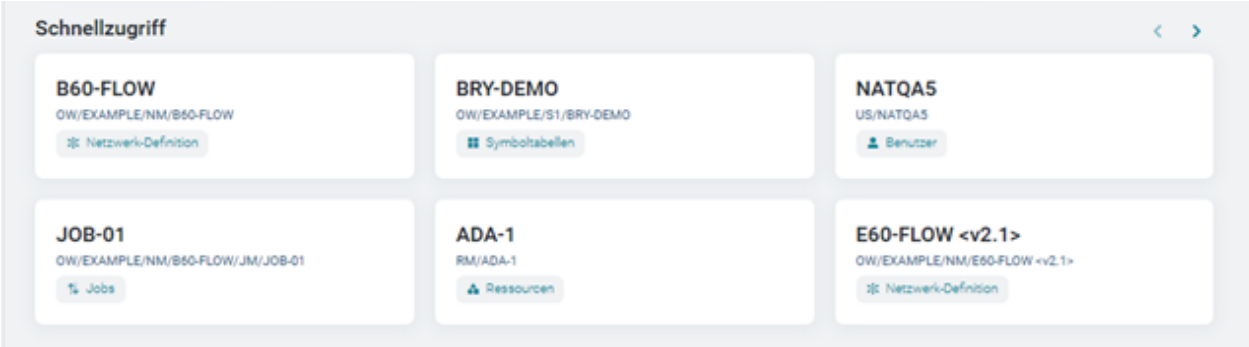
Activity History

The Entire Operations home page provides sections which give an overview of your history and quick access to the functions you have used recently:

Quick access | **Frequent actions** | **Recent activity Show all activities (history)**

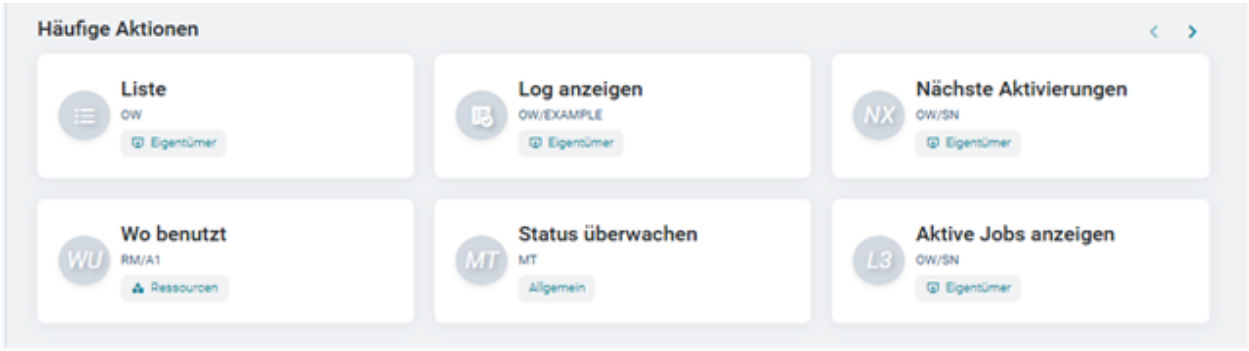
Quick Access

Example:



Frequent actions

Example:



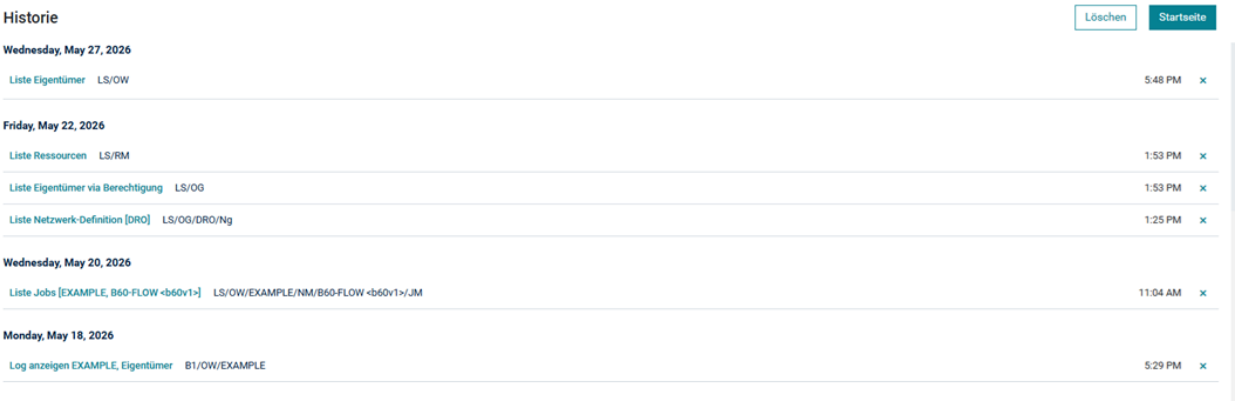
Recent activity

Example:



Show all activities (history)

Example:



8 Common and Global Functions

- Cancelling Reading 40
- Listing Objects 40
- Refreshing Object Lists 40
- Specifying Filter Criteria 41
- Saving a List View as a Report File 42
- Quick Access 42
- Displaying Objects 45
- Copying Objects 45
- Deleting Objects 46
- Using a Diagram to Monitor and Maintain a Network 46
- Importing and Exporting Objects 47
- Functions of the Metanode General 47
- Monitor Status 48
- Show Messages and Requests 49

This section describes common functions of the Entire Operations Web Client that perform the same operations for different objects.

In addition, it describes the functions of the **General** metanode that perform global operations on all objects in your Entire Operations environment.

Cancelling Reading

In Entire Operations Web Client ,, a long reading process can be cancelled by pressing ESC.

To cancel reading

- Press ESC.

Listing Objects

To view a list of objects

1. In the **object workspace**, select a metanode.
2. Open the **context menu** and choose **List** ( ) .

   All objects of the selected metanode are listed.

Refreshing Object Lists

To refresh objects listed in the object workspace

- In the **object workspace**, select a metanode and choose **Refresh**() from the context menu.

  The list of objects of the selected metanode is updated immediately.

To refresh objects from an open list window

- Open a window that contains a list of objects and choose the **Refresh** () button if available.

  The list is updated immediately.

  Or:

Open a window that contains a list of objects and choose **Auto Refresh** ( ) , if available.

In the **Automatic Refresh** window that opens, select the **Automatic refresh** check box. If required, change the time interval (the default is 30 seconds) in which you want to update the list.

Choose **OK**.

The list is updated automatically in the time interval specified.

Specifying Filter Criteria

This section describes the options provided to specify name ranges to be used as selection criteria for Entire Operations functions or drop-down list boxes.

In the following table, *value* is any combination of one or more characters.

Input	Names Selected
*	All names.
*value**	All names that start with *value*. Example: AB* Selected: AB, AB1, ABC, ABEZ Not selected: AA1, ACB
value>	All names greater than or equal to *value*. Example: AB> Selected: AB, AB1, BBB, ZZZZZZZ Not selected: AA1, AAB
value<	All names less than or equal to *value*. Example: AX< Selected: AB, AWW, AX Not selected: AXA, AY
value?	All names that start with *value* and end with any single character for each question mark (?) entered. Example: ABC? Selected: ABCA, ABCZ Not selected: AXC, ABCAA
value?*value*	All names that match and *value* combined with a question mark (?) in any order. Example: A?C Selected: ABC, ACC Not selected: ABCA, AC

Input	Names Selected
???	All names with an exact length of three characters. Example: ??? Selected: ABC, DEF Not selected: AC, DEFA

Saving a List View as a Report File

The **context menu** in a **List** window provides the **Save View as File** ( ) function to save the content of a **List** window as a report file containing columns and column headings and filter information. For detailed information, see ***Report Output Options*** in the section *Reporting*.

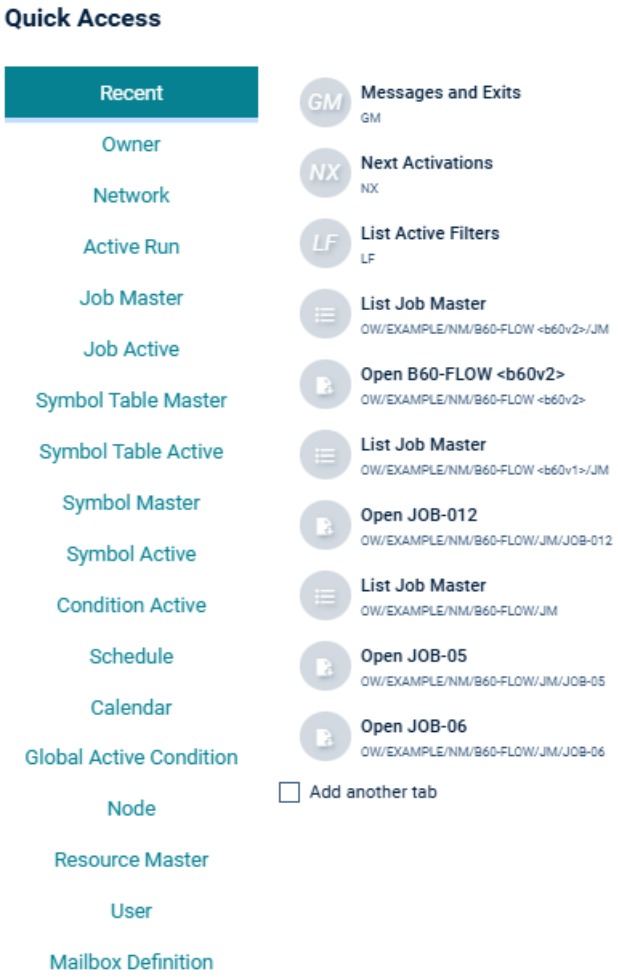
Quick Access

With function **Quick Access** you can maintain an object directly without having to search the tree structure of the **object workspace**.

To open the Quick Access dialog box:

1 In the title bar of the current screen, click **Quick Access** ( ).

The **Quick Access** dialog box is displayed as in the following example:



The **Recent** button is selected by default in the left-hand column.

The center column therefore lists the functions you have used recently.

The right column displays the date and time they were used., z.B. **Just now**, *x***m** *x* minutes ago, *x***d** *x* days ago.

2 Use the vertical scroll bar in the dialog box to display additional entries in the list.

3 You can select the **Add another tab** check box so that subsequent actions are displayed on additional tabs, allowing you to continue using the **Quick Access** dialog box in the foreground.

To access and use functions using Quick Access:

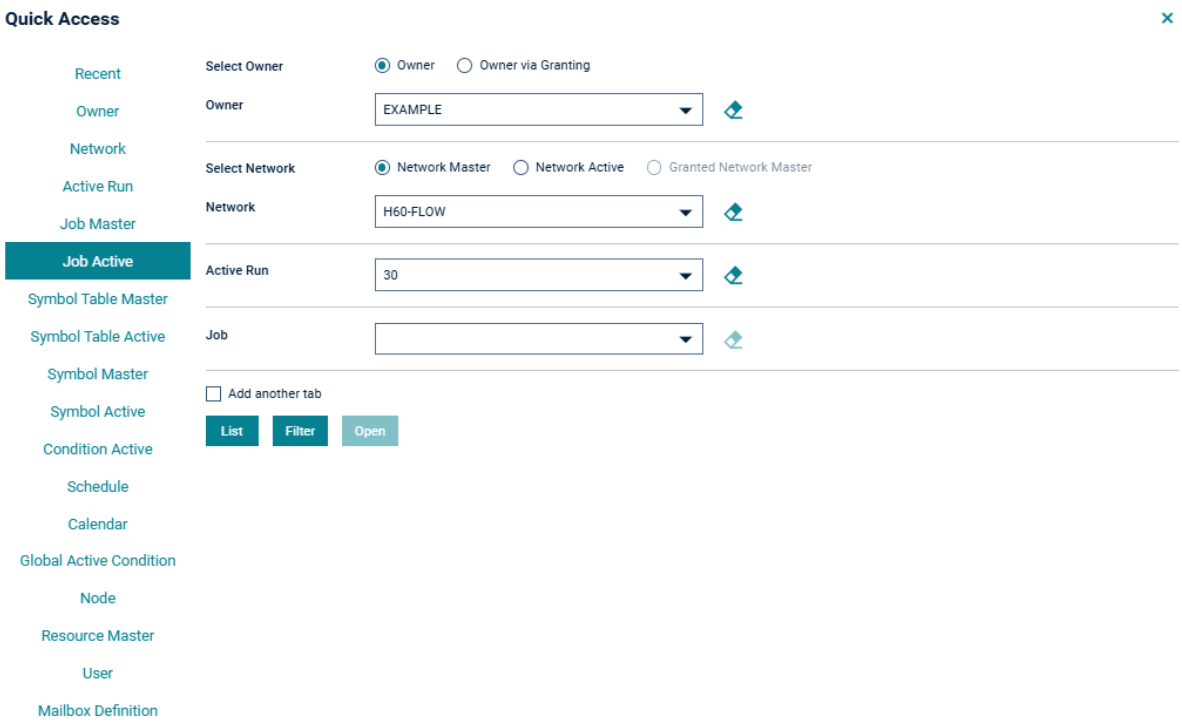
1 In the left-hand column of the **Quick Access** dialog box, select the button corresponding to the object type you want to manage, for example, **Aktive Jobs**.

The middle column displays selection fields where you can enter the information required to determine the new active job such as:

- name of owner, name of network definition and number of the active run.
- When creating a new job, the **Job** field is left empty.

2 Select the button that corresponds to the action to be performed, for example, **New**.

A tab with the **Create New Object - Active Jobs** dialog opens behind the **Quick Access** dialog box, such as in the following example:



3 You can now close the dialog window or continue using it to define selection criteria for additional functions. These functions will then be available on their respective tabs in the background.

To close a Quick Access dialog

Note: If the **Add Another Tab check box** was *not* selected, the dialog window will close automatically as soon as the tab for the desired function is displayed.

- If the check box is selected, the dialog window does not close. It remains open in the foreground, covering the tab(s) that contain the selected function(s).

  Choose **x** to close the dialog.

Displaying Objects

The **Display** function opens a window that displays information for the currently selected item. You can only view the information, not edit it. If you want to edit the item, you have to use the **Open** function.

To display an object

1 In the **object workspace**, select the object.

2 Open the **context menu** and choose **Display**.

Or:

Press CTRL+D.

A dialog appears that displays all available information for the selected object.

Note: For a detailed description of the displayed fields, see the description of the **Open** function for the respective object.

Copying Objects

Important: Copying an object always means that the complete underlying structure is also copied into the target destination.

To copy single or multiple objects

1 In the **object workspace**, select a node instance or an entire node.

Open the **context menu** and choose **Copy data** ( ).

2 Or:

In the **object workspace**, select a node for which you can choose **List** from the context menu. A **Master List** window opens.

From the list table in the **Master List** window, select single or multiple objects and choose **Copy data** ( ) .

The object data is copied to the clipboard and ready to paste at the required location within the object workspace.

3 A screen is opening. Enter **Target Object Namespace** and select **PASTE**.

Deleting Objects

The **Delete** function opens a dialog to confirm the selection for this operation.

To delete an object

1 In the **object workspace**, select the object.

2 Open the **context menu** and choose **Delete** ( ) .

A dialog appears where you have to confirm the deletion.

Using a Diagram to Monitor and Maintain a Network

Complex networks can have many components with diverse interconnections that make it difficult to monitor and maintain the networks.

For this purpose, Entire Operations provides the option to represent networks (master and active) as diagrams containing jobs (master and active), conditons and resources. Diagrams can be used to monitor all objects that belong to a network, see whether they are connected, maintain, add and delete definitions and check network execution.

To monitor and maintain a network using a network diagram

- In the object workspace, select a **Network Master** or **Job Master** instance and choose **Diagram** ( ) from the context menu.

  Or:

  In the object workspace, select an **Active Run** or **Job Active** instance and choose **Diagram** or **Network Diagram** from the context menu.

  A **Diagram Network** window like the following opens.:

  All maintenance functions available to monitor and maintain a network from a diagram are described in *Viewing and Maintaining a Job Network Diagram* in the section *Network Maintenance*.

Importing and Exporting Objects

You can import and export objects using the **Import/Export** function of the **General** metanode or the **Export** function of a selected node or node instance.

For detailed information on using the functions, see *Importing Objects* and *Exporting Objects* in the *Import/Export Functions* documentation.

Functions of the Metanode General

The **General** metanode provides menu functions that are performed on all objects available in your current Entire Operations environment.

To list all available functions for the menu "General":

- Select menu **General** in the main menu.

  The following table explains the available Commands and functions:

Function	Meaning	Described in Section
Next Activations	Display a list of future activations.	See *Viewing and Modifying Next Network Activations* in the section *Schedule Maintenance*.
List Active Jobs	List and manage all active jobs of the selected job network.	See *Listing Active Jobs* in the section *Maintaining Active Jobs*.
Activity Log	Start or cancel the display of current activities.	See *Monitoring Entire Operations Activities* in the section *Log Information*.
Browse Log	Display logged system events.	See *Displaying Logged Information - Browse Log Function* in the section *Log Information*.
Monitor Status	Open Entire Operations Monitor status window.	See *Monitor Status*.
Determination Date	Display the determination date for an evaluation function.	See *Determination Date for Report Data* in the section *Reporting*.
Reporting	Entire Operations reporting functions.	See *Report Types* in the section *Reporting*.
Cross-References	Cross-check the use of individual Entire Operations objects and produce a report from the data found.	See *Types of Cross-References* in the section *Cross-References*.

Function	Meaning	Described in Section
Import/Export	Entire Operations import/export functions.	*Importing Objects* and *Exporting Objects* in the *Import/Export Functions* documentation.
Show Messages	Show the contents of all mailboxes to which you are connected.	See ***Show Messages***.
Show Workplan	Open the workplan or bring it to the foreground.	See ***Show Workplan***.
Show all Owners	List all owners (SYSDBA access).	See ***Show all Owners***
Show linked Owners	List owners assigned to the current user.	See ***Show linked Owners***.
Set Drag And Drop Function	Drag a node from the tree view and drop it in the empty workspace. As a result of this action, a specific function is then executed.	See ***Drag & Drop***.

Monitor Status

The Monitor is the heart of Entire Operations. The Monitor is a complex program that wakes up periodically and checks the definitions stored in the master database for any work to do. It activates and processes networks and jobs according to their prerequisites and controls running job networks, even if they reside on the nodes of different computers.

The **Monitor Status** function of the **General** metanode is used, for example, to start and stop the Monitor.

Before you can monitor Entire Operations activities, you may have to log on a node.

To monitor Entire Operations

1 In the **Title Bar** choose **General**.

2 Choose **Monitor Status**.

For an inactive node, a **Logon Node dialog** opens first.

The fields contained in this dialog are explained in ***Fields: Logon Node***.

3 Enter your user ID and password and choose **OK** to log on to the specified node.

A **Monitor Status** window opens displaying all Monitor tasks.

Here you can:

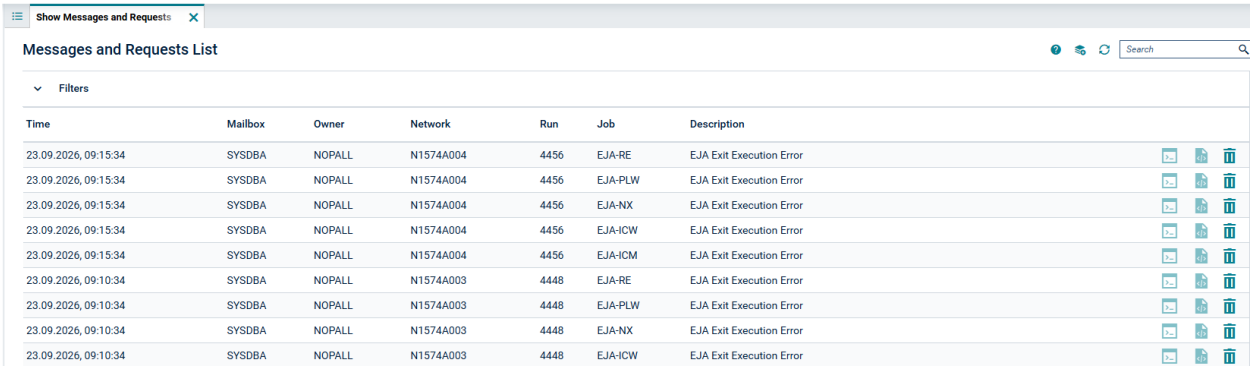
- Start and stop the Monitor.

- Hold or release a certain Monitor task.
- If you want to display the Monitor tasks before the dialog is automatically refreshed, you can refresh it manually by selecting the **Refresh** button.
- Select **OK**.

For detailed information, see *Status of the Entire Operations Monitor* in the *Administration* documentation.

Show Messages and Requests

The **Show Messages and Requests** function of the **General** is used to list mailbox messages and requests as shown in the following example:



For explanations of this window, see *Handling Messages and Requests* in the section *Working with Mailboxes*.

9 Operating System Classes and Related Operating Systems

Within Entire Operations, the term “operating system class” means one or more operating systems, which are usually handled in the same way.

Operating System Class	Operating System
B	BS2000
M	z/OS?
X	All supported UNIX operating systems
W	All supported Windows operating systems

10 Date and Time Formats

- Date Display and Input Options 54
- Time Display and Input Options 55

This section describes the date and time formats Entire Operation uses for processing system function and producing output reports. These formats are independent of the format options Windows provides in an Entire Operations GUI environment.

The date displayed in a window is represented in the Entire Operations default format selected in the **Date format** field of the *Entire Operations Defaults* (see the *Administration* documentation).

If you enter a date or specify a data as a parameter, the syntax must conform to the format set in the Entire Operations defaults. Otherwise, you receive an appropriate error message indicating that the date or time does not match the required format.

Date Display and Input Options

Possible date representations and valid input values for date fields are described in the following section.

Input fields for dates have different lengths and you may have to (or can optionally) shorten the input string. Entire Operations always treats shortened input (or output fields) with the century.

A shortened input string is always extended to a complete date (even when not representable in the window), according to the default format set for your environment. This is indicated in the following examples that assume the current date is December 30, 2016.

Date Pattern	Date Format Setting/Selection Option	Examples
DD.MM.YY	G **German DD.MM.YY** German format.	30.12.16 301216 30.12 3012
DD/MM/YY	E **European DD/MM/YY** European format.	30/12/16 301216
MM/DD/YY	A **American MM/DD/YY** American format.	12/30/16 123016
YY-MM-DD	I **International YY-MM-DD** International format (default).	16-12-30 161230 1612

Date Pattern	Date Format Setting/Selection Option	Examples
YYYYMMDD	8 **8 digits YYYYMMDD** Eight-digit format.	20161230 2016

Explanations:

- DD is a two-digit day, MM a two-digit month, YY a two-digit year and YYYY a four-digit year.

  A two-digit year is internally always processed with four digits.
- Input of a year value is optional. If you omit the year, the current year is used.

  If Entire Operations interprets an input string as a past date (relative to the current date), an appropriate message occurs.
- The century that relates to a 2-digit year value is determined by using a sliding window that assumes the date to be within a range of +50/-50 years, relative to the current year. For example:

  If the current year is 2016, the sliding window will cover the years 1966 to 2065. A 2-digit year value *nn* from 66 to 99 is interpreted accordingly as 19*nn*, while a 2-digit year value *nn* from 00 to 65 is interpreted as 20*nn*.
- Input of the separator characters slash (/), period (.) and dash (-) is optional. If you omit the separators, the input string is extended to a complete date.

Time Display and Input Options

The time is represented in one of the following formats:

Syntax	Example	Complete Time
HH:II:SS	145231	14:52:31
	1452	14:52:00
	14:53	14:53:00
	14	14:00:00

Explanations:

- HH is a two-digit hour, II a two-digit minute and SS a two-digit second.

  The hour range is 0 to 23. The range for minutes and seconds is 0 to 59.
- Input of minutes and seconds is optional. If you omit the minutes and/or seconds, the input string is extended to a complete time.

- Input of the separator character colon (:) is optional. If you omit the separators, the input string is extended to a complete time.

II Using the Entire Operations Business Functionality

11 Using the Entire Operations Business Functionality

- Structure of the Application and its Documentation 60
- Required Knowledge 60
- Authorization 61
- Overview of Objects in Entire Operations 61
- Overview of the Object-specific Maintenance Functions in Entire Operations 63
- Three Steps to an Executable Job Network 64
- Using Control and Monitoring Functions for Job Networks 65
- Performing ad hoc Actions on Active Jobs in the Active Database 65
- Creating Ancillary Objects for a Job Network or Job 66
- Using Monitoring and Evaluation Functions 66
- Using Sample Networks 66

Complementing the details on using the application's access and presentation functionality (graphical user interface, help system, etc.) described in the chapter ***Using Entire Operations GUI Client in Online Mode***, this chapter provides a brief introduction to the structure and use of the application's business logic, i.e. the operation planning and control functions provided by Entire Operations to define, update and execute job networks, jobs, and ancillary objects.

Structure of the Application and its Documentation

For the maintenance and execution of the object data, the classic character-based (CUI) maintenance screens are used in an Entire Operations (NOP) mainframe and UNIX environment. In a Windows environment, the Entire Operations GUI Client (OGC) is used, which provides a tree view and dialogs. All existing records (definitions) for an object type (owner, job network, job, etc.) are clearly arranged and readily accessible. They can be invoked and used by means of line commands (CUI) or context menu commands (GUI).

The order in which the objects are arranged in the application essentially corresponds to the workflows. These are mapped accordingly in the structure of this user manual and within the individual object-specific chapters. Each chapter describes the functions that can be performed on an object type and explains the input and output fields available.

Objects whose maintenance requires the rights of an administrator are described in the *Administration* document. You can display the associated data also as a general user, for example to view the settings in your user profile and the authorizations granted to you for a particular activity in the various functional domains of the system, or to view the definitions of system-wide resources that can be referenced in a job. If you need to change any of these details, please contact your system administrator.

Required Knowledge

The following basic knowledge is required:

- Knowledge of the operating systems used
- Knowledge of the job control language (JCL) used
- Knowledge of the Natural programming language

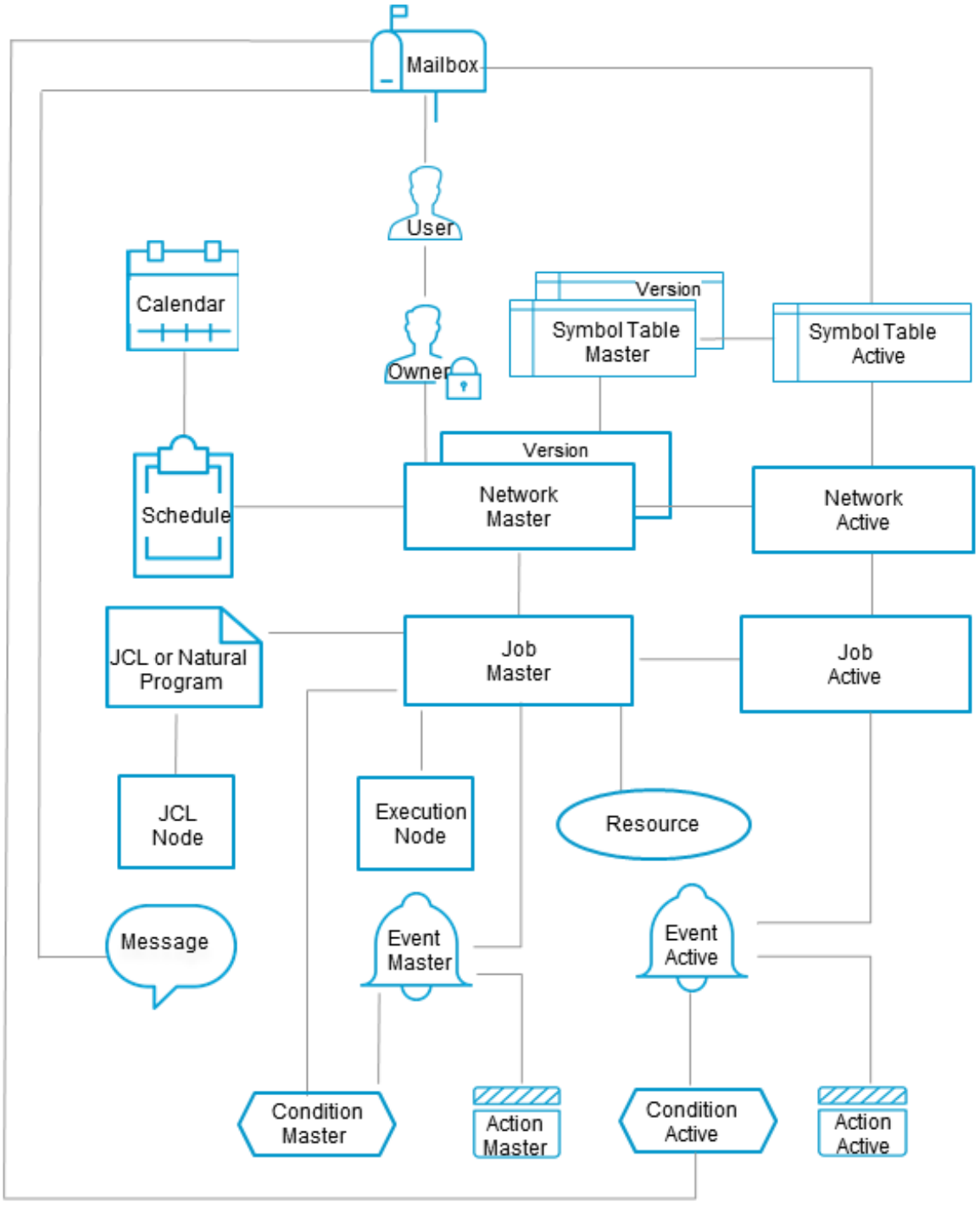
Authorization

Access to the business functionality and the definitions created can be granted in mutiple places within the application:

1. In your user profile, see *User Definition and Profile Settings* in the *Systemverwaltung* documentation.
2. In the **Owner Maintenance** function.
3. In the definition of a job network, see ***Granting Definition: Authorizing Other Users or Owners to Access a Network***.

Overview of Objects in Entire Operations

The following graphic provides an overview of objects you can maintain in Entire Objects in order to set up a job network:



For an overview of how these objects interrelate, see *Entire Operations Object Relationship* in the *Concepts and Facilities* documentation.

Overview of the Object-specific Maintenance Functions in Entire Operations

The Entire Operations application is structured in such a way that all important object-specific maintenance functions are accessible as quickly and logically as possible. The order of the objects in the **tree structure in the object workspace** of the application essentially corresponds to the workflows. This is reflected in the structure of this user manual and within the individual chapters. There, the functions available for an object are first presented in an overview, and then their use and all inputs and outputs are described in detail.

Primary Maintenance and Control Functions for Job Network and Job Definitions

- **Owner Maintenance**
- **Network Maintenance**
- **Job Maintenance**
- **Active Job Network and Active Job Maintenance**

Maintenance Functions for Global Object Definitions

- **Schedules**
- **Calendars**
- **Mailboxes**
- **Symbol Tables and Symbols**

Monitoring and Evaluation Functions

- **Log Information /Activity Monitoring**
- **Reports**
- **Cross-References**

Interface Programming

- **API Routines**
- **User Exits**

Three Steps to an Executable Job Network

The creation of a job network is essentially done in three main steps. These steps are always required to define an executable job network. The resulting definitions are stored in the master database.

- **Step 1: Determine owner**
  The central object of the Entire Operations application is the **owner** and the user(s) associated with it. See also *Owner Maintenance*.

  If the required owner is not available in the directory tree of the *Object Workspace*, refer to *Adding and Removing User/Owner Links and Owners* and/or contact your system administrator.

  In the directory tree of the *Object Workspace*, open the **Owner** node and open the node corresponding to the owner (e.g. EXAMPLE) to which you want to assign the new job network.

  Example: **Owner** > **EXAMPLE**

- **Step 2: Define job network**

  Below the **Owner** node, select the node **Network Master**.

  Open the **context menu** and choose the **New** function.

  A **Create new Network Master** window opens.

  Proceed as described in *Adding a Network Definition* or choose the **Help** button to display the corresponding instructions.

  After you have filled in all the mandatory fields, you can already save the new network definition.

  You can open the record again at any time to **display** or **modify** the existing data or enter data, e.g. to define different runtimes. You can use tabs with additional fields for special purposes.

  Close the dialog.

  You also have other options for maintaining a record in the master database. These are listed in the overview of the *Network Maintenance* chapter under *Job Network Maintenance* and *Maintenance Functions for Job Network Definitions*.

- **Step 3: Create job definition(s)**
  In the previously created job network, you can now define a job.

  In the directory tree of the *Object Workspace*, open the path **Owner** > *Owner name* > **Network definition** > *Network name* > **Jobs**. Where *Owner name* is the name of the owner you have determined in Step 1 and *Network name* is the name you have assigned to the newly created job network in Step 2.

  On the **Job Master** node, invoke the **context menu** and choose the **New** function.

The **Create new Job Master [***owner name,network name***]** dialog is displayed.

Proceed as described in *Adding a Job Definition* or choose the **Help** button to display the corresponding instructions.

After you have filled in all mandatory fields in the dialog box, you can use the tabs there with additional fields to enter further details. Confirm the entries on the individual tabs with **Apply**. Choose the **OK** button to confirm all entries you made in the dialog box and on the tabs and save the new job definition.

You also have other options for maintaining a record in the master database. These are listed in the overview of the *Job Maintenance* chapter.

Using Control and Monitoring Functions for Job Networks

A job network is activated either automatically by the Entire Operations monitor according to the defined schedule data, or by a user on demand.

After you have properly defined a job network, you can activate it, i.e., submit it for immediate or scheduled execution. The following control functions, among others, are available to you for this purpose:

- **Activating a Job Network Manually**
- **Deactivating Active Runs for a Network**
- **Displaying Next Network Starts - Next Activations**
- **Checking for a Loop in a Job Network**
- **Displaying the Network Execution History**

Performing ad hoc Actions on Active Jobs in the Active Database

Even after a job network has been activated, you can still change specifications in an active job network or an active job (in the active database) without changing the original network or job definition in the master database.

- **Maintening Active Job Networks and Active Jobs** (active jobs, active conditions, active JCL)
- **Maintaining Active Symbol Tables**

Creating Ancillary Objects for a Job Network or Job

You can define globally defined ancillary objects in Entire Operations at any time and reference them in the job network and/or job definitions.

- **Defining schedules** that can be referenced in the definitions of multiple job networks
- **Defining calendars** that can be used as a basis of time tables for jobs and job networks
- **Using mailboxes** to send network-related messages and requests to users or groups of users
- **Defining symbol tables and symbols**
- **Using functions for symbol replacement**
- Using resource maintenance functions
- Maintaining mailbox definitions

Using Monitoring and Evaluation Functions

You can use the following monitoring and evaluation functions:

- **Displaying Logged Information** or **Monitoring Entire Operations Activities**
- **Viewing Job/Network Accounting Information** to obtain information on the start and end times, elapsed time and CPU time for previous runs of a job
- **Generating Reports** to generate reports that help overview your network environment to define objects, monitor the system and plan workloads
- **Using Cross-Reference Functions** to cross-check the use of individual Entire Operations objects and produce a report from the data found

Using Sample Networks

You may ask your system administrator to load the sample networks supplied for self training. You can modify a copy to gain initial experience in using the job network maintenance features of Entire Operations.

III Entire Operations Utilities

12 Entire Operations Utilities

- Introduction 70
- Extract Log Data to Log Selection File 72
- Delete Old Data in Log Selection File 73
- Print Account Information from Entire Operations Log 74
- Mass Change of Network/Symbol Table Version 74
- Monitor or Task Wait Time Modification 75
- Monitor Shutdown 76
- Monitor Start 76
- BS2000 Jobs 76
- List or Delete TO-ACTIVATE Command Records 77
- Check the Existence of Symbol Table Definitions 79
- Mass Change of the Owner and Owner Deletion 79
- Mass Change of Access Rights Granted to Networks 82
- User ID Mass Update in Network and Job Definitions 84
- Mass Update for User Access to Nodes 86
- Mass Change of Node Numbers 87
- Mass Logon Processing in Batch Mode 88
- Bulk Execution of MACRO Commands 90
- Data Migration to the current Entire Operations Version 91

Entire Operations utilities are used to perform selected Entire Operations functions.

A utility is usually a standalone Natural program that runs either online or in batch mode when invoked by a Natural batch job.

Introduction

- Required LFILE Assignments
- Log Selection File - Format 1

Required LFILE Assignments

LFILE	Explanation
216	Entire Operations System File 1.
131	System Automation Tools log file.
215	Entire Operations System File 3 (**log selection file**).

Log Selection File - Format 1

An empty log selection file (`LFILE=215, NOP-LOG-SELECT-1`) for accounting data is part of the Entire Operations installation files.

The DDM for the log selection file is `EOR-LOG-SELECTION-1`.

The DDM is delivered with descriptors on almost every field. It is up to the user to release descriptors which are not necessary.

The file can be read and evaluated with Natural.

DB 255, File 215

This file applies to the default sequence of the `NOP-LOG-SELECT-1` file.

T	L	DB	Name	Format	S	D	Remarks
*	**EOR Log Selection**						
	1	AA	LGS-RECORD-TYPE	A6	N	D	
*							
G	1	AB	LGS-BS2-DATA				
	2	AD	LGS-BS2-USERID	A8	N	D	
	2	AE	LGS-BS2-ACCOUNT	A8	N	D	
*							

T	L	DB	Name	Format	S	D	Remarks	
G	1	AG	LGS-TIME-DATA					
	2	AH	LGS-CPU-TIME	P7.2	N	D	/* in seconds	
	2	AI	LGS-JOB-START	T12	N	D	/* date + time	
	2	A7	LGS-JOB-START-TIME-ZONE	A10	N			
	2	AK	LGS-JOB-STOP	T12	N	D	/* date + time	
	2	A8	LGS-JOB-STOP-TIME-ZONE	A10	N			
	2	AM	LGS-JOB-ELAPSED	P13.0	N	D	/* in 1/10 seconds	
*								
G	1	AN	LGS-DATA-3					
	2	AO	LGS-LOG-TIME	T12	N	D	/* date + time	
	2	BM	LGS-LOG-MICROSECONDS	P5.0	N			
	2	BP	LGS-LOG-TIME-ZONE	A10	N			
	2	AP	LGS-OWNER	A10	N	D		
	2	AQ	LGS-NETWORK	A10	N	D		
	2	AR	LGS-RUN	P13.0	N	D		
	2	AS	LGS-JOB	A10	N	D		
	2	AT	LGS-STATE	A3	N	D	Possible content:	
							CHK	Job to be checked
							ACT	Job is activated (active)
							JNO	Job ended not ok
							JOK	Job ended ok
	2	AU	LGS-JOB-TYPE	A3	N	D		
*								
	1	AY	LGS-EXECUTION-NODE-N5	N5.0	N	D		
	1	AW	LGS-JOB-ID	A10	N	D		
	1	AX	LGS-OS-JOB-NAME	A8	N	D		
	1	AZ	LGS-NETWORK-VERSION	A10	N			
	1	A1	LGS-EXEC-OPSYS-CLASS	A1	N		Possible content:	
							B	BS2000
							M	z/OS
							X	UNIX
							W	Windows
	1	A2	LGS-SUBMIT-USERID	A50	N			
	1	A3	LGS-SUBMIT-GROUP	A50	N			
	1	A4	LGS-REPEAT-COUNT	P13.0	N			
	1	A5	LGS-SPECIAL-TYPE	A8	N			

T	L	DB	Name	Format	S	D	Remarks
	1	A6	LGS-DUMMY-FLAGS	A8	N		Contains the reason(s), why an active job is a (temporary) dummy job. One or more (in arbitrary order) of the following flags are possible:
							C Condition
							D Definition (job type DUM)
							E Empty JCL
							J JCL check
							K Job to be deactivated
							M Multiple suffix
							R Recovery
							S Schedule
							T Repetition
*							
*	**Superdescriptors**						
*							
	1	AV	LGS-KEY-1	A30	N	S	
*			LGS-OWNER	1-10			
*			LGS-NETWORK	1-10			
*			LGS-JOB	1-10			

Extract Log Data to Log Selection File

- Natural Program Call
- Parameters

Natural Program Call

```
LGAX1S-P P-DATE-FROM P-DATE-TO
```

Parameters

Name	Format	Description
P-DATE-FROM (optional)	A8	Start date for the extraction in format YYYMMDD.
P-DATE-TO (optional)	A8	End date for the extraction in format YYYMMDD.

Notes:

1. Start date and end date for the extraction are optional. If wanted, both parameters must be set. If the start date is smaller than the time stamp in the private control record (see note 2 and 3), the extraction is not performed.
2. Log data extraction always starts where the last extraction ended.
3. This time stamp is kept in a private control record. (When the program is started for the first time, the start date 01.11.2001 is used.)
4. The extraction data consists of job accounting data (see *Example of Accounting Information* in the section *Reporting*).
5. New extraction data do not overwrite existing extraction data.

Delete Old Data in Log Selection File

- Natural Program Call
- Parameters

Natural Program Call

```
LGAX1D-P  P-RETENTION-DAYS
```

Parameters

Name	Format	Description
P-RETENTION-DAYS	I2	Entries older than this parameter are deleted.

Print Account Information from Entire Operations Log

- Natural Program Call
- Parameters

Natural Program Call

```
LG-AP1-P  P-TIME-FROM  P-TIME-TO
```

Parameters

Name	Format	Description
P-TIME-FROM	A14	Format: YYYYMMDDHHIISS
P-TIME-TO	A14	Format: YYYYMMDDHHIISS

Mass Change of Network/Symbol Table Version

- Natural Program Call
- Parameters

Natural Program Call

```
OV-MB--P
P-OBJECT-TYPE P-OBJECT-OWNER P-OBJECT-NAME P-OLD-OBJECT-VERSION P-NEW-OBJECT-VERSION ↩
P-ALLOW-UPD
```

Depending on the settings of the Natural parameter ID, parameters may need to be comma separated instead of using blanks.

Parameters

Name	Format	Description	
P-OBJECT-TYPE	A2	Object type:	
		NV	Network Version.
		SV	Symbol Table Version.
P-OBJECT-OWNER	A10	Owner of object.	
P-OBJECT-NAME	A10	Object name.	
P-OLD-OBJECT-VERSION	A10	Version of object to be renamed.	

Name	Format	Description	
P-NEW-OBJECT-VERSION	A10	New version of object.	
P-ALLOW-UPDATE	A1	N	Only perform a check.
		Y	Allow updates.

Notes:

1. The Natural session parameter `IM` must be set to `IM=F`.
2. The Monitor should be stopped during the execution of this utility to avoid unintended results of network activations.
3. The program can be restarted in case of an abnormal termination to complete the rename action.
4. Every object (to be) changed is reported. The report is written to Natural main output.

Monitor or Task Wait Time Modification

- Natural Program Call
- Parameters

Natural Program Call

```
MO-WTM-P  P-MONITOR-WAIT-TIME P-TASK-ID
```

Parameters

Name	Format	Description
P-MONITOR-WAIT-TIME	I4	In seconds.
P-TASK-ID	I2	Number of a Monitor task. Only the wait time of this task will be modified.

Notes:

1. If a wait time less than 5 seconds is specified, the wait time is set to 5 seconds.
2. The modifications carried out using this program only apply until the end of the current Monitor run.

Monitor Shutdown

- Natural Program Call
- Parameters

Natural Program Call

```
MO-SHD-P
```

Parameters

None.

Monitor Start

- Natural Program Call
- Parameter

Natural Program Call

```
MO-ST--P P-MONITOR-NODE
```

Parameter

Name	Format	Description
P-MONITOR-NODE	I2	Value = 1 to 99999.

BS2000 Jobs

These jobs are used to update the submit user ID, submit password and account number.

- Natural Program Call

- Parameters

Natural Program Call

```
BS2U01-P P-OWNER P-USERID-OLD P-PW-OLD P-USERID-NEW P-PW-NEW P-ACCOUNT-NEW
```

Parameters

Name	Format	Description
P-OWNER	A10	Entire Operations owner of jobs to be modified.
P-USERID-OLD	A8	BS2000 user ID to be exchanged.
P-PW-OLD	A8	BS2000 submit password to be exchanged.
P-USERID-NEW	A8	BS2000 new user ID.
P-PW-NEW	A8	BS2000 new submit password.
P-ACCOUNT-NEW	A8	BS2000 new account number.

Notes:

1. An asterisk (*) for a parameter is converted to blank.
2. Job fields are only exchanged if the Entire Operations owner, the old BS2000 user ID, and the old submit password do match.
3. Fields are replaced, only if the new value is not blank.
4. A log is written to the screen. In batch mode, the log is printed to Natural Printer 1. This printer must therefore be assigned in the JCL.

List or Delete TO-ACTIVATE Command Records

- Natural Program Call
- Parameters

- Examples

Natural Program Call

```
TA-DEL-P P-FUNCTION P-OWNER P-NETWORK
```

Parameters

Name	Format	Description	
P-FUNCTION	A1	D	Delete TO-ACTIVATE records.
		L	List TO-ACTIVATE records.
P-OWNER	A10	*	All owners.
		other	Owner selection. Wildcard is possible.
P-NETWORK	A10	*	All networks of selected owner(s).
		other	Network selection. Wildcard is possible.

Notes:

1. With this batch utility you can list or delete the internal TO-ACTIVATE command records in the Entire Operations active database.
2. Use this utility only if you have to perform a manual removal of the TO-ACTIVATE command records.
3. This utility may be used only if the Entire Operations Monitor is not running.

Examples

```
TA-DEL-P D * *
```

will remove all TO-ACTIVATE objects.

```
TA-DEL-P D AA* *
```

will remove TO-ACTIVATE objects for owners starting with 'AA'.

```
TA-DEL-P D AAAA BBBB*
```

will remove TO-ACTIVATE objects for owner 'AAAA', networks starting with 'BBBB'.

Check the Existence of Symbol Table Definitions

- Natural Program Call
- Parameters

Natural Program Call

```
CHNWST-P P-FUNCTION
```

Parameters

Name	Format	Description	
P-FUNCTION	A1	A	List all networks without a symbol table.
		S	List subnetworks without a symbol table.

The report is written to Print File 1 in batch and to screen in online.

Mass Change of the Owner and Owner Deletion

- Natural Program Call
- Parameters
- Notes

Natural Program Call

```
OW-MB--P
P-OWNER-OLD P-OWNER-NEW P-ALLOW-UPDATE
```

Depending on the settings of the Natural parameter `ID`, parameters may need to be comma separated instead of using blanks.

Parameters

Name	Format	Description	
P-OWNER-OLD	A10	Old owner	
P-OWNER-NEW	A10	New owner. If you specify ==DELETE== here, the owner will be deleted, with all references associated with the owner such as, links to users, other owners and objects maintained by the owner.	
P-ALLOW-UPDATE	A1	N	Perform a check only.
		Y	Allow updates.

Notes

- The Natural session parameter `IM` must be set to `IM=F`.
- The Entire Operations Monitor must be stopped during the execution of this utility.
- The old and new owner names may not be SYSDBA.
- The utility runs in two passes. During the first pass, the data to be changed is checked against potential problems. An error message (line with `E:`) will result in update mode (`P-ALLOW-UPD`) switched to `N`. A typical problem is “target exists”, when owner rename would cause merging of Entire Operations objects within a network. A warning message (line with `W:`) indicates situation of interest which does not prevent from object renaming.
- The second pass is either the real update or diagnostic run depending on update mode value.
- Every object (to be) changed is reported. The report is written to Natural main output.
- The utility must run with Entire Operations System File 1 opened in Adabas exclusive (EXU) mode, which means: the Entire Operations Monitor and every other Natural session with Entire Operations System File 1 must be terminated before. This prerequisite is necessary to protect the integrity of Entire Operations objects and links between them. For this reason we suggest Natural profile parameter:

  ```
  OPRB=(EXU=fnop1,UPD=fnat,flog,ACC=fsec)
  ```

  where:

  - *fnop1* = Entire Operations System File 1
  - *fnat* = Natural system file FNAT
  - *flog* = Entire Operations or System Automation Tools log file
  - *fsec* = Natural Security system file

  Other files required to run Natural should be added to `UPD=` or `ACC=` list.
- We suggest a job with the following steps to run OW-MB--P:

- Step 0: Adabas ADAORD/ADASAV/ADAULD to save Entire Operations System File 1 when following OW-MB--P runs in update mode;
- Step 1: Adabas ADADBS utility with parameters:

```
ADADBS OPERCOM
STOPF=fnop1
ADADBS OPERCOM LOCKX=fnop1
```

- Step 2: Natural session with OW-MB--P;
- Adabas ADADBS:

```
ADADBS OPERCOM
UNLOCKX=fnop1
```

- If the utility runs in a "normal" Natural session, it will (most likely) run out of space in the Adabas WORK area or max ISN hold limit or will be waiting for records being held by others (also with update mode `N`). So we strictly recommend that you run the utility in batch mode only.
- There is another Natural profile parameter, `DBUPD=OFF`, which may be used to run this utility safely and in parallel to other sessions however the utility report may indicate false error messages due to other activities on Entire Operations System File 1.
- When OW-MB--P reports the `E: ... targets exist (xx: nnn)` error, *xx* is the two-character target code and *nnn* is the number of target records found. `xx` can be the following:

  `NM` (Network Master)
  `JM` (Job Master)
  `EM` (Event Master)
  `JA` (Job Active)
  `EA` (Event Active)
  `JC` (JCL active)
  `CO` (Conditions active)
  `SA` (Symbols Active)
  `RP` (Resource Prerequisites)
  `RU` (Resource usage)

- Existing targets are reported in the sequence above, e.g. `EM` means that targets `NM` and `JM` were not found.

  If the reported target is:

  - `JM`: create manually missing network definition and jobs will appear in it. Deactivate/delete it again;
  - `CO`: these are visible in Active Conditions.

- The rest of the targets can be cleaned up with a new special "orphan" cleanup batch utility:

```
CL-ORP-P owner update-mode
```

where all parameters are mandatory:

- *owner* (A10): P-OWNER-NEW from the erroneous OW-MB--P run (SYSDBA not allowed);
- *update-mode* (L): update mode N or Y.

- Everything about execution of OW-MB--P mentioned above applies the same way to CL-ORP-P, i.e. it must be executed with Natural profile parameter OPRB to open Entire Operations System File 1 in exclusive update mode or profile parameter DBUPD=OFF may be used for diagnostic purposes. The report about (possibly) deleted "orphan" objects is written to Natural main output.
- Be sure to keep all the reports of OW-MB--P and CL-ORP-P runs with update mode Y.

Mass Change of Access Rights Granted to Networks

- Natural Program Call
- Parameters

Natural Program Call

```
LOGON SYSEOR
NV-MB--P
 P-FUNCTION,P-OWNER,P-NETWORK,P-VERSION,P-TYPE,P-NAME,P-OPTION
FIN
```

The Natural program NV-MB--P provides the option to add, modify or delete user access rights granted to networks. NV-MB--P can run online and in batch mode.

The above Natural program call assumes that the Natural profile parameter ID (input delimiter character) is set to a comma (,).

Example:

```
LOGON SYSEOR
NV-MB--P
U,EXAMPLE,*,*,O,NOPALL,DORA
```

In the example above, the access rights of owner NOPALL change to DORA for all networks and network versions of the owner EXAMPLE.

Parameters

Name	Format	Description	
P-FUNCTION	A1	D	Delete all user access rights granted to all networks specified with P-OWNER, P-NETWORK and P-VERSION.
		U	Changes user access rights granted to all networks specified with P-OWNER, P-NETWORK and P-VERSION: ■ Access rights are modified to P-OPTION for the user specified with P-TYPE and P-NAME. ■ Access rights are added if the user specified with P-TYPE and P-NAME does not yet exist.
P-OWNER	A10	Name of the network owner or a range of names specified with a wildcard (* or ?). You can only use one parameter to specify a name range.	
P-NETWORK	A10	Name of the network or a range of names specified with a wildcard (* or ?). You can only use one parameter to specify a name range.	
P-VERSION	A10	Name of the network version or a range of names specified with a wildcard (* or ?). You can only use one parameter to specify a name range.	
P-TYPE	A1	Only required for P-FUNCTION=U. User type specification:	
		O	Owner.
		U	User.
P-NAME	A8	Only required for P-FUNCTION=U. Name of the owner or user whose access rights you want to change.	
P-OPTION	A5	Only required for P-FUNCTION=U. Options for access restrictions (multiple values are possible):	
		R	Read access.
		W	Read and write access.
		D	Read, write and delete.
		O	Owner rights with permission to grant access to other users.
		A	Network activation rights.

User ID Mass Update in Network and Job Definitions

The US-UPD-P utility performs list or change operations on user IDs defined in networks and jobs.

User IDs comprise default, submit or SYSOUT user IDs, and group or domain names (UNIX and Windows only).

- Natural Program Call
- Parameters

Natural Program Call

For list operations:

```
US-UPD-P P-FUNCTION
```

For change operations:

```
US-UPD-P P-FUNCTION P-OLD-VALUE P-NEW-VALUE [P-NODE-A5] [P-OWNER] [P-NETWORK] ↩
[P-NETWORK-VERSION] [P-JOB-TYPE]
```

Parameters

Name	Format	Description	
P-FUNCTION	A3	Function codes provided to list or change user IDs:	
		NLE	List networks with empty default JCL user ID or empty submit user ID.
		NSU	Change network default submit user ID.
		NSG	Change network default submit group.
		NJU	Change network default JCL user ID.
		NJG	Change network default JCL group.
		JLE	List jobs with empty default JCL user ID or empty submit user ID for which no default user ID exists on the network level.
		JMU	Change job modification user ID.
		JSU	Change job submit user ID.
		JSG	Change job submit group.
		JJU	Change job JCL user ID.
		JJG	Change job JCL group.
		JSB	Change SYSOUT user ID (BS2000 only).

Name	Format	Description	
P-OLD-VALUE	A20	Search value for the IDs or names to be changed. Possible values are:	
		(ANY)	Select all defined user IDs.
		(EMPTY) or blank	Select all user IDs for which no names are defined (empty field values).
		text-string	Select all user IDs that match the specified text string.
		Values are treated as case-sensitive. Exception: If P-NODE-A5 is specified for a mainframe node, the given value is converted to uppercase characters.	
P-NEW-VALUE	A20	Replace value for the user IDs selected with P-OLD-VALUE. All user IDs selected with P-OLD-VALUE are replaced by the value specified with P-NEW-VALUE. Values are treated as case-sensitive. Exception: If P-NODE-A5 is specified for a mainframe node, the given value is converted to uppercase characters.	
P-NODE-A5	A5	Node name defined in Entire Operations (optional). SYSOUT definitions can only be used by jobs that have this node defined as a submit node.	
		Possible values are:	
		blank or 0	Select all networks and jobs for which a valid JCL or execution node name is defined.
		node-name	Select all networks and jobs whose JCL or execution nodes match the specified node name. The node name is either the number or short name defined for the node.
P-OWNER	A10	Owner, network and/or network version (all optional) to be selected. These fields can be used for wildcard range definition. A wildcard is allowed in the last specified field only.	
P-NETWORK	A10		
P-NETWORK-VERSION	A10		
P-JOB-TYPE	A3	Job type to be selected (optional). Possible values are:	
		blank	Select all job types.

Name	Format	Description	
		or *	
		job-type	Select all jobs of the specified three-character code, for example, DUM for dummy jobs. For valid job types, see ***Available Job Types*** in the section *Job Maintenance*.

Important: For change operations, this utility may be used only if the Entire Operations Monitor is not running.

Mass Update for User Access to Nodes

The utility NOPUNA-P is used to add or update single or multiple users for node access through defined operating system IDs.

Natural Program Call

```
NOPUNA-P
```

When running NOPUNA-P in batch mode, Work File 1 must be defined with LRECL 130. The work file can contain comment lines denoted by asterisk (*), double asterisk (**) or slash asterisk (/*) in the first and second positions of the record.

For each user ID to be added or updated, one line must be defined in the following format:

```
nop-userid,node-number,os-userid,os-group,ld-value,aj-value,as-value
```

Each successful user update is logged in the Entire Operations log.

Example:

```
EXAMPLE,517,nprusr01,Nprgrp01,Y,Y,Y
```

Fields in the Input Lines

The variable fields used for the program call are described in the following table:

Name	Format	Description
nop-userid	A8 (upper case)	ID of the Entire Operations user provided node access.
node-number	N5	Number of the Entire Operations node to be accessed.
os-userid	A50 (mixed case)	ID of the operating system assigned to the node.
os-group	A20 (mixed case)	Name of a Windows or UNIX group (if defined) to which the operating system ID belongs.
ld-value	A1	Allow (Y) or disallow (N or blank) automatic node logon.
aj-value	A1	Allow (Y) or disallow (N or blank) automatic node logon for browsing and editing JCL only.
as-value	A1	Allow (Y) or disallow (N or blank) automatic node logon for browsing SYSOUT only.

Mass Change of Node Numbers

Natural Program Call

```
ND-MB--P LOCATION NODE-OLD NODE-NEW OWNER NETWORK NETWORK-VERSION
```

Parameters

Name	Format	Description
LOCATION	A3	All occurrences of node will be changed in master definitions of network, job or event. For other values, refer to the **list of JCL locations**.
NODE-OLD	N5	Node number.
NODE-NEW	N5	Node number.
OWNER	A10	Owner filter (optional).
NETWORK	A10	Network filter (optional).
NETWORK-VERSION	A10	Network version filter (optional).

Notes

1. Both NODE-OLD and NODE-NEW must be defined and valid and of the same operating system class, i.e., you can change OS/390 to a z/OS node.
2. If no value is specified for OWNER, NETWORK, NETWORK-VERSION, an asterisk (*) is used by default.
3. You can use an asterisk (*), a greater than sign (>) or a smaller than (<) sign as a wildcard character.

   If a wildcard appears in OWNER, NETWORK contains an asterisk (*).
4. Program ND-MB--P can be run safely (only to list objects) in a Natural session with the Natural profile parameter DBUPD=OFF set.

Mass Logon Processing in Batch Mode

- Natural Program Call
- Fields in the Input Lines
- Single Node Logons without using a Work File

Natural Program Call

```
ND-LB--P
```

The utility ND-LB--P must be run in a Natural batch environment. The logging is written both to SYSOUT and to the Entire Operations log (SAT log). Passwords will not be logged.

The input is read from Natural Work File 1. For each logon to be performed, one line must be defined with the format:

```
node userid group password
```

Example:

```
517 sag users sag-pwd
```

Comment lines must have an asterisk (*) as first character. The output is written to the Natural batch job SYSOUT and to the Entire Operations log.

Fields in the Input Lines

Name	Format	Description
node	N5	The Entire System Server node, to which the logon is to be performed.
userid	A20	User ID for the logon.
group	A20	Group for the logon. For Windows nodes, *group* must contain the domain. If no group or domain is used, enter a hyphen (-) instead. Example: `148 USER - USER-PWD`
password	A20	The password for the logon. The *password* parameter must be enclosed in double quotes ("*password*"), if the password contains at least one blank. Passwords with trailing blanks are not supported.



Notes:

1. This utility must not be used if the Entire Operations Monitor is running.
2. All nodes, which are handled, must be active (running and reachable).
3. There may be one or several blanks between the parameters.
4. If no group or domain is used, enter a hyphen (-) instead.
5. For UNIX and Windows nodes, the input is case-sensitive.
6. For mainframe nodes, the text strings will be converted to upper case before the logon is performed.



Caution: Users of this utility must be aware that the input file is very security-sensitive, because it contains passwords in clear text.

Single Node Logons without using a Work File

To allow single node logons without using a work file, the parameters can be passed on the Natural command line too.

The command line parameters are:

1. node number
2. user ID
3. group (UNIX) resp. domain (Windows)

(Pass a hyphen (-) if the default group resp. no domain is to be used.)

4. password

The parameters are case-sensitive.

On return, ND-LB--P provides a return code in the AIV variable +ND-LB--P-RC (I4).

It contains 0 (zero) if the logon was successful, and any other error code in case of an error.

Bulk Execution of MACRO Commands

- Program: MAC-GE-P

Program: MAC-GE-P

The program performs the MACRO command in batch mode for a given selection of job network(s).

Natural Program Call

(with the Natural parameter ID=, being set)

```
LOGON SYSEOR
MAC-GE-P
MODE,OWNER,NETWORK,NETWORK-VERSION
FIN
```

Example:

```
LOGON SYSEOR
MAC-GE-P
A,SN,A-1,v-unnamed
FIN
```

Parameters

Name	Format	Description	
MODE	A1	A	Catalog all jobs with **JCL location** MAC.
		M	Catalog jobs with **JCL location** MAC, which were not yet cataloged.
OWNER	A10	Owner filter.	
NETWORK	A10	Network filter.	
NETWORK-VERSION	A10	Network version filter.	

The parameters P-OWNER, P-NETWORK and P-NETWORK-VERSION may contain selection wildcards:

```
*, >, <, ?
```

Notes:

1. The Program MAC-GE-P must be started in batch.
2. The Natural profile parameters `LC=ON, EDPSIZE=100` must be set.
3. On mainframe systems. The Entire Operations Monitor must be active. The task 90 must be running.

Result Checking

The SYSOUT of the batch job contains detailed information about the cataloging. The results must be checked.

Example:

```
Owner      Network    Version    Job        Library  Member   Result
SN         A-1        v-unnamed  MAC1-541   EOR-T541 MAC1     Done
SN         A-1        v-unnamed  MAC1N511   EOR-N541 MAC1     Error
>>> NAT0806 - Library not found.
```

Data Migration to the current Entire Operations Version

The data migration to the current Entire Operations version adapts various objects and fields, so that they can be used by the current Entire Operations version,

```
DMDRIV-P
```

It is recommended to use a Natural batch job to perform the migration. The Natural commands are:

```
LOGON SYSEOR
DMDRIV-P
FIN
```

Notes:

1. A data migration consists of several independent migration steps.
2. Each migration step will be performed only once. Internal control records prevent multiple executions.
3. The migration invocation may be repeated as often as necessary.
4. The migration is logged into the Natural SYSOUT.

5. The data migration may be performed only if the Entire Operations Monitor is not active.

IV Owner Maintenance

13 Owner Maintenance

- Available Functions: Owner 96
- Listing Owners 97
- Owner at Logon 99
- Linking Additional Owners 99
- Owners Granted Access to Individual Networks 99
- Adding and Removing an Owner 101

Each user ID is associated with single or multiple owners in order to group network maintenance.

You can only list, access and maintain networks that belong to associated owners (unless you have administrator rights):

- The main (mandatory) owner to which your user ID is **assigned at logon**,
- The owners **linked to your user ID**, and
- The owners who **granted you individual network access**.

This section provides an overview of the functions available to view and assign owners.

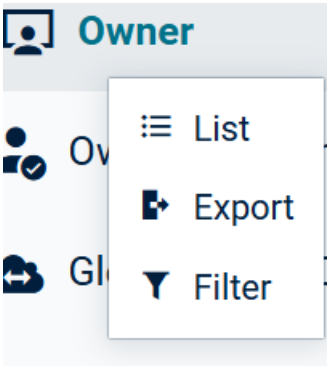
Related Topics:

- The concept of user/owner assignments is described in the section *Users and Owners* in the *Concepts and Facilities* documentation.

Available Functions: Owner

To list all available functions for the Owner metanode

- In the object workspace, select the **Owner** metanode and open the context menu.



The context menu provides the following functions:

Function	Symbol	Explanation
List		Lists all owners linked to your user ID.
Filter		see *List Owners according to Selection Criteria* and .
Export		**Export Objects** window to export all items of the metanode **Owner**: see *Exporting Objects* in the *Import/Export Functions* documentation.

To list all available functions for an Owner instance

- In the object workspace, select an owner from the **Owner** metanode and open the context menu.

  The context menu provides the following functions:

Function	Explanation
Next Activations	See *Displaying Next Network Starts - Next Activations* in the section *Schedule Maintenance*.
List Active Jobs	See *List Active Jobs*.
Browse Log	See *Browsing the Log*.
Export	Opens the **Export Objects** window to export an owner: see *Exporting Objects* in the *Import/Export Functions* documentation.

Listing Owners

- Listing Owners Linked to a User
- Listing All Owners
- List Owners according to Selection Criteria

Listing Owners Linked to a User

To list all owners linked to a current user

- Expand the **Owner** metanode node or choose **List** from the **Owner** context menu.

  All owners linked to the current user are listed in the tree node or list window of the content pane, respectively.

For explanations of the **Owner via Granting** metanode, see *Owners Granted Access to Individual Networks*.

Listing All Owners

This function requires administrator rights.

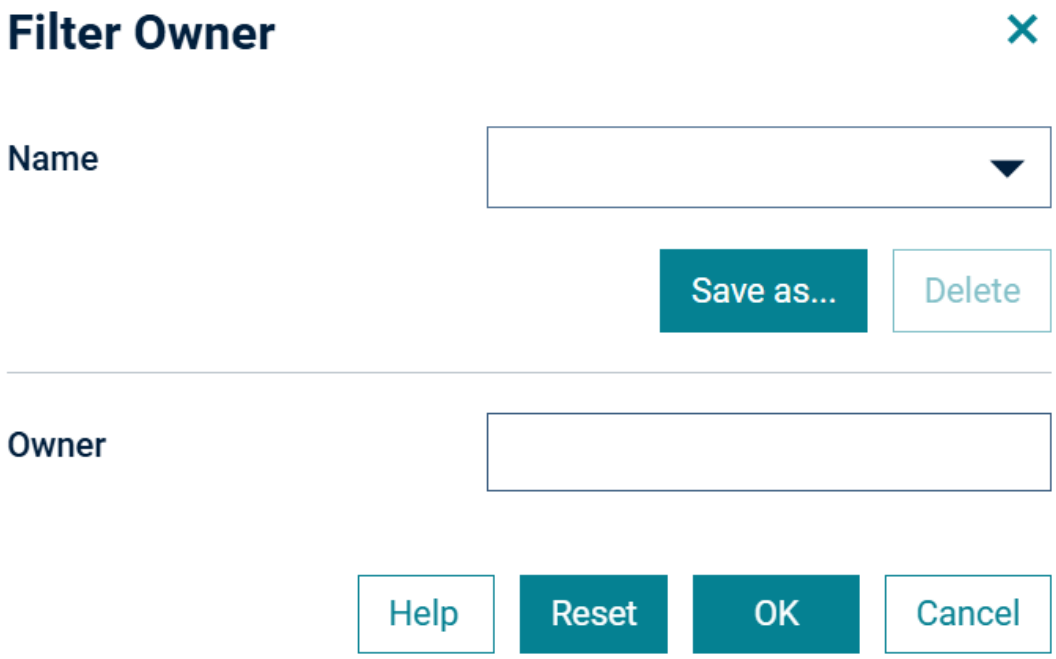
List Owners according to Selection Criteria

To list owners according to selection criteria

1 In the **object workspace**, select the **Owner** metanode and open the context menu.

2 Choose **Filter** ( )

A dialog opens.



3 Enter your search criteria.

See also *Filtering Objects*.

Owner at Logon

Your user ID is assigned an **Owner at Logon** (main owner) in your user profile as described in *Viewing, Adding and Modifying a User* (*Administration* documentation).

When you log on to Entire Operations, your main owner (in the previous example, the owner `EXAMPLE`) is set for your current session by default. The display is filtered to show only those networks belonging to the main owner and its associated owners.

To change the current session owner

- On the **Main** page of the **Maintenance User** window (see the *Administration* documentation), replace the name of the owner entered in the **Owner at logon** box.

  You can only change the session to an owner which is associated with your user ID (unless you have administrator rights).

Linking Additional Owners

(Administrator rights required)

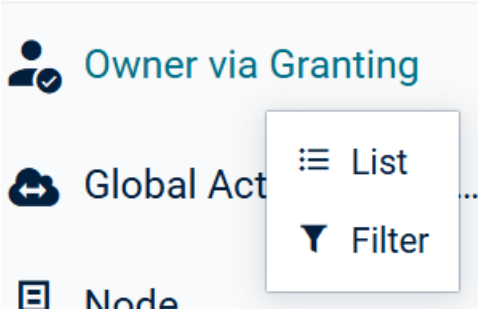
In addition to the **Owner at Logon** specified in your user profile, you can link other owners to your user ID as required. See *Adding and Removing User/Owner Links and Owners* in the *Administration* documentation.

Owners Granted Access to Individual Networks

You can grant a user or an owner unrestricted or restricted access to a selected network by using the **Granting Definition** option of the network maintenance function described in ***Granting Definition: Authorizing Other Users or Owners to Access a Network*** in the section *Network Maintenance*.

Listing Granted Owners

The **Owner via Granting** metanode lists all owners who granted network access for your user ID.



The hierarchy below the **Owner via Granting** metanode shows networks granted to your user ID or to an owner linked to your user ID if the owner of these networks is not linked to your user ID. A granted owner who is also linked in your user profile is only listed in the **Owner** node.

All networks are separated in the tree view hierarchy by their respective owner.

The context menus of the **Owner via Granting** nodes provide the following functions:

Function	Shortcut	Explanation
List		Lists all owners who granted you access. This list corresponds to the owner nodes in the tree view of the object workspace. See also *Example of an Owner via Granting List* below.
Filter		Selection criteria for listing granted network masters: see *Filtering Objects*.

The functions available for a granted network master correspond to the functions available for a network master: see *Available Functions: Network Master*.

To list all owners via Granting

1 In the **object workspace**, select **Owner via Granting** and open the context menu.

2 Choose **List** ( )

The list of the **Owners via Granting** is displayed.

To list owners via granting according to selection criteria

1 In the **object workspace**, select **Owner via Granting** and open the context menu.

2 Choose **Filter** ( )

A dialog opens.

3 Enter your search criteria.

A list of **Owners via Granting** filtered according to selection criteria is displayed.

Adding and Removing an Owner

An owner is added when entering a name for **Owner at Logon** or during a user/owner link operation.

An owner is deleted when replacing an **Owner at Logon** or removing a link to a user.

For detailed information, see *Adding and Removing User/Owner Links and Owners* in the *Administration* documentation.

V Network Maintenance

This section provides information on how to define, maintain, control and monitor job networks.

Job networks are maintained on the master database which stores all user, job network, job and scheduling definitions. It also contains all information pertaining to defined logical conditions, resources, calendars, and symbol tables. All information stored on the master database can be maintained online.

For information on all definitions made at the job and event level, see the section *Job Maintenance*.

General

Use of Job Networks

Subnetworks

Job Network Maintenance

Maintaining Job Network Definitions

- Available Functions: Network Master
- Listing all Network Definitions
- Filtering Objects

Maintaining Job Network Versions

Viewing and Maintaining a Job Network Diagram

Maintenance Functions for Job Network Definitions

Adding a Network Definition

- Fields: Network Definition
- Retention of Active Network Data
- Operating System and Environment Defaults
- Specifying User Exits for Symbol Modification

- Specifying Recipients for Network Messages
- Granting Definition: Authorizing Other Users or Owners to Access a Network
- Documenting Your Networks
- Scheduling a Network
- Viewing a Network Schedule Definition as a Calendar

Displaying a Network Definition
Modifying a Network Definition
Applying Network Defaults to Jobs (Mass Update)
Copying Job Network Definitions
Deleting a Job Network

Control and Monitoring Functions for Job Networks

Displaying Next Network Starts - Next Activations
Activating a Job Network Manually
Checking for a Loop in a Job Network
Listing Usable Symbol Tables
Listing Active Jobs
Displaying Logged Information - Browse Log Function
Displaying the Network Execution History

Related Topic

- *Defining and Using Monitor Start Networks* in *Special Monitor Functions and Batch Jobs* in the *Administration* documentation.

14 Use of Job Networks

A job network is a group of jobs that stand in defined relation to each other. This relation is composed of dependencies, which are expressed as logical conditions. In the simplest case, two jobs in a job network can be linked by the condition: If Job 1 finishes OK, start Job 2 (see ***Example of Job Linkage by Using Conditions***).

A job network is uniquely defined by its owner and network name. Each network is given a start and deadline time which determine when the network is to be activated. If your installation includes multi-CPU support, you can also specify a default node name for the jobs in the network. This node name can be overridden at the job level (see *Operating System Server Nodes* in the *Concepts and Facilities* documentation).

A user can only access a defined job network if the user's ID is associated with the same owner as the network, unless the user has special authorization to access other networks (see ***Granting Definition: Authorizing Other Users or Owners to Access a Network***).

A job network and a single job are the units of work that can be activated by Entire Operations. When a job network is activated, it is automatically given a **run number** that uniquely identifies this network activation. This feature allows several copies of the same job network to run simultaneously.

15 Subnetworks

A job network can be a **subnetwork** of another job network. This allows you to build nested networks.

The job type `NET` (subnetwork) allows you to define a whole network as a job within a main network. The subnetwork must already be defined. The same subnetwork can be defined in different jobs of the main network.

The subnetwork can be activated together with the calling network (or job of the type NET) or at the moment the job of the type `NET` is started.

Several active copies (or activations) of a network can work in parallel, since Entire Operations identifies each copy uniquely by its **run number**, which is automatically assigned to each network at activation time. Within subnetworks, other subnetworks can be called in turn. However, subnetworks must not be called in themselves, otherwise an infinite recursion would occur.

For further information refer to *Maintenance Function for Subnetworks* in the section *Job Maintenance*.

16 Maintaining Job Network Definitions

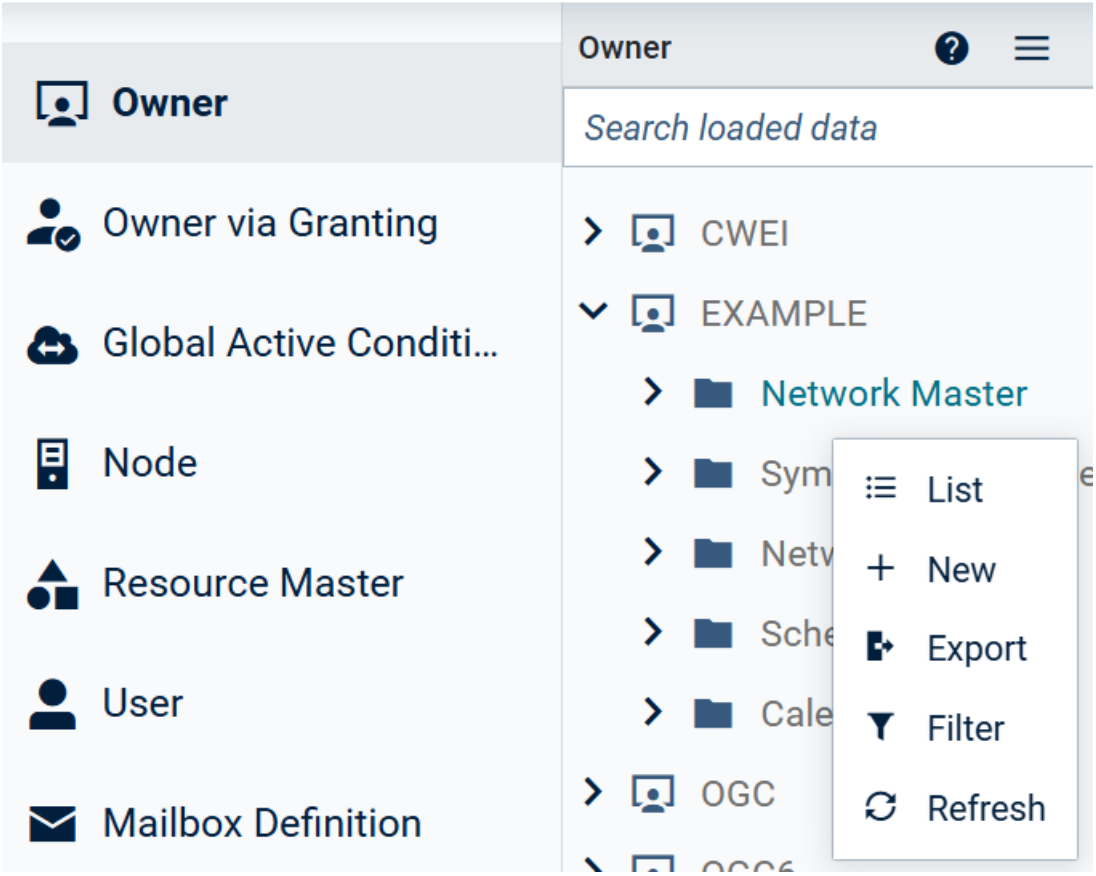
- Listing all Network Definitions 110
- Available Functions: Network Master 112
- Adding a Network Definition 114
- Displaying a Network Definition 115
- Modifying a Network Definition 115
- Fields: Network Definition 116
- OS Specials - Operating System and Environment Defaults 121
- Specifying Recipients for Network Messages 124
- Granting Definition: Authorizing Other Users or Owners to Access a Network 125
- Long Description - Documenting Your Networks 127
- Deleting a Job Network 128

Note: The **Active Run** subnode of the **Network Master** node correlates with the **Active Run** subnode of the **Network Active** node. Both subnodes control the same active runs. If you use both subnodes to change active runs during a session, consider refreshing them manually to make sure that you view the latest status.

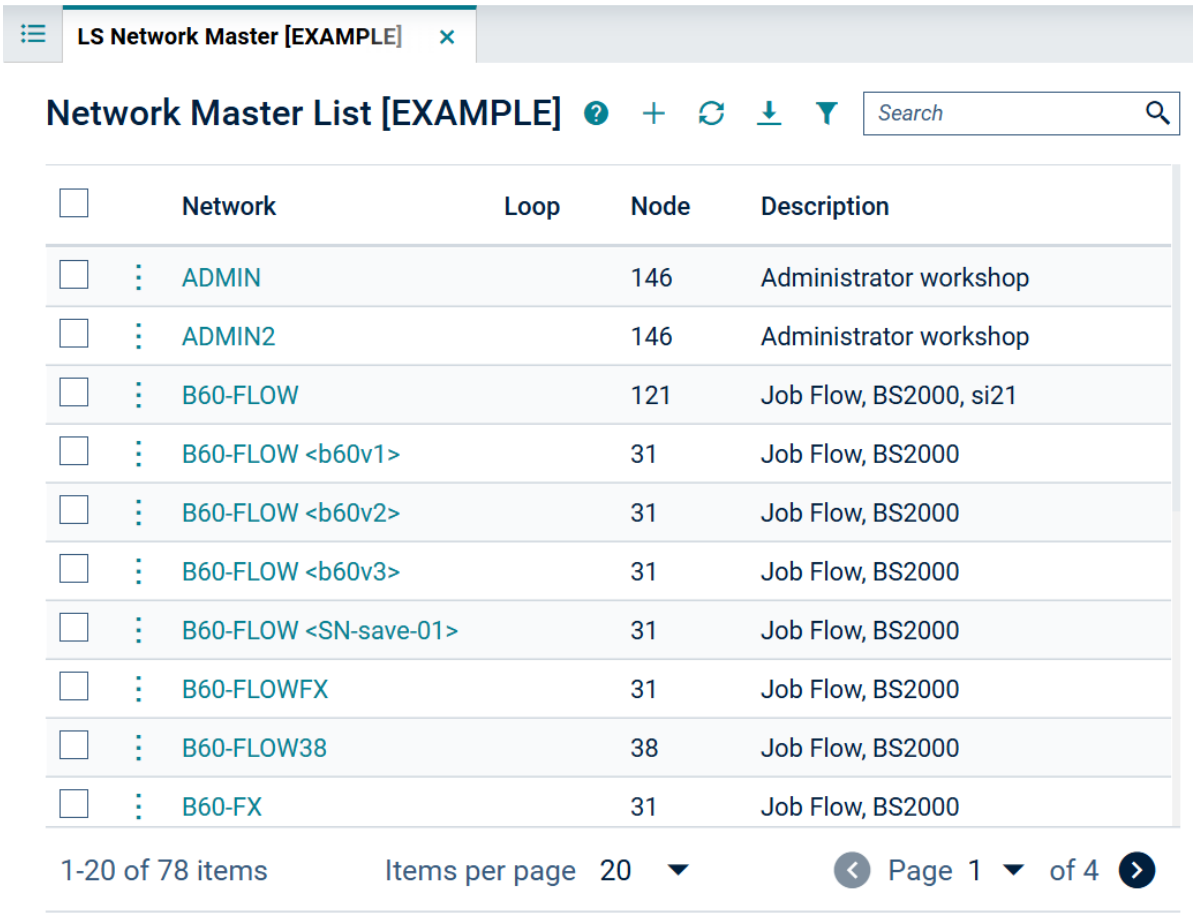
Listing all Network Definitions

To list all networks defined for an owner

- In the **object workspace**, choose **Owner** > *owner name* > **Network Master** and choose **List** ( ) from the context menu.



A **Network Master List** window like the example below opens:



The window lists all networks defined for the selected owner.

For further information, see:

- *Columns: Network Master List*
- *Available Functions: Network Master*

Columns: Network Master List

The **Network Master List** window contains the following columns:

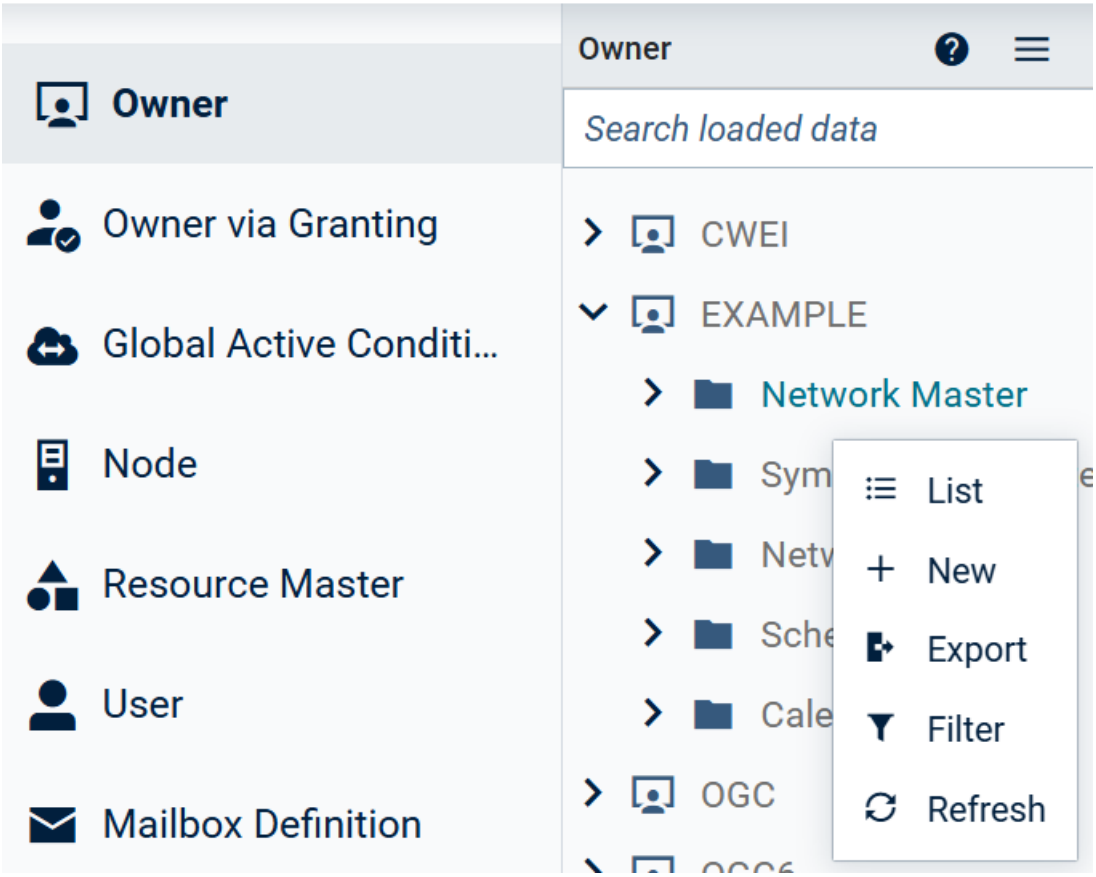
Column	Description	
Network	User-defined network name.	
Loop	blank column	No loop was found during a loop test for the corresponding network.
	`Yes`	A loop was found during a loop test for the corresponding network.
Node	Default **execution node** for the jobs in the network.	

Column	Description
Description	Short description of the network.

Available Functions: Network Master

To list all available functions for the Network Master metanode

- In the **object workspace**, choose **Owner** > *owner name* > **Network Master** and choose **List** from the context menu.



In the object workspace, select a **Network Master** metanode and open the context menu.

The context menu provides the following functions:

Function	Shortcut	Description
List		See *Listing All Network Definitions*.
New	+	Creates a new network master instance: see *Adding a Network Definition*.
Refresh		See *Refreshing Object Lists*.
Filter		Selection criteria for listing network masters: see *Filtering Objects*.
Export		See *Exporting Objects* in the *Import/Export Functions* documentation.

To list all available functions for a Network Master instance

- In the **object workspace**, choose **Owner** > *owner name* > **Network Master** > *network name* and open the context menu.

  The context menu provides the following functions:

Function	Shortcut	Description
Open	CTRL+O	**Modifies the selected network definition**.
Display	CTRL+D	Displays the selected network definition.
Diagram	---	Displays an overview of the job flow within this network. See *Viewing and Maintaining a Job Network Diagram*.
Delete	DELETE	**Deletes the selected network**, including all jobs and all other definitions for this network.
Copy data	CTRL+C	Copies all definitions of the selected network to paste them as a new network: see *Copying Job Network Definitions*.
Copy Defaults	---	Copies network attribute definitions of the selected network as the default for all jobs contained in the network: see *Applying Network Defaults to Jobs (Mass Update)*.
Activate Network	---	**Activates the selected network manually**.
History	---	**Displays the execution history** (previous network runs).
Define Scheduling	---	Opens a **Define Scheduling** window where you can add or change the schedule of the selected network as described in *Scheduling a Network*.
Display Schedule	---	Displays the **calendar schedule** defined for the selected network. See also *Viewing a Network Schedule Definition as a Calendar*.
Next Activations	---	See *Displaying Next Network Starts - Next Activations* in the section *Schedule Maintenance*.
Usable Symbol Tables	---	Opens the **Usable Symbol Tables** window to view or modify all available symbol table definitions. See also *Listing Usable Symbol Tables* in the section *Symbol Table and Symbol Maintenance*.
Check for Loop	---	**Checks for loops in the network**.

Function	Shortcut	Description
Version Usage	---	**Maintains the usage of network versions**.
List Active Jobs	---	Lists active jobs defined for this network: see *Listing Active Jobs*.
Browse Log	---	See *Displaying Logged Information - Browse Log Function*.
Export	---	See *Exporting Objects* in the *Import/Export Functions* documentation.
Add to Workplan	---	Stores a reference to the currently selected object in a list of activities to be done. See also *Add to Workplan*.
Set Drag And Drop Function	---	See *Drag & Drop Function*.

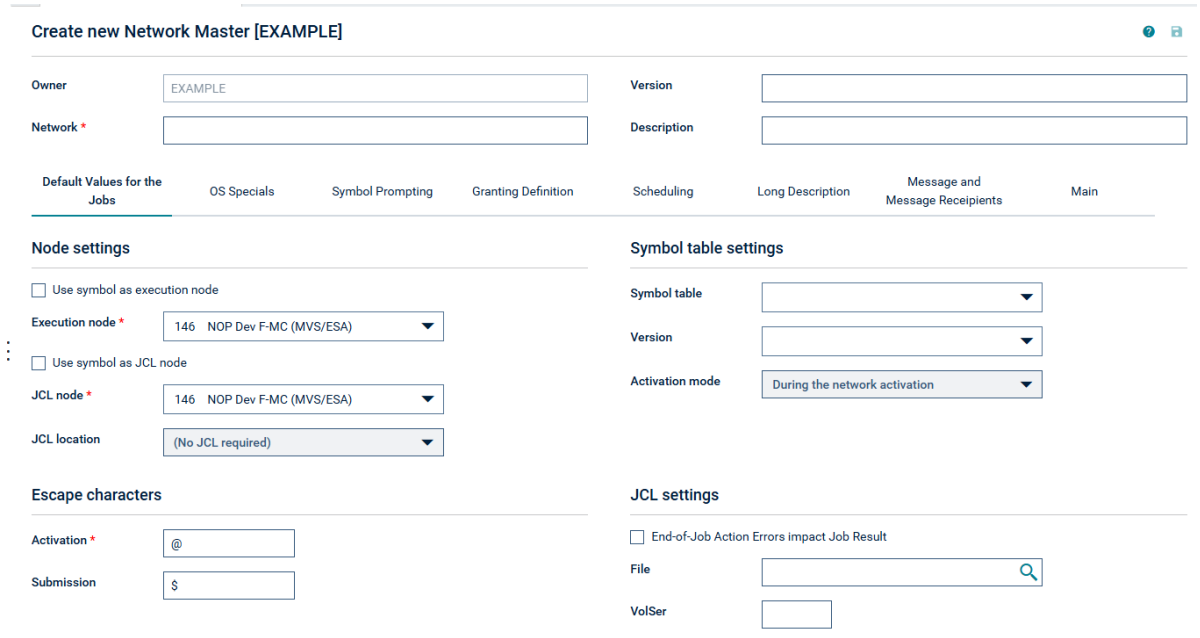
Adding a Network Definition

To create a new network master instance

1 In the object workspace, select a **Network Master** instance.

2 Invoke the context menu and choose the **New** (+) function.

A **Create new Network Master** window opens:



The fields contained in the window are explained in *Fields: Network Definition*.

3 Enter the required values and select the required options.

4 Choose **Save**( )

Your entries are saved.

Displaying a Network Definition

Displays job value, operating system special, symbol prompting and granting definition information for the currently selected Network Master.

The **Display** function invokes a dialog that displays the information for the currently selected network definition. You can only view the information, not edit it. If you want to edit the item, you have to use the **Open** function.

To display a network definition

1 In the object workspace, select an owner from the **Owner** node and then a network definition from the **Network Master** node.

2 Invoke the context menu and choose the **Open** function.

A **Network Master** window opens that displays all available information for the selected network definition.

The fields and tabs of the **Network Master window** are explained in *Fields: Network Definition*.

Modifying a Network Definition

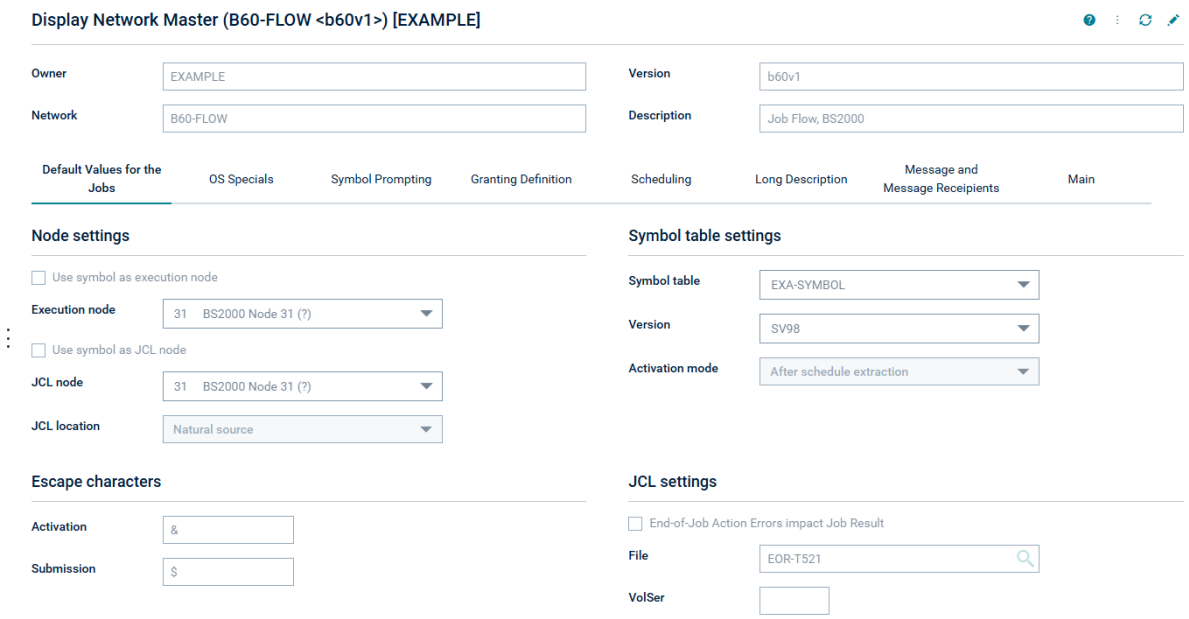
Opens job value, operating system special, symbol prompting and granting definition information for the currently selected network master in editing mode.

To modify a network definition

1 In the object workspace, select a **Network Master** instance.

2 Invoke the context menu and choose the **Open** function.

A **Network Master** window like the example below opens:



3 Choose **Edit** (✎)
The save symbol (💾) is shown in the title bar.

4 Now, you can edit the fields you want to change.

The fields and tabs of the **Maintenance Network Master** window are explained in *Fields: Network Definition*.

5 Choose **Save**(💾) .

Your changes are saved.

Fields: Network Definition

The fields and tabs of the **Create/Maintenance Network Master window** are described in the following table:

Field/Tabbed Page	Description
Owner	Owner of the network.
Network	Network name. Together with owner name, uniquely identifies the network in the master database.

Field/Tabbed Page	Description
Version	Network version. You can add a new network with a blank (`unnamed`) version or an alphanumeric version name: see *Version Names* and *Reserved Version Names for Networks*. You can use the **Create new Network Master** window for all additional versions you want to add to an existing network. New network versions can also be created by copying networks as described in *Copying Job Network Definitions*. For further information, see *Maintaining Job Network Versions*.
Description	Short description of the network. This text appears in the list of networks in the **Network List** window. A longer description of the network can be added using the editor (see *Documenting Your Networks*).
Default Values for the Jobs	The input fields on this tabbed page are used to specify default values for subsequent new job definitions in the network. Each default value specified here can be overridden at the job level.
Use symbol as execution node	Mark this option if you want to use a symbol for the **execution node** (see below).
Execution node	Default node ID on which jobs within this network are submitted. This value can be modified here or overridden at the job level (see the field **Execution Node** in *Job Maintenance*). The drop-down list box shows all nodes available for selection. The operating system type appears after a valid node number. You can define the execution node as a symbol if the **Use symbol as execution node** option (see below) is selected. For details, see *Symbols in Node Definitions* in the section *Symbol Table and Symbol Maintenance*.
Use symbol as JCL node	Mark this option if you want to use a symbol for a **JCL node** (see below).
JCL node	Node on which JCL can be accessed. The default is the same as for the **Execution node**. The operating system type appears after a valid node number. The context menu of the selected node is the same as for **Execution node**. You can define the JCL node as a symbol if the **Use symbol as JCL node** option (see above) is marked. For details, see *Symbols in Node Definitions* in the section *Symbol Table and Symbol Maintenance*.

Field/Tabbed Page	Description
JCL location	Type of storage for the JCL: see *List of JCL Locations*. The default is used in new job definitions and can be overridden there.
Symbol table	Default symbol table for those jobs in the network that use the dynamic JCL generation facility. Can be overridden at the job level and is therefore optional here. The drop-down list box shows all symbol tables available for selection. Using the context menu, you can choose various commands, see *Available Functions: Symbol*. You can also use the context menu in the empty field to choose one of the following functions: Command — Function **New** — A **Create new Symbol Table Master** window opens. Proceed as described in *Adding a Symbol Table Master*. You can then select the newly created symbol table from the selection list of the field **Symbol table**. **Paste data** — See below. **To insert a symbol table** 1 Copy the symbol table to be inserted to the clipboard as described in *Copying Objects*. 2 In the empty field **Symbol table** choose **Paste data** from the context menu. The window **Paste data Symbol Table Master** is displayed as described in *Pasting Objects*. 3 Replace the entry of column **Object name to paste** with the new name of the symbol table and choose **OK**. The new symbol table is created. It can be selected from the selection list of the field **Symbol table**.
Version (symbol table)	You can maintain several versions of a symbol table. The drop-down list box shows all symbol table versions available for selection. Reserved names: `blank` — Unknown (implicit) version. `(current)` — Is replaced by the current version for the activation date or determination date. `(nv)` — Is replaced by the version of the network used. For further information, see *Versioning of Symbol Tables* in the section *Symbol Table and Symbol Maintenance*.

Field/Tabbed Page	Description
Activation mode (symbol table)	From the drop-down list box you can choose either of the following: ■ During the network activation. No **symbol prompting** is possible. ■ After schedule extraction. Symbol prompting can be used for scheduled networks. This is the default.
Activation (escape characters)	Activation Escape Character = Network default value This escape character is the prefix for Natural code lines and symbols to be replaced at activation time. **Caution:** Existing Dynamic JCL might become invalid after changing this escape character and applying defaults to jobs.
Submission (escape characters)	Submission Escape Character = Network default value This escape character is the prefix for Natural code lines and symbols to be replaced at submission time. **Caution:** Existing Dynamic JCL might become invalid after changing this escape character and applying defaults to jobs.
End-of-Job Action Errors impact Job Result	If you select this check box, for all output condition(s) defined for job events at the step level, the default checks **All Checks ok** or **Any Check not ok** are performed after all End-of-Job actions are performed. This includes End-of-Job action (EJA) exits. Any error that occurs in the End-of-Job action processing then causes the setting of the conditions for job `not ok`. If the job was already set to `not ok`, it will remain `not ok`, regardless of the definition here. **Note:** 1. Setting this option may cause a longer elapsed time of a network, because the condition setting waits for the termination of the other End-of-Job actions. 2. The network level setting will be overridden by job settings. 3. This option does not change the settings of any conditions defined for any individual events. If the check box is not selected, errors during End-of-Job action processing will have no impact on the job result. Job level: If the field is empty, the network level definition will be inherited at activation time. This is the default. See also the field **End-of-Job Action** in the job master definition.
File	Name of the file or Natural library according to the value of the **JCL Location** field. For BS2000: The default pubset will be stripped from the file name prior to storing it. This allows easier migration to another default pubset.

Field/Tabbed Page	Description
VolSer	Volume serial number of data set (if data set is not cataloged).
OS Specials	This tabbed page is described in *OS Specials - Operating System and Environment Defaults*.
Symbol Prompting	This tabbed page is described in *Symbol Prompting during Network or Job Activation* in the section *Symbol Table and Symbol Maintenance*.
Granting Definition	This tabbed page is described in *Granting Definition: Authorizing Other Users or Owners to Access a Network*.
Scheduling	This **tabbed page** is described in *Scheduling a Network* in the section *Schedule Maintenance*.
Long Description	This tabbed page is described in *Long Description - Documenting Your Networks*.
Message and Message Recipients	This tabbed page is described in *Specifying Recipients for Network Messages*.
Main	This tabbed page is used to specify retention periods for an active network and active conditions. See *Retention of Active Network Data*.

Retention of Active Network Data

You can specify how long data from an active network run is stored in the active database by using the **Retention period for network** and **Deactivation mode for active conditions** options provided on the **Main** page of the **Create/Maintenance Network Master window**.

The settings specified here override the retention periods for active networks and active conditions, respectively, set in the Entire Operations defaults (see *Defaults for Time Ranges* in the *Administration* documentation).

Retention Period for Network

This option is used to keep active network data in the active database for the minimum number of days specified in the **days** field.

The following rules apply:

- Valid input values for **days** are 1 to 9999.

  If no value is entered in this field (default), the default retention period set for active networks in the Entire Operations defaults is used.
- If the network is not completed within the given period, a warning message is sent to the mailbox(es) associated with the network.
- For unfinished active jobs, the retention periods set in the Entire Operations defaults and the retention period set for the single network calculate the actual retention period as follows:

  (active jobs default - active networks default) + single network period = actual retention period

Example: (12 days - 10 days) + 5 days = 7 days

Deactivation Mode for Active Conditions

This option is used to remove active job conditions from the active database after a network run.

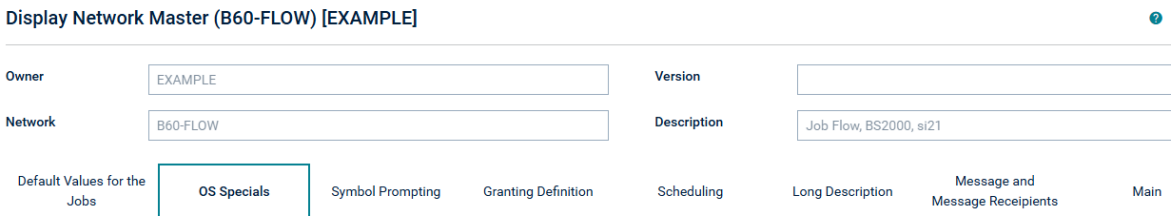
Possible option settings are:

The global retention period for active conditions applies	Remove active conditions after the period (1 or more days) specified as retention period for active conditions in the Entire Operations defaults under **Time Ranges**. This is the default setting.
For manual and API network deactivations	Remove active job conditions set by a network run immediately after the network is deactivated either manually or by an API (application programming interface).
For all network deactivations	Remove active conditions set by a network run immediately after the network is deactivated either manually or automatically.

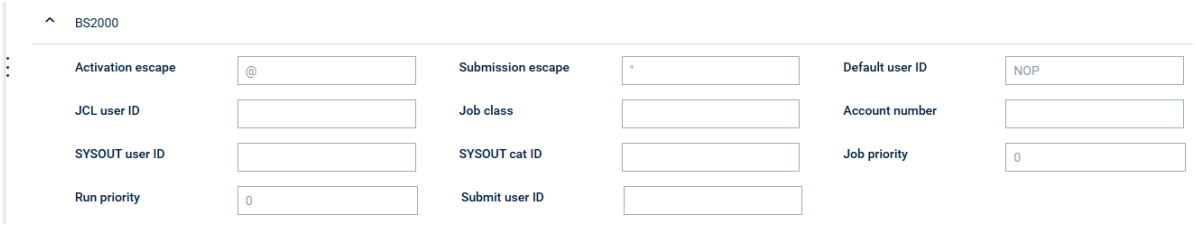
OS Specials - Operating System and Environment Defaults

To specify operating system specific defaults

1 In a **Network Master window**, select the **OS Specials** tab and then the required operating system from the **Operating System** drop-down list box.



Depending on the operating system selected, different input fields are provided on the page as shown in the example of BS2000 below:



2 Choose **Edit** ( )

3 Enter the required values.

For information on all fields available, see *Input Fields: Network Defaults*.

4 Choose **Save**( ) when you are finished.

This section covers the following topic:

Input Fields: Network Defaults

The input fields available on the tabbed page **OS Special** depend on the operating system selected as indicated in the following table:

Field	Description
Operating system	Operating system/environment: BS2000, z/OS, z/VSE or UNIX/Windows.
Activation escape	Activation Escape Character = Network default value specific for operating system This escape character is the prefix for Natural code lines and symbols to be replaced at activation time. **Caution:** Existing Dynamic JCL might become invalid after changing this escape character and applying defaults to jobs.
Submission escape	Submission Escape Character = Network default value specific for operating system This escape character is the prefix for Natural code lines and symbols to be replaced at submission time. **Caution:** Existing Dynamic JCL might become invalid after changing this escape character and applying defaults to jobs.
JCL User ID	(Not applicable to z/VSE) The user ID to be used for JCL loading
	BS2000:

Field	Description
	If this field is not empty, the JCL is loaded with the rights of this BS2000 user ID. It can be overwritten with specific definitions. TSOS may only be defined if the user himself has logged on under TSOS. Default: The user ID from the fully qualified file name. If this field is left empty in a job definition, then the **Default User ID** will be inserted during the activation of the job.
	z/OS: JCL in z/OS will be loaded under this user ID by the Entire Operations Monitor. You can define this field only if you are logged on to the JCL node with the same user ID. Default: If this field is left blank, the user ID of the last modification will be used as JCL user ID.
	UNIX and Windows: With the authorization of this user ID, the Entire Operations Monitor loads the JCL of type TXT.
	See also the default setting **User ID Definition** (*Defaults for Other Settings*, *Administration* documentation), ***Operating System User IDs*** and ***Default User ID Determination***.
Submit User ID	(Not applicable to z/VSE) The user ID to be used at job start
	BS2000: The Entire Operations Monitor starts jobs in BS2000 under this user ID. In the network definition, this is a default value for the jobs. If this field is left empty in a job definition, then the **Default User ID** is inserted during the activation of the job.
	z/OS: The Entire Operations Monitor starts jobs in z/OS under this user ID. You can only define this user ID if you have logged on to the executing node with the same user ID. Default: If this field is empty, the user ID of the last modification is taken as submit user ID.
	UNIX and Windows: With the authorization of this user ID, the Entire Operations Monitor starts the script or the executable program.
	See also the default setting **User ID Definition** (*Defaults for Other Settings*, *Administration* documentation), ***Operating System User IDs*** and ***Default User ID Determination***.
Applies to BS2000 only:	

Field	Description
Default User ID	This user ID is a default for all objects of this job network or job which are linked to a user ID.
Job Class	This job class is a default for all jobs in the network. It can be overridden by specific definitions.
Account Number	This account number is a default for the **Submit User ID** defined for the job network. It can be overridden by specific definitions.
Job Priority	If not empty, this job priority will be used during submission and will override a possible setting in the LOGON statement. The default value on network level will be used for new job definitions.
Run Priority	If not empty, this run priority will be used during submission and will override a possible setting in the LOGON statement. The default value on network level will be used for new job definitions.
SYSOUT User ID	This is the user ID under which internal SYSOUT files are created by Entire Operations. If you do not enter an ID here, the **Submit User ID** is used. See also the default setting **User ID Definition** (*Defaults for Other Settings*, *Administration* documentation), ***Operating System User IDs*** and ***Default User ID Determination***.
SYSOUT Cat ID	This is the catalog ID under which internal SYSOUT files are created by Entire Operations. This field is meaningful only if a SYSOUT user ID different from the submit user ID is used.
Applies to UNIX and Windows only:	
JCL Group	Applies to UNIX only (optional) If this field is left blank, the standard group of the UNIX user ID is used as defined under `/etc/passwd`. Otherwise, this field must contain one of the groups issued by the UNIX command groups.
Submit Group	Applies to UNIX only (optional) If this field is left blank, the standard group of the UNIX user ID is used as defined under `/etc/passwd`. Otherwise, this field must contain one of the groups issued by the UNIX command groups.

Specifying Recipients for Network Messages

You can define one or more users (recipients) to receive network-related standard messages that are sent when a specified event occurs. This is especially useful for sending information about the abnormal end of a job.

Among the events, which can cause a message to be sent, are:

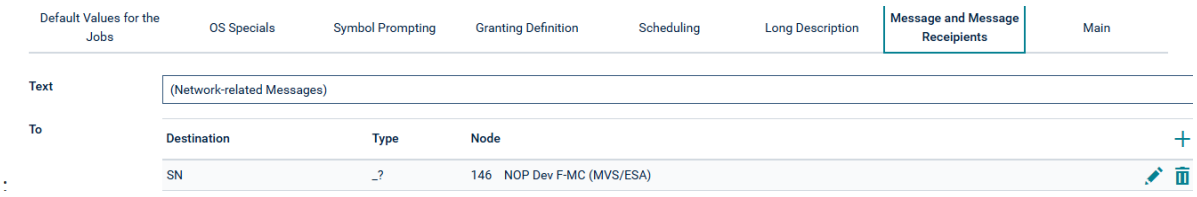
- a calendar not defined for next year;

- a network not correctly terminated;
- symbol prompting requests.

The specific message text is generated automatically by the Entire Operations Monitor.

To add, modify or remove recipients for network-related messages

1 In a **Network Master window**, open **Message and Message Recipients** .



2 Choose **Edit** ( )
The save symbol ( ) is shown in the title bar.

3 Under **To**, choose:

Add (+)	to add a new recipient. Specify a **Destination** and choose a **Type** and a **Node**
Edit ( )	to modify a recipient. Choose a **Type** and a **Node**
Delete ( )	to remove a recipient. Confirm the deletion.

4 Choose **Save**( )

Note: You cannot change the text of network-related standard messages.

Granting Definition: Authorizing Other Users or Owners to Access a Network

Entire Operations allows you to authorize other users and owners to access a network that belongs to you.

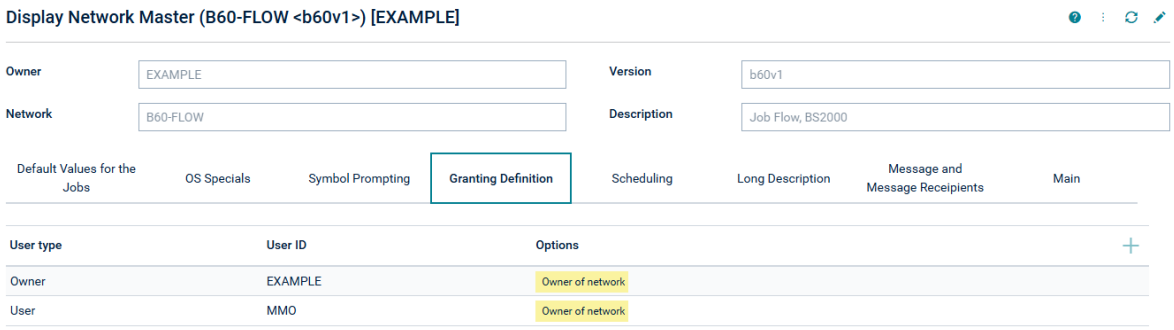
Users grouped under the owner SYSDBA have unlimited access to all networks in the system. They do not need the authorization described in this section.

Users and linked owners (lower than administrator) have limited access rights to granted networks in accordance with the rights specified in their user profiles and the restrictions specified in their individual network granting definition.

The **Owner via Granting** metanode lists all owners who granted network access for your user ID: see *Listing Granted Owners* in the section *Using Owners*.

To add, delete or modify a granting definition

1 In a **Network Master window**, open **Granting Definition**:



2 Choose **Edit** (✎)
The save symbol (💾) is shown in the title bar.

3 Choose:

Add (+)	to add a new definition.
Edit (✎)	to modify a definition.
Delete (🗑)	to remove a recipient. Confirm the deletion.

4 Specify the rights for the previously selected or added user. The fields and options available are described in *Fields: Granting Definition*.

5 Choose **Save**(💾)

The users and owners you can remove depend on your user authorization. For the delete restrictions that apply, see *Adding and Removing an Owner*.

Fields: Granting Definition

The columns on the **Granting Definition page** and the corresponding fields and options in the **Granting Definition window** are described in the following table:

Column/Field	Description	
User Type	Specifies the users to be authorized for network access. Possible options:	
	Owner	All users linked to the owner specified in the **User ID** field.
	User	A defined user.

Column/Field	Description	
User ID	User or owner to be granted network access. If the access rights are given to an owner, all linked users can obtain these access rights. You can select a user from the drop-down list box.	
Options	Authorization level for network access. You can select one or more of the following authorization levels:	
	Read only	Read authorization (maintenance not allowed).
	Write capability	Read and write authorization (maintenance allowed except delete network).
	Delete capability	Read, write and delete authorization for the user or owner entered in the **Name** field.
	Owner of network	Read, write and delete authorization. Authorization is also granted to all users and owners possibly authorized by the specified owner.
	Activate network	Activate network authorization.

Long Description - Documenting Your Networks

You can add a **brief description** of a job network when defining a network in the **Network Master** window. This short description appears in the list of networks in the **Network Master List** window.

If you wish to add more online documentation for a network, you can invoke the Entire Operations editor.

As an alternative to a descriptive text, you can also enter an URL address that be used as a hyperlink to open another page inside or outside your Windows application.

This hyperlink also works when you display the long description of a network.

To create or modify a long description or hyperlink (URL) reference

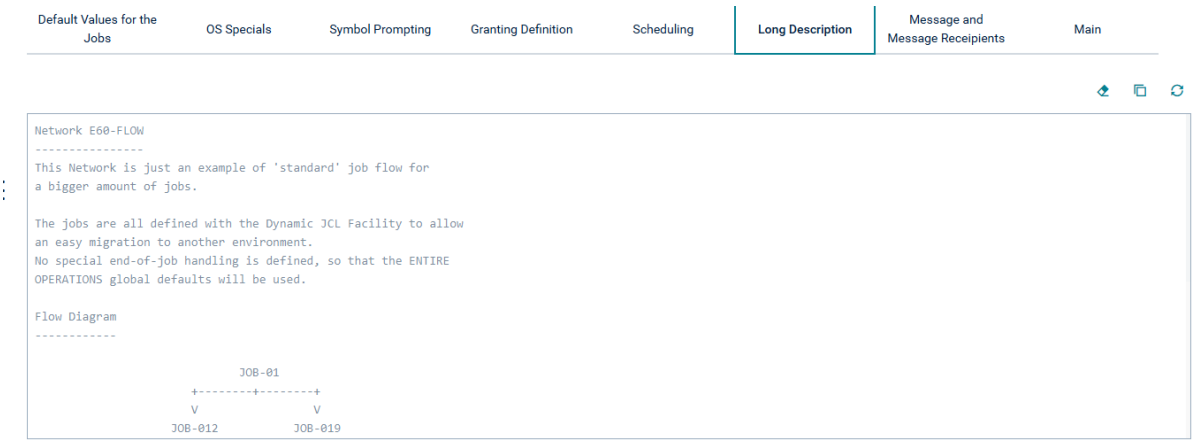
1 In a **Network Master window**, open **Long Description**.

2 Choose **Edit** ( )
The save symbol ( ) is shown in the title bar.

3 Enter an explanatory text or a hyperlink by choosing either of the following options.

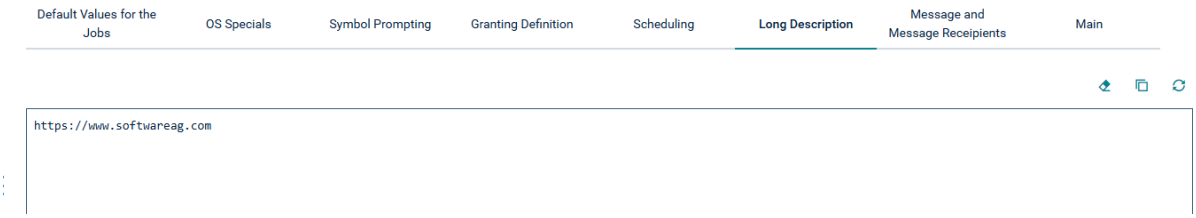
(You can choose **Clear** if you want to remove all entries on the **Long Description** page.)

Enter an explanation concerning the network master or replace existing text as shown below:



Or:

Enter a URL address that starts with `http://` or `https://` as shown below:



The URL must not contain any blank characters but it can extend over several lines. The lines are then concatenated without trailing blanks.

You can use one URL only.

4 When you are finished, choose **Save**( ).

Deleting a Job Network

Only authorized users can delete a network. Only networks without active runs and version usage can be deleted.

Deleting a network also deletes all definitions made for the network on lower levels using network and job maintenance functions.

Entire Operations saves the last run number of a deleted network. The first run of a new network under the same name receives the last run number incremented by 1.

To delete a network

1 In the **object workspace**, select **Owner** > *owner name* > **Network Master** and then a network definition.

2 Invoke the context menu and choose the **Delete** ( ) function.

A dialog prompts you to confirm the delete request.



3 Choose **Yes**.

If the delete is rejected, you receive an appropriate error message on the reason.

Note: If active objects related to this network are still existing, the network deletion will be rejected with an error message. In this case, please refer to the Entire Operations log. The first object will be logged, which prevents the network deletion.

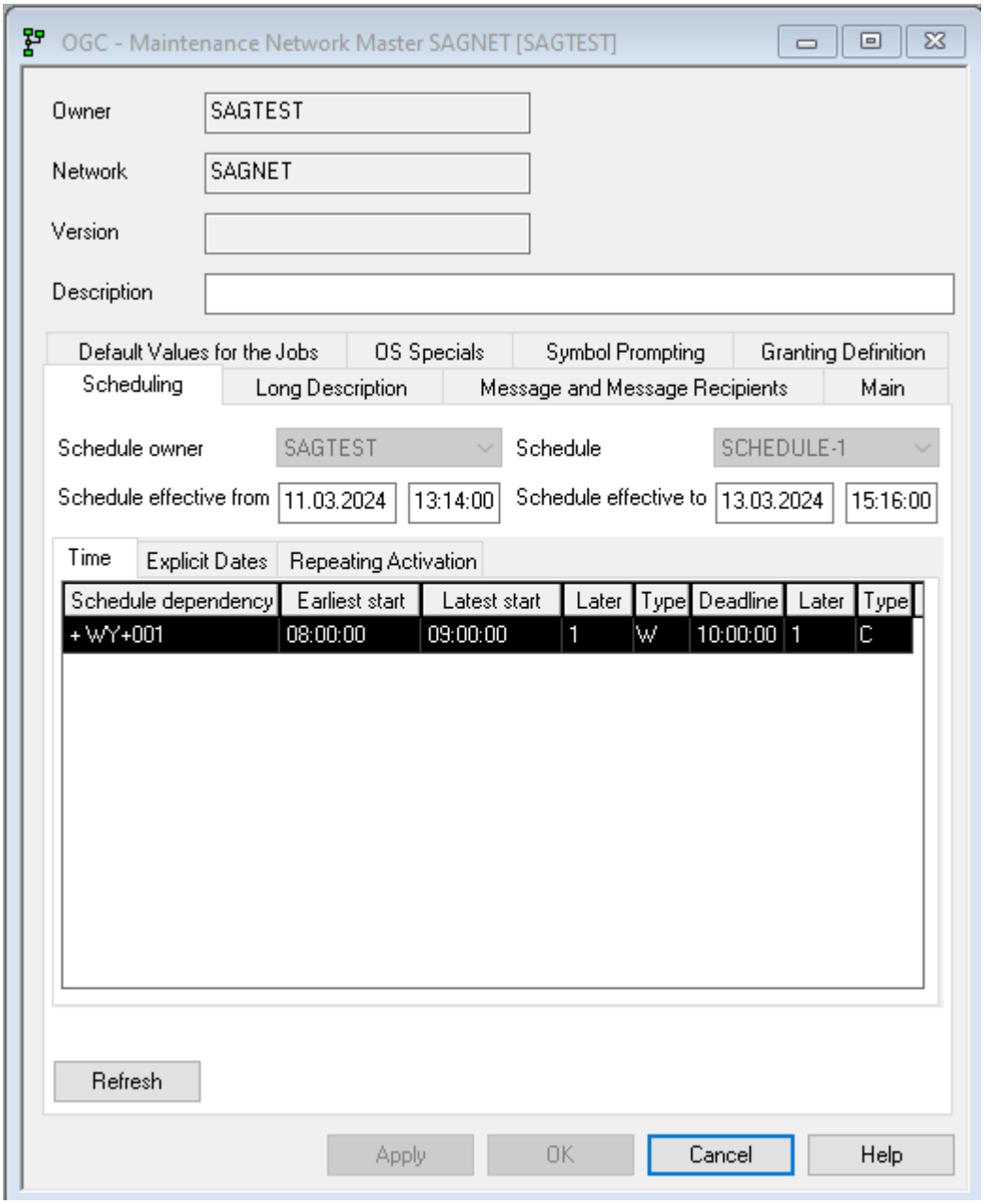
17 Scheduling a Network

- Viewing a Network Schedule Definition 132
- Defining a Network Schedule 133
- Processing of Time Frame Definitions 137
- Defining and Deleting Time Frames for a Network 139
- Defining Dates for Explicit Network Execution 141
- Defining Multiple Network Activations 145
- Viewing a Network Schedule Definition as a Calendar 148
- Producing a Network Start Summary 150
- Displaying Next Network Starts - Next Activations 150
- Displaying the Network Execution History 155
- Deleting a Scheduling Definition for a Single Network 156

Viewing a Network Schedule Definition

To display the schedule defined for a network

- In a **Network Master window**, open the tabbed page **Scheduling**:



The fields and tabs available in the window are for information only. You cannot change any entries. Changes are performed by using the **Define Scheduling** context menu function described in the following section.

The fields and tabs are explained in *Fields and Columns: Network Scheduling*.

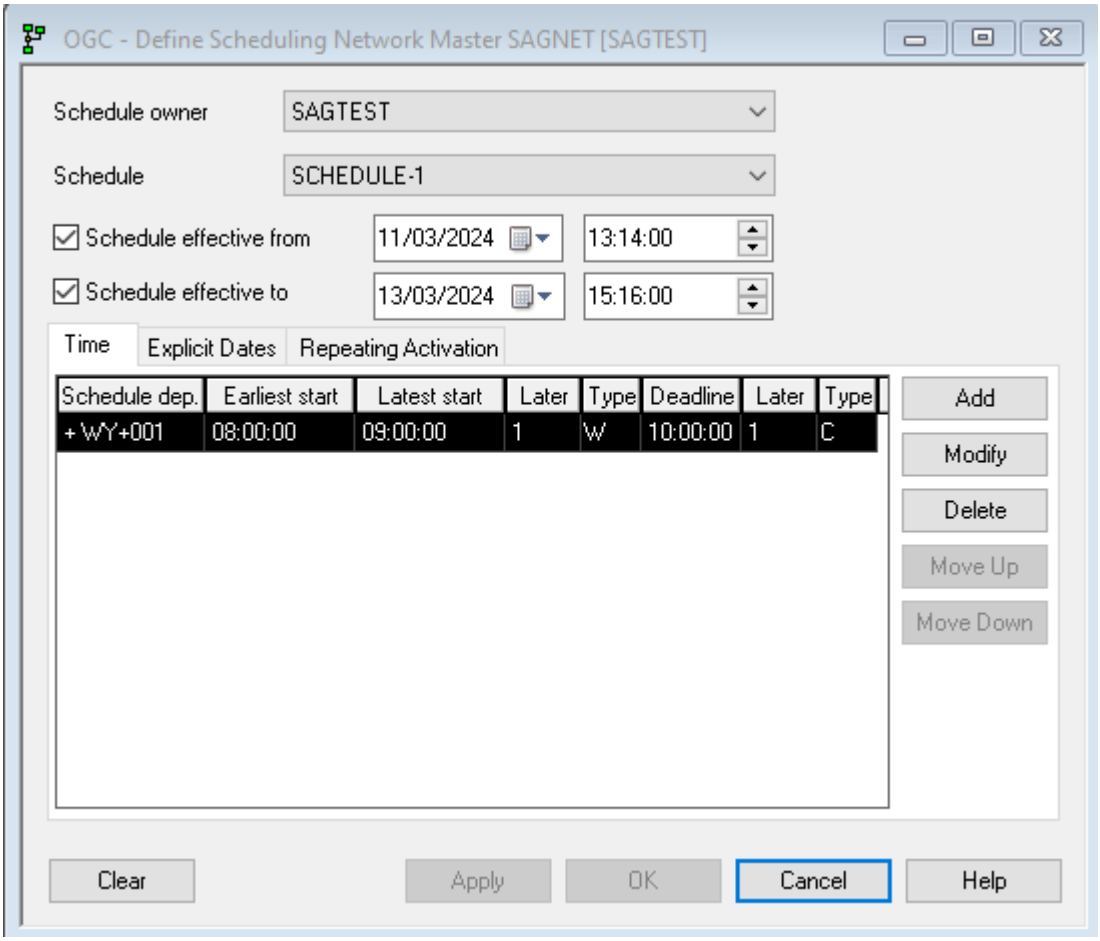
You can choose **Refresh** to view the latest definition of the schedule after most recent changes performed with the **Define Scheduling** function.

Defining a Network Schedule

To define a schedule for a network

1 In the object workspace, select a **Network Master** instance and choose **Define Scheduling** from the context menu.

A **Define Scheduling** window like the example below opens:



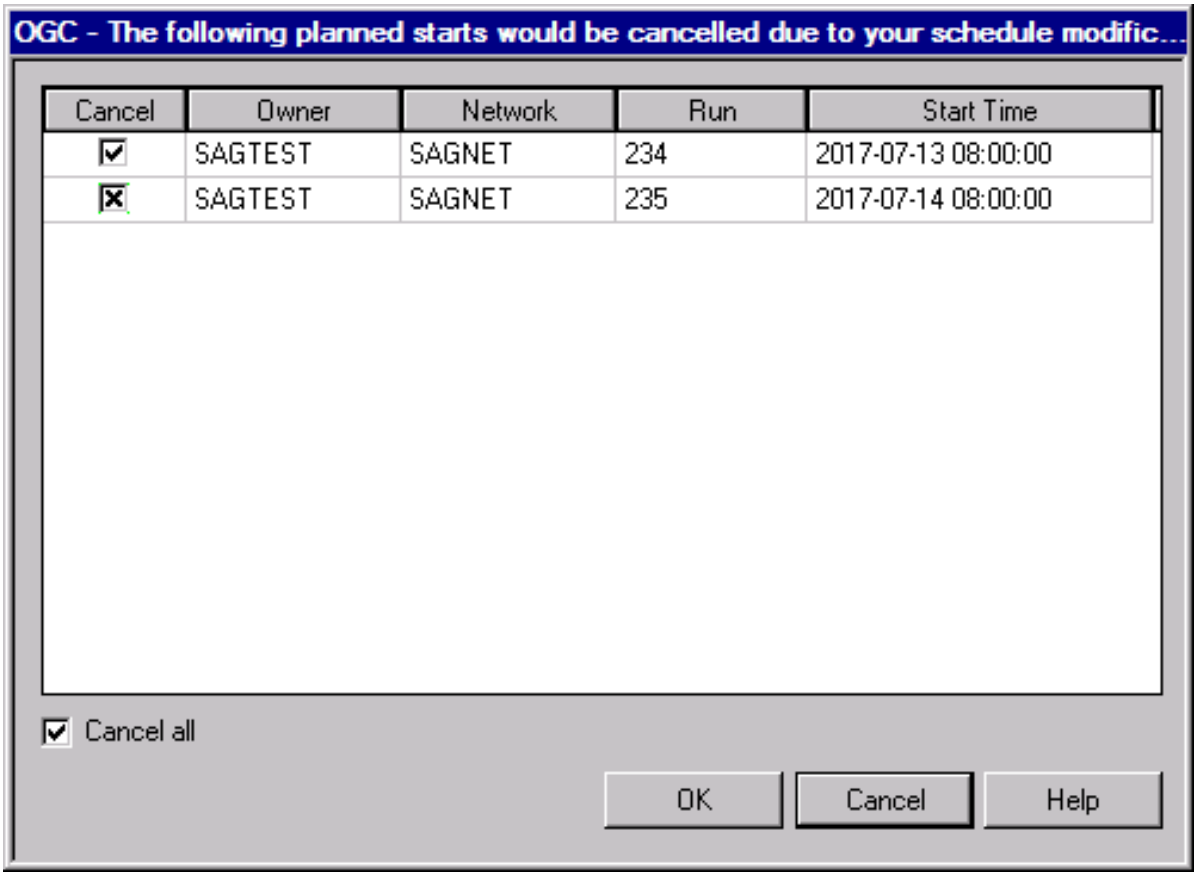
2 Enter the required values or replace existing entries. The fields and tabs available in the window are explained in *Fields and Columns: Network Scheduling*.

You can choose **Clear** if you want to remove all current definitions.

3 Choose **OK** when you are finished.

4 If you modify a network schedule and this modification affects any planned starts, you have the option to cancel the starts or to keep them active.

If you select **Cancel**, a dialog like the example below appears:



You can select the following functions:

1. Select the **Cancel all** check box to cancel all starts.
2. In the table, select the **Cancel** check box next to starts you want to cancel .
3. Select the **Cancel** button if you do not want to cancel any starts. This keeps all listed starts active, regardless of your schedule modifications.

This section covers the following topics:

- Fields and Columns: Network Scheduling

Fields and Columns: Network Scheduling

The input fields and corresponding columns (if relevant) on the tabbed pages of the **Define Scheduling window** are described in the following table:

Field | Column	Description
Schedule Owner	Owner of the schedule to be used. Default: Owner of the network
Schedule	Name of the schedule to be used. Schedule comprises a link to a calendar. Definition of dates on which the network is to be executed.
Schedule effective from	Schedule effective from (date and time) Select this field (not selected by default) to activate the schedule and/or change the date and time. If a schedule should not become effective immediately, enter the date (and time) of the validity begin here. If no time is specified, the schedule becomes effective at 00:00.
Schedule effective to	Schedule effective until (date and time) Select this field (not selected by default) to activate the schedule and/or change the date and time. If a schedule should be used only until a defined date, enter the date of the validity end here. If this field is empty, the schedule will be used infinitely many times in the future. (Default) If no time is specified, the schedule remains valid until 23:59.
Time page:	
Schedule dependency | Schedule dep.	This option is used to specify days when the network is deactivated and not executed. See *Adding and Modifying a Time Frame Definition*.
Earliest Start	Earliest start time for the network. No job in the network can start before this time. The **Earliest Start** time is used to compute network schedule times. See also *Processing of Time Frame Definitions*.

Field | Column	Description	
Latest Start	Latest start time for the network. The first job in the network must start by this time, if possible. If not, a warning message is sentl, and the job is not submitted. You can specify the number of workdays or calendar days to be used for later starts: see **Calendar days later | Workdays later** below. See also *Processing of Time Frame Definitions*.	
Deadline	Time by which the last job in the network must be finished. The deadline is used to compute the earliest and latest start times for the jobs in the network. See also *Processing of Time Frame Definitions*.	
Calendar days later | Later **Workdays later | Later**	This field is to be used if **Latest Start** and/or **Deadline** are later than 24 hours after the **Earliest Start**. Valid values: 1 - 999 days. Default for **Deadline**: 2 calendar days later. See also *Processing of Time Frame Definitions*. **Note:** The Entire Operation default settings provide the option to define later start times in units of hours instead of integer days, for time ranges that are not a multiple of 24.	
Calendar days later | Type **Workdays later | Type**	Specifies the day type:	
	Column entry: C (or empty)	Calendar day (default).
	Column entry: W	Workday as defined in the calendar linked to the schedule.
Explicit Dates page:	See *Defining Explicit Dates for a Single Network only*.	
Repeating Activation page:	See *Defining Multiple Network Activations*.	
Number of Activations	This field is used to activate a network more than once a day. In this case, select the button next to this field and enter the number of activations per day. If the network is to be activated once per schedule day only, leave this field blank. Valid input values: 2 to 999	
	every ... minutes: If a network is to be activated more than once per day, you can enter the time interval between two subsequent network activations.	

Field | Column	Description
	This field can be used without **Number of Activations**, too. In this case, the number of activations will be limited by the defined latest network start time. Unit: minutes See also *Defining Multiple Network Activations* for further information.
Activate at	In these fields, you can enter up to 10 activation times, as an alternative to **Number of Activations**. Format: HH:MM or H:MM Examples: `20:11`, `1:31` **Note:** If activation times are defined for a network, the job-specific time frames will be adapted to fit into the network time frame. See also *Defining Multiple Network Activations* for further information.
Day Deadline	If a **Day Deadline** is defined, the **Deadline** of multiple activations (**every ... minutes** or **Activate at**) is limited to this time on the day of the activation. If multiple days are needed for a perodic activation, the activations will be continued at the defined **Earliest Start** on the next day. Format: HH:MM

Processing of Time Frame Definitions

This section describes the rules and processes that apply when specifying time frames for a network or job schedule on the **Time page** of a **Define Scheduling Network** window or the **Scheduling Parameters** page of a job master or job active:

- Hierarchical Order
- Changes to Time Frames of Active Jobs
- Time Frame Selection Rules for a Network

- Time Frame Rules for Start/End Times and Day Later Definitions

Hierarchical Order

Time frames defined on the job level override time frame definitions made on the network level for this job. If no time frames are defined here, the job network defaults apply.

Changes to Time Frames of Active Jobs

All modifications to time frames of an active job affect the active database and the current job run only. Time frame definitions for the job master remain unchanged.

Time Frame Selection Rules for a Network

Time frames are selected according to the following rules:

- A table row with no time frame dependency entry is applied as the network start date if no other time frame dependency entry matches the network start date.
- From top to bottom, each table row with a defined schedule dependency condition is applied to the network. If a condition is satisfied, the respective time frame values are used for the active network.

  Example: The definition `+CW+3` means this definition applies if the network starts on Wednesday. `-CW+3` means this definition applies if the network starts on all other weekdays except Wednesday.
- If none of the defined table rows is selected, the network uses the default time settings. See also *Defaults for Time Ranges* in the *Administration* documentation.

Time Frame Rules for Start/End Times and Day Later Definitions

- If both **Latest Start** or **Deadline** and **Days later** are not specified, the Entire Operation defaults specified for **Default Latest Start after Earliest Start** and **Default Deadline after Earliest Start** on the **Time Ranges** page apply (see the *Administration* documentation).
- If **Latest Start** or **Deadline** is given and **Days later** is not specified, **Days later** is automatically filled as described above.
- If applying any of the above rules and **Days later** of **Latest Start** or **Deadline** becomes greater than 999, it remains empty and an error message is issued.
- If **Latest Start** is before **Earliest Start** and **Days later** is not specified, the **Days later** value for **Latest Start** is set to 1 by default.
- If **Deadline** is before **Earliest Start** and **Days later** is not specified, **Days later** is set to `1` so that this time follows the earliest start time.

Defining and Deleting Time Frames for a Network

- Adding and Modifying a Time Frame Definition
- Deleting Time Frame Definitions for a Network

Adding and Modifying a Time Frame Definition

To add or modify a time frame definition for a network

1 In the **Define Scheduling window**, open the **Time page**.

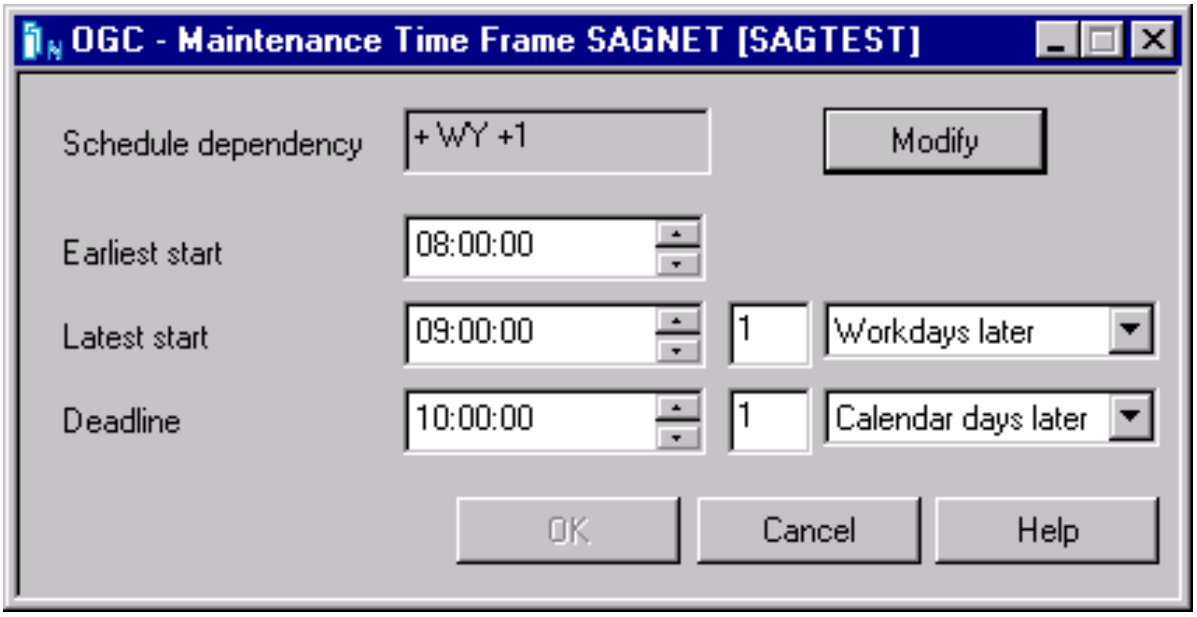
The table columns are explained in *Fields and Columns: Network Scheduling*.

2 Choose **Add** to add a new time frame definition.

Or:

Select a time frame definition from the table and choose **Modify**.

Depending on the function chosen, a **Create new Time Frame** or **Maintenance Time Frame** window like the example below opens:



3 Enter the required values. The input fields contained in the window are described in *Fields and Columns: Network Scheduling*.

See also *Time Frame Selection Rules for a Network* and *Processing of Time Frame Definitions*.

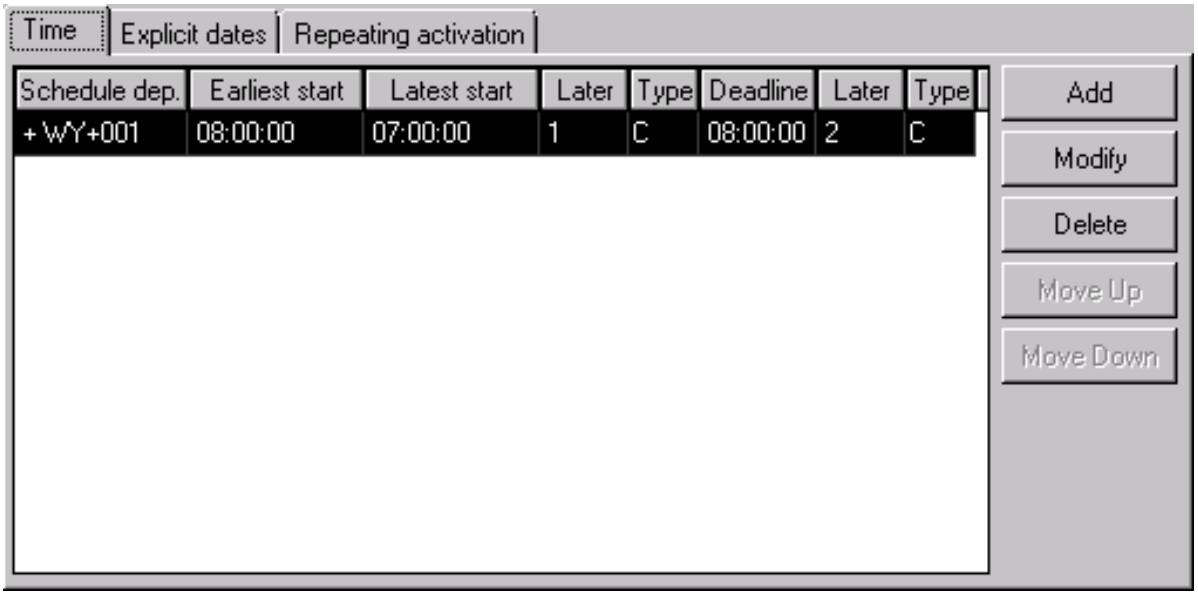
4 If you want to add or modify the schedule dependency for a time frame, choose **Modify** next to **Schedule dependency**.

A **Schedule Dependency Definition** window like the window provided for a job opens.

The input fields contained in the window are described in *Fields: Schedule Dependency Definition*.

5 Choose **OK** when you are finished.

The time frame definition is shown in the table of the **Time** page as shown in the example below:



If a schedule dependency is defined for the time frame definition, a code is shown in the **Schedule dep.** column (here: `+WY+001`).

6 You can move up or down a selected column row by choosing **Move up** or **Move down**, respectively.

You can choose **Delete** for a selected column row to remove a time frame definition.

Choose **Clear** to remove all entries from the table.

7 Choose **OK** when you are finished.

Deleting Time Frame Definitions for a Network

To delete a time frame dependency from a time frame definition

1 Open the tabbed page **Time** of the tabbed page **Scheduling**.

2 Select the table row that contains the definition you want to delete and choose **Delete**.

> **Note:** **Clear** removes all **Scheduling** entries as described in *Deleting a Schedule Definition*.

3 Choose **Apply** or **OK** to confirm the deletion and close the window.

The time frame definition is removed from the table.

Defining Dates for Explicit Network Execution

You can define explicit dates on which a network always or never executes, regardless of any other schedule definitions. Explicit dates supersede all other schedule definitions.

You can define explicit dates for all networks linked to a schedule or for a single network only.

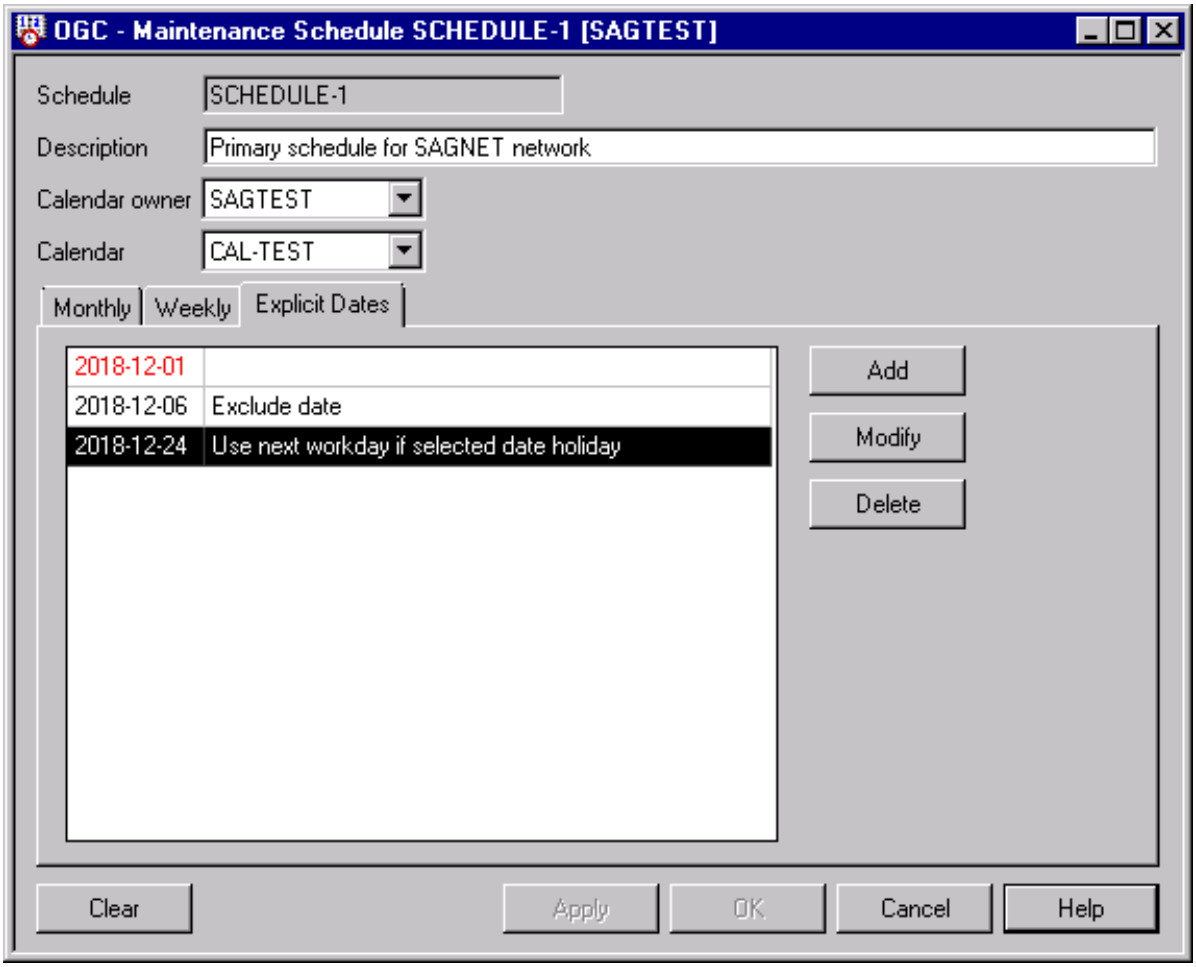
This section covers the following topics:

- Defining Explicit Dates for a Schedule
- Defining Explicit Dates for a Single Network only

Defining Explicit Dates for a Schedule

To add or modify explicit dates for a schedule

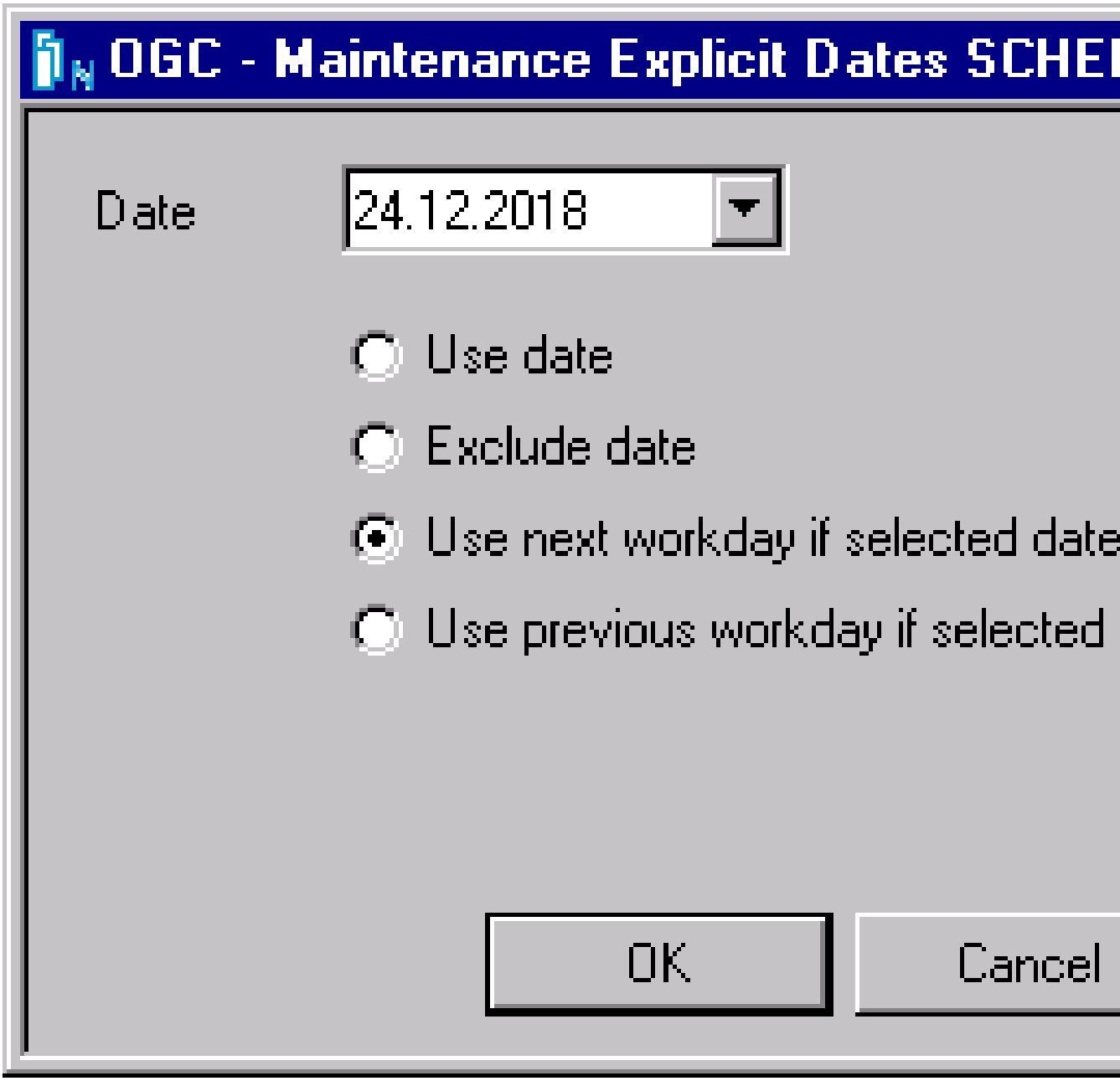
1 In the **Create new Schedule** or **Maintenance Schedule window**, open the tabbed page **Explicit Dates** shown in the following example:



2 Choose **Add** for a new definition or select an existing entry and choose **Modify**.

You can choose **Clear** to remove all table entries or **Delete** to remove a selected table entry only.

When adding or modifying a definition, an **Explicit Dates** window like the example below opens:



3 Open the drop-down list box and select the required date or replace the date entered in the date format specified within Entire Operations defaults (here: DD.MM.YYYY).

If you want to associate the date with a special condition, select one of the following options:

- Use date: Execute a network on the specified date (default).
- Exclude date: Do not execute a network on the specified date.
- Use the next workday if the specified date is a holiday: Execute a network on the workday that follows the specified date.

- Use the previous workday if the specified date is a holiday: Do not execute a network on the specified date.

4 Choose **OK** when you are finished.

The **Explicit Dates** window closes and the specified dates are shown on the **Explicit Dates** page.

A date highlighted in red denotes that the date entered is past the date currently set as default in the **Explicit Dates** window.

On the **Explicit Dates page** above, the date `2018-12-01`, was entered in the **Explicit Dates** window on December 3, 2018, for example.

5 Choose **OK** to save the date specification.

Defining Explicit Dates for a Single Network only

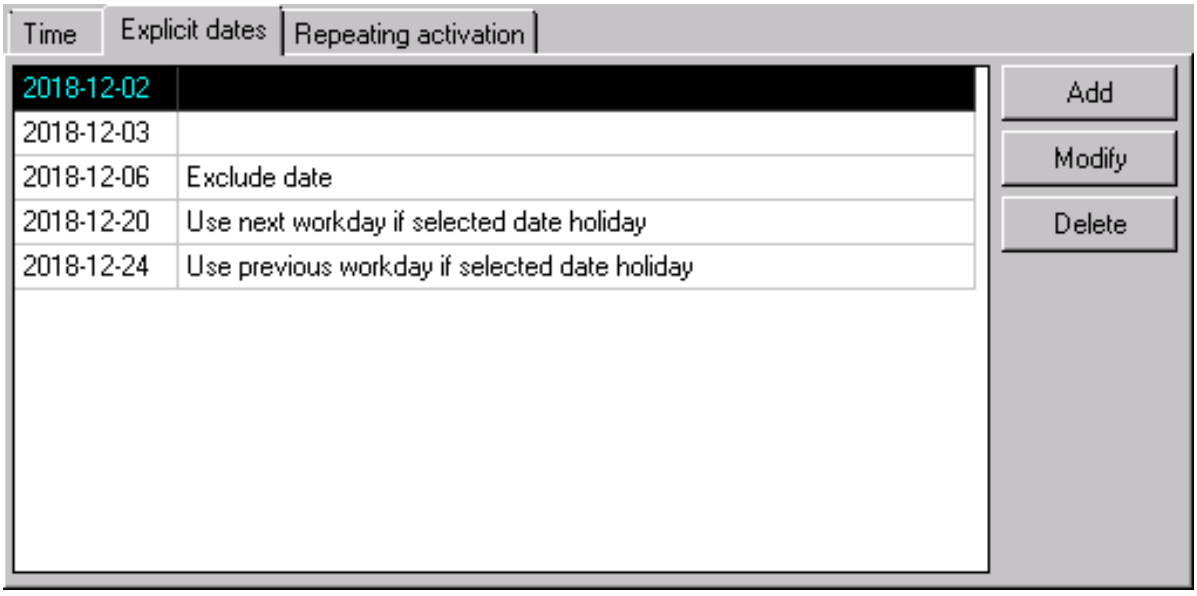
You can define explicit schedule dates on the network level. Dates defined here are network-specific in contrast to schedule definitions which affect all linked networks.

Notes:

1. Any modification made with this function causes the recalculation of the current schedule of the network.
2. Explicit dates override all definitions made in schedule(s) linked to the network.
3. Deletions of planned activations can cause that the day of deletion is added to the dates to be excluded on the **Explicit Dates** page. This is to avoid that the network is again activated on this day. (Explicit exclusion dates on the network level are set implicitly by the network activation cancelling function.)

To define explicit dates for a single network

1 In the **Define Scheduling window**, open the **Explicit Dates** page shown in the following example:



2 Proceed with **Step 2** of *Defining Explicit Dates for a Schedule*.

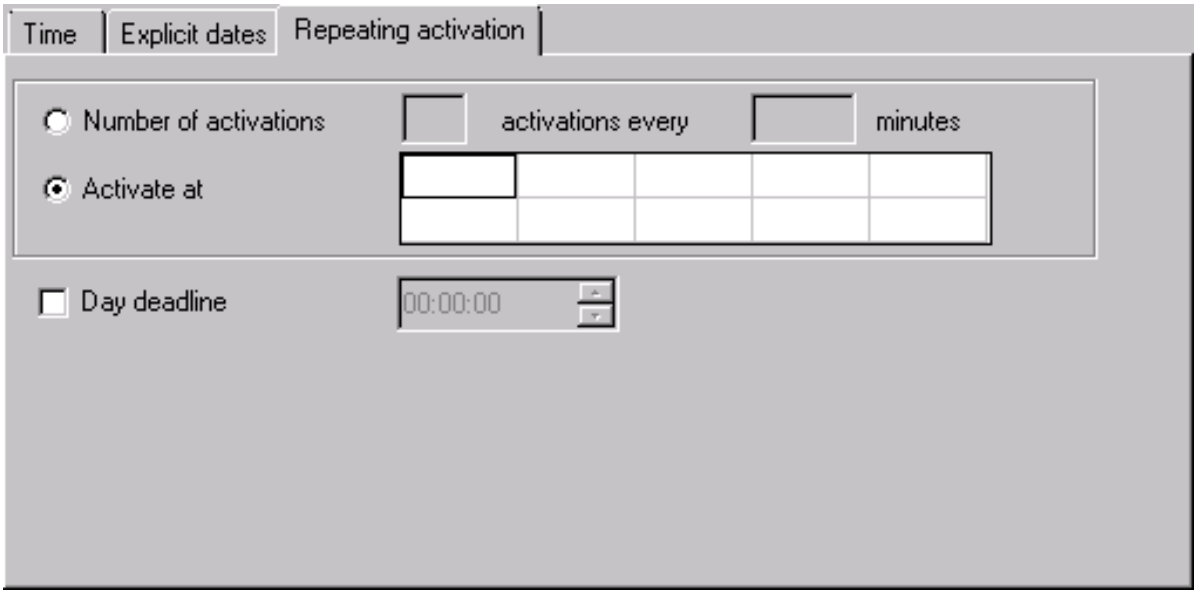
Defining Multiple Network Activations

You can repeat network activations multiple times.

Note: If activation times are defined for a network, the job-specific time frames will be adapted to fit into the network time frame.

To define multiple network activations

1 In the **Define Scheduling window**, open the **Repeating Activation** page shown in the following example:



2 Enter (or replace) either or both of the following:

- In the **Number of activations** field, enter the required number of activations.
- In the **every ... minutes** field, enter the time interval to be used for activation.

Or:

Enter (or replace) the following:

- In the input fields next to **Activate at**, enter single or multiple times to be used for network activation.

You can choose **Clear** to remove all entries from the **Repeating activation** page.

3 If required, select or enter a time limit for activation in the **Day Deadline**.

4 Choose **OK** to save your entries and close the window.

The rules that apply when defining multiple activations are described in the following section:

- Activation End

- Calculation of Time Frames for Multiple Activations

Activation End

Multiple activations are limited by

- until the end of the schedule day,
- until the latest start,
- by the defined number of activations,

whichever applies first.

Calculation of Time Frames for Multiple Activations

The following applies if both **Number of Activations** (`na`) and **every** ... **minutes** (`em`) have a non-zero value:

- The latest start time of a time frame (`tf_latest_start`) is interpreted as the latest start time of the first multiple activation.
- For activation `1 <= n <= na`, the latest start time of the specific activation will be set to `tf_latest_start + (n - 1) * em`.
- The multiple activation end time (`mae`) will be calculated as `mae = schedule_date + tf_latest_start + (na * em)`

If the multiple activation end time (`mae`) is after the defined deadline of a time frame, it will be set to the deadline of the time frame.

If the multiple activation end time (`mae`) is after the end of the schedule day, it will be set to the end of the schedule day.



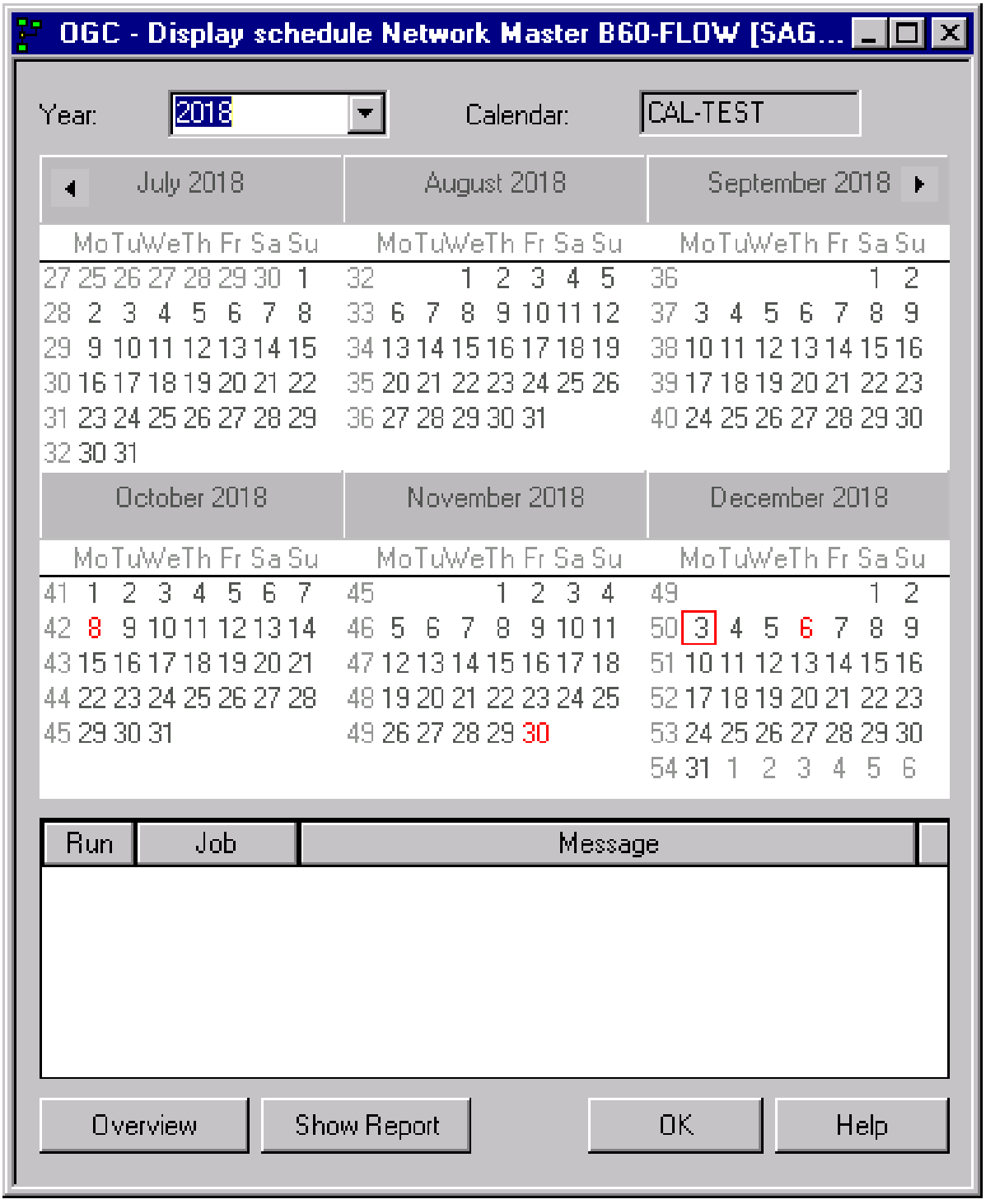
Note: A network is no longer treated as “already activated on the current day” if there was a multiple activation on the day before, which resulted in some runs that were executed after midnight. In this case, automatic activations (due to schedule) will not be blocked by the “activation on current day” check.

Viewing a Network Schedule Definition as a Calendar

To display the calendar view of a network schedule

1 In the object workspace, select a **Network Master** instance and choose **Display Schedule** from the context menu.

A **Display Schedule** window like the example below opens:



2 Select the year you want to display.

Dates outlined in red indicate that a schedule is provided.

3 Select a date and choose **Overview**.

All activities scheduled for this date are displayed for online viewing in the lower area of the window.

4 Select a date and choose **Show Report**.

A list of available reports is displayed in a separate window.

Select the required report for printing purposes.

Note: For further information on reports, see the section *Reporting*.

Producing a Network Start Summary

This section describes how to produce a network start summary for a single network. The report looks like the **example of a Network Start Summary** shown in the *Reporting* section.

If you want to produce a **Network Start Summary** for all network activations on a given day, see the corresponding report described in the section *Reporting*.

To produce a network start summary for the current network

1 In the **Display schedule window**, select a valid date and choose **Overview**.

All activities scheduled for this date are displayed for online viewing in the lower area of the window.

2 Select a date and choose **Show Report**.

A list of available reports is displayed in a separate **Reporting window**.

3 Select the required report, choose **Save as File** from the context menu and determine the file to be created from the report: see *Report Output Options*.

For detailed information on generating reports, see the section *Reporting*.

Displaying Next Network Starts - Next Activations

Scheduled starts are extracted from the network schedules and put into a pre-activation status. The extraction is usually performed at midnight, one day in advance.

Manual starts are explicitly invoked as manual activations by any user. If a special start time was not entered, the daily time frame is used. See *Activating a Job Network Manually* in the section *Network Maintenance*.

This section covers the following topics:

- Viewing and Maintaining Next Network Activations
- Columns: Next Activations

Viewing and Maintaining Next Network Activations

The context menu function **Next Activations** invokes the display of the future network activations. These activations are, in general, maintained by a schedule or a calendar but can also be invoked manually to view, modify or delete planned activations according to your selection. For example, by choosing the function **Next Activations** in the menu **General** you can globally display and manage the planned network activations.

To view and modify the next network activations

1 In the **Title Bar**, choose **General**.

Or:

In the **object workspace**, choose **Owner** ( ) and select an *owner instance*.

Or:

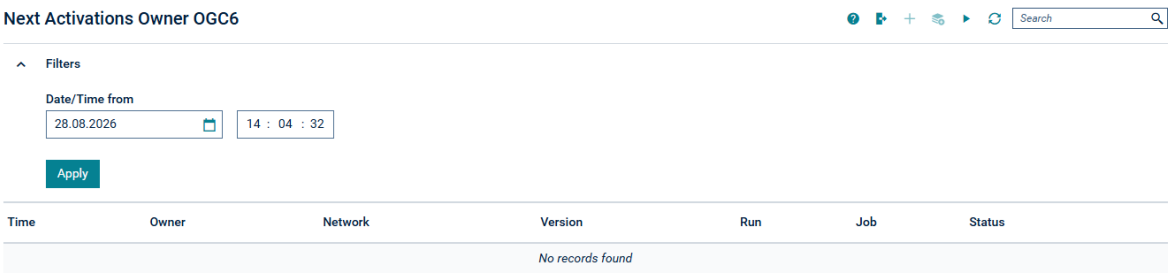
In the **object workspace**, choose **Owner** ( ) > *owner instance* > **Network Master** and select a *Network Master instance* ( ).

2 Open the context menu and choose **Next Activations**.

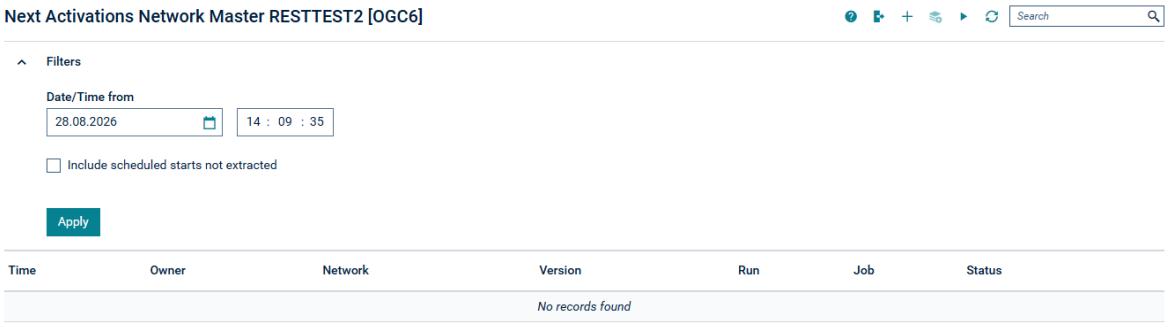
For a **General** node, the **Next Activations** window lists all network activations planned for all networks in your current Entire Operations environment as shown in the following example:



For an **Owner** instance, the **Next Activations** window lists all network activations planned for all networks that belong to the selected owner. Example:



For a **Network Master** instance, the **Next Activations** lists all activations planned for the selected network master. Example:



The **Next Activations** window displays a chronological list of all planned network or job starts, both scheduled and manual.

The columns contained in the dialog are described in *Columns: Next Activations*.

3 You can refresh the list by choosing **Refresh** ( ) and **Auto Refresh** ( ) (see also *Refreshing Object Lists*).

4 You can cancel a future activation by selecting the respective row and choosing **Deactivate** ( ).

You can only cancel a job network to which a run number is assigned.

5 You can prompt symbols (if defined) by selecting the respective row and choosing **Prompt** ( ).

This button is only active for activation with type `Aw. Symbol Prompting` and if a run number is assigned to the network.

6 You can edit a future activation by selecting the respective row and choosing **Modify** ( ).

A window like the example below opens:

This window allows you to define a new start date and time for execution, edit a symbol table if used for this activation (see also *Modifying a Symbol Table*), and select the option to **keep predefined job time frames**.

7 If you want to export the current **Next Activation** settings, choose **Export** ( ) .

An **Export Objects** window opens. Proceed as described in *Exporting Objects* in the *Import/Export Functions* documentation.

8 If you want to add a next activation, choose **Add** ( ) and continue by *Defining the time schedule for activation* using the **Activation Network Master** dialog.

Note: Next activations can only be added for a network master because the objects have to be specified in advance of any activation.

To add a next activation

Next activations can only be added for a network master because the objects have to be specified in advance of any activation.

1 In the **object workspace**, choose **Owner** ( ) > *owner instance* > **Network Master** and select a *Network Master instance* ( ).

The **Next Activations** dialog opens.

2 Choose **Add** ( ).

The **Activation Network Master** dialog opens.

3 Define the time schedule for activation.

4 Choose **Activate**.

The network is now marked and scheduled for its next activation and appears in the activations list.

To keep predefined job time frames

- If you select the check box **Keep predefined Job Time Frames**, Jobs with master time frame definitions will not be adapted.

  Or:

  If you do not select the check box **Keep predefined Job Time Frames**, all job time frames will be adapted (default setting).

Columns: Next Activations

The columns contained in the **Next Activations** window depend on whether you execute the **Next Activations** function from the **General** node or an **Owner** instance (see also *Viewing and Maintaining Next Network Activations*). All columns that can be available are described in the following table:

Column	Description
Owner	Owner of the network to be activated Above this column, you can make an owner preselection. If this field only contains an asterisk (*), all owners will be displayed. Your selection is stored in the user profile and is available when you invoke this function the next time.
Network	Network to be activated Above this column, you can make a network preselection. If this field only contains an asterisk (*), all networks for the current owner selection will be displayed. The network preselection is possible only if an owner was selected uniquely. Your selection is stored in the user profile and is available when you invoke this function the next time.
Run	Run number for this activation.
Version	Version of the network
Job	If the activation is for one job only, the job name appears in this column.
Date	Date of activation You can enter a start date and/or time in the fields above the **Date** and **Start** columns to display only those activations due to start after that time.
Start	Starting time of activation.
Type	Displays the current status of the activation. Possible status types are, for example: ■ `Schedule Table`, ■ `on request`, ■ `Schedule, active`, ■ `Aw. Symbol Prompting`, ■ `Symbol Entry in Progress`.

Displaying the Network Execution History

The execution history is a record of past schedules. It shows the days on which the network was scheduled for execution. The data in a **History** window cannot be modified.

There is history data on past network executions for the current and for the two preceding years (provided the network already existed at that time).

To view the history of a network master

1 In the **object workspace**, select a **Network Master** instance and choose **History**( ) from the context menu.

A **History Network Master** window like the example below opens:



Choose **Data Selection** to display a calender ( ) where the dates on which the network ran are highlighted in red.

2 If available, select a date from the drop-down-list box ( ) to switch the display of the calendar to the selected year.

3 If history data exists for the selected network and date, execution information on all network starts is contained in the right-hand table columns of the window as shown in the example above. The columns are explained in *Columns: Network History*.

If no history data exists, an appropriate message occurs. Log data is only retained for the number of days specified in the **Long-term log** field of the **Entire Operations Defaults** described in *Defaults for Time Ranges* in the *Administration* documentation.

This section covers the following topic:

- Columns: Network History

Columns: Network History

The columns of the **History Network Master** or **History Network Active** window are described in the following table:

Column	Description
Run	Run number of the job network that executed.
Version	Network version (if available) of the network that executed.
Job	Job name if a single job was activated for execution, or `(Network)` if the entire network was activated.
Time	Date and time when the network executed. See also *Date and Time Formats*.

Deleting a Scheduling Definition for a Single Network

To delete a scheduling definition for a single network

1 In the **Define Scheduling window**, open the tabbed page that contains the definition you want to delete.

2 On the **Time** or **Explicit Dates** page, select the table row that contains the definition you want to remove and choose **Delete**. On the **Repeating Activation** page, remove all entries.



Note: Removing the name of a schedule from the **Define Scheduling** window does not delete the corresponding schedule object. See also *Deleting a Schedule*.

Or:

If you want to delete all definitions defined in the schedule, choose **Clear**.

3 Choose **Apply** or **OK** to confirm the deletion.

The schedule definition is deleted for the selected network and the window closes.

Warning:

If you choose **Clear**, the name in the **Schedule** field and all entries on the **Time**, **Explicit Dates** and **Repeating Activation** pages are removed from the schedule. The **Schedule effective from** option is unchecked (deactivated).

18 Maintaining Job Network Versions

- Versioning of Job Networks 158
- Using Network Versions for Activations 159
- Handling Network Version Usage Definitions 161

This section describes how to maintain and use different versions of job network.

Versioning of Job Networks

This section describes the rules and guidelines for maintaining network versions.

- Version Names
- Reserved Version Names for Networks
- Version Names Exit
- Creating Network Versions by Copying Networks
- Copying Jobs
- Deleting Network Versions
- Deleting Network Versions or Single Jobs via API

Version Names

The following applies when creating version names:

- The name can contain up to 10 alphanumeric characters and uppercase or lowercase letters.
- Space characters and the following special characters are not allowed:

  ?<>*,()_
- To avoid problems when porting an Entire Operations environment to another platform, do not use special characters and umlauts.
- Do not use a **reserved version name** (see the relevant section).

Reserved Version Names for Networks

blank value; in selections and in the log also: `(unnamed)`
Is used for an unnamed version.

This is the only network version that exists after migration from an earlier Entire Operations version that does not support network versioning.

In parameter listings (e.g., for reporting) you can use also a hyphen (`-`).

`(current)`
Will be replaced by the version that is set in the activated time schedule.

`(current)` can be used in version references.

Version Names Exit

With the usage of a global version name exit you can force a user-specific version name syntax. For detailed information, see *Global Exit for Version Names* in the *Administration* documentation.

Creating Network Versions by Copying Networks

You can copy network definitions to add a new version for an existing network.

See also ***Copying Job Network Definitions***.

Copying Jobs

Jobs in any version can be copied out of the originating network master.

Deleting Network Versions

You delete a network version by using the network delete function as described in ***Deleting a Job Network***

The following applies when deleting a network version:

- You must be authorized to delete the network.
- You cannot delete a version that is defined and used in a current network activation schedule (see also ***Handling Network Version Usage Definitions***). A version defined in an outdated schedule can be deleted.

Deleting Network Versions or Single Jobs via API

By using the API `NOPUAC5N` (Function `D`, Run Number `-1`) you can delete single network versions and jobs.

Using Network Versions for Activations

This section covers the following topics:

- Evaluation of Network Versions for Scheduled Activations
- Manual Activation
- Activation of a Subnetwork
- Activation as End-of-Job Action
- Activation via API
- Versions without Schedule Activation
- Daily History of Network Activations
- Reporting

- Import/Export
- Exit Functionality (Network)
- Maximum Number of Versions per Network

Evaluation of Network Versions for Scheduled Activations

The following applies:

- If only one version exists in a network, then this version will be activated. A schedule definition will be ignored.
- If multiple versions exist in a network, then it will be checked if a version has a current activated time schedule. If this is so, then this version will be activated.
- If usage intervals are defined for a network version, but the activation date is not in this interval, then the network will not be activated although scheduled. Corresponding protocol log entries and messages will be sent.

Manual Activation

If you choose a manual activation, then any network version can be selected. If defined, the standard version for schedule activation will be offered to you first.

Activation of a Subnetwork

In the subnetwork definition, you can define any version or the reserved `(current)` name.

Activation as End-of-Job Action

For the network or job activation as End-of-Job action you can define any version or the reserved `(current)` name.

Activation via API

For the network or job activation via API `NOPUAC5N` you can define any version or the reserved `(current)` name in the field `NETWORK-VERSION`.

Note that the API may issue version-related return codes.

Versions without Schedule Activation

In Entire Operations you can save multiple versions of job networks. Versions that are not, or are no more in the usage interval of schedule activation will not be activated automatically.

Daily History of Network Activations

The history of the network activations contains the network version for every run.

Reporting

Network versions will be considered.

Import/Export

Network versions will be considered.

Exit Functionality (Network)

Entire Operations provides user exits and APIs that perform network activation functions and support network versioning, for example, the API `NOPUAC5N`.

Maximum Number of Versions per Network

The maximum number of network versions can be limited system wide as described in *Defaults for Network Options* in the *Administration* documentation.

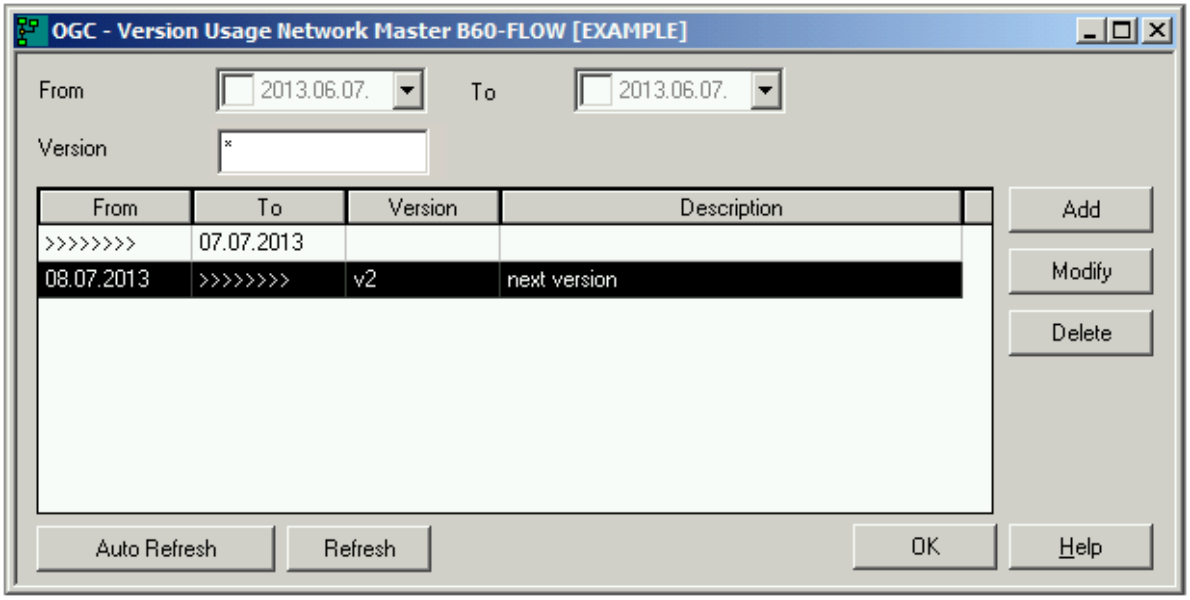
Handling Network Version Usage Definitions

You can to view and define the maintenance of date ranges, in which network versions are to be used for scheduled activations.

To list, add, modify or delete a network version usage definition

1 In the object workspace, select a network from the **Network Master** node and choose **Version Usage** from the context menu.

A **Version Usage Network Master** window like the example below opens:



The window lists date ranges in which network versions are to be used for scheduled activations.

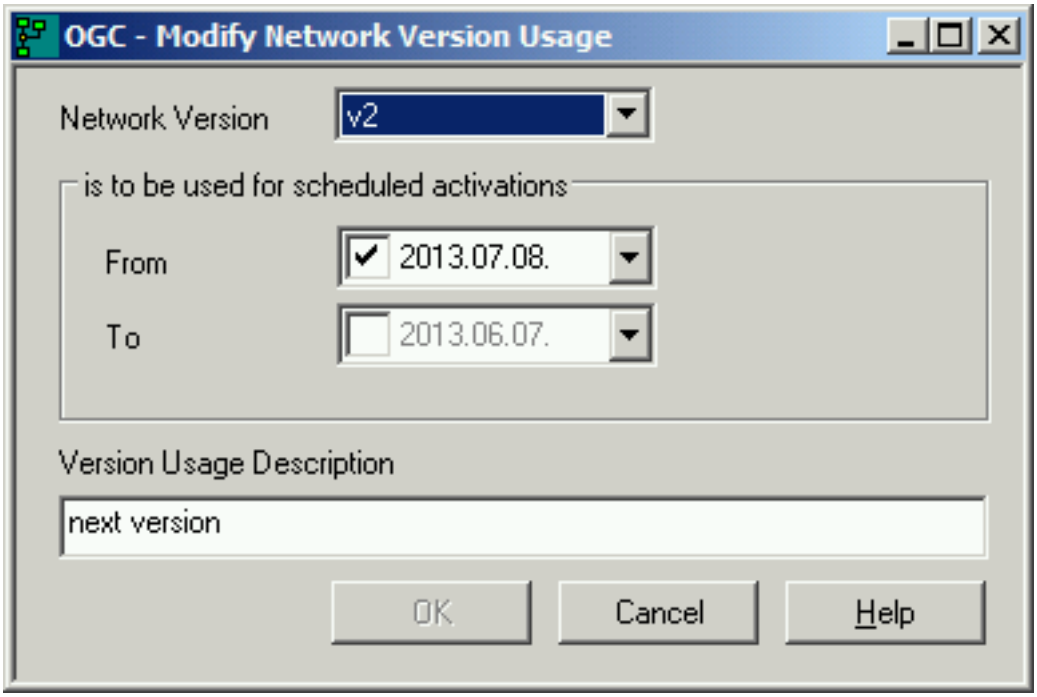
2 You can refresh the list by choosing the **Refresh** button.

3 You can modify or delete a defined version usage by selecting the respective row and choosing **Modify** or **Delete**, respectively.

Or:

You can add a version usage definition by choosing **Add**.

If you add or modify a defined network version usage, a window like the example below opens:



Add or modify the network version usage definition for scheduled activations as required. You can select a start or an end date only. If you omit the end date, the time is unlimited (infinite).

Note: The date ranges must not overlap.

19 Copying Job Network Definitions

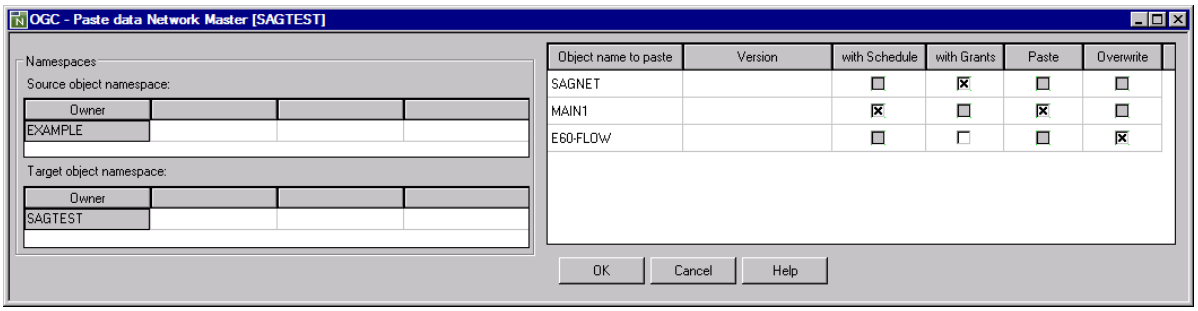
- Fields: Paste Data Network Master 167

To copy a network master

1 From a **Network Master** node in the object workspace, select the network you want to copy and choose **Copy data** from the context menu, or press you CTRL+C.

2 Select the **Network Master** where you want to add the new network and choose **Paste data** from the context menu, or press CTRL+V.

A **Paste data Network Master** window like the example below opens:



The owner of the network(s) to be copied (here: EXAMPLE) and the owner of the **Network Master** node selected for copying (here: SAGTEST) are listed in the left-hand tables.

The network(s) to be copied are listed in the right-hand table. In the example above the network:

SAGNET is not copied at all,
MAIN1 is copied (including its schedule definition) if no network of the same name exists in the **Network Master** target node, and
E60-FLOW is only copied if a network of the same name already exists in the target node.

3 Change the entries as required. The fields available in the window are described in *Fields: Paste Data Network Master*.

When you are finished, choose **OK**.

The new network master(s) are copied to the selected **Network Master** metanode.

Fields: Paste Data Network Master

The fields of the **Paste data Network Master** window are described in the following table.

Note: You cannot copy a network to an existing network, or a version to an existing version.

Field	Description	
Namespaces	The fields in the **Namespaces** section contain the name of the owner from which the new network is copied.	
Object name to paste	The name to be used for the new network.	
Version	The network version to be used for the new network. To clone a network version, make sure that owner names in the **Source/Target object namespaces** are identical, and enter a different network version.	
with Schedule	Possible check box settings:	
	unchecked	The schedule linked to the source network will not be copied (default).
	checked	The schedule of the source network will also be copied. The current day will be excluded to prevent an unwanted immediate activation. The current schedule table will be copied from the day following the current day until the end of the current year. The execution history will not be copied.
with Grants	*unchecked*	Other users/owners granted access to the source network will not be copied to the new network (default). The new network will only have access rights for the current user/owner. See also *Granting Definition: Authorizing Other Users or Owners to Access a Network*.
	checked	Other users/owners granted access to the source network will be copied to the new network, in addition to the current user/owner.
Paste	*unchecked*	A new network is not pasted. See also *Unchecked Paste and Overwrite*.
	checked	Pastes the new network into the **Network Master** node of the owner entered in **Namespaces** (default).
Overwrite	*unchecked*	An existing network with the same name entered in **Object name to paste** will not be replaced (default). See also *Unchecked Paste and Overwrite*.
	checked	Replaces an existing network with the same name entered in **Object name to paste**.

Unchecked Paste and Overwrite

If neither the **Paste** nor the **Overwrite** check box is marked, the copy function is not executed for the respective target network. This is useful, for example, if you specified multiple networks in **Object name to paste** and want to skip the copy function for a single network.

See also the **example of a Paste data Network Master** window.

20 Viewing and Maintaining a Job Network Diagram

- Explanations of Diagram Symbols 170
- Maintenance Functions for Diagrams 174
- Editing and Navigating in the Network Diagram 179
- Examples of Diagrams 182

Diagrams are provided for instances of a network master or an active network/job run. Diagrams can be used to view and edit a defined network or define a new network.

Note: You can customize the diagram view by changing the **Diagram** settings in your user profile described in the *Administration* documentation.

To view a network diagram

1 In the object workspace, select a **Network Master** instance or an **Active Run** instance.

2 Open the context menu and choose the **Diagram** function.

Depending on the node instance selected, a **Diagram Network Master** or **Diagram Active Run** window (see *Examples of Diagrams*) opens with a graphical view of the selected network.

The symbols used to represent the network components are described in *Explanations of Diagram Symbols*.

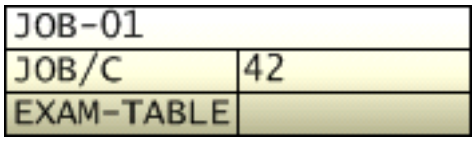
You can view and rearrange all network components as described in *Maintenance Functions for Diagrams*.

Explanations of Diagram Symbols

This section describes the symbols available in a diagram and indicates when a symbol only applies to an active job run.

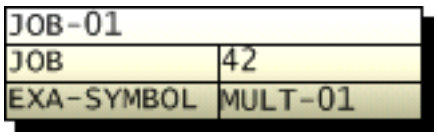
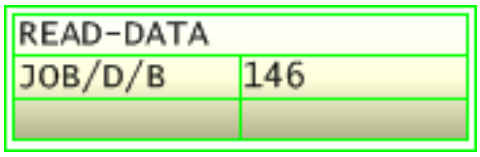
The display of the symbols in the diagram graphic depends on whether the **Use new design** option is set (default) in the user profile (see the **Diagram** page described in *Administration* documentation). In general, the new design is characterized by gradient symbol colors and icons that identify specific reasons for an active job run that ended `not ok`.

The information provided with a diagram symbol can exceed the size available to display text. In addition, a complex network can contain very small symbols impossible to read without **zooming** because the size of the diagram is adjusted to the size of the network. Therefore, **tooltips** for annotations are provided for all symbols so you can always view the complete information in a readable form.

Symbol	Description
	Job. Common entries are ■ Job name (here: `JOB-01`) ■ **Job type** (here: `JOB`)

<table>
<tr><th>Symbol</th><th>Description</th></tr>
<tr><td></td><td>

Job special type (here: <code>C</code> for a cyclic job) and/or milestone specification
Execution node number (here: <code>42</code>)
Symbol table name (here: <code>EXA-SYMBOL</code>).

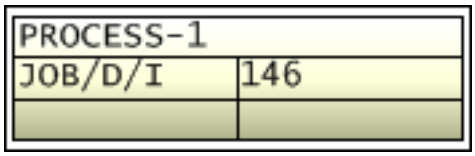
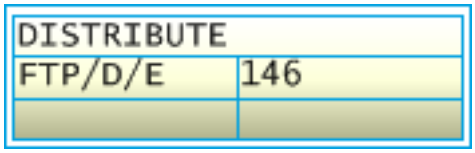
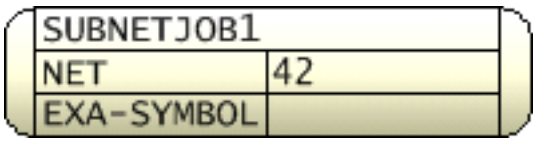
</td></tr>
<tr><td>
</td><td>Multiple suffix job (here: suffix symbol <code>MULT-01</code> in the symbol table <code>EXA-SYMBOL</code>).</td></tr>
<tr><td>
</td><td>Milestone job that performs as the first network job.

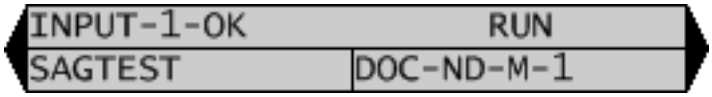
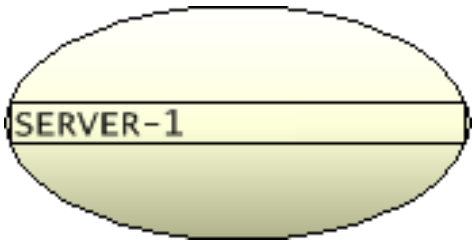
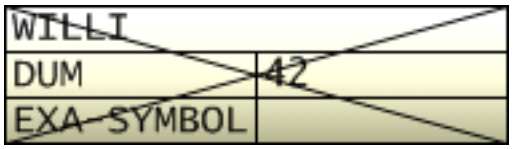
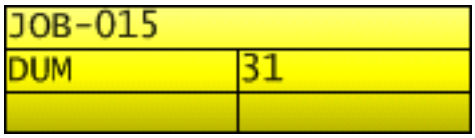
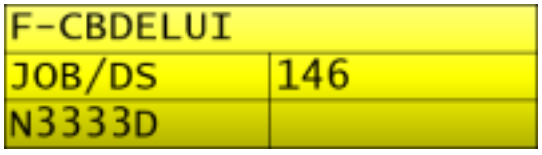
In addition to the job special type, a milestone job has one or more (separated by slashes) of the following specifications:

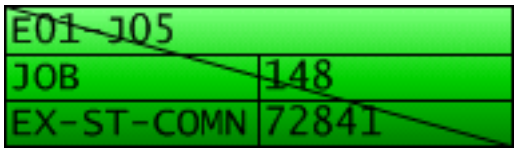
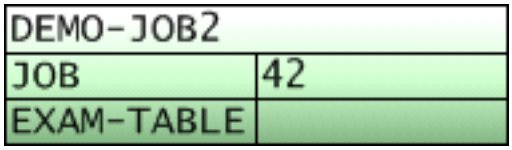
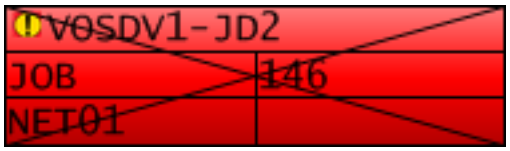
User-defined milestone:

<table>
<tr><td><code>B</code></td><td>Milestone job, network begin</td></tr>
<tr><td><code>E</code></td><td>Milestone job, network end</td></tr>
<tr><td><code>I</code></td><td>Milestone job, other</td></tr>
</table>
System milestone:

(used in active jobs only)

<table>
<tr><td><code>1</code></td><td>Milestone job, network begin</td></tr>
<tr><td><code>2</code></td><td>Milestone job, network end</td></tr>
</table>
</td></tr>
<tr><td>
</td><td>Milestone job that performs in no particular job order.</td></tr>
<tr><td>
</td><td>Milestone job that performs as the last network job.</td></tr>
<tr><td>
</td><td>Job with type Subnetwork (<code>NET</code>), here named <code>SUBNETJOB1</code>.</td></tr>
</table>

Symbol	Description
	Condition. Indicates the condition name (here: READ-DATA-OK) and the **condition reference** (here: RUN).
	External condition (new design) from another network. Common entries are: ■ Condition name (here: INPUT-1-OK) ■ **Condition reference** (here: RUN) ■ Network owner (here: SAGTEST) ■ Network name (here: DOC-ND-M-1)
	External condition (old design).
	Resource (here named FTP-SERVER-1).
	Reset condition.
	Set condition.
	Successfully ended job.
	Job waiting for execution. (active run only)
	Job executed as a **temporary dummy for one of the reasons** described in *Fields: Maintenance Job Active* (here: DS = dummy due to schedule definition). (active run only)

Symbol	Description
	Running job. (active run only)
	Job in hold or suspended after changes to the job definition. (active run only)
	Job ended `not ok`. (active run only)
	Job ended `not ok` for one of the following reasons: ■ Job missed the latest start time and/or deadline defined in the **Scheduling Parameters** of the job. ■ Latest start time and/or deadline defined in the **Network schedule** does not allow job execution. ■ Job exceeded latest start time for one of the above reasons. (shown in the new design and for the active job only)
	Job ended `not ok` because network activation was cancelled by the user. (shown in the new design and for the active job only)
	Job ended `not ok` for any reason except  and  explained above. (shown in the new design and for the active job only)

The background colors used for the symbols denote the following:

Color	Explanation
yellow	The job waits for its execution. If the allowed execution time is exceeded, the job is considered as `ended not ok` and the symbol changes to red. (active run only)
green	The job is currently executing. (active run only)
light green	The active job is affected by ■ an explicit hold function ■ an active regenerated JCL ■ a file/member that is changed in the active job definition. (active run only)
light yellow	The job ended `ok`.
red	The job ended `not ok`, for example, due to a JCL load error or an execution timeout. (active run only)
light gray	The condition is set.
dark gray	The condition is not set.

Maintenance Functions for Diagrams

This section describes the functions available in a network diagram and indicates whether a function can be used only in a network master or an active run.

The following functions are available within the free space area of the diagram:

Function	Description
Arrange	Rearranges the objects of the diagram automatically.
Refresh	Refreshes the presentation of the diagram. You can also use F5.
Auto Refresh	You can: ■ start/stop automatic refresh ■ change the interval of automatic refresh **Note:** Changes to the refresh settings remain valid for the diagram in the active window only. For persistent changes, modify the **Automatic Refresh** settings in your user profile (see the *Administration* documentation).
Object Filter	You may switch on/off: ■ Condition nodes

Function	Description
	■ Resource nodes **Note:** Changes to the filter settings remain valid for the diagram in the active window only. For persistent changes, modify the **Filter** settings in your user profile (see the *Administration* documentation).
Tooltip	You can customize the information displayed in tooltip: ■ For **Job** it is possible to switch the display on/off for Name, Type, Status (only in active run), Node, Symbol Table, Suffix symbol, Subnetwork, Job ID (only in active run). ■ For **Condition** it is possible to switch the display on/off for Name, Status (only in active run), Reference, Type, Suffix symbol, External network. ■ For **Resource** it is possible to switch tooltip on/off. ■ For **Connector** it is possible to switch the display on/off for source and target. **Note:** Changes to the tooltip settings remain valid for the diagram in the active window only. For persistent changes, modify the **Diagram** settings in your user profile (see the *Administration* documentation).
Find	Opens a search dialog where you can enter a search string. You can also use CTRL+F. See also ***To search for a string***.
Find Next	Finds the next occurrence of a search string entered in the search dialog. You can also use F3. See also ***To search for a string***.
Create Job	Adds a new job to the network. A **Create new Job Master** (see ***Adding a Job Definition***) or **Create new Job Active** (active run only; see ***Adding a New Job to the Active Network***) window opens to add a new job to the network.
Create Condition (master only)	Adds a new condition. Proceed as described in ***To add and link a condition.***
Create Active Condition (active run only)	Opens the **Condition Active** window to add a new active condition.
Paste (master only)	Paste a job object in the current network. See ***Pasting Objects***.
Browse Log	Opens the Entire Operations log for the network or active run shown in the active diagram window or a symbol selected in the active diagram. The log only contains data relevant for this network or run. See also the **Browse Log function** that can be performed on a selected node in the tree view of the object workspace.

Function	Description
Page Setup	Opens a dialog where you can specify the page setup for printing the current diagram.
Printer Setup	Opens a dialog where you can specify your printer settings.
Print Preview	Opens a print preview where you can check if the diagram is laid out for printing according to your preferences.
Print	Opens a dialog where you can send the diagram to a printer.
Zoom	You have the following possibilities to zoom the diagram: ■ in: makes the diagram larger ■ out: makes the diagram smaller ■ selection: zooms in/out single or multiple diagram symbols selected in the diagram. You can select single or multiple elements with CTRL+LEFT MOUSE BUTTON or by dragging the left mouse button over a group of adjacent symbols. ■ complete: shows the complete diagram on the screen ■ original: shows the diagram in its original size
Show World View	Opens a separate window with a minimized view of the complete diagram for easy navigation in complex diagrams. The green frame outlines the current viewport in the associated diagram. You can move the green frame to change the viewport in the diagram.
Show Legend View	Opens a separate window with a minimized view of the symbols used in the diagram for a quick overview of the objects (for example, resources and conditions) used in the job network.
Export Graphic	Opens a dialog where you can save the diagram on your hard disk.

The following functions are available for a job symbol selected in the diagram:

Function	Description
Open	Displays the currently selected job (box) and opens the **Job Master** or **Job Active** (active run only) window. Here, you can perform changes on the job.
Display	Displays the currently selected job (box) in the **Job Master** or **Job Active** (active run only) window.
New	Creates a new job (box) and opens the **Create new Job Master** (see *Adding a Job Definition*) or **Create new Job Active** (active run only; see *Adding a New Job to the Active Network*) window.
Mass update (master only)	Changes the **Execute if temporary dummy** option setting in End-of-Job actions for all jobs currently selected in the diagram. See also *Changing Execute if Temporary Dummy Settings for Multiple Jobs*.
Delete	Deletes the currently selected job (box). See *Deleting Objects*.

Function	Description
(master only)	
Copy data (master only)	Copies the currently selected job (box). See *Copying Objects*.
Paste data (master only)	Pastes the currently selected job (box). See *Pasting Objects*.
Usable Symbol Tables	Lists all symbol tables that can be used by the job. See also *Listing Usable Symbol Tables* in the section *Symbol Table and Symbol Maintenance*.
Deactivate (active run only)	You can cancel the planned activation of a job in a planned network run that is waiting for activation.
Hold (active run only)	If the job has already been submitted to the operating system, it will be held in the operating system.
Release (active run only)	Releases a job from hold status.
Cancel (active run only)	The active job is cancelled.
Resubmit (active run only)	After a job has terminated, you can modify and resubmit it while it is still in the active database. This function is useful after a job has failed.
Reactivate (active run only)	Reactivates a job.
Browse Master JCL (master only)	Browses the master JCL. See *Displaying Master JCL*.
Browse Active JCL (active run only)	A **window** opens where, you can see the actual JCL submitted to the operating system for execution. It is produced from the Master JCL when the job or network is activated. The symbols are replaced with values from the active symbol table. If it is dynamic JCL, the generation is performed at this time. The active JCL is stored in the active Entire Operations database.
Edit Master JCL (master only)	Edits the master JCL. See *Editing Master JCL and Natural Sources*.
Edit Active JCL (active run only)	A window opens where you can view and edit the active JCL of the job.
Regenerate Active JCL (active run only)	Regenerates the JCL while the job is in the active database.

Function	Description
Stop cyclic execution (active run only)	Stops execution of currently running cyclic job.
Modify Latest Start (active run only)	See *Modifying the Latest Start Time for an Active Run* in the section *Active Job Networks*.
Release edit lock (active run only)	Removes a lock from active JCL: see **Release edit lock** in the section *Active Job Networks*.
Pregenerate Active JCL (master only)	See *Pregenerating Active JCL*.
Remove pregenerated Active JCL (master only)	See *Removing Pregenerated JCL*.
Edit pregenerated JCL (master only)	See *Editing Pregenerated JCL*.
Browse SYSOUT (active run only)	A dialog opens were you can view the SYSOUT of an active job. See *Viewing Job SYSOUT*.
Extended Log / JCL (active run only)	A screen opens, where you can see the JCL which was submitted for this job.
Extended Log / SYSOUT (active run only)	A screen opens, where you can see the SYSOUT of the job.
SYSOUT Messages (active run only)	A screen opens, where you can see the SYSOUT messages of the job (z/OS only).
Activate Job (master only)	Opens the **Activation** dialog where you can activate the currently selected job (box).
Open Subnetwork (active run only)	Opens the subnetwork Network Master dialog for a subnetwork (Job with type **Subnetwork**).
Subnetwork diagram	Opens the Network diagram for a subnetwork (Job with type **Subnetwork**).
Zoom Subnetwork	Opens a list of jobs for a subnetwork (Job with type **Subnetwork**).
Waiting for (active run only)	See *Displaying Prerequisites for Active Jobs: Waiting for* in the section *Active Job Networks*.
List Active Jobs	See *Listing Active Jobs*.
Browse Log	Opens the log for the currently selected job (box). See *Displaying Logged Information - Browse Log Function*.

Function	Description
Export (master only)	See *Exporting Objects* in the *Import/Export Functions* documentation.
Add to Workplan	Adds an object to the workplan. See also ***Add to Workplan***.

Connecting Lines

The following functions are available for the relationships (connecting lines) within the diagram:

Function	Description
Set Input Condition (active run only)	Sets the input condition.
Reset Input Condition (active run only)	Resets the input condition.
Delete (master only)	Deletes the currently selected connection to the input or output condition of a job. If no other connections exist for the respective condition, the entire condition is removed from the diagram.
Open Network (master only)	Opens the corresponding network maintenance dialog (see ***Modifying a Network Definition***) for an external input condition.
Network Diagram (master only)	Opens the corresponding network diagram and to position diagram view to the proper condition for an external input condition.
Open Input Condition (master only)	Opens the **Condition** dialog. Here, you can perform changes on the currently selected input condition.
Open Output Condition (master only)	Opens the **Condition** dialog. Here, you can perform changes on the currently selected output condition.

Editing and Navigating in the Network Diagram

This section provides instructions for using the diagram editor functions to maintain objects for a network and find a particular object in the diagram.

To add and link a condition

1 From the context menu in the **Diagram Network Master** window, choose **Create Condition**.

A square symbol appears indicating the position where the new condition will be placed in the diagram.

2 In the free space of the diagram, left-click on the position where you want to place the new diagram symbol.

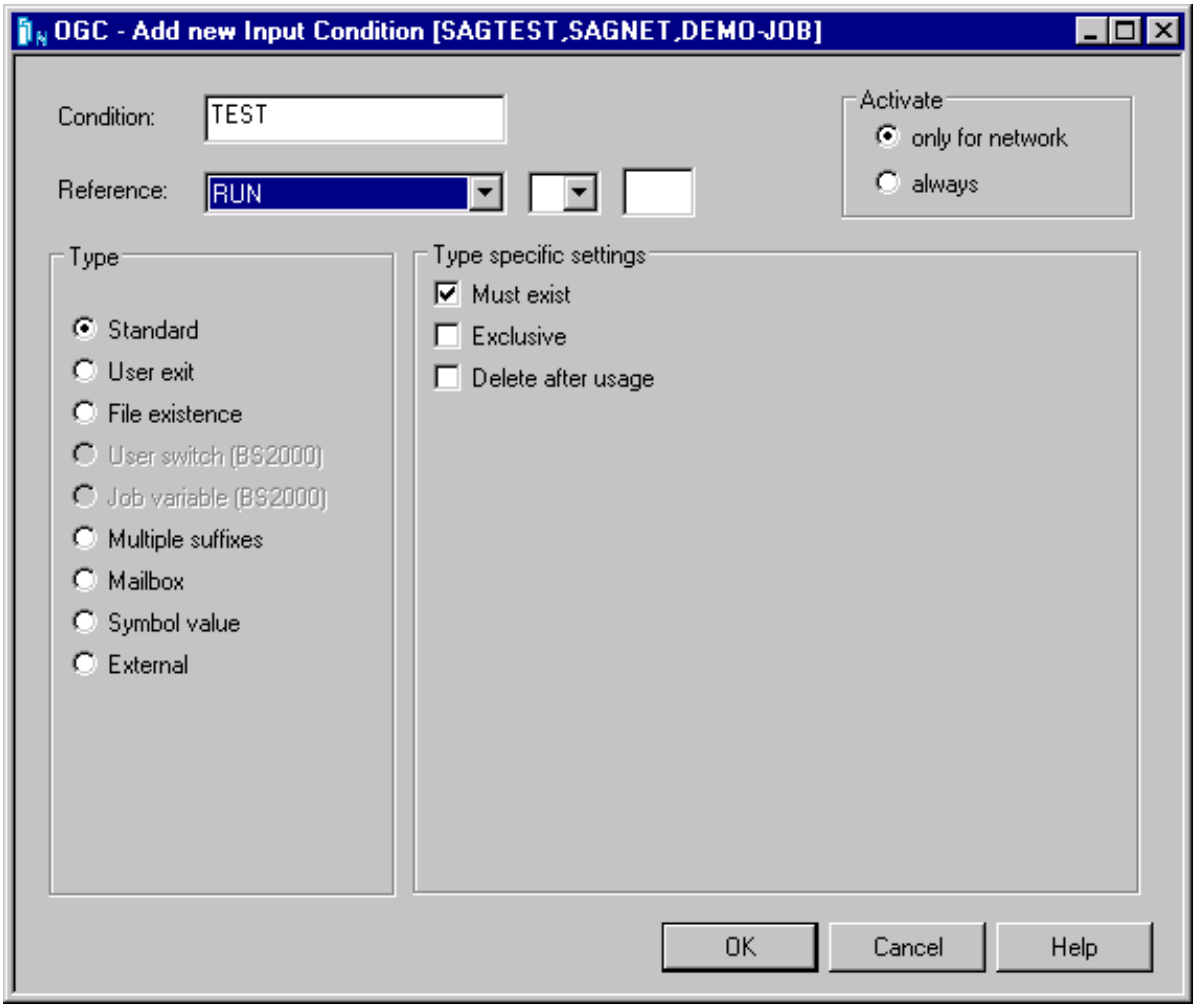
A **Create Condition** dialog appears.

3 Enter a valid condition name and choose **OK**.

The new condition is added to the diagram as a temporary (TMP) object, without any logical job links as shown in the tooltip information for the condition symbol.

4 Drag the selected condition and drop it onto the job to which you want to link the new condition as a logical input condition.

An **Add new Input Condition** window like the example below opens:



5 Enter the required definitions. The fields and options in the window are described in ***Fields and Selection Options: Input Condition*** in the section *Job Maintenance*.

6 When you are finished, choose **OK**.

The new condition is linked to the job as indicated by the connecting line between the job and the condition symbol and the tooltip information for the symbols.

To create an output condition

1 Select a job.

2 Drag the job and drop it onto a condition.

3 Select a branch (**All Checks ok** or **Any Check not ok**) to which the condition should be assigned.

To link two jobs

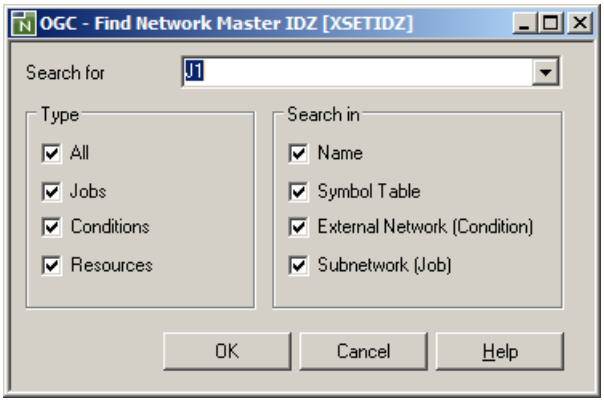
1 Select a job.

2 Drag the job and drop it onto a job.

3 Select a branch (**All checks ok** or **Any check not ok**) to which the condition should be assigned.

To search for a string

1 Call the search dialog by pressing CTRL+F or by choosing the **Find** function from the context menu:



2 Fill in your search criteria. You can specify the search string, object type where the string is searched as well as the attribute of the object where to be searched. An asterisk (“*”) is not interpreted as a wildcard.

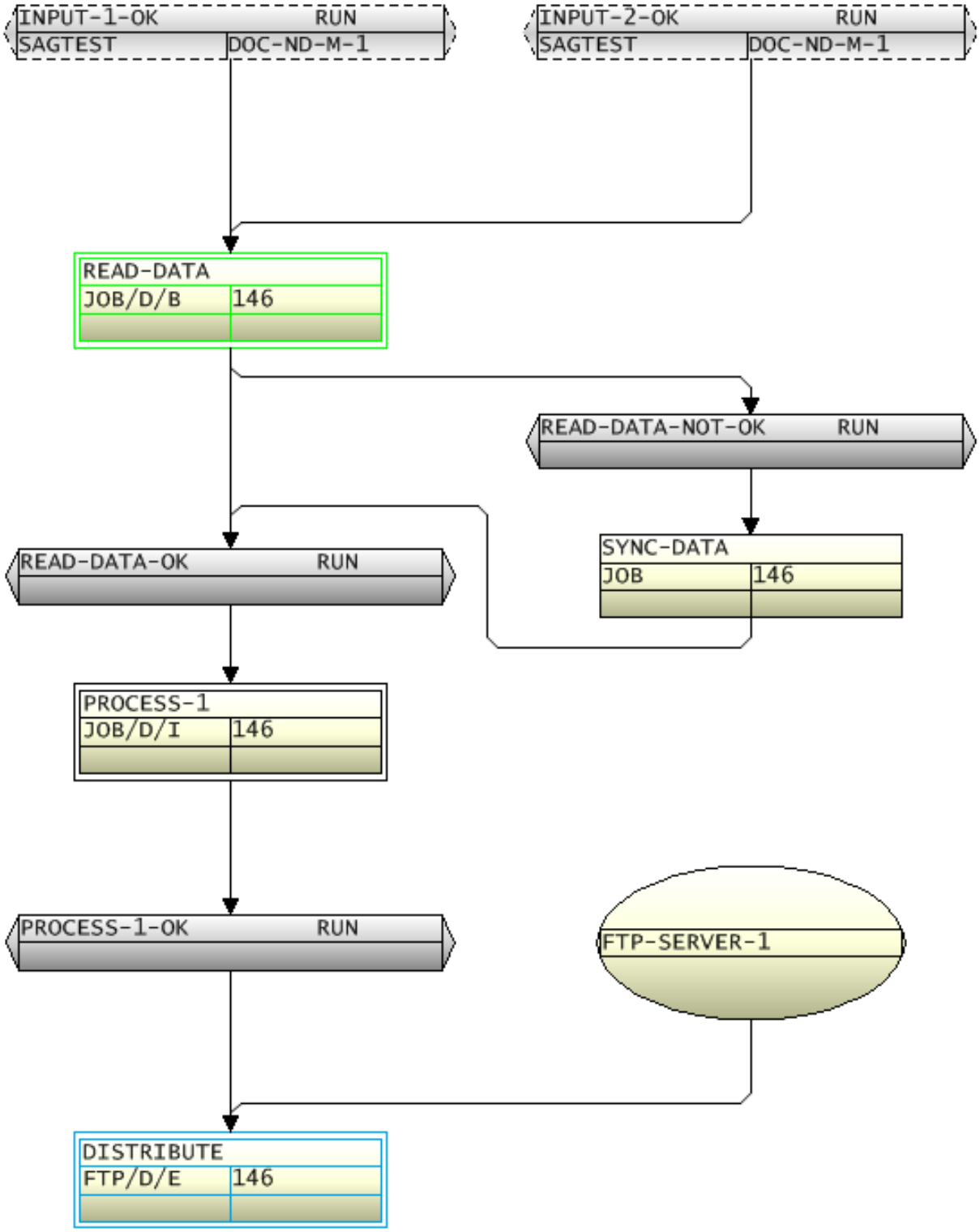
3 Choose **OK**.

Examples of Diagrams

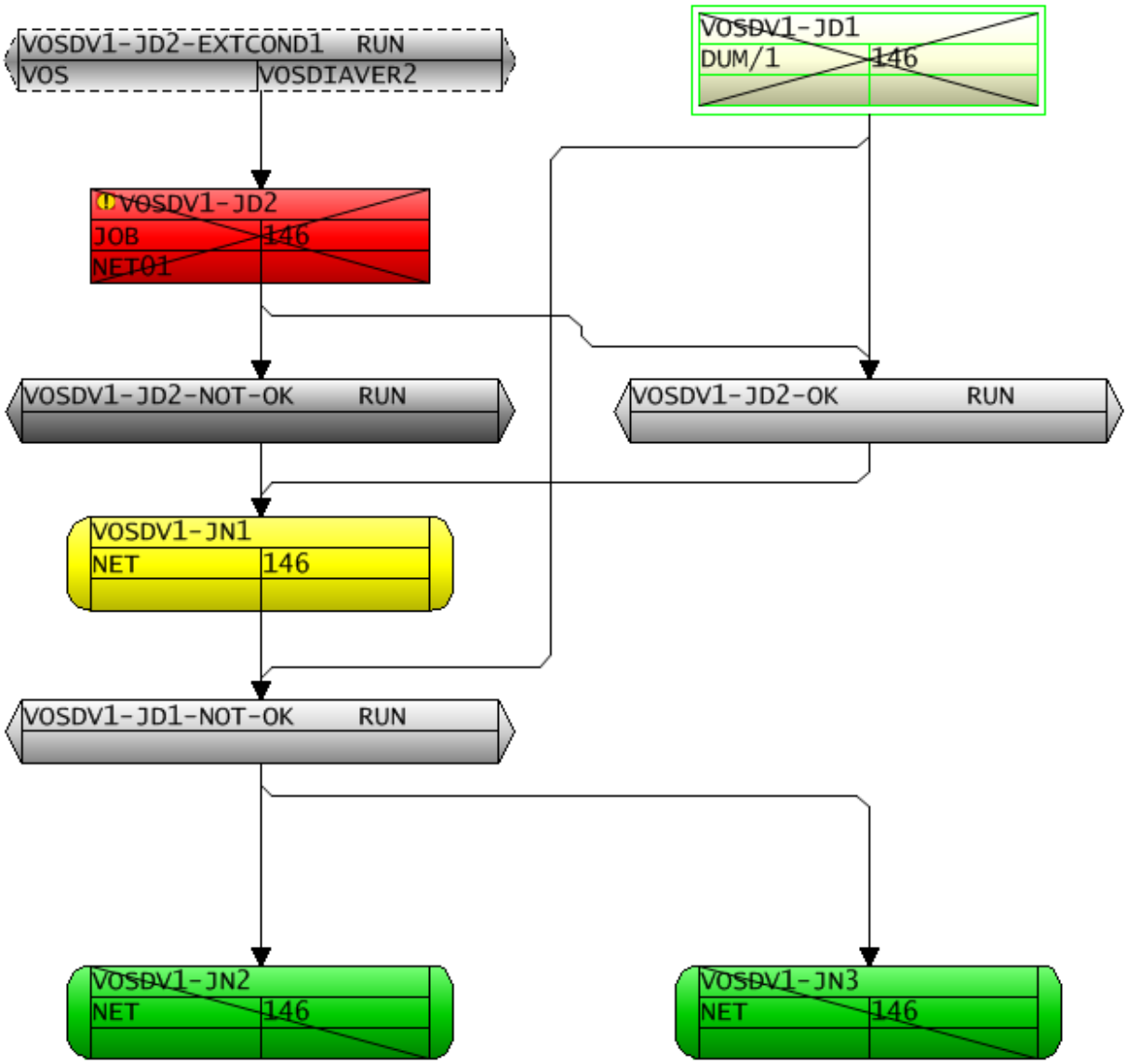
- Example of a Diagram Network Master (New Design)

- Example of a Diagram Active Run (New Design)

Example of a Diagram Network Master (New Design)


INPUT-1-OK RUN
SAGTEST DOC-ND-M-1
INPUT-2-OK RUN
SAGTEST DOC-ND-M-1
READ-DATA
JOB/D/B 146
READ-DATA-NOT-OK RUN
READ-DATA-OK RUN
SYNC-DATA
JOB 146
PROCESS-1
JOB/D/I 146
PROCESS-1-OK RUN
FTP-SERVER-1
DISTRIBUTE
FTP/D/E 146

Example of a Diagram Active Run (New Design)



21 Activating a Job Network Manually

- Fields: Network Activation 190
- JCL Check only 191

The Entire Operations Monitor will activate job networks automatically according to their defined schedules. However, it is also possible to activate a network manually from the **Network Master** node.

When a network is activated, all jobs within the network, except recovery jobs, are activated.



Note: Only authorized users can activate job networks manually.

There are several situations in which you may wish to activate a job network manually:

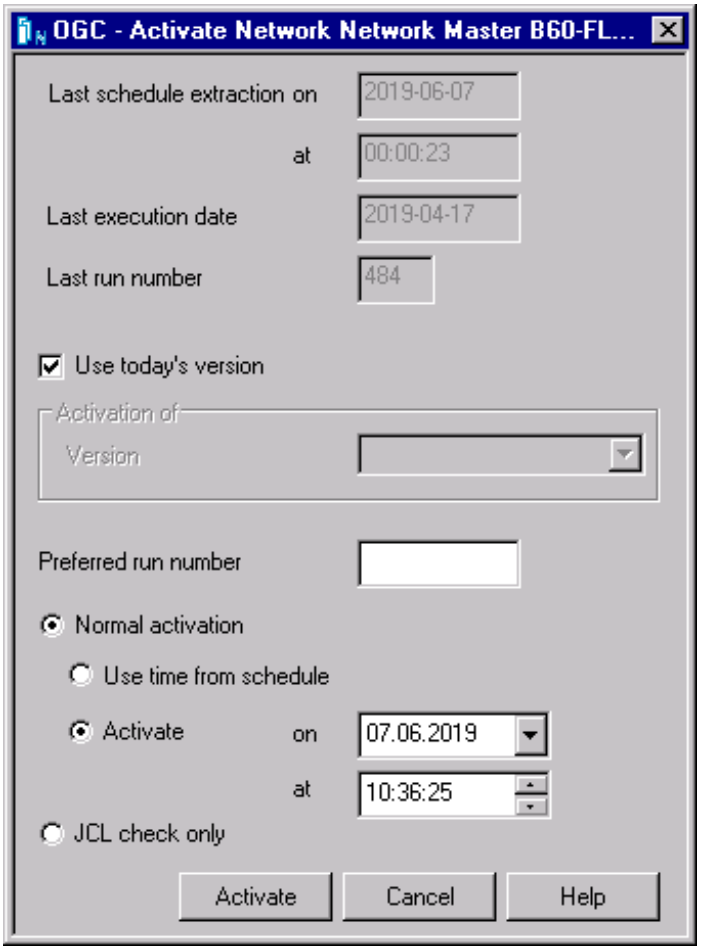
- When no schedule has been defined for the network;
- To start the same network several times (with different run numbers);
- When the network is not scheduled for the current date;
- To reactivate a network.

To activate a network manually

1 In the object workspace, select a **Network Master** instance.

2 Invoke the context menu and choose **Activate Network**.

An **Activate Network** window like the example below opens:



3 Change the input fields as required. All fields contained in the window are explained in *Fields: Network Activation*.

4 Choose **Activate** when you are finished.

If symbol prompting has been defined for the network, a **Symbol Prompting window** opens with a table of symbols used during dynamic JCL generation.

If required, change the symbols and symbol values. For further explanations, see *Symbol Prompting during Network or Job Activation* in the section *Symbol Table and Symbol Maintenance*.

5 Choose **Continue** when you are finished.

If the activation is successful, the activated network gets a new run number as returned in a message, and its execution can be tracked in the GUI Client.

If a workplan has been defined for the network, a **Workplan window** opens with a list of objects accessed during the current Entire Operations session.

6 In the **Workplan window**, change the entries as required. For further explanations, see ***Show Workplan*** in the section *Using Entire Operations GUI Client*.

7 Choose **OK** when you are finished.

Fields: Network Activation

The fields in the **Activate Network window** are described in the following table:

Field/Option	Description
Use today's Version	Version of the network to be activated.
Activation of Version	If a current version exists for the current date, it will be preset as default value. If more than one version exists for a network, select **Use today's Version** (default) or **Activation of Version** as needed. If a version is not defined, then **Activation of Version** must be used. If no version for a network exists, then **Use today's Version** and **Activation of Version** have no function.
Last Schedule Extraction on/at	Date and time of the last automatic schedule extraction for this network by the Entire Operations Monitor.
Last Execution Date	Date of the last activation.
Last Run Number	Run number of the last activation.
Preferred Run Number	If input is allowed for this field, you can enter the run number you want to be used for the activated network. Run numbers for further network activations increment from this number. If the requested run number is in use, Entire Operations assigns the next free number to the run. If this field contains a zero (0) or no value, Entire Operations (as usual) determines the number to be assigned to the run. Field input is allowed or not allowed (default) in the network default settings: see the option **Allow setting of preferred run number during activation** in the section *Defaults for Network Options* in the *Administration* documentation.
Normal activation: Use Time from Schedule	Select this option to activate the network within the **defined time frames** (**Earliest Start**, **Latest Start**, **Deadline**), even if the current date is not a scheduled date for this network. With this option, you can force the same time dependencies as if the network were scheduled and activated automatically.
Normal activation: Activate on/at	Shows the current date and time. You can modify the date and time to force activation at any time on any date. There is no limitation for future date and time settings.

Field/Option	Description
JCL Check only	If this option is selected: Only a JCL check is performed for all jobs of the job network. See *JCL Check only* for further reference.
	If this option is not selected: **Normal activation** (job submission) is initiated. This is the default.

JCL Check only

For a JCL check, the required commands are automatically inserted:

OS	Command
z/OS, JES2	`TYPRUN=SCAN`
z/OS, JES3	`EXEC PGM=JCLTEST`
BS2000:	`/MODIFY-SDF-OPTIONS MODE=TEST`
UNIX	`Script execution with set -vn`

The following applies:

- Windows: Jobs are executed as `Dummy due to JCL Check`.
- Jobs of type NAT (Natural) are executed as `Dummy due to JCL Check`.
- End-of-Job action user exits (exit type `EJA`) will be skipped.

Notes:

1. Active conditions used by the network execution for **JCL check only** do not interfere with active conditions of "real" active jobs or active networks.
2. These active conditions are marked with the prefix © in all screens and lists.

22 Checking for a Loop in a Job Network

To check for a loop in a network

1 In the object workspace, select a **Network Master** instance.

2 Invoke the context menu and choose **Check for loop**.

- If a loop in the job flow is detected, the following message appears:

  ```
  Definition Loop in xxx/yyy
  ```

  where:

  xxx is the name of the network tested for loops, and

  yyy is the name of the job in the network, in which a cyclic link was first detected.

  When a loop has been detected, a corresponding message is written to the Entire Operations log.

- If no loop is detected, the following message appears:

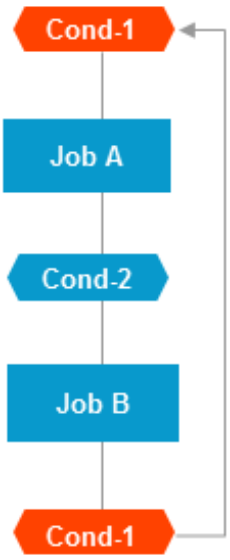
  ```
  No Loop in Network xxx
  ```

- If the network contains more than 1000 jobs and no loop has been detected in these first 1000 jobs, the following message appears:

```
Loop check limit 1000 exceeded. Loop check stops.
```

Entire Operations does not perform a loop test on more than 1000 jobs.

The following figure illustrates a loop within the job flow of a network.



23 Applying Network Defaults to Jobs (Mass Update)

- Columns: Application of Network Defaults to Jobs/Copy Defaults 198
- Functions: Application of Network Defaults to Jobs/Copy Defaults 199

Caution: Use this function with extreme care. Individual job definitions can be easily overwritten by mistake. You are not asked to confirm each update and cannot undo any changes.

You can perform a mass-update of job fields within a network provided you have write permission for networks in your user profile (see *Network Maintenance Functions* in the *Administration* documentation).

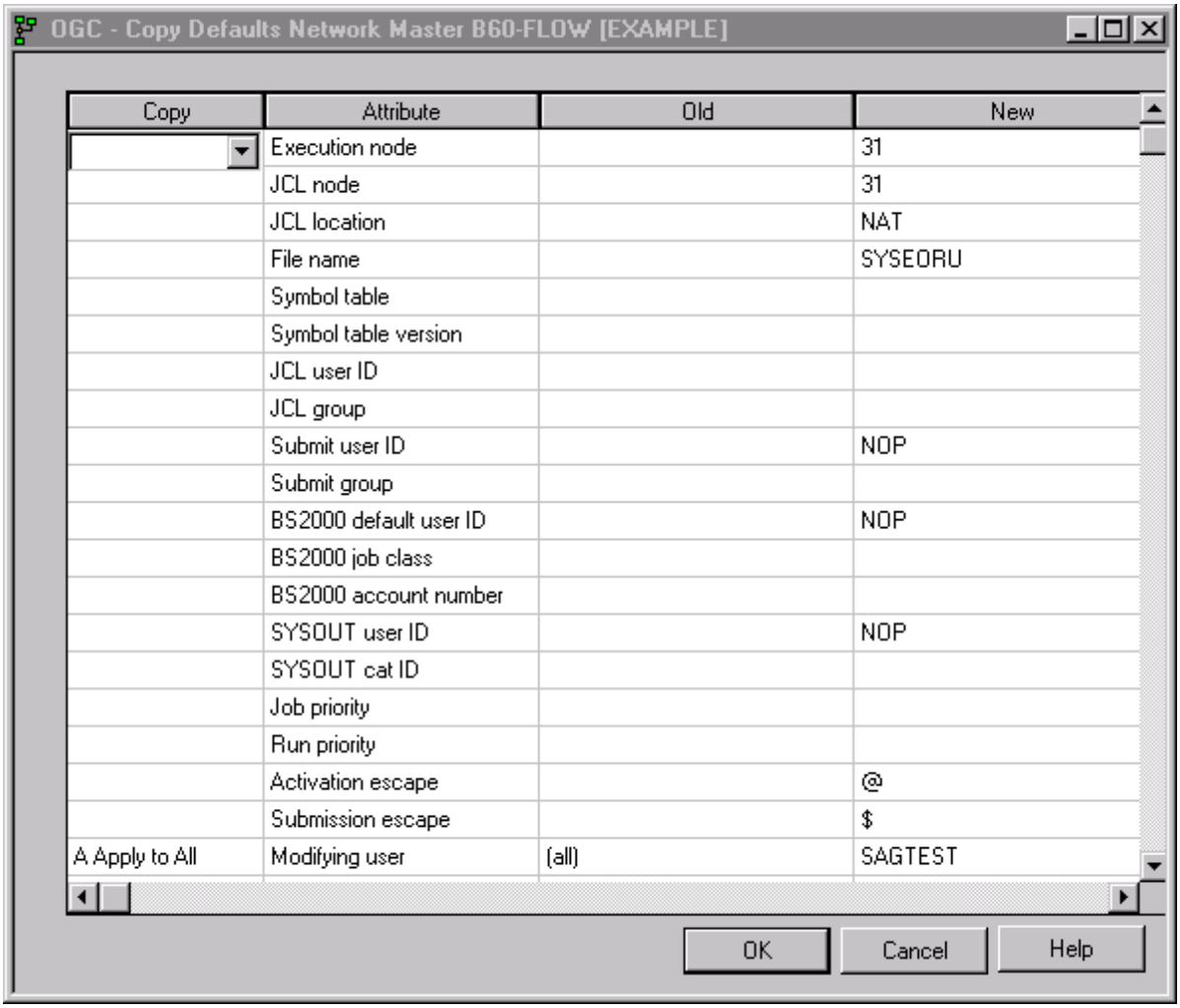
The mass update function copies values of specified network fields as default values to all jobs of a selected network.

The default values of the default network fields are also used when creating a new job.

To apply default field values to all jobs in the network

1 In the object workspace, select a **Network Master** instance and choose **Copy Defaults** from the context menu.

A **Copy Defaults** window like the example below opens:



The **Attributes** column lists all fields that can be copied to all jobs that belong to the network. The attribute values relevant for a mass update of job fields are listed in the **Old** and **New** columns.

For more information on the columns and fields contained in the window, see *Columns: Application of Network Defaults to Jobs/Copy Defaults* and *Fields/Attributes: Application of Network Defaults to Jobs/Copy Defaults*.

2 From the drop-down list box in the **Copy** column, next to the attribute definition you want to copy, select the function you want to perform: see *Functions: Application of Network Defaults to Jobs/Copy Defaults*.

3 When you are finished, choose **OK**.

The function is performed, and specified values are copied from the network fields to the corresponding job fields, if requested by the function.

A message is returned indicating the number of jobs that have been modified.

For each job for which field values were replaced, the name of the field and the new field value are listed in the Entire Operations log with a message like `Network default Activation Escape @ applied`. See also ***Displaying Logged Information*** in the section *Log Information*.

Columns: Application of Network Defaults to Jobs/Copy Defaults

The columns of the **Copy Defaults window** are described in the following section:

Column	Description
Copy	Select a function from the drop-down list box or leave the column empty: see ***Functions: Application of Network Defaults to Jobs***.
Attribute	Network field whose value can be used as the default for all jobs that belong to the selected network. New field values indicated in the **New** column can be copied into the corresponding fields of all jobs that belong to the selected network. See also ***Fields/Attributes: Application of Network Defaults to Jobs/Copy Defaults*** below.
Old	Contains the value saved from the last change of a network field.
New	Contains the current value in the network field.

Fields/Attributes: Application of Network Defaults to Jobs/Copy Defaults

The following table lists all fields in the **Copy Defaults window** and the corresponding network fields. The table also indicates the names of job fields if different from the names of the network fields.

All network fields are described in detail in ***Fields and Commands: Network Definition***.

Field to Copy	Corresponding Network Field	Remark
Execution Node	**Execution Node**	
JCL Node	**JCL Node**	
JCL Location	**JCL Location**	
Activation Escape	**Escape characters: Activation**	
Submission Escape	**Escape characters: Submission** Job field: Escape Characters: Submit	
Job Priority	**Job Priority**	Applies to BS2000 only.
Run Priority	**Run Priority**	Applies to BS2000 only.
End of job action	**End-of-Job Action Errors impact Job Result** Job field: End-of-Job Action	

Field to Copy	Corresponding Network Field	Remark
Symbol Table	**Symbol Table**	
Symbol Table Version	**Symbol Table Version** Job field: Version	
JCL User ID	**JCL User ID**	
JCL Group	**JCL Group**	Applies to UNIX/Windows only.
Submit User ID	**Submit User ID**	
Submit Group	**Submit Group**	Applies to UNIX/Windows only.
BS2000 Default User ID	**Default User ID**	Applies to BS2000 only.
BS2000 Job Class	**Job Class**	Applies to BS2000 only.
BS2000 Account Number	**Account Number**	Applies to BS2000 only.
SYSOUT User ID	**SYSOUT User ID**	Applies to BS2000 only.
SYSOUT Cat ID	**SYSOUT Cat ID**	SYSOUT Catalog ID. Applies to BS2000 only.
Modifying User	**Job field: Modified**	ID of the user who last modified a job.
File Name	**File** Job field: File/NatLib	

Functions: Application of Network Defaults to Jobs/Copy Defaults

The functions available from the **Copy** drop-down list box of the **Copy Defaults window** are described in the following table.

The functions allow you to distinguish between jobs with network default definitions and jobs with individual definitions.

Function	Description
A Apply to All	Replace all field values. Replace all values regardless of whether they are default settings or individual job definitions. This is the default setting for **Modifying User**. The value `(all)` indicates that the **Modified** field in the job is always overwritten, regardless of the command used for function execution. The **Modified** contains the ID of the user who last modified a job definition.

Function	Description
S Apply to Specific	Replace specific field values only. Replace the value only if the value in the **Old column** matches the current value in the corresponding job field. All other field values remain unchanged. Example: The value in the **Old** column for **Execution Node** is `10`, and the value in the **New** column is `20`. The **Execution Node** field of all jobs currently containing `10` changes to `20`. All other execution node numbers remain as they are.
empty column	Replace no field value. Use the empty selection option for a value you do not want to replace in a job field. A field value is not replaced when the **Copy** column is empty. This is the default for all values except **Modifying User**.

VI Job Maintenance

This section describes the purpose of jobs and the functions available to create, maintain and delete Entire Operations objects of the type job.

Jobs are maintained on the master database which stores all user, job network, job and scheduling definitions. It also contains all information pertaining to defined logical conditions, resources, calendars, and symbol tables. All information stored on the master database can be maintained online.

General

Use of Jobs

Job Maintenance

Job Maintenance

- Available Functions: Job Master
- Listing All Job Definitions of a Job Network
- Selecting a Range of Job Definitions to be Listed

Control and Monitoring Functions for Jobs

Activating a Single Job Manually

Log Information

Viewing Job/Network Accounting Information

Maintenance Functions for Job Definitions

Creating a Job Definition

- Adding a Job Definition
- Fields: Job Definition (Master)
- Defining Job Type Specific Execution Features

- Defining Operating System Specific Execution Features
- Handling Prerequisite Resources for a Job
- Defining and Managing Job Conditions
- Defining Master JCL for a Job
- Scheduling a Job
- Defining Extended Log Information for a Job
- Writing and Viewing Online Documentation for a Job
- Defining and Managing End-of-Job (EOJ) Checking and Actions

Using a Dummy Job
Listing Usable Symbol Tables
Displaying and Modifying a Job Definition
Copying Objects / Pasting Objects
Deleting a Job Definition

Maintenance Functions for Subnetworks

Defining a Subnetwork
Adding or Modifying a Subnetwork Definition
Listing Jobs of a Subnetwork

Maintenance Functions for Job Control Language (JCL)

Using Job Control in Entire Operations
Defining Master JCL for a Job
JCL Locations
Pregenerating Active JCL
Editing Pregenerated JCL
Removing Pregenerated JCL
Handling JCL during Job Submission
Editing Master JCL and Natural Sources

Maintenance Functions for Input Conditions

Defining and Managing Job Conditions
Adding and Modifying a Master Input Condition
Deleting an Input Condition Definition
Listing Jobs Linked to an Input Condition
Input Condition with User Exit
Input Condition: Multiple Suffixes
Input Condition: File Existence

Input Condition: Mailbox
Input Condition: Symbol Value
Input Condition: BS2000 User Switch
Input Condition: BS2000 Job Variable
Defining Schedule Dependencies for an Input Condition
Defining and Editing a User Exit for an Input Condition

Maintenance Functions for End-of-Job Checking and Actions

Defining and Managing End-of-Job (EOJ) Checking and Actions

File Transfers to Entire Output Management

Passing Files to Entire Output Management

24 Use of Jobs

All jobs are members of job networks and can be linked by logical conditions. Some differences arise in End-of-Job checking, depending on the job type and the operating system (see the section *Defining and Managing End-of-Job (EOJ) Checking and Actions*). However, you can always define `Job OK` or `Job not OK` as a condition for subsequent system action.

An operating system job on z/OS can consist of several steps. In these cases, Entire Operations can check the result of each job step as part of End-of-Job analyses and triggers system action accordingly.

A job is uniquely identified within a job network by its job name. The job name can, but need not be the same as the JOB or LOGON statement name (job name by which the operating system identifies the job). Before job submission, jobs can therefore only be identified by the name defined to Entire Operations. A job can only be accessed through Entire Operations by its Entire Operations name.

When defining a job, you must also specify:

- JCL location (depending on job type);
- JCL and **execution nodes** (if different from those specified for the job network);
- JCL and submit user IDs;
- Scheduling parameters (optional; otherwise, the network default is used);
- End-of-Job checking and End-of-Job action specifications (see the section *Defining and Managing End-of-Job (EOJ) Checking and Actions* for details).

Note for z/OS:

We recommend that the JCL of one Entire Operations job contains only one job statement. Entire Operations retains only the first job number assigned to a submitted job.

25 Maintaining Jobs

- Listing Jobs 208
- Selecting a Range of Job Definitions to be Listed 210
- Available Functions: Job Master 210
- Displaying and Modifying a Job Definition 211
- Creating a Job Definition 218
- Writing and Viewing Online Documentation for a Job 219
- Deleting a Job Definition 221
- Displaying Job Dependencies 221
- Activating a Single Job Manually 222
- Defining Extended Log Information for a Job 225

Listing Jobs

This section describes how to list all job masters defined for a network.

Related Topics:

- *Listing Active Jobs* in the section *Active Job Networks*
- *Listing Jobs of a Subnetwork*

To list all jobs defined for a network

- In the **object workspace**, select **Owner** > *owner instance* > **Network Master** > *network name* > **Job Master** and choose **List** ( ).

  All jobs defined for the selected network are listed in the **Job Master List** window as shown in the following example:



Job Master List [EXAMPLE, E60-FLOW <v2.1>]

Search

Job	Typ	JCL-Speicherart	Spezieller Typ	Beschreibung	File oder Library	Member
JOB-01	Standard-Job [JOB]	Makro Natural-Source [MAC]	Normaler Job []	Where it all starts	EOR-T551	E60-M02
JOB-01-PG	Standard-Job [JOB]	Makro Natural-Source [MAC]	vorgenerierte JCL [G]	Pregenerated JCL	EOR-T551	E60-M02
JOB-012	Standard-Job [JOB]	Makro Natural-Source [MAC]	Normaler Job []	Depending on Job-01	EOR-T551	E60-M01
JOB-013	Standard-Job [JOB]	Makro Natural-Source [MAC]	Normaler Job []	Depending on JOB-012	EOR-T551	E60-M01
JOB-014	Standard-Job [JOB]	Makro Natural-Source [MAC]	Normaler Job []	Depending on JOB-013	EOR-T551	E60-M01
JOB-015	Dummy-Job [DUM]	keine JCL benötigt []	Normaler Job []	Depending on JOB-014	EOR-T551	
JOB-019	Standard-Job [JOB]	Makro Natural-Source [MAC]	Normaler Job []	Depending on JOB-01	EOR-T551	E60-M01
JOB-02	Standard-Job [JOB]	Makro Natural-Source [MAC]	Normaler Job []	Dep. JOB-15, JOB-19	EOR-T551	E60-M02
JOB-021	Standard-Job [JOB]	Natural-Source [NAT]	Normaler Job []		EOR-T551	JOB-021
JOB-03	Natural-Programm [NAT]	Natural-Source [NAT]	Normaler Job []	Depending on JOB-02	EOR-T551	E60-P01
JOB-04	Standard-Job [JOB]	Makro Natural-Source [MAC]	Normaler Job []	Depending on JOB-03	EOR-T551	E60-M01
JOB-05	Dummy-Job [DUM]	keine JCL benötigt []	Normaler Job []	Depending on JOB-04	EOR-T551	
JOB-06	Standard-Job [JOB]	Makro Natural-Source [MAC]	Normaler Job []	Where it all ends	EOR-T551	E60-M02

  The window columns are described in *Columns: Job Master List*.

This section covers the following topics:

- Columns: Job Master List

Columns: Job Master List

The following table explains the columns of the **Job Master List window**.

Column	Description	
Spec. Type	Special job type. Possible indicators are:	
	C	Cyclic execution.
	D	Execute as a **dummy job** but keep definitions
	G	Pre-generated JCL is available. See also ***Pregenerating Active JCL***. This indicator is only displayed if no other indicator is already used for the job.
	P	For job type SRV: Stop a Windows service. For job type STC: Stop a started task.
	R	Recovery job, usually not activated. See also ***Defining Recovery Actions***.
Job	User-defined job name.	
Type	Job type. They are described in the section ***Job Types.*** See also the field **Job Type** described in *Fields: Job Definition (Master)*.	
Loc	**JCL location**. (Empty if no JCL is defined, or if no JCL is required for the **job type**.) For possible JCL locations, see ***List of JCL Locations***.	
Description	Short description of the job.	
File or Library	Physical storage of JCL according to the **JCL location**. For a job of the type **Subnetwork** (NET), this column contains the name of the owner and the name of the subnetwork defined for the job (see the jobs J2-SUB and SUBNETJOB1 in the **example of a Job Master List** window).	
Member	The member which contains the JCL. This refers to the **JCL location**. For possible members, see ***List of JCL Locations***. For a job of the type **Subnetwork** (NET), this column contains a letter (for example, (D)) denoting the subnetwork **activation mode**.	

Selecting a Range of Job Definitions to be Listed

To select a range of job definitions to be listed

1 In the **object workspace** select **Owner** > *owner name* > **Network Master** > *network name* > **Job Master**.

2 From the context menu, choose **Filter** ( ).

A **Filter** window opens.

For further information, refer to ***Filtering Objects*** in the section *Common and Global Functions*.

Available Functions: Job Master

To list all functions available for a Job Master metanode

- In the object workspace, select a **Job Master** metanode and open the context menu.

The following functions are available:

Function	Symbol	Description
List		See ***Listing Jobs***.
New		See ***Adding a Job Definition***.
Refresh		See ***Refreshing Object Lists***.
Filter		See ***Filtering Objects***.
Export		See *Exporting Objects* in the *Import/Export Functions* documentation.

To list all functions available for a Job Master instance

- In the object workspace, select a **Job Master** instance and open the context menu.

The following functions are available:

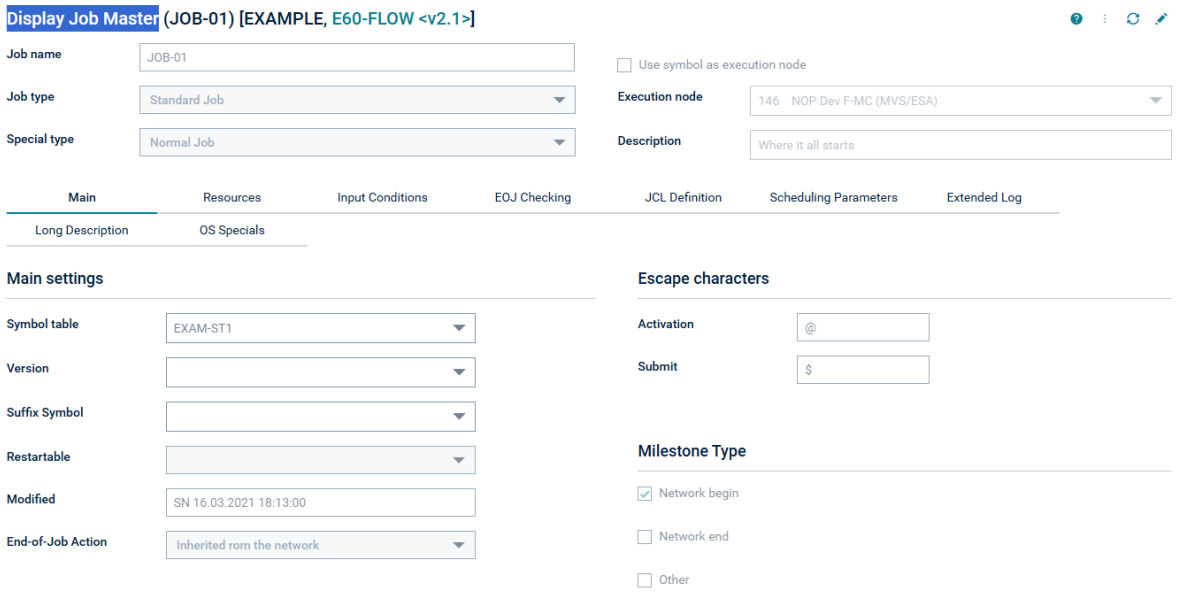
Function	Shortcut	Description
Open		See *Displaying and Modifying a Job Definition*.
Open Subnetwork	---	Open the **Maintenance Network Master** window for the network that is linked to a job of the type **Subnetwork** (`NET`).
Zoom Subnetwork	---	Open a list of jobs associated with the network to which a job of the type **Subnetwork** (`NET`) is linked. See also *Listing Jobs of a Subnetwork*.
Subnetwork Diagram	---	This function is used for navigation purposes. It opens the network diagram for the network linked to a job of the type **Subnetwork** (`NET`). You can work with the diagram as described in *Viewing and Maintaining a Job Network Diagram* in the section *Network Maintenance*.
Delete		See *Deleting a Job Definition*.
Copy data		See *Copying Objects*.
Usable Symbol Tables	---	Open the **Usable Symbol Tables** window to view or modify symbol table definitions available for the job. See also *Listing Usable Symbol Tables* in the section *Symbol Table and Symbol Maintenance* for details.
Browse Master JCL	---	See *Displaying Master JCL*.
Edit Master JCL	---	See *Editing Master JCL and Natural Sources*.
Activate Job		See *Activating a Single Job Manually*.
Pre-generate Active JCL	---	See *Pregeneration of Active JCL*.

Displaying and Modifying a Job Definition

To view or modify a job definition

1 In the **object workspace**, select **Owner** > *owner instance* > **Network Master** > *network name* > **Job Master** > *job master instance* and choose **Open** ( ) from the context menue.

A **Display Job Master** dialog is displayed such as in the following example:



2 Choose **Close** (), to close the dialog.

Or:

Choose **Edit** ( ) to switch to a **Maintain Job Master** dialog.

Edit the fields you want to change and choose **OK**. The fields are explained in *Fields: Job Definition (Master)*.

This section covers the following topics:

- Fields: Job Definition (Master)

Fields: Job Definition (Master)

The fields in the **Job Master window** are described in the following table.

Note: The fields and tabbed pages available depend on the **Job Type**, the **Special Type** and the operating system of the **Execution Node** specified for the job.

Field/Tabbed Page	Description
Job name	Job name. Entire Operations recognizes a job by this name. It can, but need not be the same name as the JCL source used by the job (the operating system recognizes a job by its JCL name). **Note:** Job names should start with a character. Job names starting with a digit are tolerated, but are not recommended.

Field/Tabbed Page	Description
Description	Short description of the job. The maximum input is 50 characters. This text appears in the list of jobs in the **Job Master List** window. You can add a long description for the job: see *Writing and Viewing Online Documentation for a Job*.
Job type	Job type as defined to Entire Operations. For further information, see the section *Job Types*.
Special type	You can select one of the following types from the list box: ■ **Normal Job** Normal job without special condition. ■ **Cyclic execution** (This is a replacement and extension of the former `CYC` **(cyclic) job type**.) Execution at certain intervals. If this type is specified, a cyclic interval must be defined in the job schedule parameters. This is checked during job activation and before job start. Each job type can be provided with this attribute. **Note:** You can use the reserved condition `P-STOPCYC-`*`jobname`* to interrupt a cyclic job execution loop. See also **reserved condition names** in *Restrictions for Condition Names*. ■ **Execute as dummy** Execution as a **dummy job**. This allows you to prevent the job from being executed without having to delete the definition. You can also set this type in an active job before a repetition. If this value is deleted in an active job, then the active JCL is generated, provided that it is already present. However, this does not have a resubmission of the job as a consequence. ■ **Stop Job** Job type **Started Task**: Stops a started task. Job type **Windows Service**: Stops a Windows service. ■ **Recovery Job** For a recovery job. You must also use this special type for jobs that are later activated with the job import **API** `NOPUJI`*`x`*`N` (see the section *API Routines*). **Note:** Recovery jobs are not activated during a normal job network activation. They are activated only if they are defined for another job and if a certain event makes a recovery necessary.
Execution node	Node on which to submit the job. The default is the value specified in the job network definition (see the field **Execution Node** in the section *Network Maintenance*). You can enter a different node for the job here.

Field/Tabbed Page	Description	
	You can define the execution node as a symbol if the **Use symbol as execution node** option (see below) is selected. For details, see ***Symbols in Node Definitions*** in the section *Symbol Table and Symbol Maintenance*.	
Use symbol as execution node	Select this check box if you want to use a symbol for the **execution node** (see above).	
Main page:		
Symbol table	Name of the symbol table to be referenced for substitution of variables in dynamic JCL. Select a name from the list. Alternatively, you can enter a new name to define a new symbol table. See ***Displaying and Modifying a Symbol Table*** in the section *Symbol Table and Symbol Maintenance* for more information. **Note:** If no symbol table is defined on the job level, the symbol table on the network level (if defined) is activated during **network activation or job activation** (if defined).	
Version (Symbol table)	Version of the symbol table to be used. Reserved names (are replaced).	
	`current`	Current version for the activation date or determination date.
	`nv`	Version of the using network.
	`svn`	Symbol table version of the using network.
Suffix symbol	(Optional field) If multiple instances of the job are to be activated in parallel, this field must contain a symbol name. The symbol will be searched in the **symbol search hierarchy** current at activation time. The symbol must contain the suffixes to be appended to active job names and output conditions. The sum of the lengths (prefix+suffix) must not exceed the maximum field lengths for job names or conditions. For example, if the job master name is `PARA`, and the symbol contains `001`, `003`, `012`, then the active jobs `PARA001`, `PARA003` and `PARA012` are created. If the suffix symbol value is not defined in a network and the network has a calling network (if it is a subnetwork), then the search for the suffix symbol is performed upward up to the highest level. If **Suffix symbol** does not contain any values, the job is activated as a temporary dummy job. For information about the definition of multiple symbol values, see ***Defining Multiple Symbol Values*** in the section *Symbol Table and Symbol Maintenance*. For information about the assignment of multiple symbol values to **Suffix symbol**, see ***Symbol Replacement with Multiple Symbol Values***.	
Restartable	Only applies to BS2000. Job can be restarted automatically. Possible selection options:	

Field/Tabbed Page	Description

	■ **Restart**
	The job is to be restarted without any recovery after a system crash.
	■ **No restart**
	No automatic restart after a system crash.
	■ **BS2000/RESTART**
	The job is to be restarted when repeated. SYSOUT files are not renamed.
	Automatic SYSOUT rerouting can be delayed with the following line right after the `/LOGON` instruction:
	`/REMARK EOR-SYSOUT-DIRECT=LATER`
	Later, the following line must appear:
	`/REMARK EOR-SYSOUT-DIRECT=NOW`
	This allows user-defined actions before using a SYSOUT file.
	Additional Restart Criteria:
	Job submission time must be before the last IPL time and the job must not be terminated. The field **Monitor JV** must contain `$R`. The system session number at the job check time must be different from the session number at submission time.

Modified	User ID, date and time of the last modification of the job definition.
End-of-Job Action	Job output conditions. Possible selection options: ■ **No impact on the job result** Errors during End-of-Job action processing have no impact on the job result. ■ **Errors impact the job result** Errors during End-of-Job action processing have an impact on the job result. For all output condition(s) defined for events at the job step level, the default checks **All Checks ok** and **Any Check not ok** are performed after all End-of-Job actions are performed. This includes End-of-Job action (EJA) exits. Any error that occurs in the End-of-Job action processing then causes the setting of the conditions defined for `job not ok`. If the job was already set to `not ok`, it remains `not ok`, regardless of the definition here. **Note:** 1. Setting this option can cause a longer elapsed time of a network, because the condition setting waits for the termination of the other End-of-Job actions. 2. The network level settings are overridden by job settings. 3. This option does not change the settings of any conditions defined for any individual events.

Field/Tabbed Page	Description
	■ **Impact as defined for the network** If the field is empty, the job uses the **End-of-Job definition** from its network. See also the field **End-of-Job Action** in the network master definition.
Milestone types	Milestone job types. You can set single or multiple milestone attributes (types) for a job that performs an important task within a job network, such as tracking significant events or controlling the successful completion of other jobs. Possible selection options for user-defined milestones: ■ **Network Begin** Milestone job performs as the first network job. ■ **Network End** Milestone job performs as the last network job. ■ **Other** Milestone job performs in no particular job order. Possible values for system milestones dynamically set by Entire Operations during job submission (applies to active jobs only): ■ **First Job** Milestone job performs as the first network job. ■ **Last Job** Milestone job performs as the last network job. You can use the application programming interface `NOPUMI1N` (see the section *API Routines*) to define your own milestones. For further information on milestone jobs, see the section *Event Store Milestones* in the *System Automation Tools* documentation.
Escape Characters: **Activation**	This is the **activation escape character**. It is a prefix for Natural code lines and symbols to be replaced at activation time. If you change this character, dynamic JCL can become invalid. See also ***Symbol Escape Characters*** (*Symbol Maintenance*) and ***Notes about Escape Characters***.
Escape Characters: **Submit**	This is the **submission escape character**. It is a prefix for symbols to be replaced at submission time. If you change this character, dynamic JCL can become invalid. Default is the dollar sign (`$`). See also ***Symbol Escape Characters*** (*Symbol Maintenance*) and ***Notes about Escape Characters***.
Other pages:	

Field/Tabbed Page	Description
Resources	This tabbed page is used to define a prerequisite resource for a job master. The fields and options on this page are explained in *Handling Prerequisite Resources for a Job*.
Input Conditions	This tabbed page is used to define input conditions for a job master. The fields and options on this page are explained in *Defining and Managing Job Conditions*.
EOJ Checking	This tabbed page is used to add, modify and delete an End-of-Job check for a job master and define a user message for the check. Furthermore, you can define an output condition for the End-of-Job check. The fields and options on this page are explained in *Defining and Managing End-of-Job (EOJ) Checking and Actions*.
JCL Definition	This tabbed page is not available for all types of jobs. The fields and options on this page are explained in *Defining JCL for a Job*.
Scheduling Parameters	This tabbed page allows you to define a schedule for the job. The fields and options on this page are explained in *Scheduling a Job* in the section *Schedule Maintenance*.
Extended Log	This tabbed page allows you to define additional (extended) information for the system log. The fields and options on the page are explained in *Defining Extended Log Information for a Job*.
Long Description	This tabbed page provides the option to write a long descriptive text for a job, in addition to the short job description you can enter in the **Description** field. See also *Writing and Viewing Online Documentation for a Job*.
Subnet	This tabbed page is only available for jobs of the type **Subnetwork** (`NET`). The fields and options on this page are described in *Fields: Job Type Specific Execution Features* in the section *Defining a Subnetwork*.
OS Specials	This tabbed page is not available for all types of jobs. The fields and options available on this page are described in *Fields: Operating System Specific Execution Features*.

Note about Escape Characters

The job escape characters are used for **symbol replacements** and **Macro JCL**.

When a job is created or when an escape character is reset, the default value is taken in the following order from:

1. **Network defaults (operating system specific)** if defined for the current operating system;
2. **Network definition** if non-blank;
3. Global operating system specific defaults for z/OS, BS2000, UNIX and Windows (*Administration* documentation).

If you change the execution node to another operating system, you might want to delete the escape character definitions, so that they will be newly preset with the operating system specific defaults for the new operating system. See also *Symbol Escape Characters* in the section *Symbol Maintenance*.

Creating a Job Definition

When defining a new job, you can either enter all required attributes individually, or copy all attributes from an existing job master including all input conditions and End-of-Job activities defined for the job.

This section covers the following topics:

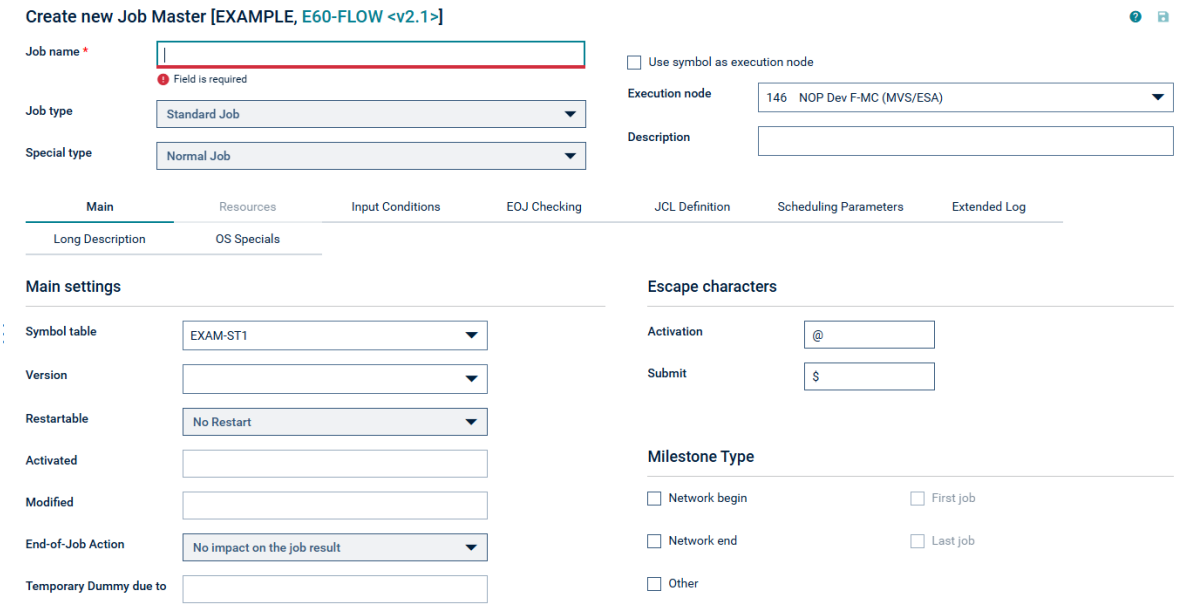
- Adding a Job Definition
- Copying Job Definitions

Adding a Job Definition

To add a job definition

1 In the **object workspace**, select **Owner** > *owner instance* > **Network Master** > *network name* > **Job Master** and choose **New** (+) from the context menu.

A dialog such as in the following example is displayed:



The dialog contains fields to specify the job, see *Fields: Job Definition (Master)* for details.

2 Choose **Save** ( ).

Your input is saved.

Copying Job Definitions

You can copy a job within the current network or between networks.

Prerequisites for copying are:

- You must be allowed to perform the copy function; see the settings of the **write** permissions in the *Network Maintenance Functions* of your user profile (*Administration* documentation).
- You must be granted access to the networks of a specified target owner.
- The target network must exist; you cannot create a new network with the copy function.

To copy a job definition

- In the object workspace, select the job you want to copy from **Job Master** node and proceed as described in *Copying Objects*.

Writing and Viewing Online Documentation for a Job

You can add a short description of a job when defining a job in the **Maintenance Job Master window**. This short description appears in the list of jobs in the **Job Master List window**.

If you wish to add more online documentation for a job, proceed as described in the following instruction.

As an alternative to a descriptive text, you can also enter an URL address that be used as a hyperlink to open another page inside or outside your Windows application.

This hyperlink also works when you display the long description of a job.

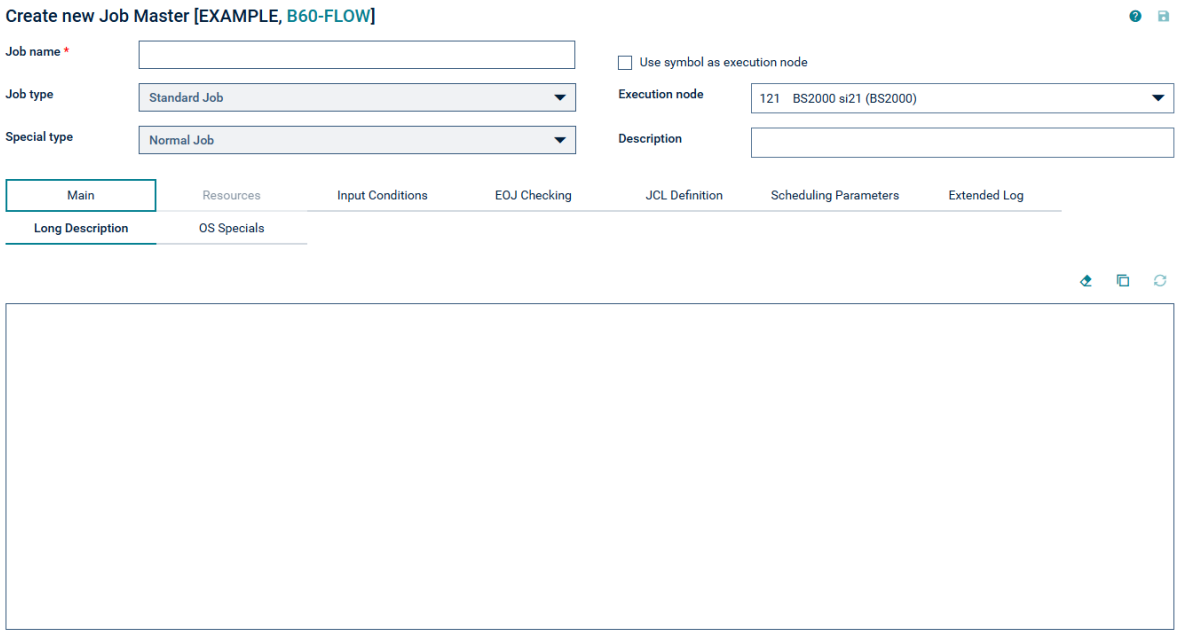
To create, view or modify a long description or hyperlink (URL) reference

1 In the **Maintenance Job Master window**, open the tabbed page **Long Description**.

2 Enter an explanatory text or a hyperlink by choosing either of the following options.

(You can choose **Clear** ( ) if you want to remove all entries on the **Long Description** page.)

Enter an explanation concerning the job master or replace existing text as shown below:



Or:

Enter a URL address that starts with `http://` or `https://`.

The URL must not contain any blank characters but it can extend over several lines. The lines are then concatenated without trailing blanks.

You can use one URL only.

Choose **Open URL** to open the default browser associated with your Windows application and view the destination page.

(This button only appears when you enter `http://` or `https://`.)

3 When you are finished, choose **Save** ( ) .

Once online documentation is written, it can be read by any user who is authorized to access the network to which the job belongs.

You can also display or print online documentation by using the **Reporting** function described in the section *Reporting*. If long job descriptions exist, they are included in the **Network Description (detailed)** report as indicated in the *Example of Network Description (detailed)*.

Deleting a Job Definition

Note: Deleting a job definition also deletes all definitions made at the job level, including its input conditions and **End-of-Job checking and actions**. The original JCL is not deleted.

To delete a job definition

- In the **object workspace**, select **Owner** > *owner instance* > **Network Master** > *network name* > **Job Master** > *job master instance* and choose **Delete** ( ) from the context menue.

  A dialog opens where you have to confirm the deletion.

Displaying Job Dependencies

The **Network Diagram** function allows you to see which jobs must run before a specific job can start, and which jobs run after it.

You can trace job dependencies through whole networks and display cross-network job links.

Displaying Job Dependencies for a Specific Job

To view job dependencies in a network

1. In the object workspace, select a job from the **Job Master** or **Job Active** node.
2. Open the context menu and choose the **Network Diagram** function.

   Depending on the node selected, a **Diagram Network Master** or **Diagram Active Run** window (see the examples in the section *Network Maintenance*) opens with a graphical view of the selected network.

   The symbols used to represent the network components and the functions available from the context menu in the diagram, are described in ***Explanations of Diagram Symbols*** and ***Maintenance Functions for Diagrams*** in the section *Network Maintenance*.

   In a complex network, you can use the **Find** function to locate the required job.

Activating a Single Job Manually

Just as you can perform an ad-hoc manual activation of a job network at any time to allow it to run outside of its scheduled times, you can also manually activate any job at any time, regardless of its position within a job network.

The activated job is assigned its own run number (in sequence with the network run number) and, if you do not change the date or time, is submitted immediately by the Entire Operations Monitor. If the job contains symbols as placeholders for variables, symbol prompting takes place as described in the section *Symbol Prompting during Network or Job Activation*. If other symbol tables are required, they are prompted too.

The differences between the manual activation of a single job and the manual activation of a job network are that when a single job is activated:

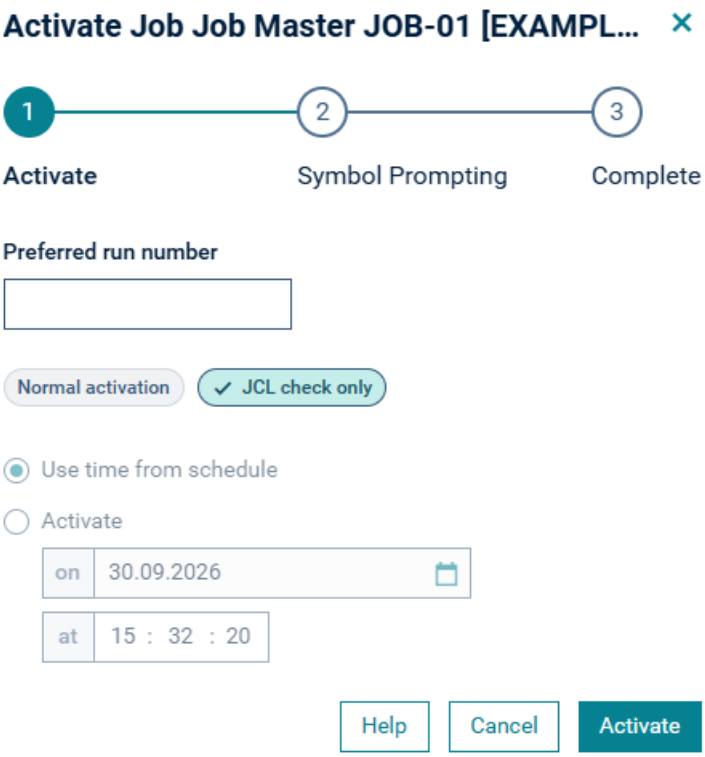
- No check for defined input conditions for the job is performed;
- No output conditions are set by the job.

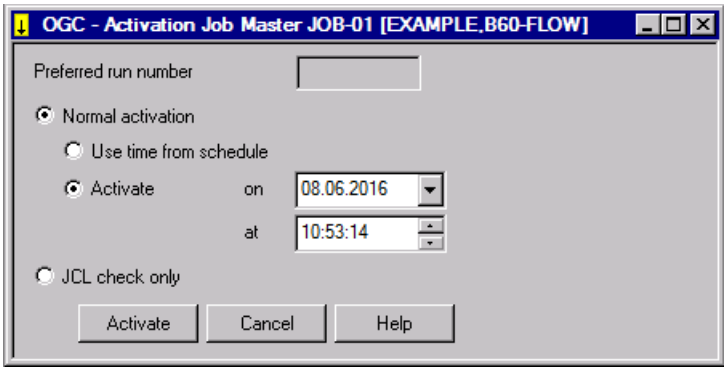
All other End-of-Job actions defined for the job are executed.

To activate a job manually

1 In the object workspace, select a **Job Master** instance and choose **Activate Job** from the context menu.

An **Activation Job Master** window like the example below opens:





The window title indicates the job (here: `JOB-01`) to be activated and the owner and network (here: `EXAMPLE, B60-FLOW`) of the job.

2 Make your definitions. The fields contained in the window are explained in *Fields: Job Activation*.

3 When you are finished, choose **Activate**.

The job is activated and listed in the **Active Run** node.

This section covers the following topics:

- Fields: Job Activation

Fields: Job Activation

The fields of the **Activation Job Master window** are described in the following table:

Field	Description
Preferred Run Number	If input is allowed for this field, you can enter the run number you want to be used for the activated job. Run numbers for further job activations increment from this number. If the requested run number is in use, Entire Operations assigns the next free number. If this field contains a zero (0) or no value, Entire Operations (as usual) automatically assigns a run number. Field input is allowed or not allowed (default) in the network default settings: see the option **Allow setting of preferred run number during activation** in the section *Defaults for Network Options* in the *Administration* documentation.
Normal activation: Use Time in Schedule	Select this option to activate the job within the **defined time frames** (earliest start, latest start, deadline), even if the current date is not a scheduled date for this network. With this option, you can force the same time dependencies as if the network was scheduled and activated automatically. For more information on network time frames, see *Activating a Job Network Manually* in the section *Network Maintenance*.
Normal activation: Activate at Date/Time	Shows the current date and time. You can modify date and time to force activation at any time on any date. There is no limitation for future date and time settings. **Note:** No symbol prompting is performed for later activations.
JCL Check only	Only a JCL check is performed for the job. See *JCL Check only* for further reference.
	If this option is not selected: **Normal activation** is used. This is the default.

Defining Extended Log Information for a Job

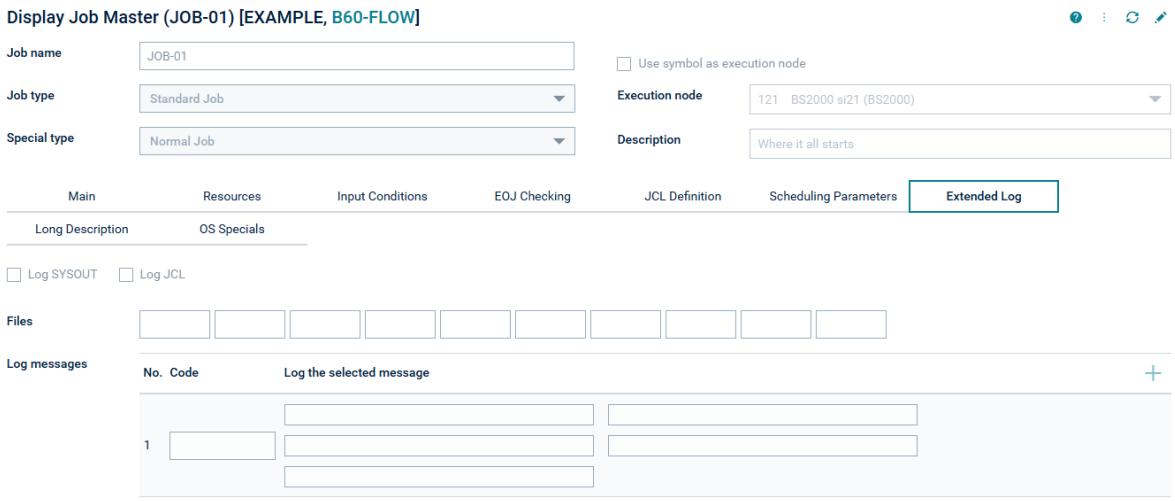
Entire Operations allows you to define which information is to be logged in addition to the Entire Operations standard system log.

Additional (extended) log information can be viewed as described in *Displaying Extended Log Information* in the section *Log Information*.

Extended log information is defined at the job level.

To define extended log information

1 In the **Maintenance Job Master window**, open the tabbed page **Extended Log**:



2 Make your definitions. The input fields and options are explained in *Fields: Extended Log*.

3 Choose **OK** to save your definitions.

The logged information can be viewed after job termination using the **Browse Log** function described in the section *Log Information*.

This section covers the following topics:

- Fields: Extended Log

Fields: Extended Log

The input fields of the **Extended Log page** are described in the following table:

Field/Column	Description	
Log SYSOUT	Specifies logging of job SYSOUT files after job termination. Possible check box settings:	
	checked	Logs SYSOUT.
	unchecked	No SYSOUT files are logged.
	SYSOUT logging can also be **defined as SYSOUT action**. If the SYSOUT file or spool data set exceeds a given line limit, the log display is truncated: see **SYSOUT line limit** in *Defaults for Network Options* in the *Administration* documentation.	
Files	(z/OS only) Input valid only if **Log SYSOUT** is specified. Enter number(s) of SYSOUT file(s) to be logged.	
Log JCL	Specifies logging of JCL after job termination. This is useful for dynamically generated JCL or if JCL is frequently modified for different job runs. Possible check box settings:	
	checked	Logs JCL.
	unchecked	No JCL is logged.
	See also ***JCL Log***.	
Log Messages	Specifies messages to be logged if returned from a SYSOUT file (BS2000, UNIX and Windows) or JES SM spool data set (z/OS). Enter a message code (for example, `IEF285`) or a code prefix in a field under the **Code** column. If used as a code prefix, it specifies a range of codes to be logged. For example: `IEF` logs all messages that begin with IEF. Asterisk (*) notation is not required.	
Log the selected message if it contains one of the following strings	Specifies additional selection strings for the message entered and selected in the **Code** column. The message is logged if no additional string is specified, or if at least one specified string is found in the message.	

JCL Log

The JCL log is generated from the following sources:

BS2000	From the ENTER file. (The ENTER file is deleted after copying into the Entire Operations Log.)
z/OS	If Entire Operations runs on a mainframe: from the SYSOUT (JES output). If Entire Operations does not run on a mainframe: from the active JCL.
UNIX	From the effectively submitted shell script.
Windows	From the effectively submitted BAT file or PowerShell script.

26 Defining Job Types and Job Execution Features

- Available Job Types 230
- Defining Job Type Specific Execution Features 232
- Defining Operating System Specific Execution Features 235

You define a job according to the **job type** you require and the operating system on which your Entire Operations environment runs.

The functions and specification options available to define a job depend on the following entries in the **Maintenance Job Master window**:

- The job type entered in the **Job Type** field,
- The job type entered in the **Special Type** field, and
- The operating system of the node entered in the **Execution Node** field.

Depending on the job type, the special type and the operating system selected in the **Maintenance Job Master window**, additional pages are provided to define individual execution features for the job.

This section describes the types of jobs you can define in Entire Operations and the type specific features provided to execute a job.

Available Job Types



Note: Only the job types `JOB`, `SRV` and `STC` result in operating system jobs when submitted.

You can define the following job types to Entire Operations:

Job Type	Description
`DAT` **Text File**	A UNIX, Windows or `DAT` text file. Jobs of type `DAT` (data file generation) can be used to create text files in the same way as active JCL is generated. Instead of a submission, the created text is just being written to a text (data) file. Symbol replacement, symbol functions, text block inclusions can be used like in other JCL. If the `DAT` output file is to be created on a mainframe node, no trigraph encoding is performed during JCL loading. If the `DAT` output file is to be created on a UNIX or Windows node, trigraph encoding is performed during JCL loading, and trigraph decoding is performed during `DAT` file writing. For more information, see *Trigraph Encoding for JCL Submission on UNIX and Windows*.

Job Type	Description
DUM **Dummy Job**	A dummy job has no JCL and is not submitted to the operating system. You can use dummy jobs to set input conditions with a time range, to provide a time window for other jobs or as placeholders for future or seldom used jobs, etc. You can define an estimated running time for a dummy job (see the field **Estimated Elapsed Time**). The dummy job is then executed for exactly this amount of time without burdening the system. **Execution node:** Even if a dummy job itself does not need it, it may be necessary to define an **execution node** with a **Submit User ID**. This is the case if any prerequisite check, End-of-Job check or End-of-Job action uses an Entire System Server node. The Monitor determines whether a node access is required, and performs Entire Systems Management logons only if they are necessary for any action. See also ***Using a Dummy Job***.
JOB **Standard Job**	This includes all standard operating system jobs with their own JCL. Jobs of the type JOB (**Standard Job**) are submitted as batch jobs. The JCL of JOB-type jobs can contain symbols as placeholders for variables defined in the symbol table specified for the job. These symbols are replaced by their current value at job activation or at job submission time. See also ***Symbol Replacement in JCL***. If the job is executed on a UNIX or Windows node, command line parameters can be passed by the symbol CMDLINE-*job*. See ***Reserved Symbols for Command Line Parameters*** in the section *Symbol Table and Symbol Maintenance*.
NAT **Natural Program**	This is a Natural subprogram. It has no JCL and is executed directly by the Entire Operations Monitor. A subprogram, for example, is used to code **user exit routines** (see the relevant section) used by a job. See also ***Editing Master JCL and Natural Sources***.
NET **Subnetwork**	This job type enables you to execute a complete network within a main network. For detailed information on defining subnetworks, see the section ***Defining a Subnetwork***.
SRV **Windows Service**	With this job type, you can define a service under Windows. A service must first be defined in Windows before it can be started. For further information, see: ■ ***Job Execution Specials for Job Type SRV*** ■ ***Passing Command Line Parameters*** Stopping a Windows service: Create a copy of the main **Windows Service** job, and define the **special type Stop Job** described in *Fields: Job Master Definition*.

Job Type	Description
`STC` **Started Task**	A started task is a special type of batch job on z/OS. Started tasks have static JCL which can reside in any PROCLIB. Only PDS is a valid JCL location for this job type. Entire Operations can activate and process started tasks like any other job type. See also ***Passing Command Line Parameters***. Stopping a started task: Create a copy of the main **Started Task** job, and define the **special type Stop Job** described in *Fields: Job Master Definition*.
`WPS` **Windows PowerShell Script**	Applies to Windows only. This job type is for Windows PowerShell Scripts (with extension `.ps1`). The Windows PowerShell (powershell.exe) is used as the command interpreter instead of the old Windows command interpreter. The script must conform to the PowerShell syntax. Command line parameters can be passed by the symbol `CMDLINE-job`. See ***Reserved Symbols for Command Line Parameters*** in the section *Symbol Table and Symbol Maintenance*.

Defining Job Type Specific Execution Features

To define job type specific execution features

- In the **Maintenance Job Master window**, open the extra page that appears when selecting one of the following job types in the **Job type** field:

  For the job type **Subnetwork** (`NET`), the tabbed page **Subnet** is provided.

  For the job type **Text File** (`DAT`), the job type dependent fields are contained on the tabbed page **OS Specials**.

  All type specific fields are described in ***Fields: Job Type Specific Execution Features***.

This section covers the following topics:

- Fields: Job Type Specific Execution Features

Fields: Job Type Specific Execution Features

The job types for which type specific fields are provided on additional pages of the **Job Master** window including the **OS Specials page** are described in the following table.

Job Type	Field	Description	
NET **Subnetwork** **(tabbed page Subnet)**	**JCL Node**	Node where the JCL of the job (if defined) is located.	
	Subnetwork owner	Owner of the subnetwork. Enter a name or select a name from the drop-down list box.	
	Subnetwork name	Name of the subnetwork. Enter a name or select a name from the drop-down list box.	
	Subnetwork version	Version of the subnetwork if defined.	
	Activation mode	Activation time: see ***Time of Activation of a Subnetwork*** in the *System Overview*. Possible selection options:	
		At activation time	At the moment of the activation of the calling job.
		At submission time	At the moment of the start of the calling job.
		Use global defaults	Use global default values as set in the Entire Operations defaults for **Network Options** (see the *Administration* documentation).
	Append multiple suffixes	Select this check box to append the suffix of the calling multiple job to the name of the subnetwork. The resulting name must not have more than 10 characters. Otherwise, the name of the active job network is the same as the name of the job master network. Uniqueness is ensured by different run numbers.	
	Use subnetwork time frames	Possible check box settings:	
		unchecked	Use the time frame from the calling job of the type NET (default).
		checked	Use the time frame defined for the subnetwork itself.

Job Type	Field	Description		
DAT **Text File** **(tabbed page OS Specials)**	**Submit User ID**	The user ID to be used to write a DAT text output file.		
	Target Location	**(No JCL required)**	Generate active JCL only.	
		SAM or ISAM file	BS2000 text file.	
		LMS library	BS2000 LMS file.	
		Natural source	Natural source object.	
		Macro Natural source	Natural source object for macro JCL.	
		Partitioned data set	z/OS partitioned data set.	
		Text file	UNIX or Windows text file.	
		Target overwrite	This option is available only for some target locations. Possible check box settings:	
			unchecked	Append the output to target file.
			checked	Overwrite target file with the output.
	Target File	Text file to which the DAT output is to be written. Symbol replacement is possible. Additionally to symbols with activation escape characters, also symbols with **submission escape characters** are resolved. Both resolutions are performed at submission time.		
	Member	The target locations LMS, NAT and PDS also require the definition of a target file member. Symbol replacement is possible. Additionally to symbols with activation escape characters, also symbols with **submission escape characters** are resolved. Both resolutions are performed at submission time.		
	input field/selection box for member type	Additional selection box or input field for BS2000. BS2000, LMS: The LMS member type. For possible selection options, see the description of the **Type** field in *Fields: Operating System Specials for JCL*.		
	SRV **Windows Service**	**Service Name**	The internal name of a Windows service. **Note:** You can obtain the internal name of a Windows service by using the following Windows	

Job Type	Field	Description	
	(tabbed page OS Specials)		command line command: `sc getkeyname <external-name>`.

Defining Operating System Specific Execution Features

To define operating system specific execution features

- In the **Job Master** window, open the tabbed page **OS Specials** .

  The input fields provided depend on the operating system of the execution node.

  (The **OS Specials** page is only available if execution features can be specified for the operating system of the selected execution node.)

  All operating system specific fields are described in *Fields: Operating System Specific Execution Features*.

  In general, these fields correspond to the operating system specific fields provided for network maintenance.

This section covers the following topics:

- Fields: Operating System Specific Execution Features

Fields: Operating System Specific Execution Features

The operating system specific input fields provided for job execution are described in the following table.

Operating System	Field	Description
BS2000	**Default User ID**	Valid logon user ID defined for BS2000. Unqualified file and job variable names in this job definition are prefixed with this BS2000 user ID. File and job variable names are not prefixed with the user ID if the name contains the **activation escape character** in input conditions and End-of-Job actions. See also *BS2000: User ID Propagated to Variables and Files*. **Note:** The user ID TSOS can be defined only if the user defining the ID is working under TSOS.

Operating System	Field	Description
	Submit User ID	Jobs in BS2000 are submitted under this user ID by the Entire Operations Monitor. This ID has no meaning for other operating systems. **Note:** The user ID TSOS can be defined only if the user defining the ID is working under TSOS. In the Monitor standard values/submit user type in the access control system, certain checks of the **Submit User ID** can be defined. See also the field **Submit Security User Type** in the section *Monitor Defaults* in the *Administration* documentation. If this field is left blank, then the default user ID is inserted at job activation. **Symbol replacement** is performed in this field if the **activation escape character** or the **submission escape character** is used. See also the default setting **User ID Definition** (*Defaults for Other Settings*, *Administration* documentation), and the sections ***Operating System User IDs*** and ***Default User ID Determination***.
	Account Number	Account number to be used for the BS2000 **Submit User ID**. **Symbol replacement** is performed in this field if the **activation escape character** or the **submission escape character** is used. If you leave this field blank, the account number is taken automatically from the BS2000 JOIN entry (only if the Entire Operations Monitor is under TSOS or under the same user ID as the job).
	BS2000 Job Class	Valid job class for job submission defined for BS2000. Symbol replacement is performed in this field if the **activation escape character** is used.
	Job Priority	If not empty, this job priority is used during submission and overrides a possible setting in the `LOGON` statement. The default value on the network level is used for new job definitions.
	Run Priority	If not empty, this run priority is used during submission and overrides a possible setting in the `LOGON` statement. The default value on the network level is used for new job definitions.
	Share SYSOUT	Make BS2000 SYSOUT file shareable. If you select this check box, the internal temporary SYSOUT file can be accessed from other BS2000 user IDs.
	Submit Password	Password for job submission. (Required only for BS2000 password encryption where submit node Entire System Server version is less than Version 3.4.1).

Operating System	Field	Description
	SYSOUT User ID	User ID under which internal SYSOUT files are created by Entire Operations. If you leave this field blank, the **Submit User ID** is used. Symbol replacement is performed in this field if the **activation escape character** or the **submission escape character** is used. See also the default setting **User ID Definition** described in *Defaults for Other Settings* in the *Administration* documentation.
	SYSOUT Cat ID	Enter the SYSOUT catalog ID. This is the catalog ID under which internal SYSOUT files are created by Entire Operations. This field is meaningful only if you specify a **SYSOUT User ID** different from the **Submit User ID**.
	Collect SYSLST	If you select this check box, and if the job contains a SYSLST assignment (like `/SYSFILE SYSLST=...` or `/ASSIGN-SYSLIST TO-FILE=...`), the SYSLST output is appended to the SYSOUT collection by Entire Operations. The SYSLST output is then also available in the SYSOUT online display.
	Monitor JV	Name of the BS2000 Monitor Job Variable to be used at job submission. If you leave this field blank, Entire Operations generates an internal unique name.
	Password	Password for the defined BS2000 Monitor Job Variable. Only alphanumeric passwords are supported.
z/OS	**Submit User ID**	In z/OS, the Entire Operations Monitor starts jobs under this user ID. You can only define this user ID if you are logged on to the executing node with the same user ID. Default: If this field is empty, the user ID from the last **Submit User ID** change is taken. Symbol replacement is performed in this field if the **activation escape character** or the **submission escape character** is used. In the Monitor standard values/job start user type in the access control system, certain checks of the job start user ID can be defined. See also the field **Submit Security User Type** in the section *Monitor Defaults* in the *Administration* documentation. See also the default setting **User ID Definition** (*Defaults for Other Settings*, *Administration* documentation), and the sections ***Operating System User IDs*** and ***Default User ID Determination***.
UNIX and Windows	**Submit User ID**	The Entire Operations Monitor sets the user ID of the shell script to this value. The user ID is converted to lower case automatically. Symbol replacement is performed in this field if the **activation escape character** or the **submission escape character** is used.

Operating System	Field	Description	
		In the Monitor standard values/job start user type in the access control system, certain checks of the job start user ID can be defined. See also the field **Submit Security User Type** in the section *Monitor Defaults* in the *Administration* documentation. See also the default setting **User ID Definition** (*Defaults for Other Settings*, *Administration* documentation), and the sections ***Operating System User IDs*** and ***Default User ID Determination***.	
	Use Symbol as SYSOUT Node	Select this check box if you want to use a symbol for the **SYSOUT node** (see below).	
	Submit Group	UNIX: If this field is empty, the user's default UNIX group (from `/etc/passwd`) is used. Otherwise, this field must contain one of the UNIX groups, which is visible in the output of the UNIX groups command. Windows: You can specify the user's domain in this field. Symbol replacement is performed in this field if the **activation escape character** or the **submission escape character** is used.	
	SYSOUT Node	Only if SYSOUT is to be copied to BS2000: Node to which internal SYSOUT files are to be copied by Entire Operations. The SYSOUT node must be different from the **execution node**, and it must be a BS2000 node. You can define the SYSOUT node as a symbol if the **Use Symbol as SYSOUT Node** option is selected. For details, see ***Symbols in Node Definitions*** in the section *Symbol Table and Symbol Maintenance*.	
	SYSOUT Cat ID	Only if SYSOUT is to be copied to BS2000: Catalog ID under which internal SYSOUT files are copied by Entire Operations.	
	SYSOUT User ID	Only if the SYSOUT is to be copied to BS2000: User ID under which internal SYSOUT files are copied by Entire Operations. See also the default setting **User ID Definition** (*Defaults for Other Settings*, *Administration* documentation), and the sections ***Operating System User IDs*** and ***Default User ID Determination***.	
	Command Line obligatory	Possible check box settings:	
		checked	The existence of the symbol `CMDLINE-job` is obligatory. If it is missing or empty, the job is not submitted. See ***Reserved Symbols for Command Line Parameters*** in the section *Symbol Table and Symbol Maintenance*.
		unchecked	A command line is not obligatory for the job (default).

BS2000: User ID Propagated to Variables and Files

The BS2000 user ID specified in the **Default User ID** field of a job master or job active definition is propagated to all job variables and file names linked to an EOJ checking and/or input condition defined for the respective job.

Examples:

Old Default User ID	New Default User ID	New User ID in Job Variable/File Name
No user ID (blank field)	`ID-A`	`ID-A` (provided the old user ID is also blank)
`ID-A`	`ID-B`	`ID-B` (provided the old user ID is also `ID-A`)
`ID-B`	No user ID (blank field)	No value changes, the old user ID is retained.

Restriction:

The ID is not propagated if the job variable or file name contains the job **activation escape character**.

Note: For jobs with the execution node BS2000, the **Default User ID** field on the tabbed page **OS Specials** is protected after a change. If you want to change the ID again, press **Apply** or **OK**.

27 Executing a Job as Dummy or Including Epilog Scripts

- Job Execution as a Dummy Job 242
- Job Execution Including an Epilog Script (UNIX and Windows) 242

Job Execution as a Dummy Job

The execution of a dummy job means that the job is running without job control and without its own action within Entire Operations. Dummy jobs can have an expected run time, which they will be waiting in the system. Dummy jobs will always terminate with the state `o.k.`.

For detailed information on dummy jobs, see ***Using a Dummy Job***.

Job Execution Including an Epilog Script (UNIX and Windows)

You can run an epilog script within the JCL frame of a UNIX or Windows job. The epilog script is called by the frame script and runs after the main script. Usually, the frame script calls the main script directly.

For detailed information on using and defining epilog scripts, see ***Reserved Symbol for Epilog Scripts (UNIX and Windows)*** in the section *Symbol Table and Symbol Maintenance*.

28 Using a Dummy Job

- Permanent Dummy Jobs 244
- Temporary Dummy Jobs 244
- Excluding a Job from Actual Execution 245
- Supervision of Running Jobs 246

The execution of a dummy job means that the job is running without job control and without its own action within Entire Operations. A dummy job is directly executed by the prerequisite checking routine. A dummy job always terminates with the state `o.k.`.

A dummy job is useful for several reasons:

- Testing the job flow when creating or changing a network.
- Finding erroneous jobs in a chain of concatenated jobs.
- Excluding or including a job from a chain of concatenated jobs without having to change job queues and adapting job definitions.
- Excluding jobs that execute tasks that are not frequently required to avoid overhead, balance workload and improve performance.

Permanent Dummy Jobs

Permanent dummy jobs are declared with the job type **Dummy Job** (`DUM`) in the job master definition.

Permanent dummy jobs execute with the estimated elapsed time, which is defined for the job. This does not block any queues within the Entire Operations Monitor.

Temporary Dummy Jobs

In principle, you can execute any type of job as a temporary dummy job. An active job that fulfills any of the conditions described under ***Dummy due to Condition*** is always executed as a dummy for a particular run.

Temporary dummy jobs with the job type **Standard Job** (`JOB`) have a master JCL definition (see ***Defining and Managing JCL for a Job***).

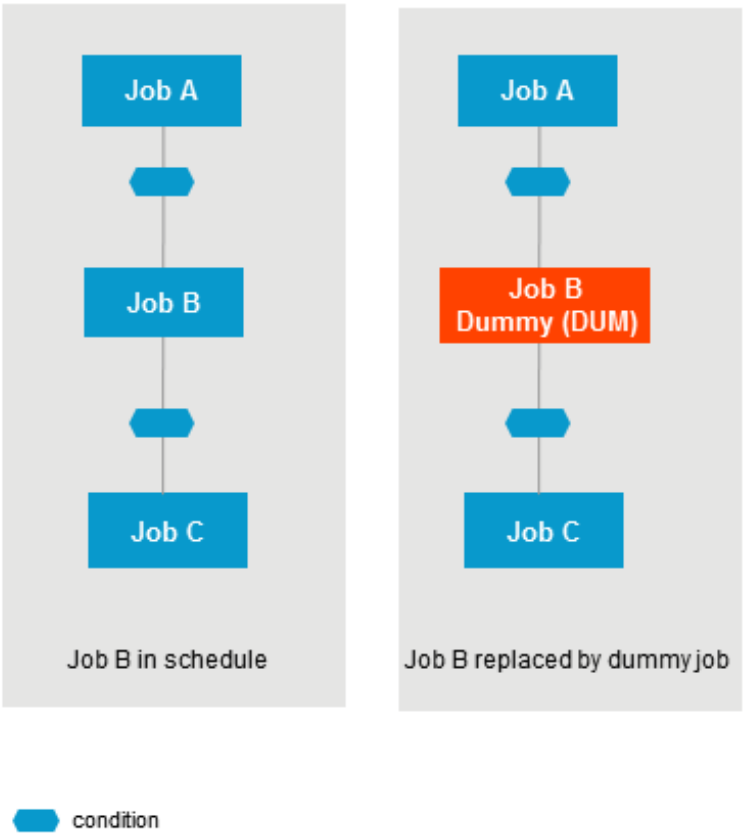
Temporary dummy jobs always have the execution time zero, regardless of which estimated elapsed time is defined for the job.

During its activation or during a prerequisite check, a job can be put into the state `temporary dummy job` for the following reasons:

Dummy due to Condition	Description
Definition	If the special type **Execute as dummy** (D) is set in the job master definition, the JCL definition can be maintained, but the execution as a dummy job can be forced.
Schedule	Depending upon the schedule of the network or upon a calendar, a job can be submitted as a temporary dummy job on particular days. For further information, see ***Maintaining Schedule Dependencies for a Job***.
Prerequisite	It can be defined that the existence or non-existence of an active prerequisite results in execution as a temporary dummy job. For further details, see ***Possible References for Input Conditions***.
Recovery Action	If a job is not to be submitted again in the case of a defined recovery action. For details, see ***Defining Recovery Actions***.
JCL check	Execution of the job control in the mode *JCL check*. Thus, jobs (see **Fields: Job Activation**) or Job Networks (see ***Fields: Network Activation***) are activated for the JCL check.
Empty JCL	If the loading process of the job control reveals that the JCL is empty, the job is submitted as a temporary dummy job. This state can result from the JCL generation by means of macro instructions, for instance.
Multiple Suffixes	If the activation is to take place by means of multiple suffixes: if the multiple-value symbol used does not contain any occurrences, then a dummy job is activated as a placeholder in the network during activation.

Excluding a Job from Actual Execution

You can exclude a job from actual execution on certain schedule or calendar days without disturbing the job network structure. If a job is not to be activated, it can be converted into a dummy job just for this run. For example:



You can use the **Schedule Dependency** feature to define job activation or input condition usage to be checked only on certain days. In this case, you can execute a job as a dummy job without changing any job definitions.

For information on using the **Schedule Dependency** feature, see *Maintaining Schedule Dependencies for a Job* in the section *Schedule Maintenance*.

Supervision of Running Jobs

Running jobs are checked as to whether they have exceeded their predefined deadline time.

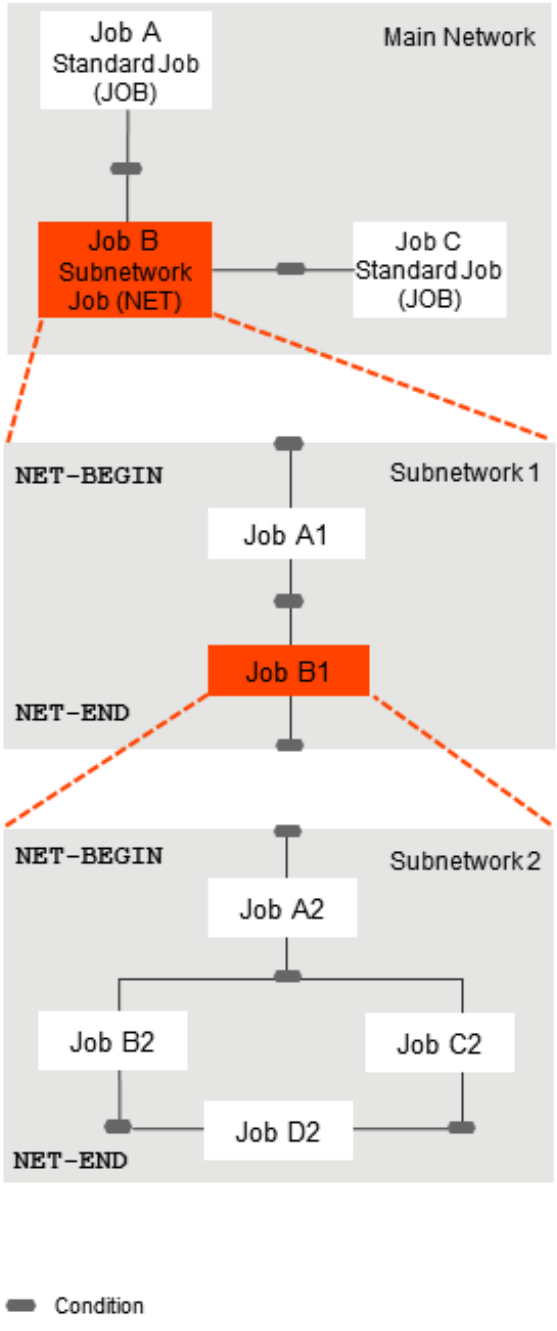
If this is the case, a message:

- is sent to all mailboxes linked to the network;
- is written to the log;
- appears in the **List Active Jobs**, **Active Job List** or **List Active Jobs Job Active** window.

29 Defining a Subnetwork

- Link to the Main Network 249
- Subnetwork Activation and Execution 250
- Restrictions for Subnetworks 251
- Adding or Modifying a Subnetwork Definition 252
- Listing Jobs of a Subnetwork 254

The job type **Subnetwork** (`NET`) allows you to define a whole network as a job. The subnetwork must already be defined. It must have an input condition `NET-BEGIN` and an output condition `NET-END` as illustrated in the following example:



The same subnetwork can be defined in different jobs of the main network.

The subnetwork can be activated together with the calling network (or job of the type NET) or at the moment the job of the type NET is started.

On activation, each active subnetwork is assigned a unique **run number**.

Link to the Main Network

The following reserved conditions are used to link a subnetwork to a main network and control the connection between the calling NET job and the subnetwork called.

Reserved Condition	Description
NET-BEGIN	You must specify NET-BEGIN to start the subnetwork and run the first job of the subnetwork.
NET-END	You must specify NET-END to run the last job of the subnetwork and indicate the end of the subnetwork.
NET-END-NOTOK	This condition is optional. If specified and fulfilled during subnetwork termination, the job is treated as ended not ok. If NET-END-NOTOK is specified in the subnetwork but NET-END is missing, the status of the NET job changes to Sub-NW executing - error indicated. In this case, the global message Job ended not ok is sent if the corresponding option is enabled in the **Global Messages for Events** settings (see the *Administration* documentation). If NET-END-NOTOK is reset in the subnetwork, the status of the NET job changes back to Subnetwork executing. In this case, the global message mentioned above is sent again.
NET-END-OK	This condition is optional. If specified at least once in your subnetwork and fulfilled during subnetwork termination, the NET job (that is, the whole subnetwork) is treated as ended ok. This has an impact on the release of resources, which are allocated on the network level. You can force the resource release by setting this condition. See also the **deallocation mode** set in the prerequisite resource definition.

Notes on Subnetwork Usage

1. If the condition NET-END-NOTOK is found together with NET-END, the calling job is set to the status not ok.
2. Jobs of the type NET can use all kinds of prerequisite checks and can have all **kinds of End-of-Job actions**.
3. The scheduling parameters of a job of the type NET are passed on to the subnetwork. See the section ***Scheduling a Job***.
4. The calling job of a subnetwork (job of type NET) should have an own symbol table definition.

Subnetwork Activation and Execution

A subnetwork is activated together with the main network. The subnetwork is assigned its own **run numbers**.

If the calling job of the type **Subnetwork** (NET) becomes a temporary dummy job due to any reason, no subnetwork activation is performed (if the subnetwork was not activated already).

Reserved symbols pass information to the jobs of the subnetwork about which job invoked the subnetwork. These symbols are described in ***Predefined Symbols for Subnetworks and Recovery Jobs*** in the section ***Symbol Table and Symbol Maintenance***.

If the subnetwork has a standard symbol table, these values are entered as symbols in the active copy of this table.

This enables you to trace, even through several steps, from where the subnetwork was invoked.

If the job of the type NET can be activated (satisfies all input conditions), the condition NET-BEGIN of the subnetwork is set and the subnetwork starts to work: the NET job is then executing. It remains in this status until the subnetwork has set the condition NET-END. Only at this point are the **End-of-Job checking and actions** performed which are defined in the invoking network.

If an execution error occurs in the subnetwork, the execution of the invoking network is also blocked. If the subnetwork sets the condition NET-END before its actual termination (or at its very start), the invoking network continues to run, and a parallel asynchronous execution of the networks is possible. However, the user him/herself then becomes responsible for later synchronizing the subnetwork with the invoking network.

This section covers the following topics:

Time of Activation of a Subnetwork

Subnetworks can be activated at two different times:

- **After the activation time of the calling network**

  The subnetwork is always activated (i.e., with the job type `NET`) after the calling network has been activated. The time discrepancy depends on the Monitor activation cycle. The subnetwork will be available from this point in time with **run number** and active JCL. This is the default.

- **At the start time (submission) of the calling job of the type NET**

  The subnetwork is only activated if the calling job of the type `NET` is really started. This can prevent a subnetwork from even being activated if the calling job is not executed at all later on. Moreover, the effort for job activations and loading the JCL is thus compensated for, and the waiting-time for prerequisites within the subnetwork is shorter.

The subnetwork activation mode can be defined as follows:

- As a global default value (see **Subnetwork activation mode** in *Defaults for Network Options* in the *Administration* documentation).
- In the definition of the calling job of the type `NET` (see **Activation mode**).

Symbol Table Activation for Subnetworks

The order of symbol table activation for subnetworks is:

1. Set caller info into the active table (`P-C-`... symbols).
2. Activate symbol table(s).
3. Perform symbol modification exit.

The order of symbol table activation for subnetworks is the same as for main networks.

Restrictions for Subnetworks

This section covers the following topics:

- Recursion Checking

- Maximum Nesting Level

Recursion Checking

Subnetworks can in turn be invoked within subnetworks; however, a subnetwork must not invoke itself directly or indirectly in a circle because this would cause an infinite recursion.

The definition of a network as its own subnetwork is prohibited. If this is attempted, error messages such as the following are issued:

- `Recursive Subnetwork Definition prohibited`
- `Max. Subnetwork Level :1: reached`

Maximum Nesting Level

The maximal supported nesting level for subnetworks is 20.

Adding or Modifying a Subnetwork Definition

To display or modify a subnetwork definition

1 From the table in a **Job Master List** window, select a job of the **Type** `NET`, and choose **Display** or **Open** from the context menu.

2 Open the **tabbed page Subnet**.

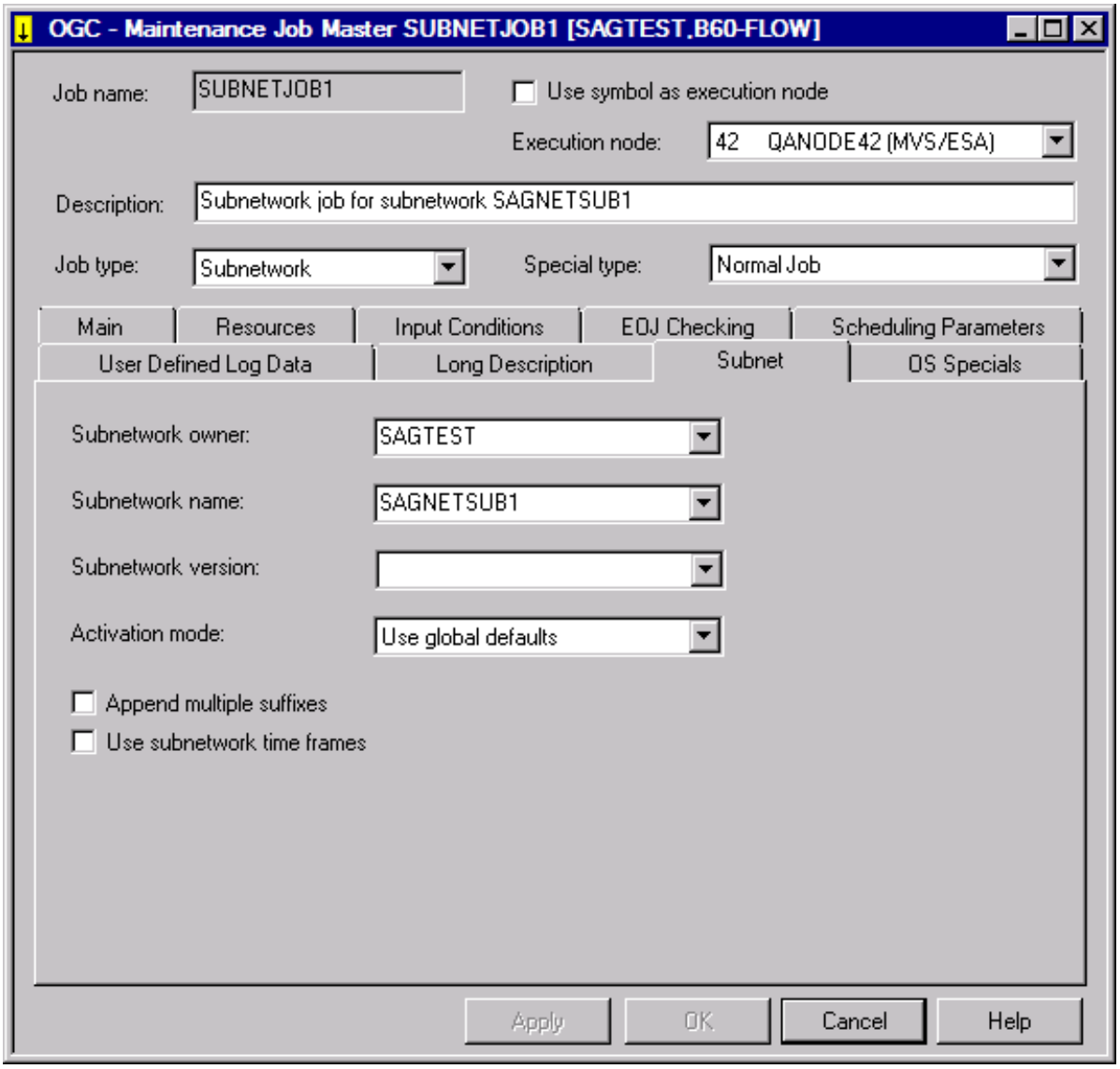
The fields on the page are described in *Fields: Job Type Specific Execution Features*.

To add a subnetwork definition

1 From the **Job Master window**, select **Subnetwork** from the **Job type** drop-down list box.

The tab **Subnet** appears.

2 Open the tabbed page **Subnet** like the example below:



3 Make your definitions. The input fields available on the page are described in *Fields: Job Type Specific Execution Features*.

4 Choose **OK** when you are finished.

Listing Jobs of a Subnetwork

If a master job has the type NET, and if the subnetwork is defined properly, you can invoke the master job list of the defined subnetwork directly.

If the subnetwork was not defined yet, the subnetwork definition screen is invoked instead.

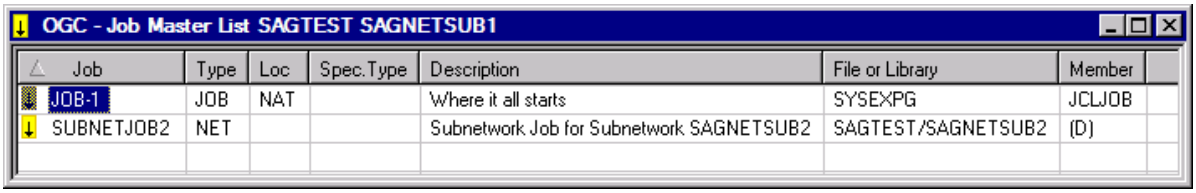
To list the jobs of a subnetwork

- From a **Job Master** or **Job Active** node, select a job of the type **Subnetwork** (NET), and choose **Zoom Subnetwork** from the context menu.

  Or:

  From the table in a **Job Master List** or an **Active Job List** window, select a job of the **Type** NET, and choose **Zoom Subnetwork** from the context menu.

  The jobs of the subnetwork defined for the selected NET job are listed in a (separate) **Job Master List** or **Active Job List** window, as shown in the following example:



30 Defining and Managing JCL for a Job

- Using Job Control in Entire Operations 256
- Dynamic JCL Generation (JCL Location MAC) 257
- Job Control for Jobs under BS2000 262
- Job Control for Jobs under UNIX 262
- Job Control for Jobs under Windows 262
- Defining Master JCL for a Job 266
- Displaying Master JCL 270
- JCL Locations 270
- Defining Operating System Dependent JCL Specifications 274
- Handling JCL during Job Submission 275
- Symbol Replacement in JCL 277
- Pregenerating Active JCL 277

A job needs job control language (JCL) instructions to perform a task. The required JCL is contained in the JCL member of a library/file of the operating system, or in a Natural object contained in a Natural library/system file.

This section provides information on defining JCL for a job master and editing the JCL source referenced by a job.

Related Topics:

- *Editing Master JCL and Natural Sources*
- *Regenerating Active JCL* in the section *Active Job Networks*

Using Job Control in Entire Operations

Entire Operations handles JCL as follows:

- **Job Master Control**
  Entire Operations used JCL in its original format on the original storage medium. The usual JCL storage locations of the various operating systems are supported. The source texts for **dynamic JCL generation** are also considered job master control.
- **Active Job Control**
  Entire Operations used the actual JCL submitted to the operating system for execution. It is produced from the master JCL when the job or network is activated. The symbols are replaced with values from the active symbol table. If it is **dynamic JCL**, the generation is performed at this time. The active JCL is stored in the active Entire Operations database.
- **Pregenerated Active Job Control**
  For reasons of performance, it might be necessary to generate active JCL in advance. See also *Pregenerating Active JCL*.

  JCL must be pregenerated again when:

  - The definition of the master JCL storage has been modified;
  - The master JCL has been edited;
  - The corresponding symbol table has been modified.
- **Submitted Job Control - Header Information**
  Entire Operations inserts a header with Entire Operations environment information as a comment into the submitted JCL.

  For UNIX, the header is inserted into the frame script.

  For Windows, the header is inserted into the frame BAT file.

- **Submitted Job Control - Environment Variables for UNIX and Windows**
  The generated and submitted JCL frame script (UNIX) or frame BAT file (Windows) can contain predefined symbols provided as environment variables. For details, see ***Reserved Symbols for UNIX and Windows Environment Variables*** in the section *User Exits*.

Dynamic JCL Generation (JCL Location MAC)

When defining a job within a network, you can specify that its JCL is to be generated dynamically either at job activation time or at job submission time.

Dynamic JCL generation is achieved by using the Entire Operations edit macro function. The text strings contained in the JCL can contain **escape characters** (see ***Symbol Escape Characters***) followed by variables that are replaced by their current values during dynamic generation.

These current values are taken from a **symbol table** which must contain the current values to be substituted. The symbol table to be used can be determined by choosing the **Usable Symbol Tables** function from the context menu of a network or job node (see ***Listing Usable Symbol Tables***).

If any symbol specified in the dynamic JCL is not found in the symbol table defined for the job, the symbol is searched for in the symbol table next in the search hierarchy for symbol tables (see ***Symbol Table Types and Symbol Search Order***.

Additionally, Entire Operations passes **predefined symbols** from the parameter section to the dynamically generated program, such as job owner, network name, current job name and original scheduling date. The same applies to Natural system variables such as `*DATE`, `*TIME` and `*USER`. As these parameters can be replaced in any part of the JCL, different JCL configurations can be generated depending on time, date, user ID etc.

Entire Operations provides dynamic JCL generation for all supported operating systems (z/OS, BS2000, UNIX and Windows) as shown in the following examples.

This section covers the following topics:

- Example 1: Dynamic JCL in a z/OS Environment
- Example 2: Dynamic JCL in a BS2000 Environment
- Example 3: Dynamic JCL in a UNIX Environment

Related Topics:

- ***Handling Macro Sources for Dynamic JCL Generation***

Example 1: Dynamic JCL in a z/OS Environment

The following is the symbol table specified for the macro program:

Symbol Name	Current Value
STEPLIB	SN.SYSF.SOURCE
CLASS	G

The variable from the parameter section is assumed to have the following value:

P-OWNER	NET1

The system variables are assumed to have the following values:

*TPSYS	COMPLETE
*DEVICE	BATCH
*INIT-USER	SN

The following is a macro Natural program including a parameter section and JCL with the escape character (#) followed by variable names from the symbol table:

```
# DEFINE DATA PARAMETER USING NOPXPL-A
# LOCAL /* MUST BE CODED
# END-DEFINE
//SNMAC4 JOB ,#P-OWNER,MSGCLASS=X,CLASS=#CLASS //STEP01 EXEC
PGM=NOPCONTI,PARM='C0004' //STEPLIB  DD DISP=SHR,DSN=#STEPLIB
//* DEVICE: *DEVICE, INIT-USER: *INIT-USER //* TPSYS: *TPSYS
# IF CLASS = 'G'
//* THE MSGCLASS IS REALLY 'G'
# ELSE
//* ANOTHER MSG-CLASS FOUND
# END-IF
//*
```

The resulting dynamically generated JCL will be:

```
//SNMAC4 JOB ,NET1,MSGCLASS=X,CLASS=G
//STEP01 EXEC PGM=NOPCONTI,PARM='C0004' //STEPLIB  DD
DISP=SHR,DSN=SN.SYSF.SOURCE //* DEVICE: BATCH, INIT-USER: SN
//* TPSYS: COMPLETE
//* THE MSGCLASS IS REALLY 'G'
//*
```

Example 2: Dynamic JCL in a BS2000 Environment

The fields taken from the DB-INFO view are assumed to have the following values after the FIND statement:

Field	Value
NUCLEUS	055
LP1	1000
NU1	100
ACCOUNT	EXAMPLE
NH1	4000
MSG	FHL
VERSION	524

The variables taken from the parameter section have the following current values:

Variable	Value
P-OWNER	OS
P-JOB	NUC055
P-EXECUTION-NODE	055

No symbol table was defined for this example job.

The following is the example JCL written using the edit macro function, including variables to be substituted from the DB-INFO view and the parameter section. Variables are preceded by the escape character (#):

```
# DEFINE DATA PARAMETER USING NOPXPL-A
#  1 L-JOB
# 1 REDEFINE L-JOB
# 2  L-JOB-A    (A3)
# 2  L-JOB-NUC  (N3)
# LOCAL   /* LOCAL VARIABLES START HERE
# 1 DB-INFO VIEW OF DB-INFO
# 2  NUCLEUS
# 2  LP1
# 2  NU1
# 2  ACCOUNT
# 2  NH1
# 2  MSG
# 2  VERSION    /* E.G. 524
# 1 LWP  (N7)
# 1 NUC  (N3)
# 1 SPOOL  (A10) INIT <'NOSPOOL'>
# END-DEFINE
```

```
# *
# MOVE P-JOB TO L-JOB-A
# MOVE P-EXECUTION-NODE TO NUC
# F1. FIND DB-INFO WITH NUCLEUS = NUC
/.NUC NUC LOGON #P-OWNER,#ACCOUNT
/OPTION MSG=#MSG
/REMARK
/REMARK  NUCLEUS #NUC
/REMARK
/SYSFILE  SYSLST = NUC NUC..LST.NUC
/SYSFILE  SYSDTA = SYSCMD
/FILE  ADA VERSION..MOD,LINK=DDLIB
/FILE  *DUMMY,LINK=DDLOG
/FILE  *DUMMY,LINK=DDSIBA
/FILE  ADA NUC..ASSO,LINK=DDASSOR1,SHARUPD=YES
/FILE  ADA NUC..DATA,LINK=DDDATAR1,SHARUPD=YES
/FILE  ADA NUC..WORK,LINK=DDWORKR1,SHARUPD=YES
/EXEC  (ADARUN,ADA VERSION..MOD)
# COMPUTE LWP = F1.LP1 * (F1.NU1 + 100)
ADARUN PROG=ADANUC,LP=F1.LP1,LU=65535,LWP=#LWP ADARUN
DB=#NUC,NU=#NU1,NC=20,TT=600,TNAE=1800 ADARUN NH= NH1
/SYSFILE  SYSLST = (PRIMARY)
/SYSFILE  SYSDTA = (PRIMARY)
/SYSFILE  SYSOUT = (PRIMARY)
/LOGOFF  SPOOL
# END-FIND
```

The resulting dynamically generated JCL will be:

```
/.NUC055 LOGON OS,EXAMPLE
/OPTION MSG=FHL
/REMARK
/REMARK  NUCLEUS 055
/REMARK
/SYSFILE  SYSLST = NUC055.LST.NUC
/SYSFILE  SYSDTA = SYSCMD
/FILE ADA524.MOD,LINK=DDLIB
/FILE *DUMMY,LINK=DDLOG
/FILE *DUMMY,LINK=DDSIBA
/FILE ADA055.ASSO,LINK=DDASSOR1,SHARUPD=YES
/FILE ADA055.DATA,LINK=DDDATAR1,SHARUPD=YES
/FILE ADA055.WORK,LINK=DDWORKR1,SHARUPD=YES
/EXEC (ADARUN,ADA524.MOD)
ADARUN PROG=ADANUC,LP=1000,LU=65535,LWP=200000 ADARUN
DB=055,NU=100,NC=20,TT=600,TNAE=1800 ADARUN NH=4000
/SYSFILE  SYSLST = (PRIMARY)
/SYSFILE  SYSDTA = (PRIMARY)
/SYSFILE  SYSOUT = (PRIMARY)
/LOGOFF NOSPOOL
```

Note: Any JCL generated at activation time using the macro language can be modified by the user until the job is actually submitted. Of course this modification is valid only for the current network run.

Example 3: Dynamic JCL in a UNIX Environment

The following example illustrates dynamic symbol replacement within a Bourne shell script (escape character §):

```
#
# Bourne shell script for checking the number of users
# entered in /etc/passwd.
# If more than §USER-LIMIT entries appear,
# the script will be ended with exit 1.
#
#!/bin/sh
set -x
USER_COUNT='wc -l < /etc/passwd'
echo Number of users on node 'hostname' : $USER_COUNT
if test $USER_COUNT -gt §USER-LIMIT
then
    echo USER_COUNT_WARN
    exit 1
else
    echo USER_COUNT_OK
fi
```

The symbol table to be used should appear as follows:

Symbol Name	Current Value
USER-LIMIT	100

The result is the following executable shell script:

```
#
# Bourne shell script for checking the number of users
# entered in /etc/passwd.
# If more than 100 entries appear,
# the script will be ended with exit 1.
#
#!/bin/sh
set -x
USER_COUNT='wc -l < /etc/passwd'
echo Number of users on node 'hostname' : $USER_COUNT
if test $USER_COUNT -gt 100
then
    echo USER_COUNT_WARN
    exit 1
else
```

```
    echo USER_COUNT_OK
fi
```

Note: Any JCL generated at activation time using the macro language can be modified by the user until the job is actually submitted. Of course this modification is valid only for the current network run.

Job Control for Jobs under BS2000

This section covers the following topics:

- Naming Conventions for Work Files
- User Exit for Work File Names

Naming Conventions for Work Files

The name generation for work files under BS2000 is described in the section *Naming Conventions for Work Files* of the *Installation and Setup* documentation.

User Exit for Work File Names

Names of BS2000 work files can also be generated via an exit (see ***Generating SYSOUT File Names for BS2000*** in the section ***API Routines***).

Job Control for Jobs under UNIX

The environment variable `$EOR_WORK` of Entire System Server/UNIX contains the name of the Entire Operations work directory. Within this directory, the work files are hierarchically stored.

Job Control for Jobs under Windows

This section covers the following topics:

- Job Control Characteristics under Windows
- Components Required
- Executing Operating System Functions
- File Names
- SYSOUT Redirection
- Entire Operations Work Directory
- Environment Variables

- Job Control (JCL)
- Job Start and Job Control
- End-of-Job Check
- End-of-Job Actions

Job Control Characteristics under Windows

The job control system Entire Operations also runs in a Windows environment, with the following characteristics:

- Operating system neutral modeling of job networks.
- Entire Operations functionality and flexibility is also transparently available on Windows.
- Support of DOS batch files and executables (EXE).
- It is not necessary to directly enter Windows DOS commands.
- Can also run in mixed mainframe/Windows/UNIX environments.
- Job control on several Windows machines at the same time.

Components Required

- **Entire Operations Monitor**

  The Monitor can run under the operating systems BS2000, z/OS, or UNIX, concurrently controlling jobs executed under BS2000, z/OS, and UNIX.

- **Entire System Server - For Mainframes, UNIX and Windows**

  Required to access mainframe, UNIX or Windows operating systems.

  On every machine to be controlled, an Entire System Server/Windows server has to be installed. This is installed as a Windows service and is administrated with the Windows services administration.

- **Entire Net-work/EntireX Broker**

  Used as the transport layer.

Executing Operating System Functions

For executing operating system functions, there is a server of type Entire System Server/Windows on each Windows node. This server runs as a background Windows process.

The Entire Operations Monitor and the Entire Operations online application use the following components for server communication:

- The System Automation Tools (SAT) communication layer
- The EntireX Broker for transmitting client/server requests
- Entire Net-work as the transport layer
- Up to 740 Windows nodes can be served concurrently

File Names

As there is no backslash (`\`) available on mainframes, Windows file names can be written with a slash (`/`) as an alternative as well if the character sequence `+F+` directly precedes the file name. This applies to file names within JCL as well.

Example:

Original Windows	c:\jcl\script1.bat
Alternative representation	+F+c:/jcl/script1.bat

SYSOUT Redirection

From the Entire Operations Monitor, all jobs are started which redirect their output to a file. The SYSOUT files are stored in the Entire Operations work directory. If the job is repeated, the old SYSOUT file is renamed.

Entire Operations Work Directory

The Entire System Server/Windows environment variable `%EOR_WORK%` contains the name of the Entire Operations work directory. Within this directory, the work files are stored hierarchically.

The name generation for work files under BS2000 is described in *Naming Conventions for Work Files Created by Entire Operations, Windows* in the section *Installing Entire Operations on Mainframes* in the *Installation and Setup* documentation.

The name of the work directory for an active network is available in the predefined symbol `P-NADIR`. You may store application-specific work files there, provided there are no naming conflicts with files created by Entire Operations.

Work files created by Entire Operations or by the application are deleted by the Entire Operations Monitor on network or job deactivation.

Environment Variables

You can use Windows environment variables freely within file names. This corresponds to the behavior of BAT files. You can combine environment variables and variables from symbol tables.

Job Control (JCL)

The Windows JCL can be stored in any place. Among other places, it can be stored in Natural text objects or in mainframe files. Symbol replacement and JCL generation (jobs with JCL location `MAC`) are available.

Job Start and Job Control

The Entire Operations Monitor starts jobs based on conditions and on time. **Accounting data** is retrieved and stored. You can manually cancel the job from the online environment.

End-of-Job Check

Entire Operations adds some echo commands to the Windows job control. This allows control of certain messages in the SYSOUT:

- Starting and ending message with timestamp
- Elapsed job time

These messages are used to check whether a job did run completely. Additionally, **accounting information** is retrieved. For checking the job, strings can be searched for in SYSOUT and End-of-Job check routines can be used.

End-of-Job Actions

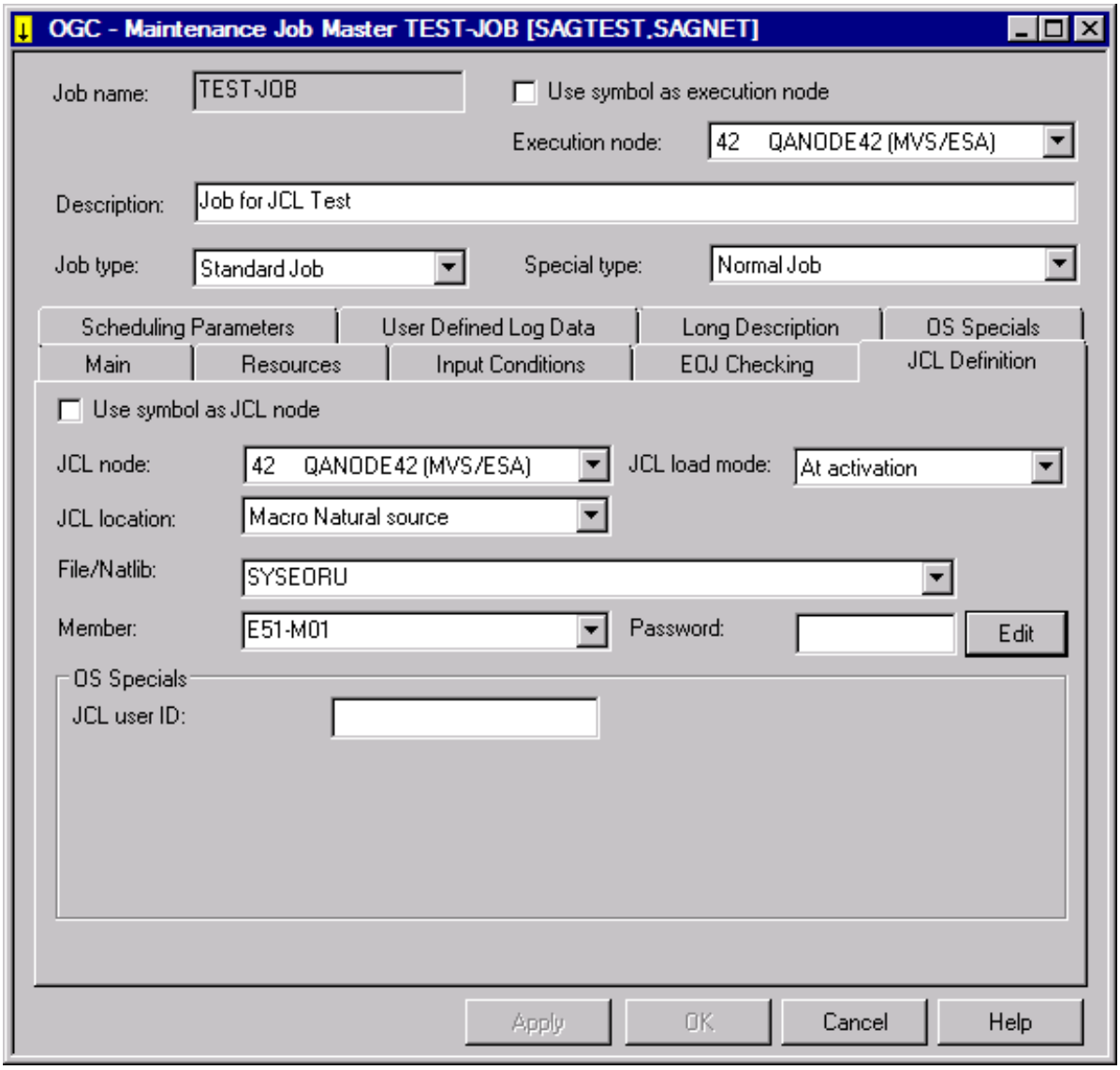
You can send messages (for example by e-mail) to other users from the Windows node. In the Windows node definition, a program for sending mail can be defined. It is required that this program can be started from the DOS command prompt. One example for such a program is the shareware wsendmail. All other forms of news broadcasting, such as the Entire Operations Mailbox, can still be used.

You can define printing of files and of SYSOUT lists as an End-of-Job action. For each Windows node, a Windows print command with a variable for the file name can be defined. User exits can help start further actions.

Defining Master JCL for a Job

To define master JCL for a Job

1 In the **Maintenance Job Master window**, open the tabbed page **JCL Definition**:



2 Make your definitions. The input fields and special functions available are explained in *Fields: Master JCL Definition* and *Functions: Master JCL Definition*.

3 Choose **OK** to save your definitions.

If you want to edit the defined JCL, proceed as described in *Editing Master JCL and Natural Sources*.

This section covers the following topics:

- Fields: Master JCL Definition
- Functions: Master JCL Definition

Fields: Master JCL Definition

The fields on the **JCL Definition page** are described in the following table:

Field	Description	
Use symbol as JCL node	Select this check box if you want to use a symbol for a **JCL node** (see below).	
JCL Location	Type of JCL storage location. For selection options, see *JCL Locations*. In a network definition, this field serves as a default for the entire network. See also *Fields: Network Definition* in the section *Network Maintenance*.	
JCL Node	JCL for the job is read from this node. The job is started on the **execution node**. The default is the value specified in the job network definition (see *Fields: Network Definition*), which can be overwritten here. You can define the JCL node as a symbol if the **Use symbol as JCL node** option is marked. For details, see *Symbols in Node Definitions* in the section *Symbol Table and Symbol Maintenance*.	
JCL Load Mode	Possible selection options:	
	At activation	Load JCL at activation time (default). **Note:** If the job is defined as cyclic, the job's active JCL will be loaded only once. Only *Symbol Replacement* with *Symbol Escape Characters* will be performed before each submission.
	At submission	Load JCL at submission time. **Note:** 1. JCL load errors are detected very late. 2. Network execution might slow down because of the time necessary for JCL load before submission. 3. **JCL pregeneration** also works with JCL loading at submission time.

Field	Description	
		4. If **resources** are defined, they are allocated at submission time. If the JCL loading at submission time fails, the allocated resources are freed immediately. 5. JCL loading at submission time is also intended to allow you the generation of ***Macro JCL*** as late as possible. 6. The JCL of a cyclic job will be loaded again before each ***Cyclic execution***. 7. For other types of JCL, it is still recommend to use activation time symbol replacement, which causes less system overhead.
File/Natlib	Storage type of the JCL corresponding to the job type. For Natural programs and dynamic JCL (storage location `NAT`): the Natural library where the program or the variable JCL resides. The Natural library must reside in the FUSER system file and must not have the same name as the SYSEOR system library. For all others: the file name. You can select the required library or file from the drop-down list box. For BS2000: The default pubset is stripped from the file name prior to storing it. This allows easier migration to another default pubset. For UNIX: Be aware of upper and lower case. You may use environment variables and symbols within names. Symbols may be used. If a JCL file contains the **activation escape character** (see *Escape Characters*), symbol replacement is performed: ■ When editing JCL (temporary, using the symbol table master); ■ When loading JCL (permanently, using the active symbol table). For detailed information, see the section ***Symbol Table and Symbol Maintenance***.	
Member	Name of the Natural object or library/file member of the operating system that contains the JCL. This applies to all JCL locations which support members: see ***List of JCL Locations***. This field cannot be used and is protected for other JCL locations. When you add a new job definition, the name of the job is entered in this field by default. You can select the required member from the drop-down list box.	

Field	Description
	New Members: You can also enter the name of a non-existing member if, for example, no JCL has yet been defined for the job. Entire Operations accepts the member name assuming a member with this name will be created. Before activating a network and before starting a job, Entire Operations checks whether a member has been defined. If there is no member, an error message is issued. Symbols may be used. If a JCL file contains the **activation escape character** (see *Escape Characters*), symbol replacement is performed: ■ When editing JCL (temporary, using the symbol table master); ■ When loading JCL (permanently, using the active symbol table). For detailed information, see the section ***Symbol Table and Symbol Maintenance***. **Note for BS2000:** JCL location `PRC` (**Callable procedure**): If no member is specified, the JCL file must be a BS2000 sequential file.
VolSer	Volume serial of the file. Only required if the file has not been cataloged (z/OS only).
Password	The file password if the file has been protected by a password.
OS Specials	Operating system-dependent input fields: see ***Fields: Operating System Specials for JCL.***

Functions: Master JCL Definition

The following special function is available on the **JCL Definition page**:

Function	Description
Edit	Opens the **editor** for the source entered in the **Member** field or the file entered in the **File/Natlib** field depending on the specified JCL location. This command is only available if you are authorized to edit the specified source and if the **Apply** and **OK** buttons are disabled in the window. This command corresponds to the **Edit Master JCL** or **Edit Active JCL** function available from the context menu of a selected job instance. See also ***Editing Master JCL and Natural Sources***.

Displaying Master JCL

To display the master JCL source defined for a job

- In the object workspace, select a **Job Master** instance and choose **Browse Master JCL** from the context menu.

  A **Browse** window like the **Edit window** opens displaying the source of the JCL.

JCL Locations

This section covers the following topics:

- List of JCL Locations
- Restrictions for Job Types
- Location on another Node
- Replacing Symbols
- JCL Location PRC: BS2000 Procedure
- JCL Frames for BS2000 DO Procedures

List of JCL Locations

Entire Operations offers a broad range of possible locations for its original (master) JCL.

The following table describes the physical locations you can select for JCL from the **JCL location** drop-down list box on the **JCL Definition page**.

The locations `NAT` (**Natural source**) and `MAC` (**Macro Natural source**) are available on all operation systems supported by Entire Operations. Other locations are available for individual operating systems only as indicated in the following table:

JCL Location/Selection Option	Applies to Operating System	Description
blank field **(No JCL required)**	n/a	None (no JCL required).
`BS2` **SAM or ISAM**	BS2000	SAM or ISAM file. See also **BS2000** in *Fields: Operating System Specials for JCL*.

JCL Location/Selection Option	Applies to Operating System	Description
EXE **Executable program**	UNIX, Windows	Executable, for direct file execution. Not intended for shell scripts or BAT files. You can pass parameters to the executable file with the CMDLINE-*job* reserved symbol. See ***Reserved Symbols for Command Line Parameters*** in the section *Symbol Table and Symbol Maintenance*.
LIB **Librarian**	z/OS	Librarian.
LMS **LMS library**	BS2000	LMS library. See also **BS2000** in *Fields: Operating System Specials for JCL*.
MAC **Macro Natural source**	all	For macro JCL. A Natural source object is used.
NAT **Natural source**	all	Natural source object. See also ***Restriction for Job Types***.
PDS **Partitioned data set**	z/OS	Member of a partitioned data set. See also ***Restriction for Job Types***.
PRC **Callable procedure**	BS2000	BS2000 procedure. See ***JCL Location PRC: BS2000 Procedure*** for more information.
TXT **Text file**	UNIX, Windows	Text file. See also **UNIX and Windows** in *Fields: Operating System Specials for JCL*.

Restrictions for Job Types

- Jobs of the type `STC` (**Started Task**) must have the location `PDS` (**Partitioned data set**) on z/OS.
- Jobs of the type `NAT` (**Natural Program**) must have the location `NAT` (**Natural source**).
- Jobs of the type `DUM` (**Dummy Job**) must not have any JCL location.

During definition, it is checked whether the defined JCL exists at all. If the JCL cannot be found, a warning appears in the Job JCL Definition (Master) window.

Location on another Node

The JCL can be saved on another machine, not identical with the executing machine and even on another operating system, not identical with the executing operating system.

Replacing Symbols

In all JCL locations, you can easily replace symbols as described in ***Symbol Replacement*** in the section *Symbol Table and Symbol Maintenance*.

JCL Location PRC: BS2000 Procedure

If this location has been defined, Entire Operations generates a BS2000 ENTER job in the active JCL calling this procedure.

The following rules are applied:

- If no JCL member is specified, the JCL file must be a BS2000 sequential file. If a JCL member is specified, the JCL file must be a BS2000 LMS library.
- The BS2000 job name adopts the Entire Operations job name and if it is longer than 8 characters, it cuts the name to 8 characters.
- SDF statements (`CALL-PROC`) are used.
- The symbol table for which the job has been defined must contain all parameters of the BS2000 procedure. The symbol values are used for calling the BS2000 procedure.
- Positional and keyword parameters are supported.
- Numbers and names of the parameters are automatically retrieved from the procedure (`PROC / BEGIN-PROC` statement).
- The generated `CALL-PROCEDURE` statement has the parameter `LOGGING=*YES`.
- BS2000 job submissions are performed with the generated line `/MODIFY-SDF-OPTIONS CONTINUATION=*NEW-MODE`.

JCL Frames for BS2000 DO Procedures

For BS2000 procedures, a user-defined JCL frame can be used. It must be saved under the name PRCFRAME in the library SYSEORU. If PRCFRAME is not found in SYSEORU (or in SYSEOR, for compatibility), Entire Operations generates a standard frame.

Special Commands

Command	Description
`#ESC-FRAME` *escape-character*	Defines the escape character the JCL frame uses to replace symbols. This command must be at the top and is compulsory.

`#CALL-PROC`	The procedure call (`/CALL-PROC`) is generated here.
	Adding User-Defined Parameters to the /CALL-PROC Statement
	
	It is possible to add user-defined parameters to the `CALL-PROC` statement in the PRCFRAME text object. The following applies:
	
	■ The `#CALL-PROC` line must look like `#CALL-PROC -`. The minus sign (`-`) indicates a continuation. In the generated JCL, it will be placed in column 72.
	■ The user-defined continuation(s) follow the `#CALL-PROC` line. They must begin with the following characters: `/ ,.`
	
	Example:
	
	`#CALL-PROC -`
	`/ ,LOGGING=YES`

Example of PRCFAME Contents

```
#ESC-FRAME §
/.SN§P-RUN LOGON SN,1
/REMARK === PRC FRAME EXAMPLE
/REMARK §AAAA YYYYYYYYYYYY
#CALL-PROC
/STA L
§BBBB
/LOGOFF
```

Notes:

1. All symbols used in the JCL frame must be contained in the active symbol table of the job. The active symbol table must continue to contain all symbols for calling the procedure.
2. If a symbol does not exist, job activation is discontinued.

Defining Operating System Dependent JCL Specifications

Some operating systems or JCL locations may require some additional definitions.

To define JCL parameters specific to an operating system

- **On the JCL Definition page**, enter valid values in the input fields provided for the relevant operating system in the **OS Specials** section.

  The fields contained in the window are explained in ***Fields: Operating System Specials for JCL***.

This section covers the following topics:

- Fields: Operating System Specials for JCL

Fields: Operating System Specials for JCL

The operating system specific input fields provided in the **OS Specials** section of the **JCL Definition page** are described in the following table.

Operating System	Field	Description
BS2000	**Default User ID**	Valid logon user ID defined for BS2000. This ID has no meaning for other operating systems. Unqualified file and job variable names in this job definition are prefixed with this BS2000 user ID. **Note:** The user ID TSOS can be defined only if the user defining the ID is working under TSOS. See also the default setting **User ID Definition** described in *Defaults for Other Settings* in the *Administration* documentation.
	JCL User ID	If this field is not empty, the JCL is loaded under the rights of this BS2000 user ID and may be overridden by specific definitions. TSOS may only be defined, if the defining user is working under TSOS. Default: If this field is left blank, the user ID from the fully qualified JCL file name is used as **JCL User ID**. Symbol replacement is performed in this field if the **activation escape character** or the **submission escape character** is used. See also the default setting of **User ID Definition** (*Defaults for Other Settings*, *Administration* documentation), and the sections ***Operating System User IDs*** and ***Default User ID Determination***.

Handling JCL during Job Submission

This section covers the following topics:

- JCL Modifications during Job Submission
- Trigraph Encoding for JCL Submission on UNIX and Windows

Related Topics:

- ***Maintaining Active JCL (Job Control Language)*** in the section *Maintaining Active Jobs and Networks*

JCL Modifications during Job Submission

Jobs defined and scheduled in Entire Operations are submitted automatically under the control of the Entire Operations Monitor. During the submission process, the submitted JCL can be handled in any of the following ways:

- Completion or modification of the job JCL according to the Entire Operations defaults;
- Checking of all submitted JCL by a global user exit (to be defined in the Entire Operations defaults; see the *Administration* documentation);
- Insertion of **header information** into the submitted JCL. This is always performed if the **Generate Header in submitted JCL** option is activated in the Entire Operations defaults (see *Defaults for Other Settings* in the *Administration* documentation).
- Insertion of information on **replaced symbols** if the **Log Symbol Values in submitted JCL** option is activated in the Entire Operations defaults (see *Defaults: System/Log Files* in the *Administration* documentation).
- **For z/OS only:**
  If a submission time symbol replacement error occurs, a JCL error is forced to prevent the job from executing. Lines similar to the following force a JCL error:

  ```
  // ###### Entire Operations Symbol Replacement Error ######
  ```

  The job remains in an error status such as the following:

  ```
  JobId 51058 - Symbol Replacement Error
  ```

  End-of-Job processing is not completed, because this is treated as a submission error.

See also ***Working with Entire System Server Nodes*** in the section *Operating System User ID*.

Related Topic:

- ***Header and Symbol Information in SYSOUT*** in the section *Maintaining Active Jobs and Networks*

Trigraph Encoding for JCL Submission on UNIX and Windows

Trigraph encoding is used to avoid translation failures during ASCII/EBCDIC text conversion. An ASCII character for which no equivalent EBCDIC character exists is then substituted by a trigraph and the text can be encoded successfully.

Entire Operations uses trigraphs when generating JCL for UNIX and Windows. For Entire System Server execution nodes on UNIX and Windows, trigraph encoding is switched on by default using the question mark (?) as the default escape character.

Switching Trigraph Encoding On and Off

You can switch on and off trigraph encoding in the JCL with the following meta commands:

#EOR-TRIG=YES	Enables trigraph encoding.
#EOR-TRIG=NO	Disables trigraph encoding.

Each meta command must be in a single line as shown in the following example for UNIX and Windows.

Defined JCL:

```
...
echo '??( test ??)'
#EOR-TRIG = OFF
echo '??( test ??)'
#EOR-TRIG = ON
...
```

Submitted JCL:

```
...
echo '[ test ]'
echo '??( test ??)'
...
```

Supported UNIX and Windows Trigraphs

Entire System Server execution nodes on UNIX and Windows support the trigraphs listed in the following table. A trigraph always begins with two escape characters; the table below assumes the question mark (?) to be the default escape character.

ASCII	Trigraph	Remarks
[	??(	
\	??/	
]	??)	
^	?? '	
{	??<	
|	??_	
}	??>	
~	??-	
@	??%	
`	??;	
!	??:	
\f	??+	Form feed
\t	??&	Tab

Symbol Replacement in JCL

Symbols for replacement can be defined for all storage locations of master JCL.

Escape characters for symbols can be assigned system-wide, and they can also be defined individually for each job. Defaults for escape characters can be defined for each operating system.

Escape characters for symbol replacement must not conflict with other characters used in the JCL.

For detailed information, see the following sections in *Symbol Table and Symbol Maintenance*:

- *Defining Symbols and Symbol Values*
- *Symbol Replacement* and *Defining Escape Characters*

Pregenerating Active JCL

For performance reasons, it might be necessary to pregenerate active JCL to be used for job submission. This function is conceived mainly for macro jobs, whose **dynamic JCL generation** can become time-consuming, because macro language must always be executed. JCL can also be pregenerated for all other job types.

Note: In the case of a **resubmission of the active job**, the pregenerated JCL is not reloaded if the active JCL of the job has been edited in the meantime.

To pregenerate active JCL

- In the object workspace, select a **Job Master** instance and choose **Pre-generate Active JCL** from the context menu.

  If no generation errors occur, a message indicates that active JCL is pregenerated for the job.

  - Editing Pregenerated JCL
  - Removing Pregenerated JCL

Related Topic:

- *Regenerating Active JCL* in the section *Active Job Networks*

Editing Pregenerated JCL

This function allows editing the pregenerated JCL.

To edit pregenerated JCL

1. In the object workspace, select a **Job Master** instance for which pregenerated active JCL exists.
2. Choose **Edit pre-generated Active JCL** from the context menu.

   An **Edit window** opens where you can change the pregenerated JCL as required.

   See also *Editing Master JCL and Natural Sources*.

Removing Pregenerated JCL

To remove pregenerated active JCL

1. In the object workspace, select a **Job Master** instance for which pregenerated active JCL exists.
2. Invoke the context menu and choose the **Remove pre-generated Active JCL** function.

   The message `Pregenerated JCL deleted` indicates that the pregenerated JCL has been removed.

31 Editing Master JCL and Natural Sources

- General Editing Considerations 280
- UNIX and Windows: Handling of Tab Characters (H'09') within JCL Lines 282
- Starting an Edit Session 283
- Handling Macro Sources for Dynamic JCL Generation 287
- Usage of Text Objects in JCL 294
- Locking of Natural Sources 296
- Avoiding Exceptions while Editing JCL Location (NAT, MAC) 297

This section provides information on editing JCL and Natural source objects.

By editing the master JCL, you change the master definition that affects each future executed active job.

Before writing the file back, the editor creates a backup copy of the edited file.

Related Topics:

- *Editing Active JCL* in the section *Active Job Networks*

General Editing Considerations

This section covers the following topics:

- Automatic Logon to the Operating System Server
- Logon to a Node
- Windows: File Names
- UNIX and Windows: Text Files
- UNIX: Usage of Profile Scripts
- z/OS: JES2 /* ROUTE Statement

Automatic Logon to the Operating System Server

Before JCL is edited, Entire Operations checks whether the user may access the JCL file. If it is an operating system file, then an automatic logon with the defined JCL user ID (**BS2000** and **UNIX and Windows**) is performed, if required. If this automatic logon fails (e.g., if a password is required), an appropriate message is displayed. In this case, the user must first logon to the corresponding operating system server.

Logon to a Node

While editing the master JCL, an implicit logon is triggered to establish the connection to the required node. The logon is not prompted if you have already logged on to this node within this session.

For further information, see the **logon function** described in the section *Using Entire Operations GUI Client*.

Windows: File Names

File names in Windows can be written in an alternative notation if the backslash character (\) is not available, like in mainframe environments. See ***File Names* in *Job Control for Jobs under Windows* in the section *System Overview*.**

UNIX and Windows: Text Files

If a file to be edited or loaded as JCL is not a text file, the error message `EOR0325 - No Text File` is returned. For JCL loading, this is treated as a permanent error.

UNIX: Usage of Profile Scripts

Entire Operations determines usage of profiles in non-login UNIX scripts (as they are submitted by Entire Operations) during submission of the shell script as described in the following section:

- The Entire Operations Monitor checks whether the symbol `ETC-PROFILE` exists in the symbol table of the active job, or in another symbol table in the standard **symbol search hierarchy**, up to SYSDBA/A.
  - If the symbol `ETC-PROFILE` is found and if it contains the **value** `Y`, the frame script (*job*.current.sh) sources the profile script(s) executed in the currently running shell (other scripts are not sourced). These scripts are retrieved from */etc/profile* and */etc/profile.local*.
  - The user script (*job*.current.frame.sh) locates the environment variables of the profile script being set.
- The Entire Operations Monitor checks whether the symbol `ENV` exists in the symbol table of the active job, or in another symbol table in the standard **symbol search hierarchy**, up to SYSDBA/A.
  - If the symbol `ENV` is found and contains a **value**, its content is assumed to be a profile script like $HOME/.profile.
  - If a profile script exists, its name is assigned to the environment variables `ENV` and `BASH_ENV` by the frame script (*job*.current.sh).
  - The frame script (*job*.current.frame.sh) sources the profile script.
  - The user script (*job*.current.sh) locates the environment variables of the profile script being set.

For more information on `ETC-PROFILE` and `ENV`, see the section ***Reserved Symbols for Profile Scripts (UNIX)***.

Notes:

1. Several profile scripts can be sourced sequentially if the symbol `ETC-PROFILE` contains the value `Y` and if the symbol `ENV` contains the name of a profile script.

2. The user is responsible to make the used profile scripts proof against multiple execution, e.g., by using the PROFILEREAD variable like in Linux.

z/OS: JES2 /* ROUTE Statement

If a z/OS JES2 JCL contains a statement

```
/* ROUTE XEQ target
```

the job executes on the *target* target machine.

As long as the SYSOUT is passed back to the submission machine, the running job is not accessible. Entire Operations detects a rerouting, and behaves differently for such jobs.

For rerouted z/OS jobs, some features do not work, for example:

- Direct execution tracking.
- Browsing of SYSOUT while the job is executing.
- Cancelling.

However, if SYSOUT is available again, all End-of-Job checking and End-of-Job actions can be performed.

UNIX and Windows: Handling of Tab Characters (H'09') within JCL Lines

- If a master or active JCL is edited, each tab character is converted to a single blank before the editing. This prevents strange effects during editing. The JCL is written back without tab characters in any case.
- **Extended logging**, JCL logging: for the extended logging of JCL tab characters are also converted to blanks.
- If Entire Operations is running on a mainframe, the middleware translation tables convert tab characters to blanks anyway. This means that each UNIX and Windows JCL is submitted without tab characters.
- If Entire Operations is running on UNIX or Windows, tab characters remain in master and active JCL, as long as the master or active JCL is not edited from within Entire Operations.

Starting an Edit Session

This section describes how to edit JCL and Natural sources including user exits.

To edit JCL of a standard operating system job or a Natural source

1 In the object workspace, select a **Job Master** instance and choose **Edit Master JCL** or **Edit Active JCL** from the context menu.

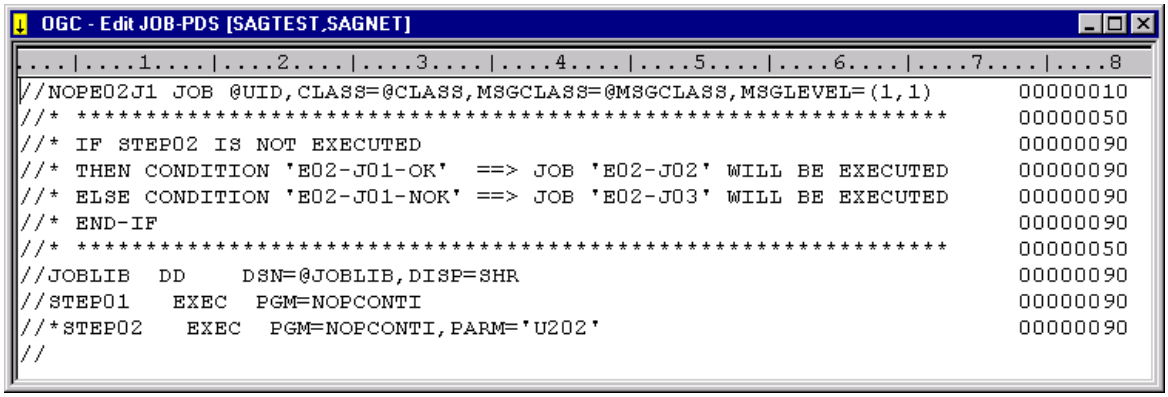
Or:

On the **JCL Definition page** in the **Maintenance Job Master** or **Maintenance Job Active** window, choose **Edit** (see also *Functions: Master JCL Definition*).

An **Edit** window like the following examples opens.

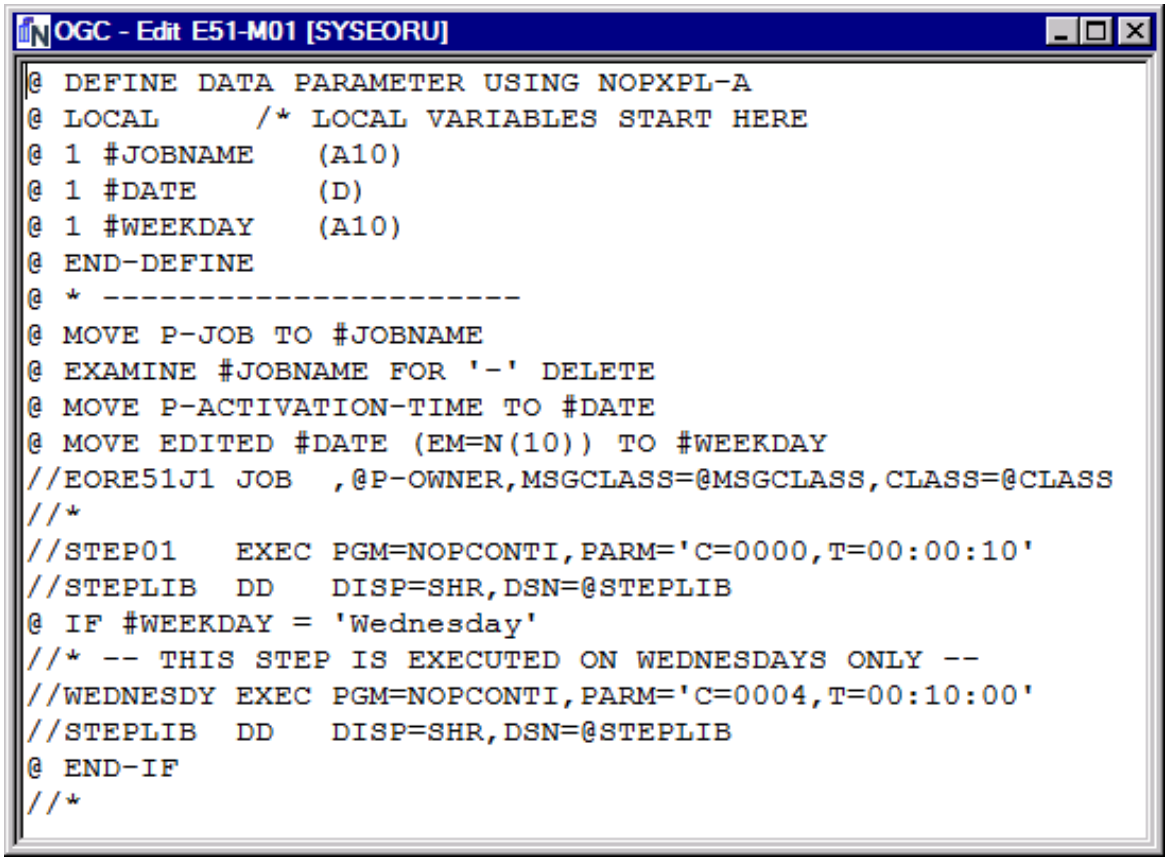
(If the source of the specified JCL member does not yet exist, the **Edit** window is empty.)

- Example of JCL for the **location Partitioned data set** (`PDS`):



The example above shows the JCL member `NOPE02J1` of the job `JOB-PDS`.

- Example of JCL for the **location Macro Natural source** (`MAC`):



The example above shows the macro Natural source `E51-M01` contained in the Natural library `SYSEORU`.

For details on using macro sources and special edit functions, see *Handling Macro Sources for Dynamic JCL Generation*.

2 Edit the source as required.

You can use the editor functions provided in the context menu of the **Edit** window (or the toolbar of the **main application window**) to find and change text, navigate in the source or perform special functions for macro sources. See *Available Functions: Edit Window* and *Handling Macro Sources for Dynamic JCL Generation*.

3 When you are finished, choose **Save** from the toolbar or **Object** menu to save any changes.

To edit the source of a user exit

1 In the object workspace, select a **Job Master** instance and open the **Input Conditions** page.

Select an input condition of the type **Exit**.

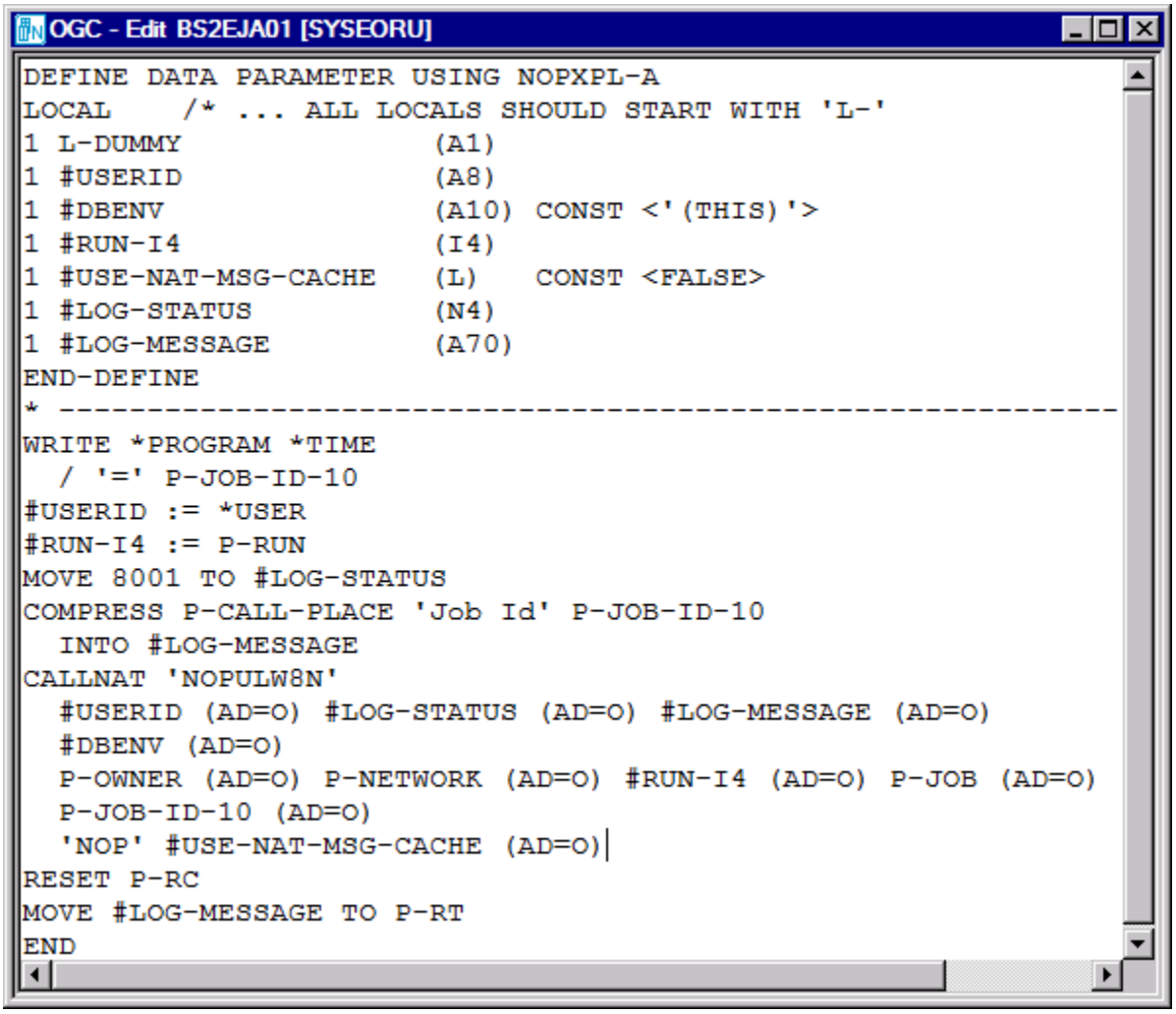
Choose **Edit**.

Or:

In the object workspace, select a **Job Master** instance and open the **EOJ Checking page**.

Select an action user exit (indicated by an **X** in the **Action** column) or a checking user exit and choose **Edit Action Exit** or **Edit User Exit**, respectively.

An **Edit** window like the following opens:



(If the source of the selected user exit has not yet been created and cataloged, the **Edit** window only contains an incomplete `DEFINE` sample statement.)

2 Edit the source as required.

In order to use a user exit, it must be saved and cataloged as a Natural object in a Natural library.

For information on coding action user exits, see also *User Exits for End-of-Job Checking and Actions* in the section *User Exits*.

You can use the editor functions provided in the context menu of the **Edit** window (or the toolbar of the **main application window**) to find and change text, navigate in the source or perform special functions for macro sources. See *Available Functions: Edit Window*.

This section covers the following topics:

- Available Functions: Edit Window

Available Functions: Edit Window

The following functions are provided from the context menu in an **Edit window** or the toolbar in the **main application window**:

Function	Shortcut	Description
Undo	CTRL+Z	Undo the last action.
Redo	CTRL+Y	Redo the last action.
Cut	CTRL+X	Cut the selected text string to the clipboard.
Copy	CTRL+C	Copy the selected text string to the clipboard.
Paste	CTRL+V	Paste the text string previously cut or copied to the clipboard.
Delete	DELETE	Delete the selected text string.
Select All	CTRL+A	Select the whole text contained in the **Edit** window.
Upper	CTRL+SHIFT+U	Convert the selected text string to uppercase letters.
Lower	CTRL+SHIFT+L	Convert the selected text string to lowercase letters.
Find	CTRL+F	Open the **Find** dialog to enter a search string and specify search options (case-sensitive and/or whole word only) and the search direction backward (scrolls up in the source) or forward (scrolls down in the source). By default, the currently selected text string is entered in the **Find** dialog.
Find Next	F3	Find the next occurrence of the search string specified in the **Find** dialog.
Replace	CTRL+H	Open the **Replace** dialog to enter a **Find** search string and a **Replace** text string and specify search options and the search directory as described for the **Find** dialog. By default, the currently selected text string is entered in the **Find** dialog.
Replace Next	CTRL+F3	Replace the next occurrence of the search string specified in the **Replace** dialog.
Go To	CTRL+G	Open the **Go To** dialog to place the cursor on the source line entered in the dialog.

Function	Shortcut	Description
Toggle ruler	CTRL+R	Show or hide a vertical ruler at the top of the **editor window** as shown in a previous example. If you want to switch on the ruler display by default, set the corresponding option in your user profile: see *GUI General* in the *Administration* documentation.
Print	CTRL+P	Output the text on a printer.
Check	---	Applies to Natural sources only. Check the syntax of the source.
Catalog	---	Applies to Natural sources only. Catalog the source.
Stow	---	Applies to Natural sources only. Check the syntax and catalog the source.
Macro	---	Applies to macro sources only. See *Special Editor Functions for Macro Sources*.
Test	---	Applies to macro sources only. See *Special Editor Functions for Macro Sources*.

Handling Macro Sources for Dynamic JCL Generation

Jobs with the **JCL location** **Macro Natural source** (MAC) use the edit macro function of Entire Operations, which allows the use of variables anywhere in the JCL. Dynamic generation of JCL refers to the automatic substitution of these variables with their current values either when the job network is activated or when the job is submitted. The same job can thus have different JCL for different activation.

The **Macro** editor function must be used to generate the final JCL. The **Test** editor function can be used to test the generated JCL.

This section covers the following topics:

- Special Editor Functions for Macro Sources
- Escape Character for Natural Source Lines
- Parameter List

- Variables in Dynamically Generated JCL

Special Editor Functions for Macro Sources

The following special editor functions are available for a macro source:

Function	Description
Macro	Compile the macro JCL into an executable Natural object. This object is executed by Entire Operations during JCL load. It generates the executable active JCL. (This command is similar to the `STOW` command used for standard Natural programs.) **Note:** The **Macro** function can be executed in batch mode, e.g., for whole networks. See also ***Bulk Execution of MACRO Commands***.
Test	Generate JCL for testing purposes. The generated JCL is displayed in a new window. **Note:** 1. The macro source is not saved during execution of the **Macro** function. 2. The generated object is not compiled with the Natural `STOW` command during execution of the **Test** function. Since no changes are made to the original macro source and the existing generated object, the `TEST` command can also be performed by users who have read access only for the macro source.

Run Number used for the Test Function

The reserved run number used during the **Test** function is `-3`. During **Test**, a temporary active symbol table with this run number is used. This avoids any influence on the defined symbol table master. In the log display, this reserved run number is not displayed.

Escape Character for Natural Source Lines

Natural source lines must be preceded by the **activation escape character** of the job (see *Symbol Escape Characters* in the section *Symbol Table and Symbol Maintenance*).

The activation escape character at the beginning of a line distinguishes the line as a Natural statement from the JCL. The use of Natural statements provides full Natural functionality in dynamic JCL, including access to Adabas and Entire System Server. This means you do not need to learn any special control statements. All Natural statements used in dynamic JCL must be coded in structured mode.

Variables are user-defined and can be used in any part of the JCL. Variables are distinguished in the JCL by preceding them with an escape character: the activation escape character denotes variables to be substituted at activation time; the **submission escape character** denotes variables to be substituted at job submission time (see *Symbol Escape Characters* in the section *Symbol Table and Symbol Maintenance*).

Note: These escape characters can be changed by the system administrator. However, this should only be done if absolutely necessary, for example for countries with a different alphabet. It is not advisable to use escape characters which have a defined meaning in a particular operating system or which are already defined as escape characters, e.g., `$` in BS2000 or UNIX.

Parameter List

Each JCL for a job with the JCL location **Macro Natural source** (`MAC`) must start with the declaration of the common user exit parameter list `NOPXPL-A`:

```
§ DEFINE DATA PARAMETER USING NOPXPL-A
```

where the first character (here: `§`) is the **activation escape character** of the job.

The `P-CALL-PLACE` parameter in `NOPXPL-A` contains the value `MAC`.

See also ***Common User Exit Parameter Data Area NOPXPL-A*** and ***User Exit Types, P-CALL-PLACE*** in the section *User Exits*.

Variables in Dynamically Generated JCL

You can use these types of variables:

- A variable from the symbol table specified for the job;
- A variable from the parameter section (`P-...`);
- A local variable defined in this Natural program, which can be computed in your program (`L-...`);
- A Natural system variable (`TIME`, `DATE`, etc.) which is distinguished by an asterisk (`*`), for example, `*TIME` and `*DATE`;
- The variables `*TIMN`, `*TIMN6`, `*TIMA6` and `*TIMA7` which will be handled as described in the ***Table of Predefined Symbols***.

Note: All variables with prefixes other than `P-`, `L-` or `*` are assumed to be found in the symbol table. Symbol replacement itself (without embedding Natural statements) is also available for standard JCL (`JOB` (**Standard Job**) type jobs). See ***Symbol Replacement*** in the section *Symbol Table and Symbol Maintenance*.

This section covers the following topics:

- Sample JCL
- #GET-SYMBOL
- #SET-SYMBOL
- #SET-SYMBOL-M
- General Notes on #GET-SYMBOL, #SET-SYMBOL and #SET-SYMBOL-M

- Examples of Using #GET-SYMBOL, #SET-SYMBOL and #SET-SYMBOL-M

Sample JCL

The following sample JCL illustrates the use of variables in the dynamically generated JCL of a job with JCL location **Macro Natural source** (MAC):

```
§ DEFINE DATA PARAMETER USING NOPXPL-A
§ LOCAL    /* ... ALL LOCALS SHOULD START WITH 'L-'
§ 1 L-01    (A30)
§ 1 CLASS   (A01)     /* FROM SYMBOL TABLE, FOR #GET-SYMBOL
§ END-DEFINE
§ * ----------------------------------
#GET-SYMBOL CLASS
§ COMPRESS P-NETWORK P-JOB INTO L-01
//SNNOPEX  JOB  ,§P-OWNER,MSGCLASS=§MSGCLASS,CLASS=§CLASS
//STEP01   EXEC PGM=NOPCONTI,PARM='C=0004'
//STEPLIB  DD   DISP=SHR,DSN=§STEPLIB
//* DEVICE: §*DEVICE, INIT-USER: §*INIT-USER, TIME: §*TIME
//* L-01  : §L-01
§ IF CLASS = 'K'      /* SYMBOL USED IN STATEMENT -> #GET-SYMBOL
//* THE CLASS IS §CLASS
§ ELSE
//* ANOTHER MSGCLASS FOUND
§ END-IF
//*
```

The variables used in the JCL are assumed to have the following current values:

Symbol Table Variables

STEPLIB	NOP.SYSF.DEV.LOAD
CLASS	K
MSGCLASS	X

Natural System Variables

*DEVICE	BATCH
*INIT-USER	EORMON

Variables from the Parameter Section

P-NETWORK	EX131A
P-JOB	EX-1-24
P-OWNER	SN

When the job is activated, Entire Operations substitutes the variables with their current values. The following JCL is generated as a result:

```
//SNNOPEX  JOB  ,SN,MSGCLASS=X,CLASS=K
//STEP01   EXEC PGM=NOPCONTI,PARM='C0004'
//STEPLIB  DD   DISP=SHR,DSN=NOP.SYSF.DEV.LOAD
//* DEVICE: BATCH, INIT-USER: EORMON
//* L-01  : EX131-A EX-1-24
//* THE CLASS IS K
//*
```

#GET-SYMBOL

#GET-SYMBOL can be used within macro JCL to put an active symbol value into a local variable.

The symbol is taken from where it is found first in the symbol search path.

The syntax is as follows:

```
#GET-SYMBOL variable [value]
```

value is optional. If it is not specified, the variable is filled with the contents of a symbol with the same name.

value must be of the format A (alphanumeric). To convert *value* to a numeric field, use the Natural VAL system function.

```
#GET-SYMBOL variable(index)
```

With this syntax, an instance of a multiple value symbol can be obtained. The variable must be defined as a local array with a sufficient array size.

Example:

```
#GET-SYMBOL L-SYMBOL(L-IND)
```

See also *Examples of Using #GET-SYMBOL, #SET-SYMBOL and #SET-SYMBOL-M*.

#SET-SYMBOL

#SET-SYMBOL can be used within macro JCL to set an active symbol and its value from a local variable or string.

The symbol name itself can consist out of symbols, also partially and / or nested.

The symbol is set in the active symbol table which belongs to the active job.

For #SET-SYMBOL, a job must have a locally defined symbol table.

The syntax is as follows:

```
#SET-SYMBOL symbol [value]
```

value is optional and can contain one of the following values:

Value	Meaning	Example
none	Set the value of the symbol *symbol* to the content of a local variable with the name *symbol*.	`#SET-SYMBOL S1`
'*text*'	Set the value of the symbol *symbol* to '*text*'. '*text*' can consist out of symbols, also partially and / or nested.	`#SET-SYMBOL S2 'v002'`
variable	Set the value of the symbol *symbol* to the content of the local variable with name *variable*.	`#SET-SYMBOL S3 S30`

See also *Examples of Using #GET-SYMBOL, #SET-SYMBOL and #SET-SYMBOL-M*.

#SET-SYMBOL-M

#SET-SYMBOL can be used within macro JCL to set a symbol master and its value from a local variable or string.

The symbol is set in the symbol table master which belongs to the active job.

For #SET-SYMBOL-M, a job must have a locally defined symbol table.

The syntax is as follows:

```
#SET-SYMBOL-M symbol [value]
```

value is optional and can contain the same values as described for #SET-SYMBOL.

See also *Examples of Using #GET-SYMBOL, #SET-SYMBOL and #SET-SYMBOL-M*.

General Notes on #GET-SYMBOL, #SET-SYMBOL and #SET-SYMBOL-M

- The function are executed during JCL loading. Make sure that required active symbols exist at that time.
- Any symbol used within these functions must be preceded by the **activation escape character** of the job. The **submit escape character** can be used, too. In these functions, it will also be handled at JCL loading time.
- For #SET-SYMBOL and #SET-SYMBOL-M, text strings can contain blanks.
- Blanks are used as separators for the parameters of #GET-SYMBOL and #SET-SYMBOL.
- To avoid truncation, make sure that the receiving symbol or variable is at least as long as the symbol or the variable of the origin.

Examples of Using #GET-SYMBOL, #SET-SYMBOL and #SET-SYMBOL-M

The **activation escape character** used in the examples is the section sign (§).

Statement	Description
#GET-SYMBOL J	Move the contents of symbol J to the local variable J.
#GET-SYMBOL #J §BB	Move the contents of symbol BB to the local variable #J.
#GET-SYMBOL #J '§BB'	Move the contents of symbol BB to the local variable #J.
#GET-SYMBOL L-MULT '§?MV<M1,§I>'	Move the result of a symbol function call into the local variable L-MULT.
#SET-SYMBOL J	Set the active symbol J to the contents of the local variable J.
#SET-SYMBOL I #A	Set the active symbol I to the contents of the local variable #A.
#SET-SYMBOL AA '*text*'	Set the active symbol AA to the value *text*.
#SET-SYMBOL BB §I	Set the active symbol BB to the contents of symbol I.
#SET-SYMBOL BB '§I'	Set the active symbol BB to the contents of symbol I.
#SET-SYMBOL CC '§D.§E'	Concatenate symbol values of D and E (without a dot in between), and put the result into the active symbol CC.
#SET-SYMBOL CC '§D.xyz'	Concatenate symbol value of D and the string 'xyz' (without a dot in between), and put the result into the active symbol CC.
#SET-SYMBOL CC '§D..§E'	Concatenate symbol values of D and E (with one dot in between), and put the result into the active symbol CC.
#SET-SYMBOL D§E 'text'	If the active symbol E contains 'YY': Set the active symbol DYY to the value text.
#SET-SYMBOL-M DD '*value*'	Set the symbol master DD to the value *value*.
#SET-SYMBOL-M TAGYYMMDD '&!D<T,YYMMDD>'	Sets the result of the symbol function &!D<T,YYMMDD> (e.g. 150211) into the symbol master TAGYYMMDD.

Usage of Text Objects in JCL

You can include text objects in JCL. The text objects can have their own local parameters. Text objects can invoke other text objects, i.e., nesting is possible.

This feature is not limited to jobs with JCL location `MAC` (macro Natural source), but can also be used from within standard JCL.

This section covers the following topics:

- Inserting Text Objects into JCL
- Parameters for Included Text Objects
- Nested (Recursive) #EOR-INCL Statements
- Replacement of Parameters within the Text Object
- Examples of #EOR Statements

Inserting Text Objects into JCL

You can insert Natural text objects anywhere in your JCL.

The meta commands described here are stripped from the active JCL and the submitted JCL.

The syntax is as follows:

```
#EOR-INCL LOC=NAT LIB=library MEM=text-object
```

Parameters of the #EOR-INCL Statement

Parameter	Description
`LOC`	Location `NAT` Natural text object. This is the only location which is currently allowed for text objects.
`LIB`	Library of the text object.
`MEM`	Name of the text object.

If the text object cannot be read, the JCL generation is aborted with an error message.

Symbol replacement is possible within the `#EOR-INCL` statement.

Parameters for Included Text Objects

You may invoke included text objects with specific parameters.

For each parameter you want to pass to the text which is included by a #EOR-INCL statement, you must code a #EOR-PARM line in front of the #EOR-INCL statement.

These parameters are in effect only for the text included (and all nested inclusions) in the following #EOR-INCL statement.

The syntax is as follows:

```
#EOR-PARM parameter = value [parameter = value]...
```

Multiple parameters can be passed on the #EOR-PARM line. This is limited by the line size.

Parameter values implicitly have the format A (alphanumeric).

Parameter values may contain blanks. In this case, they must be included in apostrophes (' ') or double quotes (" ").

Nested (Recursive) #EOR-INCL Statements

You may use nested #EOR-INCL statements inside text objects which have already been included. These nested inclusions may have their own parameter lists.



Note: Be aware that the total number of inclusions for JCL is limited by the Natural editor buffer size for the Natural task executing.

Replacement of Parameters within the Text Object

Parameters for text objects can be used within the included text object like any other symbol. They have precedence over the symbols of the current symbol table of the job (and the other symbol tables in the search hierarchy).

Therefore, parameters for included text objects can temporarily override symbols with the same name. The scope of a parameter is only the included text object, and the text objects which are invoked from within this text object by nested calls.

The lifetime of parameters defined via #EOR-PARM is JCL load time only. Normal symbol replacement can be used within included text objects like everywhere else in the JCL.

Examples of #EOR Statements

Example 1:

```
...
#EOR-INCL LOC=NAT LIB=JCLLIB MEM=§OBJECT
...
```

Include the Natural text object the name of which is in the symbol `OBJECT` from the library JCLLIB.

Example 2:

```
...
#EOR-PARM DBID = 9
#EOR-PARM TEXT-1 = "a string with blanks"
#EOR-INCL LOC=NAT LIB=USERLIB MEMBER=BLOCK1
//* JCL statement
#EOR-PARM DBID = 10
#EOR-PARM TEXT-1 = "another string"
#EOR-INCL LOC=NAT LIB=USERLIB MEMBER=BLOCK1
...
```

Include the Natural text object `BLOCK1` from the library USERLIB. The parameters `DBID` and `TEXT-1` are passed to the text object with different values for the two invocations.

Locking of Natural Sources

If you start Natural with the profile parameter `SLOCK` set to `PRE`, a lock avoids that two or more users can concurrently edit a Natural source object from an Entire Operations, an Entire Operations GUI Client or within a native Natural environment.

If `SLOCK` is set to `POST` or `OFF`, no locking is performed but the Entire Operations editor detects that several users edit a source at the same time.

You can use the Natural system command `UNLOCK` to check or unlock locked Natural source objects.

For detailed information on `SLOCK` and `UNLOCK`, refer to the relevant sections in the Natural *Parameter Reference* documentation.

Additional Prerequisites for Source Locking

- In a UNIX or Windows environment, the Natural Development Server must be installed in the current Natural FNAT system file.

- If you want to avoid concurrent source editing among users in a local Entire Operations GUI Client environment, in addition to Natural, you also have to start the Natural RPC server with `SLOCK` set to `PRE`.

Avoiding Exceptions while Editing JCL Location (NAT, MAC)

In the Software AG Editor, any four-digit number prefixed with an opening parenthesis "`(`" must be used exclusively for line number referencing within a Natural program. Such notation is prohibited for any other use.



Caution: In the Software AG Editor, "`(####`" is reserved for Natural program line numbers only. Do not use for any other purpose.

32 Defining and Managing Job Conditions

- Use of Input and Output Conditions 300
- Maintaining Input Conditions for a Job 302
- Adding and Modifying a Master Input Condition 305
- Accessing another Network Defined for an Input Condition 310
- Input Condition References 310
- Global Conditions 313
- Input Condition with User Exit 314
- Input Condition: Multiple Suffixes 317
- Input Condition: File Existence 318
- Input Condition: Mailbox 321
- Input Condition: Symbol Value 322
- Input Condition: BS2000 User Switch 325
- Input Condition: BS2000 Job Variable 326
- Listing Jobs Linked to an Input Condition 328
- Deleting an Input Condition Definition 330

Logical conditions are dependencies between jobs. Jobs within a job network are linked by user-defined logical conditions. A logical condition can be added, deleted or modified. A logical condition can have either of two statuses that determine how Entire Operations is to continue processing: true (condition exists) or false (condition does not exist).

During execution of networks and submission of jobs, Entire Operations automatically checks the status of logical conditions and triggers system actions accordingly. Alternatively, logical conditions can be set by an **API routine** (see the relevant section).

All conditions are identified by name and a reference date to allow the Entire Operations Monitor to distinguish between the same event occurring on different dates. Condition names must be unique within a job network. Dates can be specified as relative dates or explicit dates. All relative dates are converted to real dates when the job is put in the active queue.

Apart from a name and reference date, the user can also assign a mailbox to a condition. Entire Operations will automatically notify each user of all pending conditions assigned to any mailboxes associated with his user ID.

Jobs are linked by defining the output condition (End-of-Job checking) of one job as the input condition (prerequisite) of the subsequent job.

Related Topics:

- ***Defining Schedule Dependencies for an Input Condition*** in the section *Schedule Maintenance*
- ***Maintaining Active Job Conditions*** in the section *Active Job Networks*
- *Logical Conditions* in the *Concepts and Facilities* documentation

Use of Input and Output Conditions

There are two ways of using logical conditions:

- As input conditions;
- As output conditions.

This section covers the following topics:

- Input Conditions
- Output Conditions

- Example of Job Linkage by Using Conditions

Input Conditions

Input conditions are prerequisites for job submission. Entire Operations does not submit a job until all input conditions and other prerequisites are set (fulfilled). Before job submission, all input conditions defined for the job are checked automatically by the Entire Operations Monitor. If you want the checking to be done by a Natural user exit, this routine must also be defined as an input condition.

An input condition can be set by the occurrence of an event detected by Entire Operations or manually by the user when **maintaining active job conditions**. It can also be set by a reply to a mailbox request.

You can set an input condition to true or false. The job then must wait until the condition is fulfilled before it starts running. This is useful, for example, to avoid that two or more jobs with the same input condition run at the same time. You can also specify whether an input condition is reset after job submission.

If no input condition is defined for a job, Entire Operations assumes a virtual true input condition. This means that this job can be submitted immediately at the (earliest) starting time defined for it, unless the job has other prerequisites such as resources.

It is possible to define a maximum of 40 input conditions per job. If you need more input conditions, you must use intermediate dummy jobs to collect the conditions. See also ***Job Execution as a Dummy Job***.

Input conditions can refer not only to the current run of a job network, but also to given time frames in the past or to previous runs.

You can also use an input condition to turn a job into a dummy job when it occurs. For further information on this topic, see ***Using a Dummy Job***.

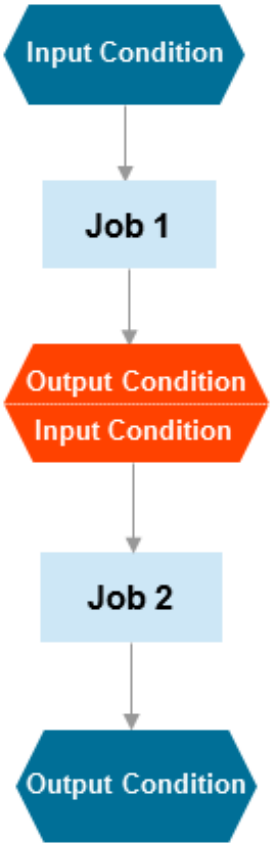
Output Conditions

Output conditions can be set or reset during End-of-Job checking of Entire Operations. For each job or job step (operating system job), you can specify any number of possible events. Each event can be associated with up to 20 output conditions. When any of these events occur, Entire Operations automatically sets the associated output conditions and starts those jobs which have these conditions as input conditions (see also ***Defining and Managing End-of-Job (EOJ) Checking and Actions***).

As in the case of input conditions, output conditions are defined by name and reference. Additionally, you can specify whether the output condition is to be set (to true) or reset (set to false) when the associated event occurs.

Example of Job Linkage by Using Conditions

The figure below illustrates a simple example of two jobs linked by logical conditions:

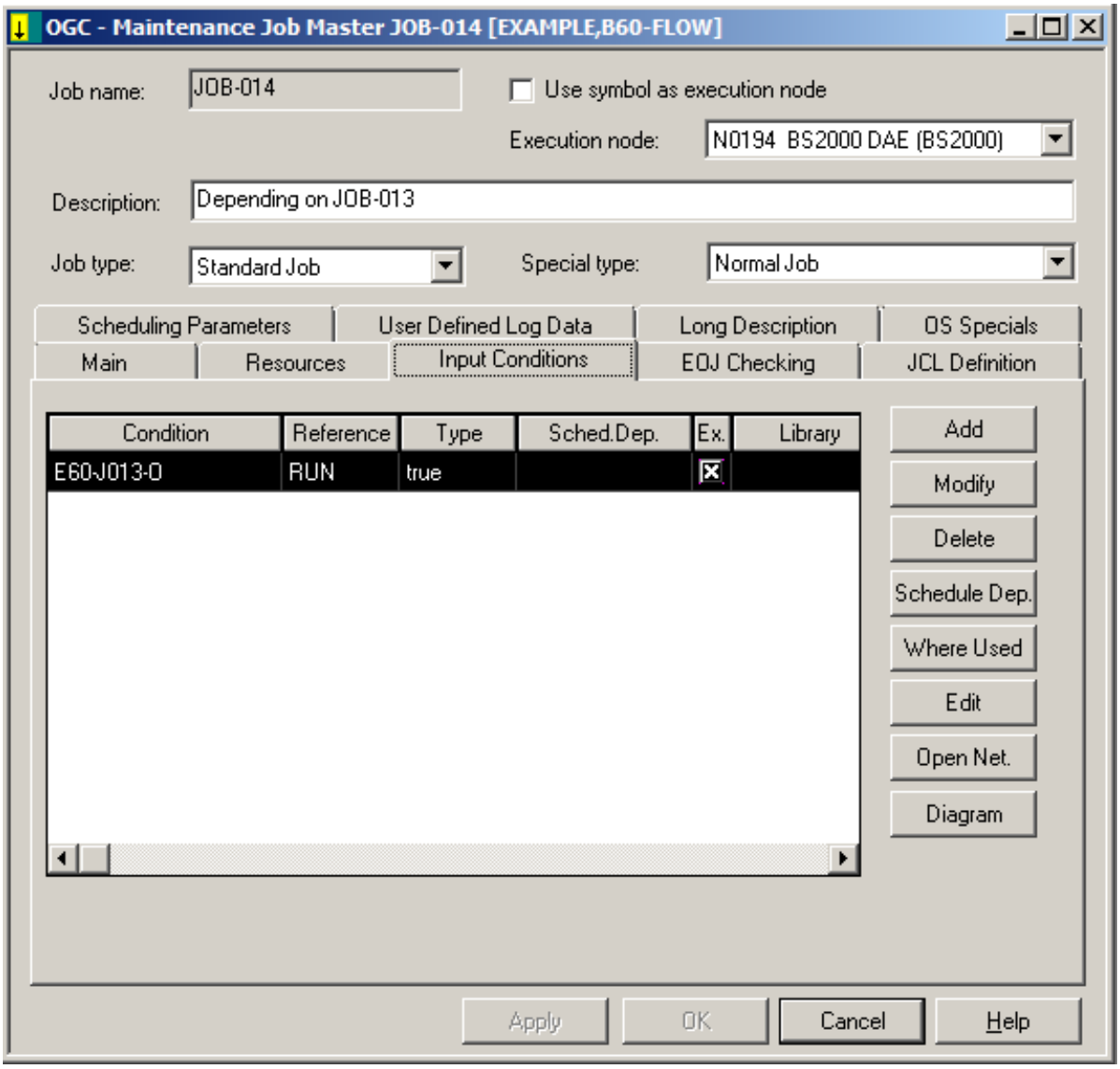


To link the two jobs: an Output Condition of Job 1 is defined as an Input Condition for Job 2.

Maintaining Input Conditions for a Job

To list input conditions

- In the **Maintenance Job Master window**, open the tabbed page **Input Conditions** like the example below:



All input conditions defined for the job are listed on the page.

The columns and functions available on the page are explained in ***Columns: Input Conditions Maintenance*** and ***Functions: Input Conditions***.

This section covers the following topics:

- Columns: Input Conditions

- Functions: Input Conditions

Columns: Input Conditions

The following table explains the column headings for the data listed on the **Input Conditions page** of the **Maintenance Job Master** window:

Column	Description	
Condition	User-defined condition name. See also *Restrictions for Condition Names* and *Global Conditions*.	
Reference	Reference date used to refer to a certain occurrence of the input condition. For possible entries, see *Possible References for Input Conditions* in the section *Input Condition References*.	
Type	The values shown in this column refer to the condition defined in the **Input Condition** window:	
	`true`	Condition must exist for the job to be submitted.
	`false`	Condition must not exist for the job to be submitted.
	`exclusive`	Exclusive use of the condition.
	`destruct.`	Condition is deleted after use.
	`extern +`	Condition from another network must exist.
	`extern -`	Condition from another network must not exist.
	`Exit`	Condition depends on the result of a user exit.
	`File +`	File must exist.
	`File -`	File must not exist.
	`User Sw +`	User switch must exist (BS2000 only).
	`User Sw -`	User switch must not exist (BS2000 only).
	`Job Var.`	Condition depends on a job variable (BS2000 only).
	`Symbol`	Condition depends on the value of a symbol in a symbol table.
	`mult.Sfx.`	Condition depends on multiple suffixes.
	`Mailbox +`	Condition must exist. It is prompted in the mailbox and must be set or reset to continue job execution.
	`Mailbox -`	It is prompted in the mailbox and must be set or reset to continue job execution.
	`Recov.tmp.`	Condition is used for recovery (only temporary created by the Entire Operations Monitor for active jobs only).
Sched.Dep.	If the condition is defined to be schedule-dependent, a short summary of the dependency appears in this column. For more information, see *Defining Schedule Dependencies for an Input Condition* in the section *Schedule Maintenance*.	

Column	Description	
Ex.	**x** mark in the table	Condition must exist.
	no mark in the table	Condition must not exist.
Library	Natural library where a defined user exit resides.	

Functions: Input Conditions

The following functions are available on the **Input Conditions page**:

Function	Description
Add	**Add an input condition definition**.
Modify	**Modify the input condition definition**.
Delete	**Delete the input condition definition**.
Schedule Dep.	Add or modify schedule dependency. See *Defining Schedule Dependencies for an Input Condition* in the section *Schedule Maintenance*.
Where Used	Display jobs for which the condition is used as an input or output condition. See *Viewing the Usage of a Prerequisite Resource*.
Edit	**Edit user exit to set input condition**.
Open Net.	Open the network definition for an external input condition. See *Accessing another Network Defined for an Input Condition*.
Diagram	View the network diagram for the input condition. For further information, see *Viewing and Maintaining a Job Network Diagram*.

Adding and Modifying a Master Input Condition

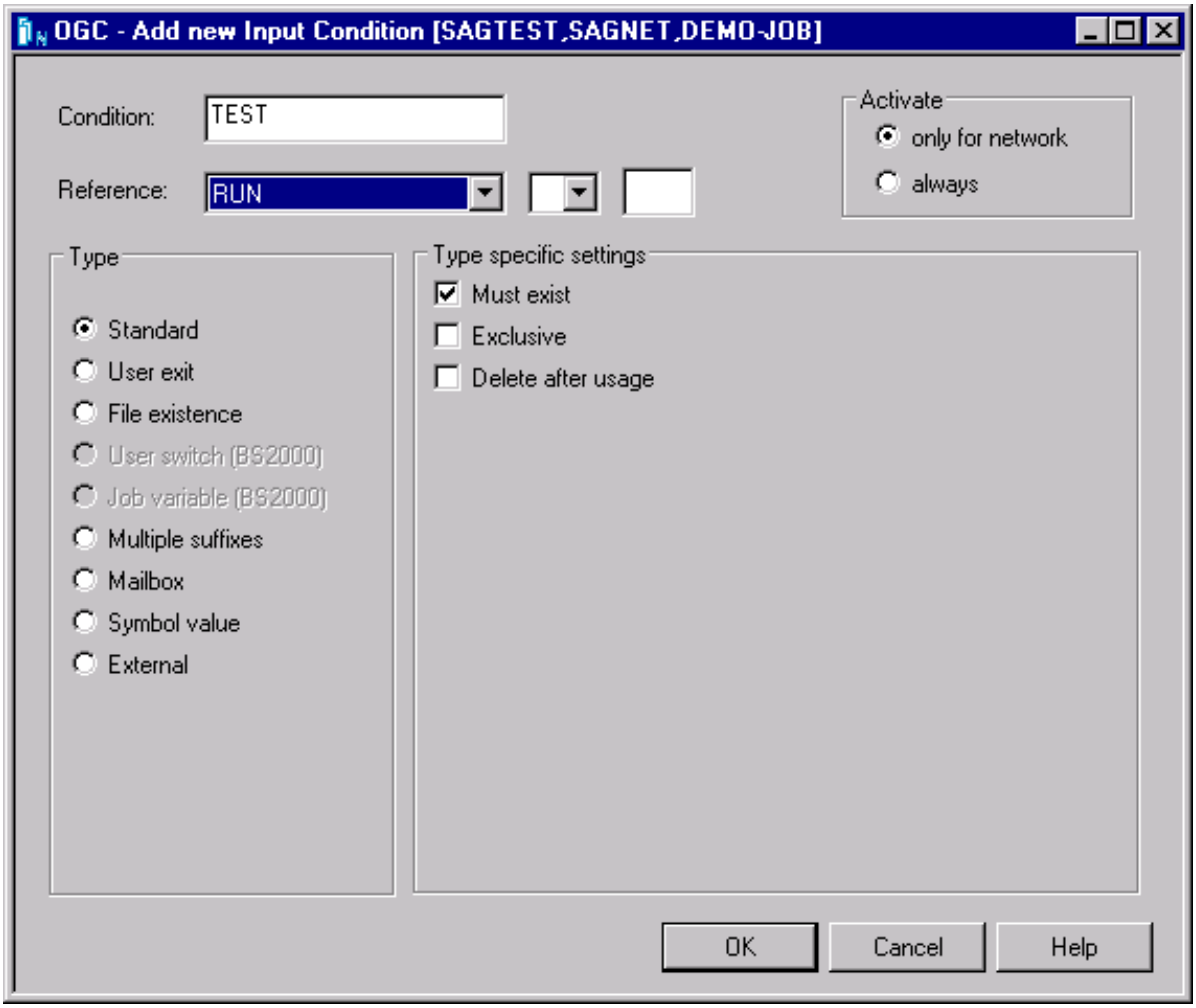
To add or modify an input condition

1 On the **tabbed page Input Conditions**, choose **Add** to create a new input condition.

Or:

On the **tabbed page Input Conditions** select an input condition from the table and choose **Modify** to change an input condition.

An **Input Condition** window like the example below opens:



2 Make your definitions. The fields and options in the window are described in *Fields and Selection Options: Input Condition*.

3 Choose **OK**.

The new input condition is now allocated to the job master.

Note: After an input condition has been defined or modified, a loop check is performed for the network. The same conditions apply as described in *Checking for a Loop in a Job Network* in the section *Network Maintenance*, with one exception: if a loop is detected in the job flow, no corresponding message appears.

- Fields and Selection Options: Input Condition

Fields and Selection Options: Input Condition

The input fields and selection options in the **Input Condition window** are described in the following table.

The fields and options provided in the **Type specific settings** section of the window depend on the **Reference** and **Type** selected for the condition.

Field	Description	
Condition	Name assigned to the condition. The condition name and its reference date uniquely identify an active condition. See also *Restrictions for Condition Names*.	
Run	Current run number (for active jobs only).	
Reference	Reference date to specify which occurrence of this definition the job uses. For possible selection options, see *Possible References for Input Conditions* in the section *Input Condition References*.	
Activate	*checked*	Input condition definition is always activated (for job activations as well).
	unchecked	Default. Input condition definition is activated for network activations only.
Type specific settings section:		
Must Exist	*checked*	Specifies that the condition must exist (be true) as a prerequisite to job submission.
	unchecked	Specifies that the condition must not exist (be false) as a prerequisite for job submission. Alternatively, this field also controls the setting of the condition according to the existence or non-existence of a file specified in the **File Existence** field (file or member in a file).
Exclusive	*checked*	Specifies that when this condition is in use, no other job can access this condition until it is free (job finished).
	unchecked	Any job can use the condition at any time. This feature is useful to prevent simultaneous execution of jobs with the same input conditions. Default.
Delete after usage	*checked*	Specifies that the condition is automatically reset after the job is submitted.

Field	Description	
	unchecked	Do not reset condition: later job runs can use this condition according to the **Reference** date. Default.
Activate	*checked*	Input condition definition is always activated (for job activations as well).
	unchecked	Input condition definition is activated for network activations only. Default.
Type section:		
Standard	Select this radio button (default) to set the standard condition usage options (must exit, exclusive and/or delete after usage).	
User Exit	Select this radio button if the condition is to be set by a user exit. For further information and the fields/options available in the **Type specific settings** section, see ***Input Condition with User Exit***.	
File Existence	Select this radio button to define an input condition dependent on the existence or non-existence of a file. For further information and the fields/options available in the **Type specific settings** section, see ***Input Condition: File Existence***.	
User Switch (BS2000)	User switch (BS2000 only). Select this radio button to define an input condition dependent on the existence or non-existence of a user switch. For further information and the fields/options available in the **Type specific settings** section, see ***Input Condition: BS2000 User Switch***.	
Job Variable (BS2000)	Job variable (BS2000 only). Select this radio button to define an input condition dependent on a comparison with the contents of a BS2000 job variable. For further information and the fields/options available in the **Type specific settings** section, see ***Input Condition: BS2000 Job Variable***.	
Multiple Suffixes	Select this radio button to define a symbol to be used for the active condition name. For further information and the fields/options available in the **Type specific settings** section, see ***Input Condition: Multiple Suffixes***.	
Mailbox	Select this radio button to define a user prompt to a mailbox. For further information and the fields/options available in the **Type specific settings** section, see ***Input Condition: Mailbox***.	

Field	Description
Symbol Value	Select this radio button to define an input condition dependent on a comparison with the value of a symbol in a symbol table. For further information and the fields/options available in the **Type specific settings** section, see *Input Condition: Symbol Value*.
External	Select this radio button to define an input condition dependent on another network. See also *Accessing another Network Defined for an Input Condition*.

Restrictions for Condition Names

The name of a condition can contain numbers and letters as required. The maximum name length is 20 bytes.

The following restrictions apply:

- Umlauts are not allowed.
- The use of special characters is restricted to the following:

```
-+/§#$_&
```

- Names of **global conditions** begin with a plus sign (+).
- The **activation escape character**, the **submission escape character** and a period (.) symbol delimiter are still accepted if symbol replacement is allowed for the relevant name fields.
- The following condition names are reserved for special purposes and may not be used for common conditions:

Reserved Condition Name	Explanation

`NET-BEGIN`
 	
`NET-END`
 	
`NET-END-NOTOK`
 	
`NET-END-OK`
	Used for subnetwork control.
	
	These reserved conditions are described in detail in the section *Link the Main Network*.

`P-STOPCYC - jobname`	If this condition is set in the active symbol table of a job with the special type **C**, the cyclic execution is stopped. For detailed information, see the field **Special Type** in the section *Fields: Job Master Definition*.
jobname`-MAX-RETRY`	The special condition *jobname*`-MAX-RETRY` is set by the Entire Operations Monitor when the message `EOR5316 (Recovery Retry Maximum:1: exceeded)` is issued during a **job recovery**.

Accessing another Network Defined for an Input Condition

To access another network defined for an input condition

- From the table on the **tabbed page Input Conditions**, select an input condition of the **Type** `extern` and choose **Open Net.**.

  The **Maintenance Network Master** window of the network defined for the input condition opens.

  If required, you can change the network definitions. The fields and tabs of the **Network Master window** are explained in ***Fields: Network Definition*** in the section ***Modifying a Network Definition***.

Input Condition References

To check an input condition, you must know which reference is meant. References can result in time or run number intervals.

The simplest reference is `RUN`, which refers to conditions set in the current network run. However, if you define an external input condition (which is not produced by the current network), you should always remember that different networks usually have different run numbers, which implies that `RUN` makes no sense in this case.

Run numbers are not assigned sequentially in chronological order. For references to previous network runs use `LNR`.

With the exception of `RUN`, all references described in this section also apply to **global conditions**.

- Possible References for Input Conditions
- Relative Values

Possible References for Input Conditions

The following table describes all references you can select from the drop-down list box next to the **Reference** field of the **Input Condition window**.

Reference	Unit of Relative Value	Description
AAC		Job uses condition only if there is at least one active run in the active database for the referred owner and network, see *Active Database*. A later activation of the referred network does not trigger an automatic recheck.
ABS		Job uses condition only if it is absolute. Absolute conditions are independent of run numbers and can exist only once under the same name.
ANY		Job uses any occurrence of the condition, except ABS (absolute), which has a reserved run number.
ANT		Job uses condition only if there is no entry in the active database for this owner, network and job.
DAT	Days	Job uses the condition as set by the network run on the current date.
Explicit date		A date entered in the format YYYYMMDD. Job uses condition only if set on the explicit date. The job then uses the condition as set by the network run on this date (does not apply when job can run more than once daily).
DST		Job uses the condition as set during the network run on the date specified as the job start time.
DUM		If this condition is satisfied, the job is started as a temporary dummy job. If this condition is not satisfied, the job is started normally. If several conditions with the reference DUM are defined for a job, only one condition must be satisfied for the job to be executed as a dummy. The condition can have a special dependency (for example, on a file). The active condition is also accepted if it has the reference ABS (absolute). If a job is started as a temporary dummy job because of a condition, then this is written to the log.
HRC	Hours	Job uses the condition only if it was set a defined number of hours before the check time of the condition. This reference can only be entered with a **relative hour value**. HRC-24 is the default value if this field is left blank and if the condition is set in a different network. (RUN is the default in the same network.)
LNR	Hours	■ If the condition was set by another network: Job uses the condition if it was set by the most recent run in the previous *nnn* hours. ■ If the condition was set by an earlier run of the same network: Job uses the condition if it was set by an earlier run in the previous *nnn* hours.

Reference	Unit of Relative Value	Description
		■ The condition is not set if an error occurred during the most recent or earlier run. This reference is recommended for constructing chains of networks and must be followed by a relative value (see ***Relative Values***).
LNT	Hours	This reference is used like LNR. Additionally, the condition is set to true if the creating job network did not have an active run in the previous *nnn* hours. Network runs that were already deactivated are indicated in the **accounting data**.
MON	Months	Job uses the condition as set by the network run of the current month.
NSD		Job uses the condition as set during the network run on the date specified as the network start time.
PDA	Days	Job uses the condition only if set on the production date of the job activation [minus relative days]. The production date end time can be defined in the Entire Operations default settings: see *Defaults for Time Ranges* in the *Administration* documentation. **Note:** This reference does not evaluate schedules or calendars.
PDS	Days	Job uses the condition only if set on production date of the job start [minus relative days]. The production date end time can be defined in the Entire Operations default settings: see *Defaults for Time Ranges* in the *Administration* documentation. **Note:** This reference does not evaluate schedules or calendars.
RCA		A job with multiple active subnetworks uses the same input conditions defined for a predecessor job that runs in the primary subnetwork. If **Multiple suffixes** are used as an input condition (see ***Input Condition: Multiple Suffixes***) for the predecessor, the multiple suffixes are appended to the job. If RCA is specified, the output condition of the corresponding predecessor job must be referenced with RCM: see RCM in the section *Field Descriptions: Output Conditions*.
RUN	Run numbers	Job uses the condition as set by the current network run. This is the default value if this field is left blank and if the condition is set in the same network. (HRC-24 is the default in a different network.)
WEK	Weeks	Job uses the condition as set by the network run of the current week.
WCC	Days	Real date, relative to the current day.

Reference	Unit of Relative Value	Description
WCW	Days	Calendar day, relative to the current day, in a linked calendar (workday).
WCS	Days	Schedule day, relative to the current day, in a linked schedule.
YYYYMMDD		Job uses condition only if it was set on the specified day. Example: 20220627

Relative Values

Some references can be followed by a minus (-) or plus (+) sign and a numeric offset. This is called a relative value. For example:

DAT-1	Refers to yesterday.
HRC-2	Refers to the previous 2 hours.
WEK-1	Refers to the previous week.

In this case, additional fields appear next to the **Reference** field where you can select the required sign (+ or -) and enter a number.

Global Conditions

Logical conditions are either set for a single job network or independently of any networks. Independent conditions are referred to as global conditions.

A global condition is not restricted to an owner, a network or a job but reflects the current value of a condition set for the given environment. It is defined once and can be used in several networks and job environments.

The following applies to a global condition:

- A global condition has the prefix + (plus sign).
- A global condition is assigned to the owner SYSDBA and to the network SYSDBA.
- A global condition gets the reference abs. (absolute). The reference RUN is accepted but is converted to abs. at runtime.

This document covers the following topics:

- Restrictions for Global Conditions

Related Topic:

- ***Maintaining Active Job Conditions*** in the section *Active Job Networks*.

Restrictions for Global Conditions

For global conditions, only the following **references** are allowed:

With the definition of an active condition	ABS, ANY, RUN
If used as input condition	HRC, DAT, PDA, WEK, MON, ABS, ANT, DUM, RUN, ANY
If used as output condition	ABS, RUN

Input Condition with User Exit

Input conditions can depend on the result of a user exit (P-CALL-PLACE set to ICO; see ***Parameters Used for Different Call Places***). If a user exit is defined for an input condition, Entire Operations automatically executes the exit when checking the status of input conditions during the prerequisite check. The user exit can perform any database or Entire System Server call to obtain the necessary information. This allows Entire Operations to react to complex or user-specific dependencies.

User exits are Natural subprograms and can be edited with the Entire Operations editor. See also the section ***User Exits***.

When defining a user exit as an input condition, consider the following:

- The Entire Operations Monitor sets the parameter field P-RC (return code) to 0 (zero) before the user exit is called.
- You can also specify an input condition user exit for an input condition with the reference DUM (execute the active job as a temporary dummy job). See ***Return Code Settings for an Input Condition User Exit*** for the meaning of the return codes used for the input condition reference DUM.

- Defining and Editing a User Exit for an Input Condition
- Fields: Input Condition User Exit
- Return Code Settings for an Input Condition User Exit
- Example of a User Exit

Defining and Editing a User Exit for an Input Condition

To define a user exit for an input condition

1 In the **Input Condition window**, select **User exit** from the **Type** section.

In the **Type specific settings** section, enter the name of the user exit and the name of the Natural library in which the user exit resides.

The input fields available are described in ***Fields: Input Condition User Exit***.

2 When finished, choose **OK** to save your entries and close the window.

To edit the user exit of an input condition

1 From the table on the **tabbed page Input Conditions**, select an input condition of the type **Exit** and choose **Edit**.

An editor window opens containing the source of a Natural subprogram like the *Example of a User Exit*.

(If no user exit is specified for the selected input condition, an appropriate message occurs instead.)

2 Modify the user exit as required.

For detailed information on handling user exits, see the section *User Exits*.

Fields: Input Condition User Exit

The input fields provided for a user exit in the **Type specific settings** section of the **Input Condition** window are described in the following table:

Field	Description
Natural Library	Name of the Natural library where the user exit resides. This library must be different from the Entire Operations system library.
Exit	Name of the user exit which sets the condition. The user exit coding must start with `DEFINE DATA PARAMETER USING NOPXPL-A`. For further information, see *Common User Exit Parameter Area NOPXPL-A*.

Return Code Settings for an Input Condition User Exit

When defining a user exit as an input condition for a job, you must set the return code as follows:

Input Condition Reference	Return Code	Meaning
`DUM`	`0`	Job executes as `dummy due to condition`.
	`99`	Job waits for the input condition, for example, until an ICO user exit sets another return code.
	other	Job executes normally.
other	`0`	Job executes normally.
	other	Job waits for the input condition, for example, until an ICO exit sets another return code.

Example of a User Exit

Below is an example of a user exit which sets an input condition:

```
*
Entire Operations


* USER EXIT TO SET AN INPUT CONDITION
*
* THIS ROUTINE CHECKS THE EXISTENCE OF A FILE, DEPENDING ON
* GIVEN PARAMETERS
*
DEFINE DATA PARAMETER USING NOPXPL-A
LOCAL                    /* LOCAL VARIABLES START HERE
1 CATALOG VIEW OF CATALOG        /* An Entire System Server VIEW
  2 NODE
  2 DSNAME
  2 ERROR-CODE
  2 ERROR-TEXT
*
1 #DSNAME         (A54)
END-DEFINE
* ----------------------
RESET P-RC                 /* ASSUME GOOD RETURN -> SET CONDITION
COMPRESS P-OWNER '.SYSF.SRCE' INTO #DSNAME LEAVING NO SPACE
CAT. FIND CATALOG WITH NODE = P-EXECUTION-NODE
    AND DSNAME = #DSNAME
  IF CAT.ERROR-CODE NE 0
    MOVE CAT.ERROR-CODE TO P-RC      /* BAD RETURN
    MOVE CAT.ERROR-TEXT TO P-RT
    ESCAPE ROUTINE
  END-IF
END-FIND        /* (CAT.)
END
```

The user exit must set a return code in `P-RC`.

If `P-RC` is not equal to `0`, the condition is reset (false) and the user is notified with a message. In the example above, the returned condition code (`ERROR-CODE`) sets (fulfills) the input condition for which the user exit is defined if the routine finds a file with the string *`owner`*`.SYSF.SRCE`.

Input Condition: Multiple Suffixes

If you define a symbol for multiple suffixes, its contents are separated, and the single fields are concatenated to the active condition name. These multiple conditions are used to wait until all parallel executing predecessors are finished.

The active conditions are created during activation of the job network. For example, if the condition name is `COND` and if the specified symbol contains `001003012`, the active conditions `COND001`, `COND003` and `COND012` are created.

To define an input condition dependent on multiple suffixes

1 In the **Input Condition window**, select **Multiple suffices** from the **Type** section.

2 In the **Type specific settings** section, enter a symbol name and select a symbol table/version, if required. See also ***Fields: Input Condition Multiple Suffixes***.

3 When finished, choose **OK** to save your entries and close the window.

This section covers the following topics:

- Fields: Input Condition: Multiple Suffixes

Fields: Input Condition: Multiple Suffixes

The input fields provided for multiple suffixes in the **Type specific settings** section of the **Input Condition window** are described in the following table:

Field	Description	
Always use Job Symbol Table	Specifies whether the symbol table defined for the job is used. Possible check box settings:	
	checked	The multiple suffix is taken from the symbol table defined for the job (default). (A symbol table defined for the network to which the job belongs is ignored.)
	unchecked	The multiple suffix is taken from the symbol table specified for this input condition only.
	Note: In the case of a job or network copy, it is recommended to select this check box. By this you make sure that always the defined symbol table of the job is used, even if it was changed in the job definition.	

Field	Description
Symbol Table	(optional) Name of the symbol table with the symbol that contains the suffix(es) to be used for the condition when the job network is activated. You must specify the same symbol table in the predecessor job definition. If no symbol table is specified, the suffix symbol is being searched in the symbol table hierarchy current at activation time.
Version	(optional) Version of the symbol table.
Symbol	Name of the symbol that contains the suffix(es) to be used for the condition when the job network is activated. You must specify the same symbol in the predecessor job definition.

Input Condition: File Existence

An input condition value can be dependent on the existence or non-existence of a file or of one of its members. The Monitor checks for the file or member on the job's **execution node** until the condition is satisfied.

To define an input condition that requires a file

1 In the **Input Condition window**, select **File existence** from the **Type** section.

2 In the **Type specific settings**, enter a file and a member name. The input fields and options are described in ***Fields: Input Condition: File Existence***.

3 When finished, choose **OK** to save your entries and close the window.

This section covers the following topics:

- Fields: Input Condition: File Existence

- Rules for File Names and File Checking

Fields: Input Condition: File Existence

The input fields provided for a file existence check in the **Type specific settings** section of the **Input Condition window** are described in the following table:

Field	Description	
File	Name of the file that must or must not exist. If the file is not cataloged, specify the volume serial number in the format *file/volume*. **Note:** When entering a file name, remember to observe the rules for upper and lower case which are specific to some operating systems. See also ***Rules for File Names and File Checking***.	
Member	(Optional field) If the input condition is dependent on the existence or non-existence of a member in the file specified in the **File** field, enter the member name. **Using a Wildcard in the Member Name:** A wildcard (*) can be used at the string end. The condition is set (or not set) if at least one member is found. **Note:** Only specify a member where necessary and possible. If this field is left blank, the existence of the whole file is checked. See also ***Rules for File Names and File Checking***.	
Must exist	Possible check box settings:	
	checked	The file (or member) must exist as a prerequisite to job submission.
	unchecked	The file (or member) must not exist as a prerequisite.

Rules for File Names and File Checking

The following rules apply when specifying a file as an input condition check:

BS2000 Files

The condition is satisfied only if the file is closed. For opened BS2000 files, the condition is not satisfied.

Migrated (archived) Files

Migrated (archived) files are recognized like files that are actively used. If a member is included in the file existence check, the active job is set to a permanent error, with the error text `Prerequisite File Check - Library containing member is archived`.

Entire System Server Node used for File Check

The node used for the file check is always the **execution node** of the job. The file is checked with the access rights of the **Submit User ID** (on UNIX and Windows: submit and submit group).

If you must check a file on a different node, use a predecessor dummy job with a different **execution node** and/or **Submit User ID** for this purpose.

Variable File Name: Using Escape Characters

The fields **File** and/or **Member** can contain symbols preceded by an **activation escape character**.

If the activation escape character is used, symbol replacement is performed during the first existence check.

Symbol replacement can be used, for example, for:

- file generation groups;
- changing input files;

After successful symbol replacement, these fields will contain the replaced value in the active job. This reduces the effort with symbol replacements.

The symbols are taken from the active symbol table assigned to the job. The symbol replacement in the file name is performed only once and the result is written back to the active input condition definition for further check. A missing symbol causes a permanent error.

It is also possible to use the **submission escape character**. In the case of an unsatisfied condition, the symbol replacement in the file name is performed before each file check. The result is not written back. This allows more flexible use of symbols, but may cause more system overhead.

File in Use

The case `file in use` is handled as a temporary error. The file check is repeated as long as the file is in use. The waiting job is not sent to passive wait.

z/OS: HSM Migrated Libraries

The following applies only if the operating system of an Entire System Server node is z/OS, and if the Entire System Server version is greater than or equal to Version 3.2.1. The Entire Operations Monitor performs the initialization of a file recall.

The file member check is repeated in intervals of two minutes, until the file is reloaded. A reload is not initiated if the file check is on file level only.

Input Condition: Mailbox

Mailboxes are defined to the system and assigned to user IDs by using the appropriate context function of the **Mailbox Definition** metanode. For more information on defining mailboxes, see *Mailbox Definition* in the *Administration* documentation.

For more information on how mailboxes can be used, see ***Working with Mailboxes***.

- Using Mailboxes with Input Conditions
- Defining an Input Condition of the Type Mailbox

Using Mailboxes with Input Conditions

Each logical condition can be assigned to a mailbox.

- If the condition is the only one pending (unfulfilled) and is therefore delaying the start of the subsequent job, a message is automatically sent to the mailbox.
- If an input condition is dependent on manual action(s), a message is sent to a mailbox that asks a user to confirm completion of the action(s).

Each user linked to this mailbox receives this message.

A user can be associated with up to ten mailboxes.

Defining an Input Condition of the Type Mailbox

To send a message to a mailbox for an input condition that is not satisfied during network execution

1 In the **Input Condition window**, select **Mailbox** from the **Type** section.

2 In the **Mailbox** field of the **Type specific settings** section, enter the name of the mailbox to which you want to send a message.

Or:

Select a name from the drop-down list box next to the **Mailbox** field.

3 When finished, choose **OK** to save your entry and close the window.

Input Condition: Symbol Value

An input condition can be dependent on a comparison with the contents of a symbol (symbol value) or the substring of a symbol value in a symbol table. The Monitor checks the value of the symbol on the job's **execution node** until the condition is satisfied.

You can specify the **instance** of a symbol table to be used for the symbol check: the active symbol table or the symbol table master .

This section covers the following topics:

- Defining an Input Condition of the Type Symbol Value
- Fields: Input Condition Symbol Value
- Nested Symbol Evaluation

Defining an Input Condition of the Type Symbol Value

To define an input condition that depends on a symbol value

1 In the **Input Condition window**, select **Symbol value** from the **Type** section.

In the **Type specific settings** section you can enter the symbol to be compared and further parameters to specify the symbol.

The input fields available are described in *Fields: Input Condition Symbol Value*.

2 When finished, choose **OK** to save your definitions and close the window.

Fields: Input Condition Symbol Value

The input fields of the **Type specific settings** for the type **Symbol value** in the **Input Condition window** are described in the following table:

Field	Description
Symbol	Valid symbol name. **Predefined symbols** can also be used. The symbol name itself can contain a symbol . The symbol replacement will be done with the escape characters of the using job. The value of this symbol, or a part of it, is to be compared with the given value.

Field	Description	
Symbol table	Valid symbol table name. If you leave this field blank, the symbol search procedure starts with the active symbol table of the job. Otherwise, the active symbol table with this name is searched instead of the symbol table of the job. If the symbol is not found there or in the caller's symbol tables, the owner's symbol master table with this name is searched too. See also ***Symbol Table Types and Symbol Search Order*** in the section *Symbol Table and Symbol Maintenance*.	
Version	Symbol table version. Possible version names:	
	`(current)`	Current version at determination date.
	`(nv)`	Same version as network version.
	`(svn)`	Symbol table version of network.
	`(svj)`	Symbol table version of job.
	`(unnamed)`	Unnamed version (without name).
Symbol instance	Instance of the symbol table where to perform the symbol value check. Valid selection options:	
	Check symbol in the symbol table active	Perform symbol check in the active symbol table (default). This setting has no effect on the symbol tables SYSDBA/A and *owner*/A. They only exist as symbol table masters, and are therefore always checked as symbol table masters. (See also ***Symbol Table Types and Symbol Search Order*** in the section *Symbol Table and Symbol Maintenance*.)
	Check symbol in the symbol table master	Perform symbol check in the symbol table master.
Position	Position of the substring of the symbol value to be checked. (Checked only if **Format** is set to `A`.) Possible values: 1 to 120 characters.	
Length	Length of the substring of the symbol value to be checked. (Checked only if **Format** is set to `A`.) Possible values: 1 to 120 characters.	

Field	Description	
Format	Format in which the substring of the symbol value is to be checked against the comparison string. Possible selection options:	
	A	Alphanumeric.
	D	Date in the format YYYYMMDD. See also *Date and Time Formats*.
	N	Numeric (zoned).
Symbol is	Comparison operator. Specify a logical operator for the comparison of the defined symbol against the comparison string (see below). Possible selection options:	
	= or EQ	Code is equal to specified value.
	>= or GE	Code is greater than or equal to specified value.
	> or GT	Code is greater than specified value.
	<= or LE	Code is lower than or equal to specified value.
	< or LT	Code is lower than specified value.
	<> or NE	Code is different from specified value.
compared to *comparison string*	Comparison string. In the input lines below **compared to**, enter the string or field to be compared with the substring of the symbol value. The strings are compared in the defined **Format**. Symbol replacement is possible in this field. ■ If an activation escape character is used, the replacement is performed once during activation. A symbol replacement error is treated as a permanent error in this case. ■ If a submission escape character is used, the replacement is performed directly before each prerequisite check. This causes more system overhead. A symbol replacement error is treated as temporary error in this case.	

Nested Symbol Evaluation

The symbol value may contain other (nested) symbols, prefixed by both activation escape character and submission escape character.

Symbols prefixed by the activation escape character are evaluated only once, at job activation.

Symbols prefixed by the submission escape character are evaluated at each prerequisite check of the active job.

Input Condition: BS2000 User Switch

An input condition value can be dependent on the existence or non-existence of a BS2000 user switch. The Monitor checks for the user switch on the job's **execution node** until the condition is satisfied.

This section covers the following topics:

- Defining an Input Condition of the Type User Switch
- Fields: Input Condition User Switch

Defining an Input Condition of the Type User Switch

To define an input condition that depends on a user switch

1 In the **Input Condition window**, select **User switch (BS2000)** from the **Type** section.

In the **Type specific settings** section, enter a user switch and BS2000 user ID.

The input fields available are described in *Fields: Input Condition User Switch*.

2 When finished, choose **OK** to save your entries and close the window.

Fields: Input Condition User Switch

The input fields of the **Type specific settings** for the type **User switch (BS2000)** in the **Input Condition window** are described in the following table:

Field	Description	
User Switch	Number of a user switch.	
BS2000 User ID	BS2000 user ID to which the specified user switch belongs.	
Must exist	Possible check box settings:	
	checked	The user switch must exist as a prerequisite to job submission.
	unchecked	The user switch must not exist as a prerequisite.

Input Condition: BS2000 Job Variable

An input condition can be dependent on a comparison with the contents of a BS2000 job variable. The Monitor checks for the job variable on the job's **execution node** until the condition is satisfied.

This section covers the following topics:

- Defining an Input Condition of the Type Job Variable
- Fields: Input Condition Job Variable (BS2000)
- Using Symbols

Defining an Input Condition of the Type Job Variable

To define an input condition that depends on a job variable

1 In the **Input Condition window**, select **Job variable (BS2000)** from the **Type** section.

In the **Type specific settings** section, enter a job variable and further parameters to specify the input condition.

The input fields are described in *Fields: Input Condition Job Variable (BS2000)*.

2 When finished, choose **OK** to save your entries and close the window.

Fields: Input Condition Job Variable (BS2000)

The input fields provided for a job variable in the **Type specific settings** section of the **Input Condition window** are described in the following table:

Field	Description	
Job variable	Enter the name of a valid BS2000 job variable. ■ If the specified job variable does not exist, a content comparison is done by Entire Operations, when the Monitor detects that the job variable has been created. ■ If a job variable does not exist, a job cannot be declared "dummy due to condition". Instead, the Entire Operations Monitor waits until the job variable exists and then performs the check. ■ If the job variable is specified without an explicit user ID, the job default BS2000 user ID is used as prefix. ■ Symbol replacement: see ***Using Symbols***.	
Position	Enter the position of the substring of the job variable value to be checked. Possible values: 1 to 253.	
Length	Enter the length of the substring of the job variable value to be checked. Possible values: 1 to 253.	
Format	Specify the format in which the substring of the job variable value is to be checked against the comparison string. Possible selection options:	
	`A`	Alphanumeric.
	`N`	Numeric (zoned).
Variable is	This is the comparison operator. Specify a logical operator for the comparison of the defined job variable substring against the comparison string (see below). Possible selection options:	
	`=` or `EQ`	Code is equal to specified value.
	`>=` or `GE`	Code is greater than or equal to specified value.
	`>` or `GT`	Code is greater than specified value.
	`<=` or `LE`	Code is lower than or equal to specified value.
	`<` or `LT`	Code is lower than specified value.
	`<>` or `NE`	Code is different from specified value.
compared to *comparison string*	In the input lines below **compared to**, enter the string or field to be compared with the substring of the job variable value. The strings are compared in the defined **Format**. The content of this field is compared with the substring of the job variable value, or it is inserted into the substring of the job variable value. The content is treated as blank if `''` (2 single quotes, no space) or `' '` (single quote, space, single quote) is defined. The comparison is made in the defined format. Symbol replacement: see ***Using Symbols***.	

Field	Description
Password	(Optional field) If the job variable is read password-protected, specify the password here.

Using Symbols

Resolving symbols in the job variable name produces the same behavior as resolving symbols in the job variable value:

- If the activation escape character is used:
  - The symbol is replaced once during the job activation.
  - The active job variable name is the resolved string.
- If the submission escape character is used:
  - The symbol is resolved during each performed prerequisite check.
  - This allows symbol setting shortly before the usage.



Note: The submission escape character option causes more system overhead.

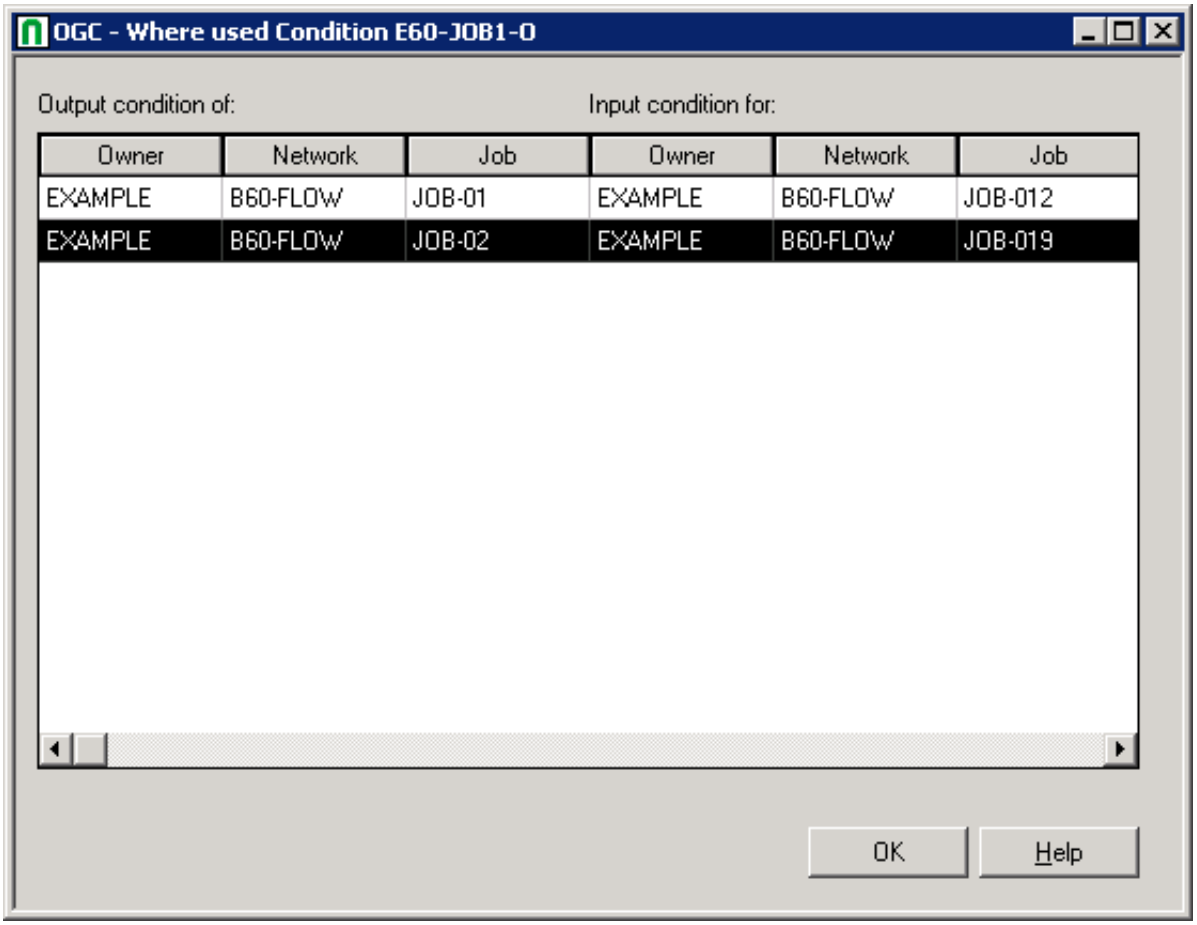
Listing Jobs Linked to an Input Condition

You can list jobs that also use a selected input condition as an input condition, or that use this input condition as an output condition (see also *Defining Output Condition Actions*).

To list jobs linked to a condition

- From the table on the **tabbed page Input Conditions**, select an input condition and choose **Where Used**.

  A **Where used Condition** window like the example below opens:



The window displays the name of the selected condition and two lists of jobs:

- one lists the jobs which use the condition as input condition;
- the other lists the jobs which use the condition as output condition.

The jobs are listed according to owner, network and job name.

Note: For an active input condition, a similar window opens: see the **Active Usage Condition** window described in the section *Active Job Networks*.

Deleting an Input Condition Definition

To delete an input condition

1 On the **tabbed page Input Conditions**, select an input condition in the table and choose **Delete**.

2 Select an input condition in the table.

3 Choose the **Delete** button.

The input condition is now deleted for this job master.

33 Handling Prerequisite Resources for a Job

- Use of Resources and Resource Allocation 332
- Listing Prerequisite Resources Defined for a Job 336
- Viewing the Usage of a Prerequisite Resource 337
- Displaying, Modifying and Adding a Prerequisite Resource Definition 338
- Columns and Fields: Prerequisite Resource Definitions 339
- Deleting a Prerequisite Resource Definition 340

The functions described in this section are used to list and define single or multiple resources for a job that are a prerequisite for submitting the job.

Related Topics:

- *Resources* (maintenance of resource masters) - *Administration* documentation
- ***Viewing and Modifying Resources Used by Active Jobs*** - *Active Job Networks*

Use of Resources and Resource Allocation

You can specify certain amounts of a defined resource master as a prerequisite for job submission. Entire Operations does not submit the job until the specified amount is available.

Before you can define a resource as a prerequisite for a job, it must be defined as resource master to Entire Operations. You define resources to Entire Operations by using the maintenance functions of the **Resource Master** metanode described in the *Administration* documentation.

Resources can be

- Quantitative or absolute;
- Reusable or not reusable.

Some examples of resources are listed below:

Resource	Type
Print forms	Quantitative, not reusable
Main storage	Quantitative, reusable
Line to a remote machine	Absolute
Availability of a device	Absolute

This section covers the following topics

- Scope of Resource Allocation
- Ordering of Resource Allocation
- Resource Amount Determination by User Exits
- Periods of Resource Allocation and Deallocation
- Resource Allocation Modes

- Resource Deallocation Modes

Scope of Resource Allocation

Prerequisite resources are allocated for all **job types** (including) and for all kinds of temporary dummy jobs. You can set an **allocation mode** to disallow resource allocation for temporary dummy jobs.

Ordering of Resource Allocation

The following rules apply for the ordering of resource allocations:

1. If a resource is requested by the same owner, network, job, but different runs (at the same time), the active job with the lowest **run number** or earliest activation time will get the resource first.
2. If several different jobs of one active network or of several active networks wait for the same resource: The available quantity of the resource will be allocated to as many as possible jobs in parallel, but under the restriction of item 1.

If a resource waiter with a higher priority is found during a prerequisite resource check, the message `Res. resource available, other jobs have priority` will be written to the log. It is followed by `Res. resource - waiting with higher priority`: and one or several lines with context information.

Resource Amount Determination by User Exits

The available amount of a resource can be determined by the usage of an exit. A resource master determination exit (described in the *Administration* documentation) will be invoked by the Entire Operations Monitor before prerequisite resource checks.

Resource master amounts can be modified by API calls. Note that this is possible only if the amount is not determined by a resource master determination exit. For more information, see ***Handling of Resource Allocations (NOPURE2N)*** in the section *List of Available Entire Operations API Routines*.

Periods of Resource Allocation and Deallocation

A resource is usually allocated for the duration of job execution.

If you want to allocate a resource for a longer time, you can change the resource deallocation mode set for a job as described in ***Resource Deallocation Modes***.

The reasons and rules for resource allocation or deallocation are described in the following section:

- **Retention period of condition reached**

  A resource is deallocated if the retention period for an active condition is reached.
- **Deallocation mode set**

If you want to allocate a resource for a longer time, you can change the resource deallocation mode set for a job as described in ***Resource Deallocation Modes***.

- **Deactivated network or job**

  If an active network or job is deactivated, all resources allocated by it are deallocated too.

  This is performed regardless of the defined **deallocation mode**.

  This means that even resources with deallocation mode `K` (keep until explicit release) are released in such cases.

- **Database cleanup performed**

  Resources are deallocated when an automatic cleanup of the active database is performed (see also *Defaults for Time Ranges* described in the *Administration* documentation).

- **Allocation kept for jobs that ended not ok**

  It is possible to prevent a resource from deallocation if a job ended `not ok`. This can be used, for example, to keep a resource during a recovery for an unsuccessful run.

  See the option **Deallocate if job not ok** in the **Resource Prerequisite window** described in ***Columns and Fields: Prerequisite Resource Definitions*** in the section *Job Maintenance*.

- **Forced deallocation**

  You can force freeing of a resource allocation.

  All current resource allocations can be inquired in usage lists. From these active resource usage lists it is possible to force the deallocation of a given resource allocation.

  For further information, see *Listing Jobs Currently Using a Resource* in the section *Resources* in the *Administration* documentation.

  Use this feature with care. Be aware that one or several active jobs may be started immediately, which were way for this resource.

- **Allocation or deallocation determined by API**

  Resources can be allocated for a job by an **API call**. These additionally allocated resources are handled in the same way as defined prerequisite resources.

  Resources can be deallocated by **API calls**. Preliminary releases of resources are allowed for deallocation modes.

  For more information, see ***Handling of Resource Allocations (NOPURE2N)*** in the section *List of Available Entire Operations API Routines*.

All resource allocations and deallocations are logged (see also the section ***Log Information***).

Resource Allocation Modes

This section describes the resource allocation modes that can be set for the prerequisite resource defined for a job (see also *Displaying, Modifying and Adding a Prerequisite Resource Definition*). Allocation modes are also indicated on resource lists.

Mode/Selection Option	Explanation
A **Always**	Allocate the resource (default).
T **Only if the active job does not become a temporary dummy job**	Do not allocate the resource if the job executes as a **temporary dummy job**.

Resource Deallocation Modes

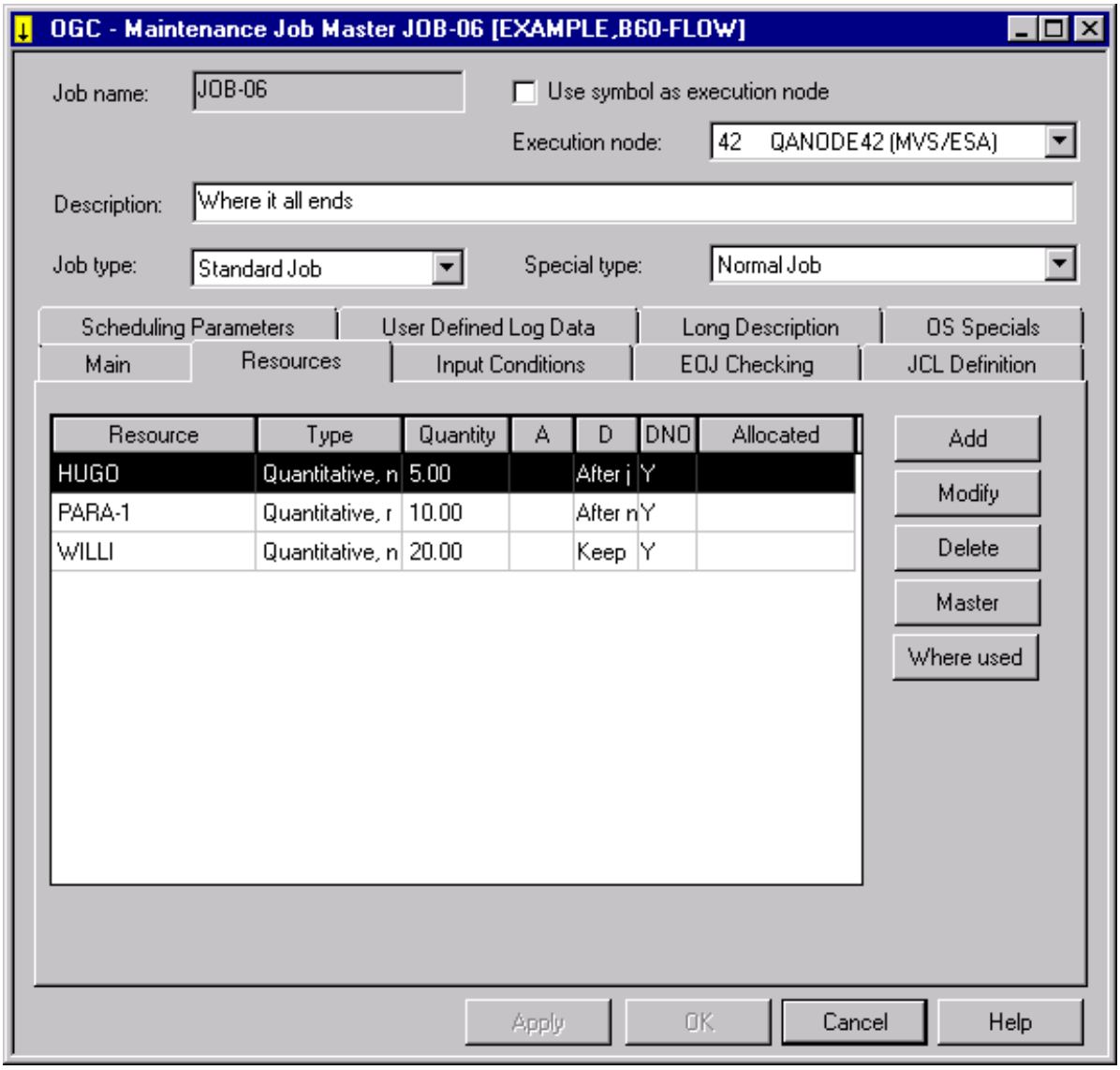
This section describes the resource deallocation modes that can be set for the prerequisite resource defined for a job (see also *Displaying, Modifying and Adding a Prerequisite Resource Definition*). Deallocation modes are also indicated on resource lists.

Mode/Selection Option	Explanation
J **After job termination**	Release resource at job termination (default). The resource is released when the allocating job terminates.
N **After network termination**	Release resource at network termination. The resource is released when the Entire Operations Monitor detects that all jobs of a job network are terminated. The resource is released also if a currently executing network is being deactivated. **Note:** To override the automatic detection of `network ended ok`, you must set the **reserved condition** `NET-END-OK` at least once in your network.
E **Keep until network termination**	Release resource at network termination, but release earlier after an erroneous job. The resource is kept until network termination. If any job ends `not ok`, it is released immediately after this job's termination.
K **Keep until explicit release**	Keep resource until manual release. The resource is not released automatically at job or network termination. If the retention period for active conditions is reached, the resource is released automatically.

Listing Prerequisite Resources Defined for a Job

To list all prerequisite resources defined for a job

- In the **Maintenance Job Master window**, open the tabbed page **Resources**:



The table in the window lists all resources defined as prerequisites for the job. The table columns are described in *Columns and Fields: Prerequisite Resource Definitions*.

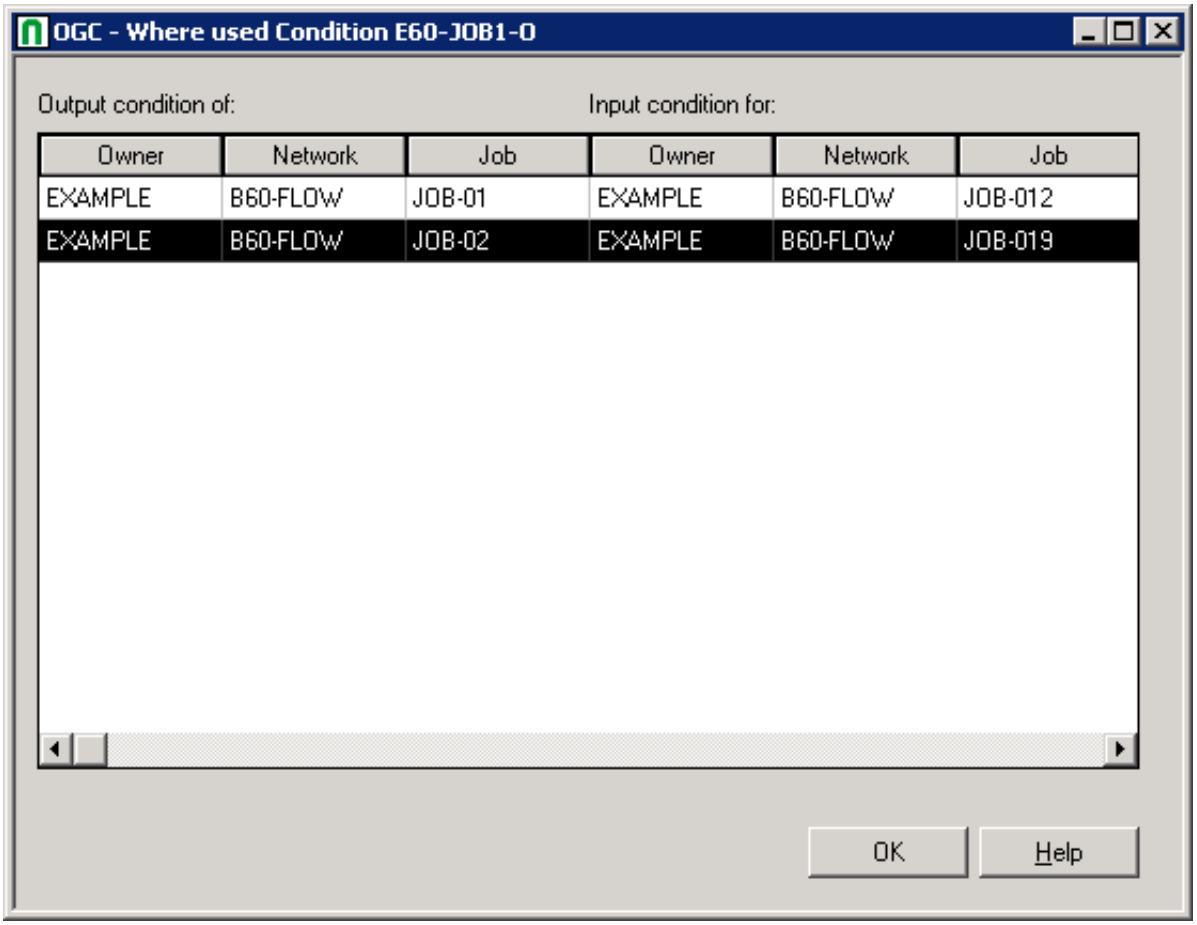
Viewing the Usage of a Prerequisite Resource

To find out where a predefined resource is used

1 From the table on the **Resources** page, select the required resource.

2 Choose **Where Used**.

A **Where used Condition** window like the example below opens:



If invoked for a job master definition: Shows the usage of this resource as a prerequisite in all job master definitions.

If invoked for an active job definition: Shows the current usage of this resource by active jobs.

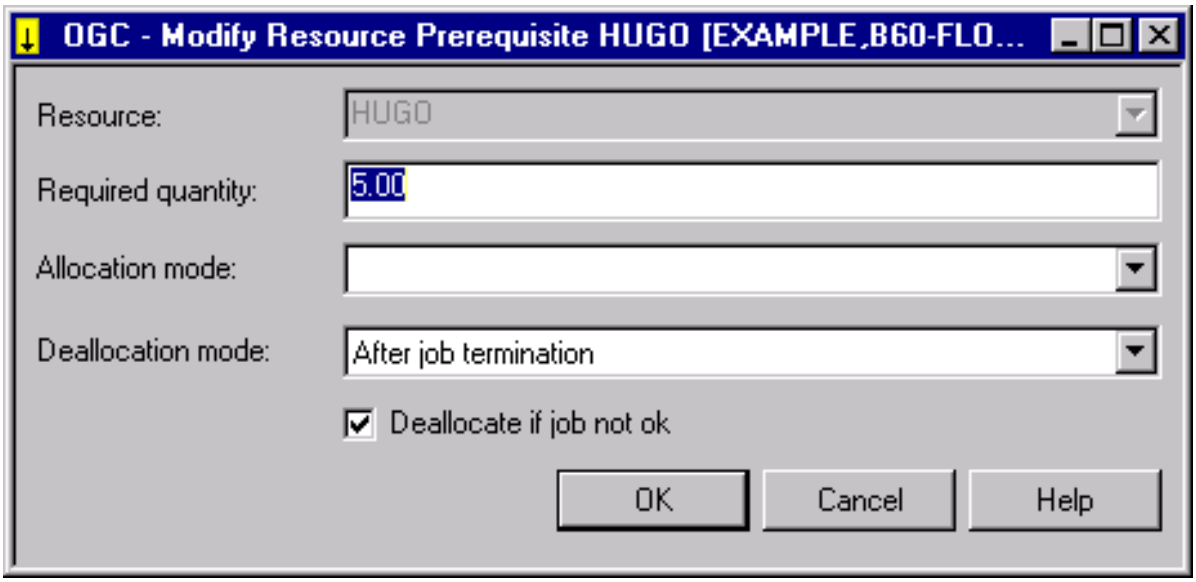
The fields and columns contained in the window correspond to the fields and columns of the **Defined in Jobs** page. They are described in *Field and Columns: Resource Defined in Jobs* in the section *Resources* in the *Administration* documentation.

Displaying, Modifying and Adding a Prerequisite Resource Definition

To display or modify the prerequisite resource definition of a job

1 From the table on the **tabbed page Resources**, select the resource definition you want to view or change and choose **Modify**.

A **Resource Prerequisite** window like the example below opens:



2 Make your definitions. The input fields and selection options available in the window are described in *Columns and Fields: Prerequisite Resource Definitions*.

3 Choose **OK** when you are finished.

To add a prerequisite resource definition for a job

1 On the **tabbed page Resources**, choose **Add**.

A **Resource Prerequisite window** like the previous example opens.

2 Make your definitions. The input fields and selection options available in the window are described in *Columns and Fields: Prerequisite Resource Definitions*.

3 Choose **OK** when you are finished.

Columns and Fields: Prerequisite Resource Definitions

The following table explains the columns on the **Resources page** and the corresponding input fields/options of the **Resource Prerequisite window**:

Column	Input Field	Description
Resource	**Resource name**	Name of the resource as defined in Entire Operations and listed in the **Resource Master List** window (see *Resources* in the *Administration* documentation).
T	n/a	Resource type as defined for the resource master: see the description of the **Type** field in the section *Resources* (*Administration* documentation).
Quantity	**Required Quantity**	Resource quantity required for job execution. Input field: Enter the resource quantity required for job execution. The required quantity must be greater than or equal to zero. A required quantity that equals zero has the same effect as if the prerequisite resource would not be defined. Entire Operations does not submit the job until this amount of resource is available.
A	**Allocation Mode**	Allocation mode of this prerequisite resource: see ***Resource Allocation Modes***.
D	**Deallocation Mode**	Deallocation mode of this prerequisite resource: see ***Resource Deallocation Modes***.
DNO	**Deallocate if Job not ok**	Applies to **deallocation mode** **After job termination** only. Deallocate if job is `not ok`:
		`Y` — The resource is always deallocated after job termination, even if the job ended `not ok`.
		`N` — The resource is not deallocated if the job ended `not ok`. (However, the job will still be deallocated at network termination.)
		Input option: select this check box if you want to deallocate the job.
Allocated	**Allocated**	Applies to active prerequisite resources only. Date and time when the resource was allocated for the current job. If the resource has not yet been allocated or if it has already been deallocated, this field is empty. See also ***Date and Time Formats***.

Deleting a Prerequisite Resource Definition

To delete a resource definition

- From the table on the **tabbed page Resources**, select the resource definition you want to remove and choose **Delete**.

  The selected resource is deleted for this job master.

34 Defining and Managing End-of-Job (EOJ) Checking and Actions

- End-of-Job (EOJ) Checking and Actions - General Information 342
- Maintaining End-of-Job Events and Actions 346
- Adding and Modifying an Event Definition 349
- Deleting an Event Definition 356
- Operating System Dependent Defaults for Event Checking 356
- Creating and Viewing Online Documentation for Events 358
- Examples of Event Definitions 360
- Handling End-of-Job Actions 368
- End-of-Job Actions after Execution as a Temporary Dummy Job 372
- Defining Output Condition Actions 373
- Defining Actions for Symbol Value Modification 377
- Defining Actions for Modifying Job Variable Values 380
- Defining Action User Exits 382
- Defining Other Actions: Deactivate Job Automatically 384
- Defining Network and Job Activation Actions 385
- Defining Recovery Actions 387
- Defining SYSOUT Actions 392
- Defining Notification Messages 393
- Defining Release Actions for Kept Resources 404

This section provides general information on end-of-job (EOJ) checking and actions that are triggered after a job has been completed. It also describes corresponding online functions that are available for maintaining *job master definitions* as well as *active job definitions*.

Related Topics:

- ***Passing Files to Entire Output Management***.
- For conceptual information, see *Events and Actions: End-of-Job Checking* in the *Concepts and Facilities* documentation.

End-of-Job (EOJ) Checking and Actions - General Information

Defining End-of-Job checking and actions means instructing Entire Operations what actions to trigger after a job has terminated. The triggered action is referred to as an End-of-Job action.

End-of-Job checking and actions consists of two steps:

1. Analysis of job results (determination of End-of-Job status);
2. Triggering of appropriate system actions.

The End-of-Job action depends on the status of the whole job or any other event which can be checked by Entire Operations on termination. The status of the job on termination is determined by the occurrence of certain events during job execution which Entire Operations can check.

Before Entire Operations can check for an event, it must be defined to the system, together with instructions as to the action to be triggered. If no events are specified, Entire Operations performs End-of-Job checking automatically using **system defaults**.

The following topics are covered below:

- Events
- Event Check Method
- Possible End-of-Job Actions

Events

In the terminology of Entire Operations, an event is the occurrence of a defined situation which is recognized during the End-of-Job analysis. Entire Operations automatically triggers system action, depending on the occurrence of events during job processing.

If you do not specify any event, Entire Operations provides a default event expressed as `job OK` or `job not OK`, depending on whether a received condition code is greater or less than a default condition code, or, for BS2000, whether certain system messages are received.

Some examples of possible defined events are:

- Exit code of a UNIX job equals 2;
- STEP2 of JOB1 ends with a condition code greater than 8;
- No job step ends with a condition code greater than 0;
- A defined message appears in the job SYSOUT;
- A database or file contains or does not contain certain expected data;
- The result of a user exit (expressed by its return code).
- A job variable contains certain expected data (BS2000).

Event Check Method

Events are checked on a check `OK`/`not OK` basis. End-of-Job actions can be triggered by:

- the result of a single event check (for example, a job step checking in z/OS);
- the result of all End-of-Job checks (including the default checks), which always causes either the event `job OK` (**All Checks ok**) or `job not OK` (**Any Check not ok**) to occur.

Both event definitions are always predefined for every job.

In the latter case, you can define two sets of actions:

- one set is performed if all checks finished `OK` (**All Checks ok**, i.e., `job OK`);
- the other is performed if at least one check finished `not OK` (**Any Check not ok**, i.e., `job not OK`).

All Checks ok and **Any Check not ok** are mutually exclusive parameters to be checked.

All events can be combined as required. You can specify up to 30 different event checks for a job.

This section covers the following topics:

- End-of-Job Checks for Operating Systems and Job Types
- Job SYSOUT Check

- Retrying End-of-Job Checking

End-of-Job Checks for Operating Systems and Job Types

The following table provides an overview of the availability of End-of-Job checks for the supported operating systems and **job types**:

End-of-Job Check	z/OS	BS2000	UNIX	Windows		
Termination Codes	*					
Job Variables		*				
User Exit	*	*	*	*		
String Search	*	*	*	*		*
User Switch		*				
Exit Code			*	*		

An asterisk (*) indicates that the End-of-Job check is available for the operating system or job type. These checks are described later in this section.

Job SYSOUT Check

- On z/OS: The job result check will be retried by the Monitor up to 10 times, when the message `Job disappeared from Spool Queue` appears.

  The wait interval between SYSOUT read attempts is constantly at least 30 seconds (not to be confused with the Monitor wait time, because it may be very short).
- On BS2000: Entire Operations can only check job SYSOUT if it is assigned to a file. JCL of jobs that are to run under Entire Operations control must therefore not contain SYSOUT assignments to `*dummy`, `primary` or to a temporary file, otherwise no End-of-Job checking is possible.

See also *Defining SYSOUT Actions*.

Retrying End-of-Job Checking

For the operating system z/OS the following applies:

- In case of incomplete SYSOUT, the SYSOUT reading will be retried 10 times, with intervals not shorter than 30 seconds. Interval can be longer if the monitor task wait time is longer.

Possible End-of-Job Actions

End-of-Job actions refer to all actions performed after termination of a job. These actions can be performed automatically by Entire Operations or manually by the user.

All End-of-Job actions can be defined for an event (job step) or at the job level.

For each of the user-specified or default events, you can define how Entire Operations is to act.

Possible End-of-Job actions are:

- Set/reset output conditions to continue the job flow.

  See *Defining Output Condition Actions*.
- Set/modify symbol values.

  See *Defining Actions for Symbol Value Modification*.
- Release resources held by a job.

  See *Defining Release Actions for Kept Resources*.
- Set/reset/modify job variables (BS2000 only).

  See *Defining Actions for Modifying Job Variable Values*.
- Execute End-of-Job action user exits.

  See *Defining Action User Exits*.
- Deactivate job.

  See *Defining Other Actions: Deactivate Job Automatically*.
- Activate other job networks or single jobs.

  See *Defining Network and Job Activation Actions*.
- Start job recovery.

  See *Defining Recovery Actions*.
- Handle job SYSOUT data.

  See *Defining SYSOUT Actions*.
- Pass output files to Entire Output Management.

  See *Passing Files to Entire Output Management*.
- Send message to user or console with information about any abnormal event or pending condition.

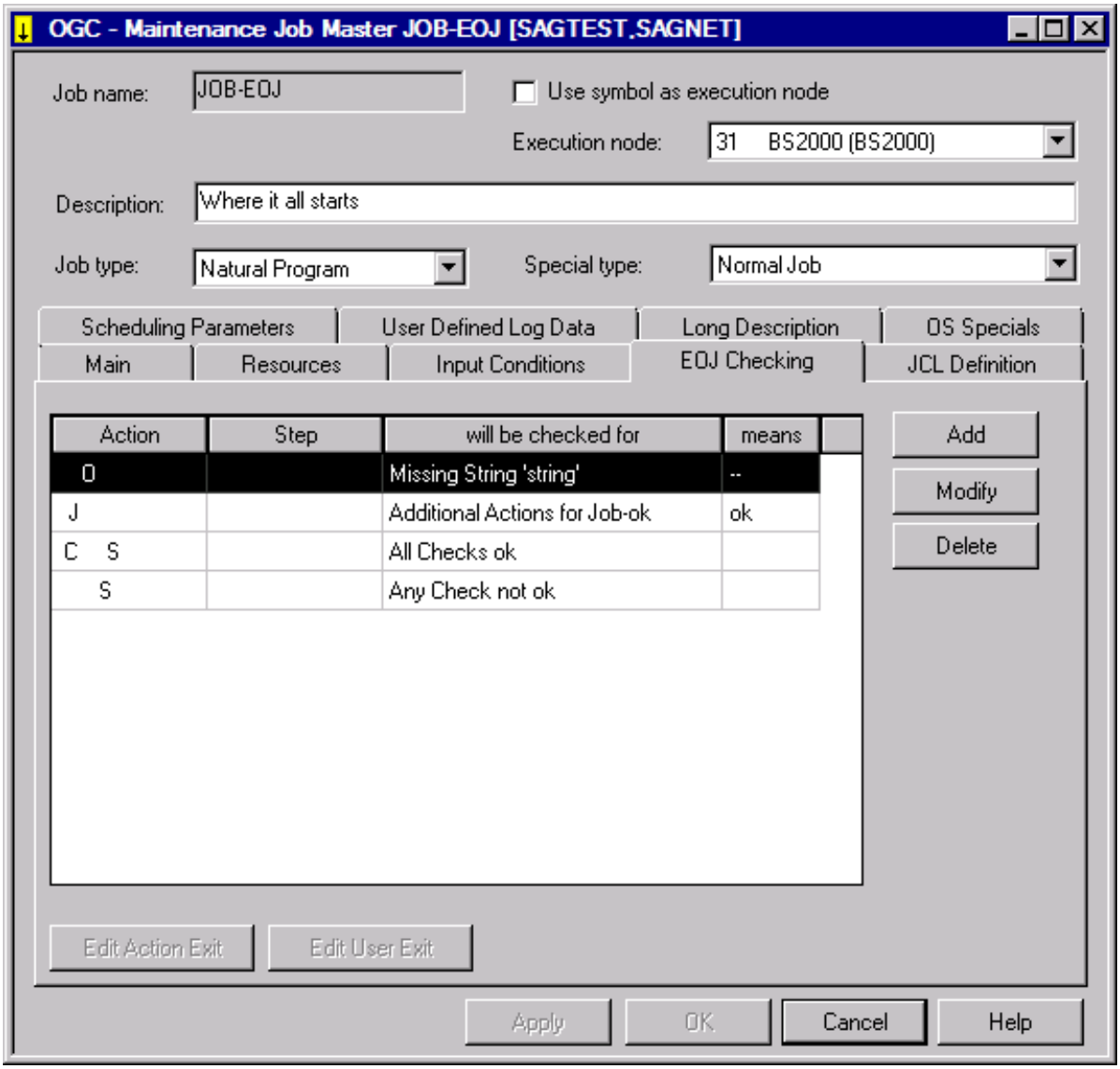
  See *Defining Notification Messages*.

Only one action of the same type can be defined for an event (except in the case of output conditions). If you need several actions of the same type for the event `job OK` or `job not OK`, you can define these by adding an **Add. Job-ok, Job-not-ok** event.

Maintaining End-of-Job Events and Actions

To view End-of-Job events and actions defined for a job

- In the **Job Master** window, open the tabbed page **EOJ Checking** as shown in the following example:



The page lists all End-of-Job actions and events defined for the selected job.

The **columns** and **functions** available on the page are described in the relevant sections.

This section covers the following topics:

- Columns and Rows: End-of-Job (EOJ) Checking and Actions
- Action Indicators: EOJ Checking Page
- Functions: EOJ Checking Page

Columns and Rows: End-of-Job (EOJ) Checking and Actions

The **tabbed page EOJ Checking** provides the following columns:

Column/Row	Description		
Action	Summary of defined actions for the event. Each character listed in this column represents an action defined for the event. All characters possibly shown here are listed in *Action Indicators: EOJ Checking Page*.		
Step	Applies to z/OS only. Step name in JCL (`ANYSTEP` means all steps).		
will be checked for	Event to be checked (condition code, system code, user abend code, system message, etc.).		
means	Event status after the check:		
	`ok`	All checks ended OK.	
	`n.ok`	At least one check ended not OK.	
	`--`	Event has no influence on the job result. (Corresponds to **No influence**.)	
	See also *Handling End-of-Job Actions*.		
OA	This column only appears on the **EOJ Checking** page in the **Maintenance Job Active** window of an active job. This column only contains entries if an event occurred during job execution. The entries under **O** indicate the status of the event check, the entries under **A** indicate whether the action defined for the event was performed:		
	Entry	**O**	**A**
	blank	Event was not checked.	No action performed.
	`-`	Event did not occur.	n/a
	`+`	Event occurred.	Action performed.
	`E`	Error occurred during event check.	One or more actions performed erroneously.
	`M`	Event definition was modified.	Action definition was modified.

Column/Row	Description		
	E	Error occurred during the event check.	One or more actions failed.

Action Indicators: EOJ Checking Page

The character(s) shown in the **Action** column of the **EOJ Checking page** indicate the action(s) that have been defined for the event. For example: PSU indicates that a description (P), a SYSOUT action (S) and a notification message (U) have been defined for the event.

Action Indicator	Action Indicated
A	**Job or job network activated.**
C	**Output condition defined.**
J	**Job variable defined.**
L	**Kept resource released.**
O	**Other actions** defined.
P	**Description of event** exists. See also *Creating and Viewing Online Documentation for Events*.
R	**Recovery action** defined.
S	**SYSOUT action** defined.
T	**SYSOUT or other files passed to Entire Output Management.**
U	**User message** defined.
X	**Action user exit** defined.
Y	**Symbol or symbol settings** modified.

Functions: EOJ Checking Page

The following functions are available for a table item selected on the **EOJ Checking page**:

Function	Description
Add	Define a new End-of-Job event and action(s). See *Adding and Modifying an Event Definition*.
Modify	Edit the selected End-of-Job event and action(s). See *Adding and Modifying an Event Definition*.
Delete	Delete the selected End-of-Job event and action(s).
Edit Action Exit	Edit the action user exit to be executed if the selected event occurs. This command is only available if you are authorized to modify the selected user exit. See also *User Exits for End-of-Job Checking and Actions* in the section *User Exits*.

Function	Description
Edit User Exit	Edit the user exit specified for the selected checking user exit. This command is only available if you are authorized to modify the selected user exit. See also ***User Exits for End-of-Job Checking and Actions*** in the section *User Exits*.

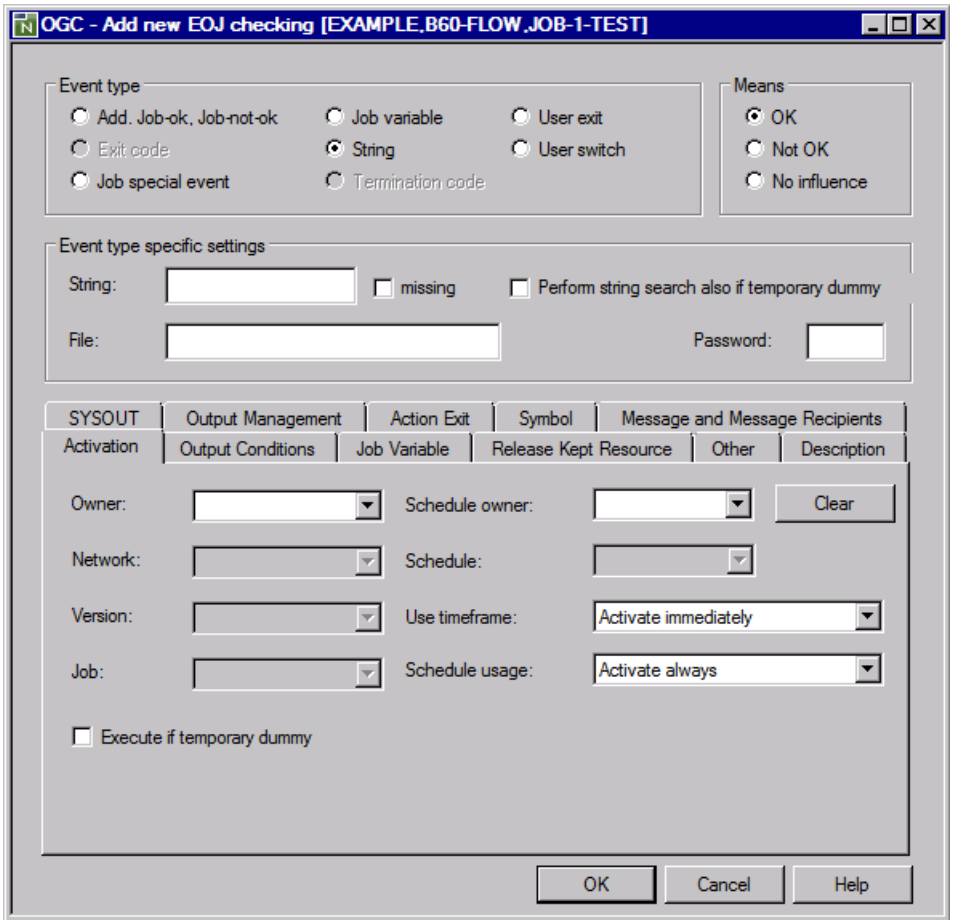
Adding and Modifying an Event Definition

To add or modify an End-of-Job (EOJ) check and action definition

1 In the **Job Master** window, open the tabbed page **EOJ Checking**.

2 Choose **Add** or **Modify** depending on whether a definition already exists.

An **EOJ checking** window like the example below opens:



The input fields and tabs provided in the **EOJ checking** window depend on the operating system (here: BS2000) of the execution node and the **Event type** (here: **String**) specified for

the job. All fields and tabs that can be available in the window are explained in *Fields: EOJ Checking Window*.

3 Make your definitions and choose **OK** to save your entries.

This section covers the following topics:

- Fields: EOJ Checking Window

Fields: EOJ Checking Window

This section describes the fields and tabs contained in the **EOJ checking window**.

The fields in the **Event type**, **Means** and **Event type specific settings** group boxes are used to define the job event.

The fields on the tabbed pages in the lower window section are used to define the action to be taken if the event occurs. The following table indicates when a field in a group box has a corresponding field on a tabbed page.

Note: The fields and tabbed pages available to define an event or action depend on the operating system of the execution node and the **Event type** selected for the job. This is also indicated in the table below.

Field | Page	Description
Event type	**A - Add. Job-ok, Job-not-ok:** Additional definition for **All Checks ok** or **Any Check not ok** to execute multiple End-of-Job actions of the same type. This event is required to define several actions of the same type for the event `job ok` or `job not ok`. **Note:** You can use these additional event definitions to set conditions, but it is not recommended, because they are not evaluated for the display of **job dependencies**. See also *Handling End-of-Job Actions* and *Example of an Additional Job OK/not OK Event (Event Type A)*.
	(z/OS only) **C - Termination Code:** Termination code received during job execution (job step). For possible values, see the field **Event** field. See also *Example of a Termination Code Event (Event Type C)*.
	(BS2000, UNIX and Windows only) **J - Job Special Event:**

Field | Page	Description
	Special event during job execution. When such an event occurs this always means `Job not ok`. This event occurs if an operating system abend or hardware failure occurs during job execution. This is always used when a job did not run on LOGOFF (job ended normally) or ABEND (job interrupted with error). The defined actions are taken after the start of the Monitor. See also *Example of a Job Special Event (Event Type J)*.
	R - User Exit: Job SYSOUT is to be checked by a user exit. See also *Example of a User Exit Event (Event Type R)*.
	S - String: Occurrence of a specific string in the job SYSOUT or a specified file. See also the field **String** and *Example of a String Event (Event Type S)*.
	(BS2000 only) **U - User Switch:** Event depends on a user switch. See also *Example of a User Switch Event (Event Type U)*.
	(BS2000 only) **V - Job Variable:** Event depends on the contents of a job variable. The input fields available to define a job variable are explained in *Fields: Job Variable Definitions (BS2000 only)*. See also *Example of a Job Variable Event (Event Type V)*.
	(UNIX and Windows only) **X - Exit Code:** Exit code check. The result of the event comes from the exit code of a script under UNIX or Windows or from the exit code of a program executable under Windows. Specify a **logical operator** and an exit code to compare the received exit code with the specified values. **Note:** A standard check for the exit code can be defined in the Entire Operations defaults. If a SYSOUT file read error occurs while the Monitor is trying to read the exit code from the job SYSOUT, this error is treated as a temporary error. The Monitor attempts to read the exit code up to 10 times. The time between the attempts is equal to the Monitor wait

Field | Page	Description	
	time. If there is still no success after 10 attempts, the job is set to the `status permanent` error. See also *Example of an Exit Code Event (Event Type X)*.	
Means	Specifies the event check status if the defined event occurs. Possible check-box settings:	
	OK	Check `OK`.
	Not OK	Check `not OK`.
	No influence	No effect on the job result.
	See also *Handling End-of-Job Actions*.	
Event type specific settings:		
Stepname (z/OS only)	(z/OS and z/VSE only) JCL step to be checked for the return code specified in the **Operator** and fields. Special step names:	
	`ANYSTEP` (or `/ANYSTEP`)	Event occurs if the definition is true for any step of the jobs.
	`MAXCC` (or `/MAXCC`)	Event occurs if it is true for the maximum termination code (condition code) of the job.
	See also *Example of a Termination Code Event (Event Type C)*.	
Operator (z/OS only)	(z/OS only) Specifies the **logical operator** used to compare the received return code with the value entered in the **Event** field. See also *Example of a Termination Code Event (Event Type C)*.	
Event (z/OS only)	(z/OS only) Select a termination condition and enter a value, if required:	
	Condition	4-character condition/return code
	User Abend	4-character user abend code (z/OS only).
	System Abend	4-character system abend code (z/OS only).
	Job deleted by operator	Job deleted by operator.
	Job failed JCL error	Job failed, JCL error. **Note:** If this event occurs and a **Job failed JCL error** event definition with `OK` exists, this can be treated as `OK`.
	Job execution interrupted	Job execution interrupted, for example, system crash (z/OS only).
	'Not cataloged' error	Not cataloged (compiled) error (z/OS only).

Field | Page	Description	
	Job not run JCL error	**Note:** If this event occurs and a **Job not run JCL error** event definition with `OK` exists, this can be treated as `OK`.
	See also *z/OS Defaults for Event Checking* and *Example of a Termination Code Event (Event Type C)*.	
User exit	Name of the End-of-Job checking user exit which is to run on job termination. **Note:** Entire Operations performs an Entire System Server logon to the job's **execution node** with the job's submit user ID prior to the execution of the exit. To display a selection list of user exits, first enter the name of a Natural library in the **Natural library** field. You can then select a user exit from the drop-down list box of the **User exit** field. Depending on your permissions, you can **Edit** or **Delete** a checking user exit on the tabbed page **EOJ Checking**. See also: ■ *Example of a User Exit Event (Event Type R)*. ■ *User Exits for End-of-Job Checking and Actions* in the section *User Exits* and	
Natural library	Name of the Natural library which contains the required user exit. This library should be different from the Entire Operations system library. See also *Example of a User Exit Event (Event Type R)*.	
Execute user exit asynchronously	Possible check-box settings:	
	unchecked	User exit is executed synchronously (default).
	checked	User exit is executed asynchronously (in a Natural task).
String	Specify the string to be searched for in SYSOUT or any defined file. If the string is found, the event occurs. The following applies: ■ All checks of the SYSOUT file and actions which refer to the SYSOUT file are not executed if no SYSOUT file is available. This case only applies if the job is submitted as a temporary dummy job. ■ A string search in a non-existent file is handled like `string not found` in an existing file. ■ The search string may contain symbols. ■ Symbols with **activation escape characters** are replaced at job activation time. If the string contains the activation escape character, a symbol replacement is performed (from the active symbol table). The updated string is stored in the active database.	

Field | Page	Description	
	■ Symbols with **submission escape characters** are replaced at End-of-Job checking time. The symbol remains in the field, so that it will be replaced again during an eventual job re-submission. ■ Missing symbols cause a permanent activation error or permanent End-of-Job checking error. **Note for BS2000 and UNIX:** If the search string contains at least one blank at the beginning, in the middle or at the end, it must be enclosed in quotes. For example: `'  leading blanks'` See also *Example of a String Event (Event Type S)*.	
missing (string)	**String** missing. Possible check-box settings:	
	unchecked	The event occurs if the string is found (default).
	checked	The event occurs if the string is not found.
Select (string) (z/OS only)	(z/OS only) z/OS: You can limit the search for **strings** by specifying file numbers in this field according to the file type (`SM`, `SO`, `JL` and `CC`). Enter the file type followed by a number, a range of numbers or an asterisk (*) as a wildcard. For example:	
	`SM 1`	Search system messages 1.
	`SM 1:5`	Search system messages 1 to 5.
	`SO *`	Search in all SYSOUT files.
	`CC 1`	Search condition codes 1.

Fields: Job Variable Definitions (BS2000 only)

The fields available to define a job variable event in the header section of the **EOJ checking** window and a job variable action on the tabbed page **Job Variable** are explained in the following table:

Field	Description
Job variable	Name of a valid BS2000 job variable. If the job variable is specified without an explicit user ID, the job default BS2000 user ID is used as a prefix. Symbol replacement is performed in the name if the name contains the **activation escape character** at least once. Symbol replacement is possible. The **submission escape character** is to be used.

Field	Description	
Position	Position of the substring of the job variable value to be checked. Possible values: 1 to 253.	
Length	Length of the substring of the job variable value to be checked. Possible values: `1` to `253`.	
Format	Format in which the substring of the job variable value is to be checked against the *`comparison-string`*. Possible selection options:	
	`A`	alphanumeric
	`N`	numeric
is	**Logical operator** to be used for comparison of the defined BS2000 substring of the job variable value against the *`comparison-string`*.	
compared to *`comparison-string`*	Enter the string or field to be compared with the substring of the job variable value. The strings are compared in the defined format. The content of this field is compared with the substring of the job variable value, or it is inserted into the substring of the job variable value. The comparison is made in the defined format. Symbol replacement is possible. The submission escape character is to be used.	
Password	(Optional field) If the job variable is read password-protected, specify the password here.	
Execute if temporary dummy	See *End-of-Job Actions after Execution as a Temporary Dummy Job*.	

See also *Example of a Job Variable Event (Event Type V)*.

Valid Comparison Operators

You can use one of the following comparison (relational) operators to compare two character strings and return either the value true or false:

Operator	Explanation
=	Equal to the specified value.
EQ	
>=	Greater than or equal to the specified value.
GE	
>	Greater than the specified value.
GT	
<=	Less than or equal to the specified value.
LE	

Operator	Explanation
<	Less than the specified value.
LT	
<>	Not equal to the specified value.
NE	

Deleting an Event Definition

To delete an event definition

1 On the tabbed page **EOJ Checking**, select the required event and choose **Delete**.

A window prompts you to confirm the deletion.

2 Choose **Yes** to confirm the deletion.

(**No** cancels the action.)

The page closes and the event definition is removed.

Operating System Dependent Defaults for Event Checking

This section covers the following topics:

- Common Defaults for Event Checking
- z/OS Defaults for Event Checking
- BS2000 Defaults for Event Checking
- UNIX and Windows Defaults for Event Checking

Common Defaults for Event Checking

- Symbol replacement error during job submission:

  If a symbol replacement error occurs at submission time, the event **Job not run JCL error** is set to `occurred`.

  If the event **Job not run JCL error** is not defined, it is added to the active job.

- If a string defined in the Global Message Code Table (see the *Administration* documentation) is found anywhere in the SYSOUT, its severity equivalent is compared with the highest condition code value accepted as `OK`. The job is `not OK` if the severity equivalent is higher than this default.

z/OS Defaults for Event Checking

If no events are defined for a job, Entire Operations provides a default check of the return code type (for termination codes):

- If a system code that is not equal to `0` (zero) has occurred at any job step, the job is evaluated as `not OK` if this event is not explicitly defined. This also applies if an event has occurred which has been defined using `ANYSTEP`. The occurrence of such an event has the priority over definitions, which apply for any steps.
- You can set the default for how to treat the message `IEF201I job terminated` if it occurs. The appropriate option in the **Operating System Specials** (see the *Administration* documentation) determines whether to treat this message as `not OK`. If treated as `not OK`, a log is written, and the job is set to the status `Job interrupted`. If treated as `OK`, the occurrence of IEF201I in SYSOUT has no consequences and the job continues.
- If the message `IEF287I data set not cataloged` is received, it is written to the log and the job is treated as `not OK`.
- If a condition code or user code, which is not explicitly defined, is received at any job step, it is compared with the highest value accepted as `OK`. The job is `not OK` if the code is higher than this default. The job is also `not OK` if no explicit check has returned an `OK`. For more information, see the **Operating System Specials** option.
- Events like JCL error, Job not run and Job interrupted cause the status `not OK` by default. Dedicated event definitions can override this.
- In all other cases, the job is accepted as `OK`.
- Precedence of event checks:
  - If a `MAXCC` or `ANYSTEP` check has been defined and this event occurs, no check is done on the highest value accepted as still `OK`.
  - `MAXCC` precedes `ANYSTEP`.

BS2000 Defaults for Event Checking

Entire Operations either uses a defined Monitor job variable for a job or creates its own job variable if job variables are available.

- If the Monitor job variable displays an abnormal end, the job is `not OK`.
- By default, if defined message keys appear, the job is regarded as faulty and is set to `not OK`. These message codes (see *BS2000 Default Message Codes* in den *Administration* documentation) are default settings which are in effect after the installation of Entire Operations. However, they can be freely adapted. It is possible to completely delete this table.



Note: Each time this table is modified, possibly faulty jobs may no longer be set to `not OK`.

- If messages like `program dump` appear in the SYSOUT, the job is `not OK`.
- In all other cases, the job is accepted as `OK`.

UNIX and Windows Defaults for Event Checking

Entire Operations inserts start and end messages (EOR0301, EOR0302) in the SYSOUT.

- If the end message EOR0302 is missing, the job is treated like an interrupted job. Because it is possible that the SYSOUT of an asynchronous child process is written behind the end message, the message EOR0302 is searched in the whole SYSOUT file. If the End-of-Job checking detects that there is unexpected SYSOUT behind the end message, you cannot obtain the process times for accounting. A note is written to the log:

  ```
  Process times for accounting not found ... SYSOUT of background processes may exist.
  ```

- UNIX and Windows JCL frames generate the message EOR0339 if the **Maximum SYSOUT Size (in MB)** is exceeded:

  EOR0339 - Maximum SYSOUT Size :1: exceeded

  If this message is detected, no further End-of-Job checking and no End-of-Job actions will be performed for the job at all. As a result, a job network execution will be interrupted.
- Unix only: If the SYSOUT contains `segmentation fault` (in German: `Speicherzugriffsfehler`), the job is treated as `not OK`.
- Unix only: If the SYSOUT contains `bad interpreter` (in German: `Defekter Interpreter`), the job is treated as `not OK`.
- Unix only: If the SYSOUT contains `command not found` (in German: `Kommando nicht gefunden`), the job is treated as `not OK`.
- If no special event checking was defined for a job (UNIX and Windows), then the event is compared to a system-wide default value for UNIX or Windows. Depending on this check, the job result can be set to `not OK`.
- In all other cases, the job is accepted as `OK`.

Creating and Viewing Online Documentation for Events

You can browse text description of active event and write or modify the text description of a master event.

To add, view or change an event description

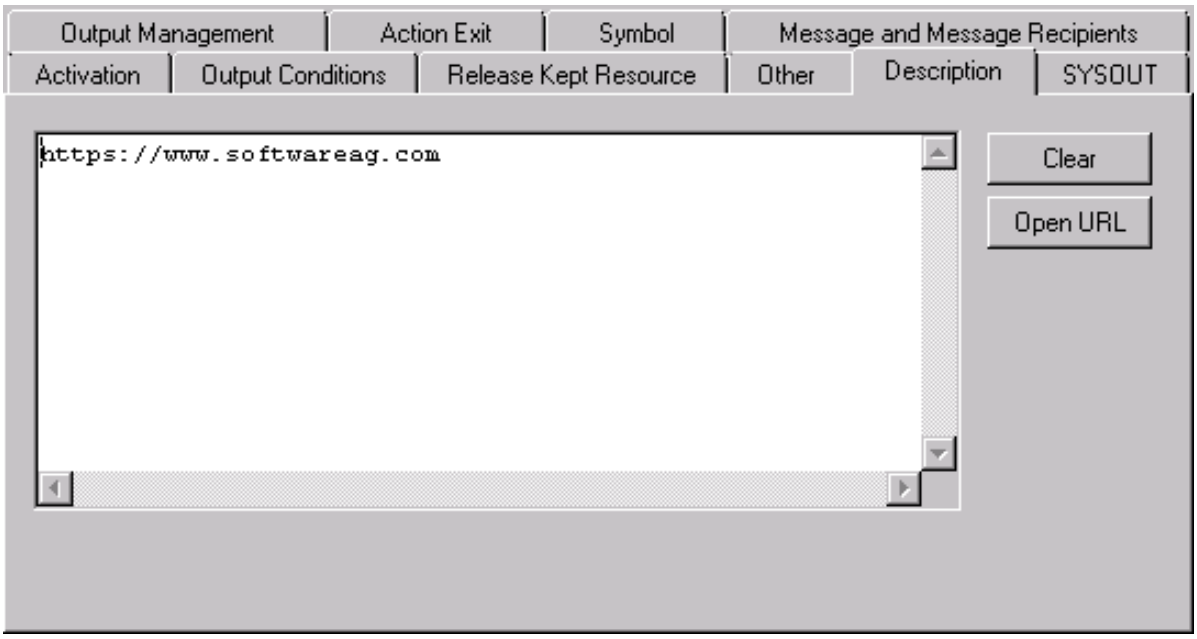
1 On the **EOJ Checking page** select the event for which you want to add a description or whose description you want to view or change and choose **Modify**.

The letter **P** in the **Action** column indicates whether a description already exists for the event.

The **EOJ Checking window** opens.

2 Open the tabbed page **Description**.

The page contains either descriptive text for the event or an URL address as shown in the example below:



If an URL address is displayed, you can choose **Open URL** to open the page associated with the URL.

3 Write new or replace existing text as required.

Choose **Clear** if you want to remove the whole text contained in the editor area.

4 Choose **OK** when you are finished.

Note: After you add or modify an output condition definition, a loop check is performed on the affected network. The same conditions apply as described in *Checking for a Loop in a Job Network* in the section *Network Maintenance* with one exception: if a loop is detected in the job flow, no corresponding message appears.

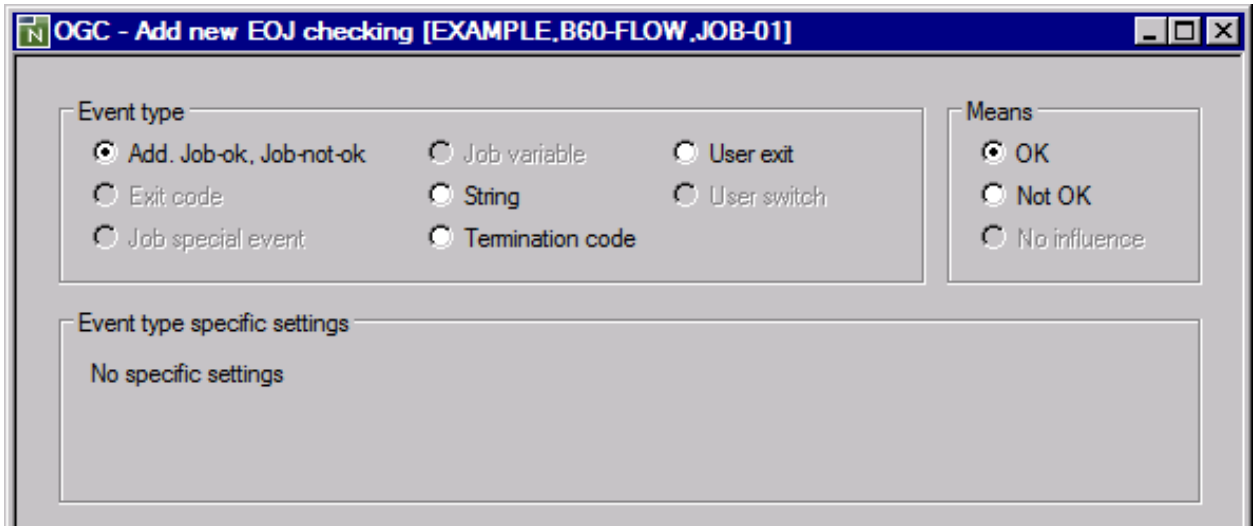
Examples of Event Definitions

This section contains examples of event definitions:

- Example of an Additional Job OK/not OK Event (Event Type A)
- Example of a Termination Code Event (Event Type C)
- Example of a Job Special Event (Event Type J)
- Example of a User Exit Event (Event Type R)
- Example of a String Event (Event Type S)
- Example of a User Switch Event (Event Type U)
- Example of a Job Variable Event (Event Type V)

- Example of an Exit Code Event (Event Type X)

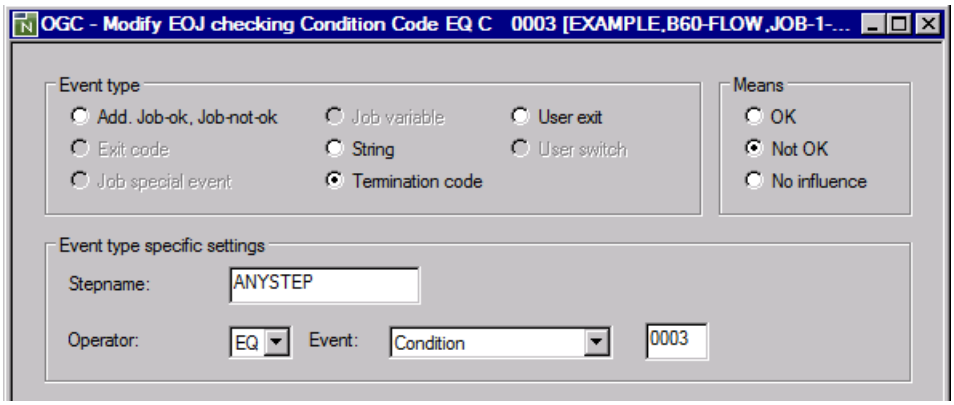
Example of an Additional Job OK/not OK Event (Event Type A)



Explanation:

This example defines an additional `Job OK` event to execute another End-of-Job action of the same type.

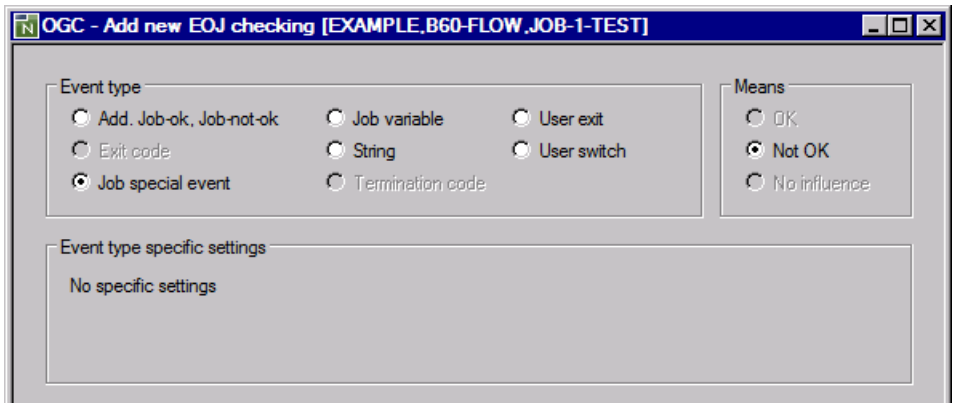
Example of a Termination Code Event (Event Type C)



Explanation:

If Condition Code 3 is received at any step (`ANYSTEP`) during job execution, the event result is `not OK`.

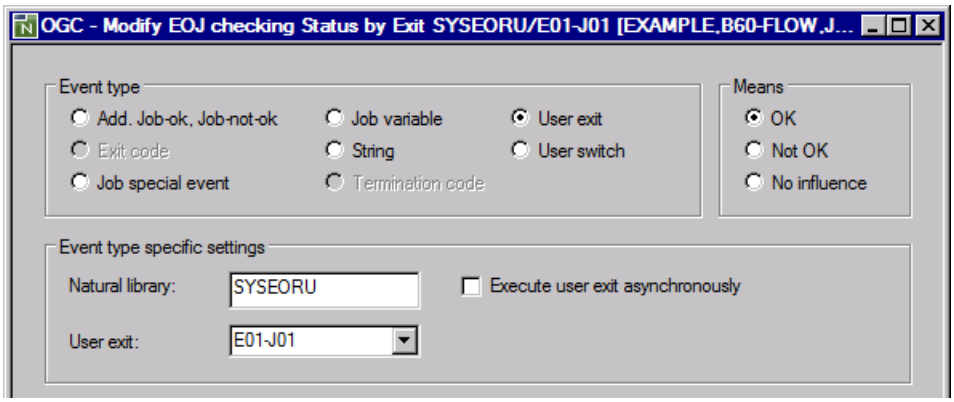
Example of a Job Special Event (Event Type J)



Explanation:

This event occurs if an operating system abend or hardware failure occurs during job execution.

Example of a User Exit Event (Event Type R)

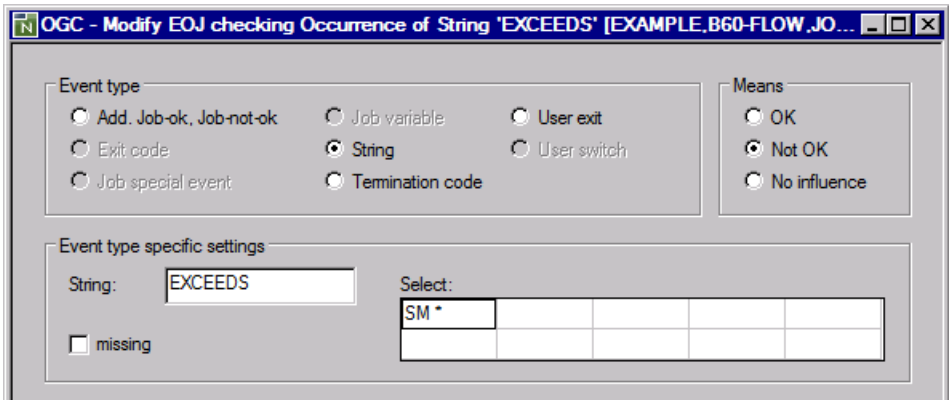


Explanation:

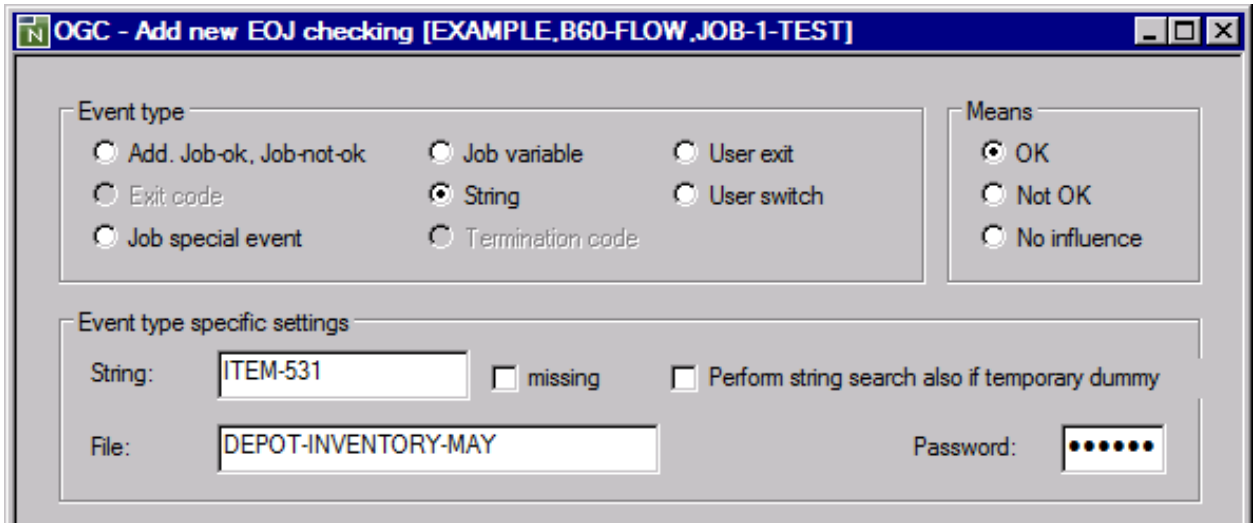
User exit `E01-J01` checks the job SYSOUT at job completion. This user exit resides in the Natural library `SYSEORU`.

Example of a String Event (Event Type S)

Example for z/OS:



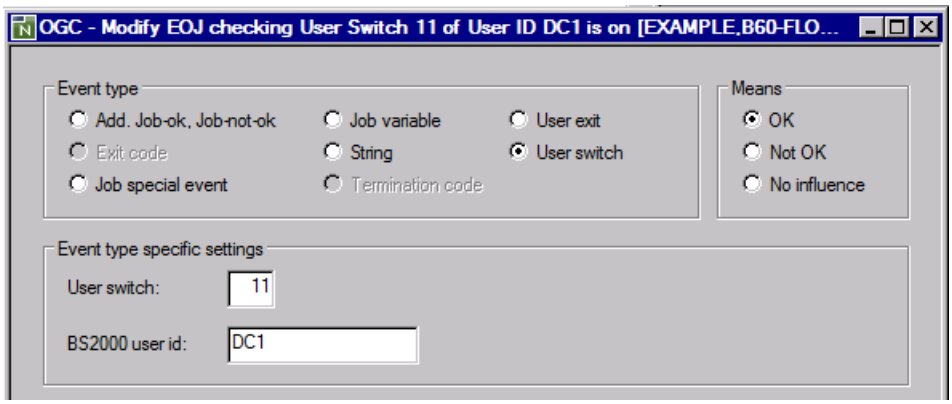
Example for BS2000:



Explanation:

- Example for z/OS: If the string `EXCEEDS` is found in any system message (`SM *`), the event result is `not OK`.
- Example for BS2000: If the string `ITEM531` is found in the BS2000 file `DEPOT-INVENTORY-MAY`, the event result is `OK`.

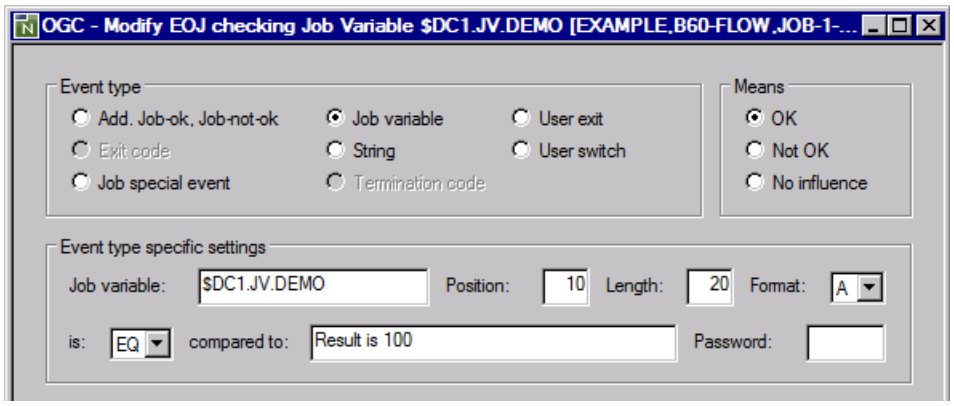
Example of a User Switch Event (Event Type U)



Explanation:

The event is `OK` if user switch `11` of the BS2000 user ID `DC1` is on at job termination time.

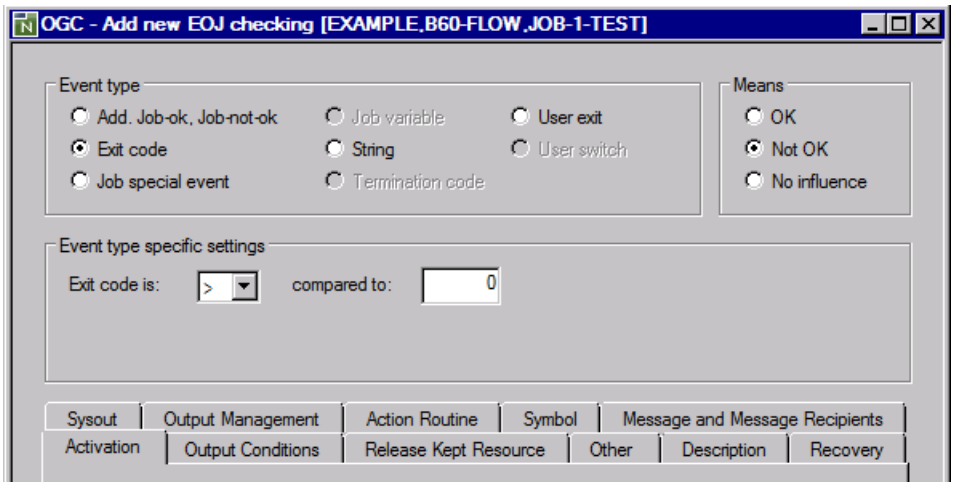
Example of a Job Variable Event (Event Type V)



Explanation:

The event is `OK` if the job variable `$DC1.JV.DEMO` contains `Result is 100` at the position `10`, in the length `20`, in alphanumeric (`A`) format.

Example of an Exit Code Event (Event Type X)



Explanation:

The event is `OK` if an exit code greater than `0` (zero) occurs.

Handling End-of-Job Actions

An End-of-Job action can be performed for a check event defined at the job step level and/or the default check event **All Checks ok** or **Any Check not ok** defined at the job level.

The default check events are always provided in the last two table rows of the **EOJ Checking page**.

All checks ok can be used to define actions to perform if no error occurred during job execution.

Any Check not ok can be used to define actions to perform if at least one error occurred during job execution.

This section covers the following topics:

- Defining Actions
- Changing Execute if Temporary Dummy Settings for Multiple Jobs

- Removing Actions

Defining Actions

To define an action

1 On the **EOJ Checking page**, select an event at the job step level for which you want to define an action, or select the default event check **All Checks ok** or **Any Check not ok**

2 Choose **Add** or **Modify** depending on whether you want to add or change an action, and proceed as described in the action-specific instructions in the remainder of this chapter.

The example below demonstrates how the same action can be defined for several purposes:

Action	Step	will be checked for	means
U		Additional Actions for Job-ok	ok
U	ANYSTEP	Condition Code = C0004	n.ok
		All Checks ok	
U		Any Check not ok	

In the example above, a notification message (**U**) is sent after job termination if

- No error occurred during job execution (`Additional Actions for Job-ok`),
- Condition Code 4 occurred during job execution, and
- Any error occurred during job execution (`Any Check not ok`).

Changing Execute if Temporary Dummy Settings for Multiple Jobs

You can modify the setting of **Execute if temporary dummy** for all relevant End-of-Job actions defined for a selected set of job masters.

Prerequisite: Read and write permissions must be checked in the **EOJ Checking + Action** option of your user profile (see *Network Maintenance Functions* described in the *Administration* documentation).

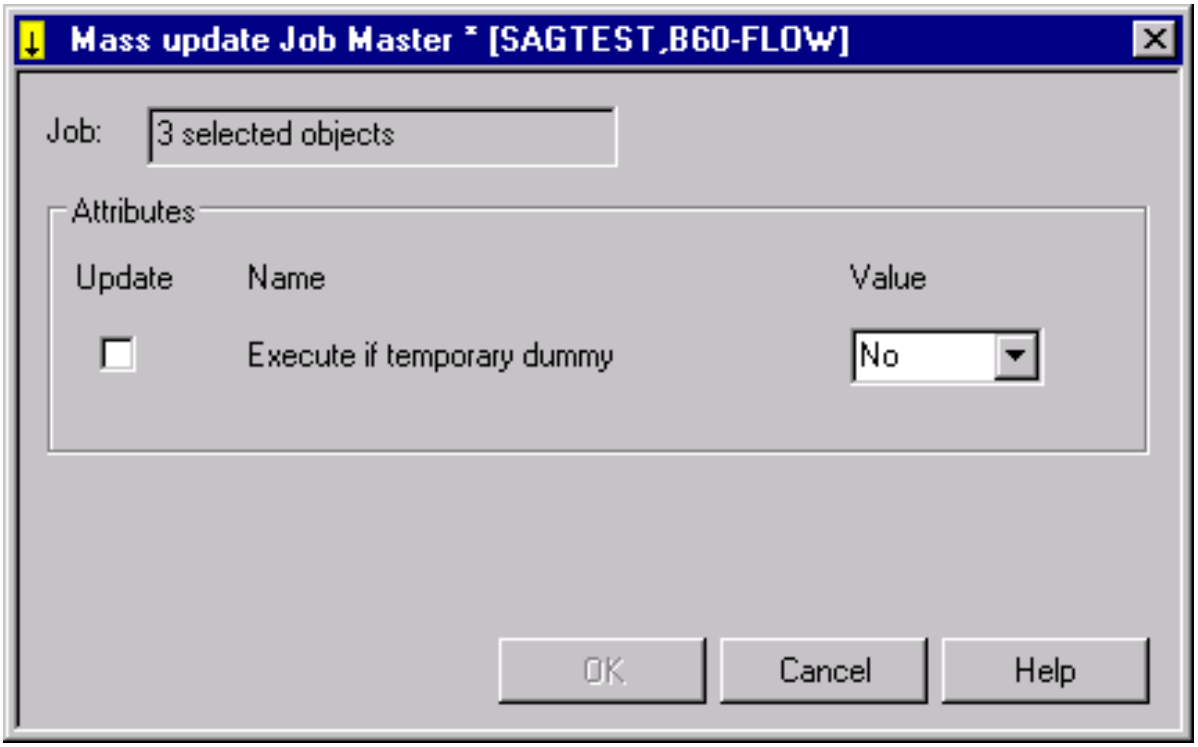
To update "Execute if temporary dummy" for multiple jobs

1 From the table in the **Job Master List window**, select the jobs whose **Execute if temporary dummy** settings you want to change and choose **Mass update** from the context menu.

Or:

Open the diagram of the network master containing the jobs to be updated, select the job masters whose **Execute if temporary dummy** settings you want to change and choose **Mass update** from the context menu (see also *Maintenance Functions for Diagrams*).

A **Mass update Job Master** (or **Diagram Job Master**, respectively) dialog like the following opens:



The **Job** field indicates the number of job masters selected for the update operation.

2 If you want to perform the execute operation, select the **Update** check box (not selected by default) and choose **Yes** from the drop-down list box to check the **Execute if temporary dummy** option for all End-of-Job actions of the selected jobs.

Or:

If you do not want to perform the execute operation, select the **Update** check box and choose **No** (default) from the drop-down list box to uncheck the **Execute if temporary dummy** option for all End-of-Job actions of the selected jobs.

3 Choose **OK**.

The setting of the **Execute if temporary dummy** option is changed for the following End-of-Job actions:

- **Activation**
- **Action User Exits**
- **Job Variable**
- **Symbol**
- **Release Kept Resource**
- **Message and Message Recipients**

Removing Actions

Related Topic:

- *Deleting an Output Condition*

To remove an action

1 On the **EOJ Checking page** select the event for which you want to remove an action and choose **Modify**.

The **EOJ checking window** opens.

2 Open the page that contains the action you want to remove.

3 On the appropriate action page, choose **Clear**.

The entries in the window are removed.

Choose **OK**.

Or:

On the appropriate action page, choose **Delete** (if available).

A confirmation window opens.

Choose **Yes**.

(**No** cancels the action.)

The page closes and the action is removed.

To remove an event with a single action

- If a single action is defined for an event, you may want to remove the entire event by selecting the required event on the **EOJ Checking page** and choosing **Delete**.

End-of-Job Actions after Execution as a Temporary Dummy Job

For **temporary dummy jobs**, End-of-Job actions are partially carried out as if the jobs were running in normal mode.

Several End-of-Job action definitions contain a field **Execute if temporary dummy** which can be used to execute the End-of-Job action according to this definition if the job was executed as a temporary dummy job.

Possible check-box settings:

checked	Perform the End-of-Job action if the job executed as a temporary dummy job.
unchecked	Do not perform the End-of-Job action if the job executed as a temporary dummy job (default in most cases).

Note: You can check or uncheck the **Execute if temporary dummy** option for multiple jobs at the same time: see *Changing Execute if Temporary Dummy Settings for Multiple Jobs*.

This section covers the following topics:

- Default End-of-Job Actions for Temporary Dummy Jobs

- SYSOUT Actions

Default End-of-Job Actions for Temporary Dummy Jobs

If the **Execute if temporary dummy** option is not selected in an event action definition, the following defaults apply:

Output conditions	Output conditions are always set or deleted even if the job executed as a temporary dummy job.
Network or job activation	Network or job activation is performed by default even if the job executed as a temporary dummy job.
Dummy due to schedule changes	Additional End-of-Job actions are not executed.
Dummy for other reasons	Additional End-of-Job actions are executed.
End-of-Job Action User Exit	Exit type EJA: End-of-Job action user exits are not executed

SYSOUT Actions

Actions that refer to the SYSOUT file cannot be executed if no SYSOUT file exists. This case applies if the job is executed as a temporary dummy job.

Defining Output Condition Actions

You can set or reset output conditions as End-of-Job actions for any job events.

You can define up to 50 output conditions for any event at the job step level or at the job level according to **All checks ok** and **Any check not ok**.

- Adding and Modifying an Output Condition
- Deleting an Output Condition

Adding and Modifying an Output Condition

To add, view or change an output condition

1 On the **EOJ Checking page**, select the event for which you want to create a condition and choose **Add**.

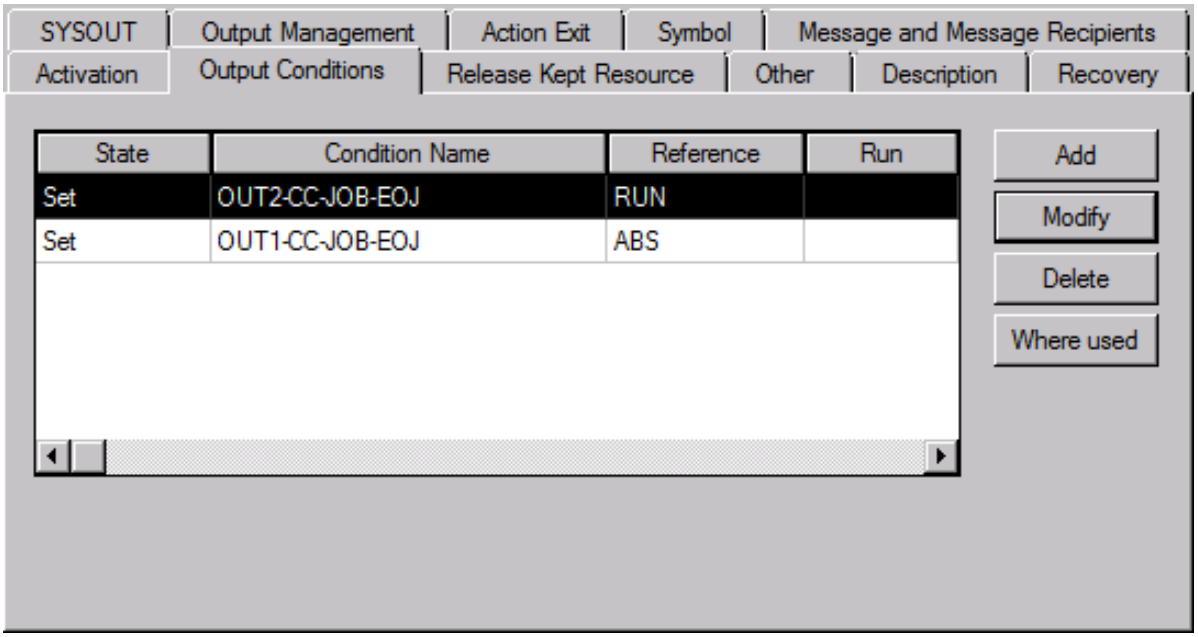
Or:

On the **EOJ Checking page**, select the event whose output condition you want to view or change and choose **Modify**.

The letter **C** in the **Action** column indicates whether an output condition exists for the defined event.

The **EOJ checking window** opens.

2 Open the tabbed page **Output Conditions** show in the following example:



If output conditions already exist, they are listed in the table. In the example above two conditions are set for a selected **Termination code** event.

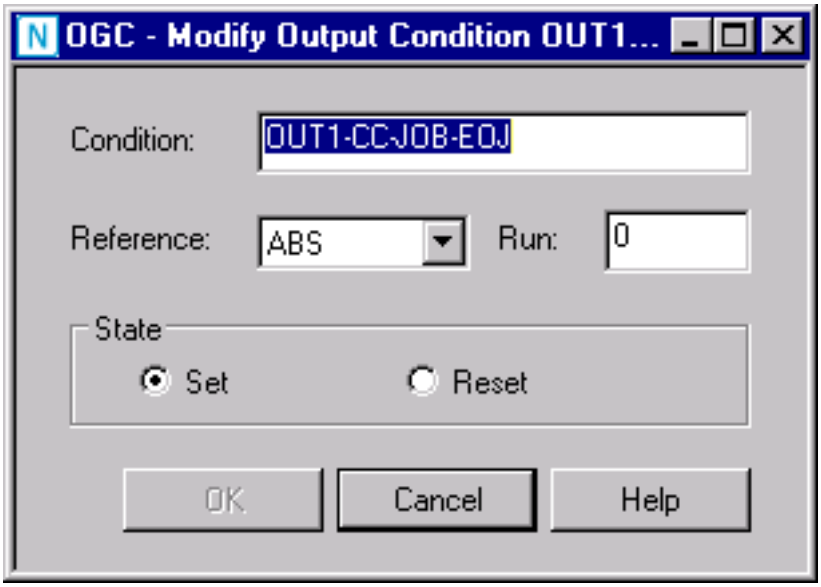
The column headings are explained in *Columns and Fields: Output Conditions*. The commands are explained in *Commands: Output Conditions*.

3 Choose **Add** to define a new condition.

Or:

Select an existing output condition (here: `OUT1-CC-JOB-EOJ`) and choose **Modify**.

An **Output Condition** window like the example below opens:



4 Enter the required values.

The fields and valid input values are explained in *Columns and Fields: Output Conditions*.

5 Choose **OK** when you are finished.

The **Output Condition** window closes.

6 On the **Output Conditions** page, choose **OK**.

7 On the **EOJ Checking** page, choose **OK**.

The definition of the new or modified output condition is saved.

Columns and Fields: Output Conditions

The columns available on the **Output Conditions page** and the corresponding fields in the **Output Condition window** are explained in the following table.

Column/Field	Description	
State | **Set/Reset**	Determines whether the defined output condition must be true (**Set**) or false (**Reset**) if the associated event occurs. Possible values:	
	Set	Condition is true (default).
	Reset	Condition is false (ignored).
Condition Name | **Condition**	User-defined name for the condition. Use this name if you wish to define this output condition as an input condition for another job. Symbol replacement is possible. The submission escape character is to be used.	

Column/Field	Description
	If you want to add or modify a global condition that can be used across networks, consider the rules and restrictions for global conditions described in the section *Global Conditions*.
Reference	An output condition can be set with different references (usually the current network run number). The reference is evaluated and set when the active condition is created by the Monitor according to the End-of-Job definition. Possible values:
	`ABS` — Absolute condition. Exists only once, because it is independent of run numbers.
	`RCM` — Sets an active condition with owner, network, run number of the calling network and appends multiple suffixes. `RCM` can be used to determine that the conditions defined for a predecessor job apply to all jobs that reference this predecessor in multiple activated subnetworks. In this case, the input condition of the corresponding successor job must be referenced with `RCA`: see `RCA` **in the section** *Possible References for Input Conditions.*
	`RUN` or blank — Current run number of the job network is assigned (default).
Run	Run number of the job to which this condition applies when modifying the condition for an active job. If left blank, the current run number is assumed by default.

Commands: Output Conditions

The commands available on the **Output Conditions page** are explained in the following table.

Function	Description
Add	Create a new output condition. See *Adding, Displaying and Modifying an Output Condition*.
Delete	Delete the condition as an output condition for the job. See *Deleting an Output Condition*.
Modify	Modify the output condition definition. See *Adding, Displaying and Modifying Output Condition*.
Where used	Open a **Where used Condition window** with a list of jobs for which this condition is defined as an input or output condition. For more information, see the corresponding function for input conditions described in *Listing Jobs Linked to an Input Condition*.

Deleting an Output Condition

To delete an output condition

1 On the **Output Conditions page**, select the output condition you want to delete and choose **Delete**.

A confirmation window opens.

2 Choose **Yes**.

(**No** cancels the action.)

The confirmation window closes and the selected output condition is removed.

Defining Actions for Symbol Value Modification

You can define a value to be assigned to a symbol or the substring of a symbol value (analogous to a BS2000 job variable) as End-of-Job action depending on the occurrence of an event. The value assignment can be performed in the master or active symbol table, or in both tables. It is possible to modify symbol tables different from the ones specified for the job or the network.

To add, view or change a symbol value setting

1 On the **EOJ Checking page**, select the event for which you want to set a symbol value and choose **Add**.

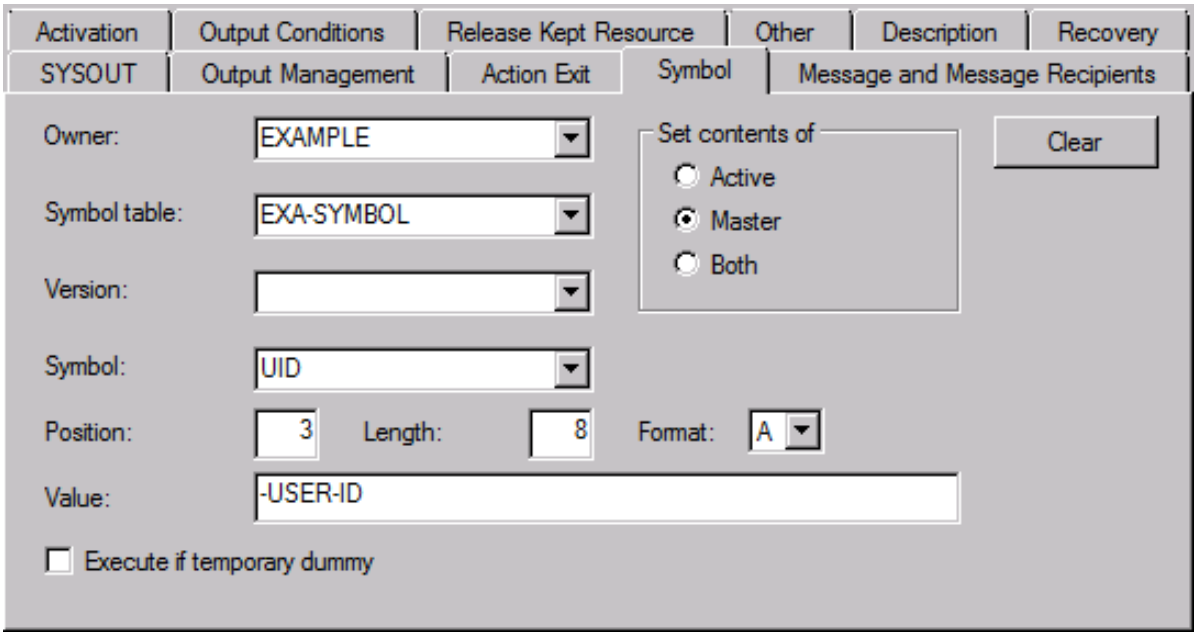
Or:

On the **EOJ Checking page**, select the event whose symbol setting you want to view or change and choose **Modify**.

The letter **Y** in the **Action** column indicates whether a symbol value has already been set for the event.

The **EOJ checking window** opens.

2 Open the tabbed page **Symbol** like the example below:



In the example above, the string `-USER-ID` replaces or sets the value for the symbol `UID` in the `EXA-SYMBOL` master table of the owner `EXAMPLE`:

- Assuming the old value is `NOP`, the new value in the master table will be `NOP-USER-ID`.
- Assuming the old value is `VVVVVVVVVVVVVVVVVVV`, the new value in the master table will be `VVV-USER-IDVVVVVVVVV`.
- Assuming an old value does not exist (`empty` value), the value in the master table will be set to

```
   -USER-ID
```

(with three leading blanks).

3 In the input fields, enter the required values. They are described in *Fields: Symbol Modification*

You can remove all current entries from the window by choosing **Clear**.

4 Choose **OK** when you are finished.

The symbol value setting is saved and the **Symbol** page closes.

To remove a symbol value setting

- Proceed as described in *Removing Actions*.

This section covers the following topics:

- Fields: Symbol Modification

Fields: Symbol Modification

The fields available on the **Symbol page** are explained in the following table.

Field	Description
Owner	Owner of the symbol table. When a symbol is set to **Active** or **Both**, the owner of the symbol table and the owner of the job must be the same. This is indicated by `(SAME)` or an empty field, which is the default value.
Symbol table	Name of the symbol table that contains the symbol whose value you want to set or change. The symbol table must be defined for the network master of the job master. `(SAME)` or an empty field indicates that the owner of the symbol table is also the owner of the job.
Version	Version of the specified symbol table. The current version (empty field) is the default.
Set contents of	Instance of the symbol/symbol table where to perform the symbol value setting/modification. Valid selection options: **Active** — Perform modification in the active symbol table (default). **Master** — Perform modification in the symbol table master. **Both** — Perform modification in both the active symbol table and the symbol table master.
Symbol	Name of the symbol. **Symbol replacement** is possible. Example: `@P-JOB.-STATUS` If @ (commercial at) is used as the submission escape character, for a job with the name `JOB-1`, a symbol with the name `JOB-1-STATUS` is set.
Value	Value to be set into the complete symbol value or a substring of the symbol value. **Symbol replacement** is possible. If the effective value length (after symbol replacement) is longer than the value specified in **Length**, the value will be shortened to the length specified there.
Position	Position at which the specified symbol value or a substring of the symbol value is set. Possible values: 1 (default) to 120 characters.

Field	Description
Length	Length of the substring of the symbol value to be set. Possible values: 1 to 120 characters. **Note:** The effective **value** length may be shortened due to this definition. If you want to make sure that an old value is entirely replaced, enter a new value of 120 characters.
Format	Format in which the substring of the symbol value is to be set. Possible selection options: A – Alphanumeric. L – Alphanumeric (lower case). U – Alphanumeric (upper case). N – Numeric.
Execute if temporary dummy	See *End-of-Job Actions after Execution as a Temporary Dummy Job*.

Defining Actions for Modifying Job Variable Values

(Applies to BS2000 only)

You can define a value to be assigned to a job variable (analogous to a symbol) as End-of-Job action depending on the occurrence of an event.

To add, view or change a job variable setting

1 On the **EOJ Checking page**, select the event for which you want to define a job variable setting and choose **Add**.

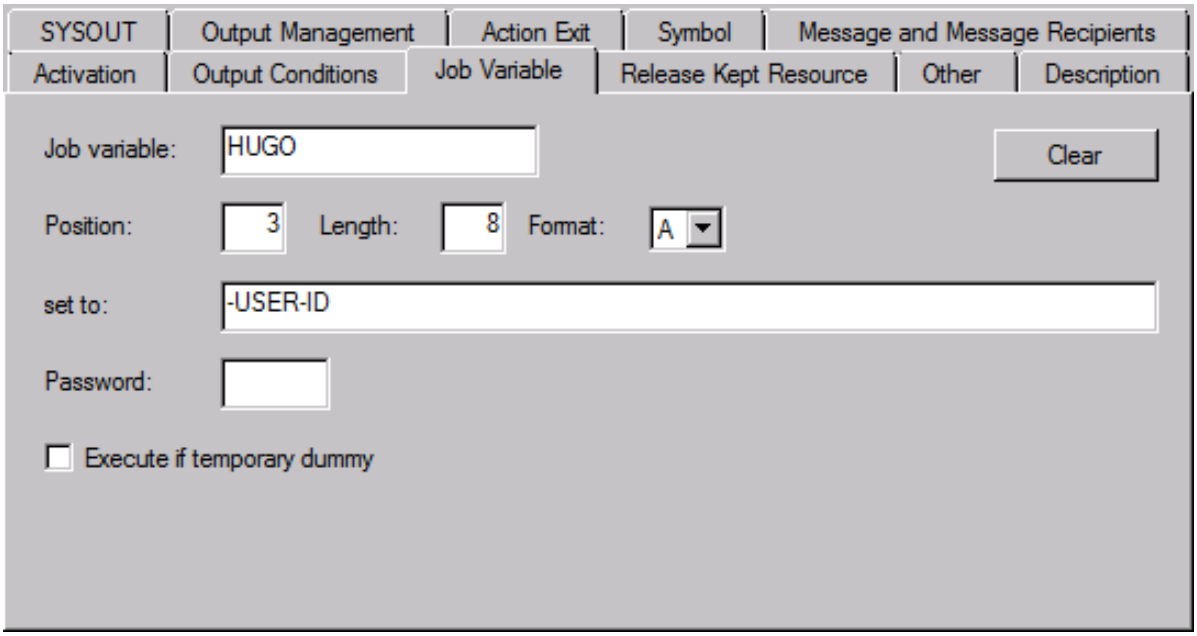
Or:

On the **EOJ Checking page**, select the event whose job variable setting you want to view or change and choose **Modify**.

The letter **J** in the **Action** column indicates whether a variable setting has already been defined for the event.

The **EOJ checking window** opens.

2 Open the tabbed page **Job Variable** like the example below:



In the example above, the string `-USER-ID` replaces or sets the value for the job variable `HUGO`:

- Assuming the old value is `NOP`, the new value will be `NOP-USER-ID`.
- Assuming the old value is `VVVVVVVVVVVVVVVVVVV`, the new value will be `VVV-USER-IDVVVVVVVVV`.
- Assuming an old value does not exist (`empty` value), the value will be

```
   -USER-ID
```

  (with three leading blanks).

If no job variable has been defined, the fields in this window are empty; you can set a job variable, by entering the values here.

3 In the input fields, enter the required values. They are described in *Fields: Job Variable Modification*.

You can remove all current entries from the window by choosing **Clear**.

4 Choose **OK** when you are finished.

The job variable setting is saved and the **Symbol** page closes.

To remove a job variable setting

- Proceed as described in *Removing Actions*.

Fields: Job Variable

Field	Description
Job variable	Name of a valid BS2000 job variable. If the job variable is specified without an explicit user ID, the job default BS2000 user ID is used as a prefix. **Symbol replacement** is performed in the name if the name contains the **activation escape character** at least once.
Position	Position of the substring of the job variable value to be checked. Possible values: 1 to 253.
Length	Length of the substring of the job variable value to be checked. Possible values: 1 to 253.
Format	Format in which the substring of the job variable value is to be checked against the comparison string. Possible selection options:
	A alphanumeric
	N numeric
set to	Enter the string or field to be set as the job variable value or as a substring of the job variable value. **Symbol replacement** is performed if the field contains the **activation escape character** at least once.
Password	(Optional field) Password to be entered if the job variable is password-protected.
defined	Information field that indicates whether a **Write Password** is defined.
Execute if temporary dummy	See *End-of-Job Actions after Execution as a Temporary Dummy Job*.

Defining Action User Exits

You can define user exits for End-of-Job checking and End-of-Job actions.

For information on defining user exits, see *User Exits for End-of-Job Checking and Actions* in the section *User Exits*

This section describes how to specify an action user exit for an event. Depending on your permissions, you can also add or edit an action user exit.

Related Topics:

- *User Exits for End-of-Job Checking and Actions* in the section *User Exits*

To add, view or change an action user exit definition

1 On the **EOJ Checking page**, select the event for which you want to define an action user exit and choose **Add**.

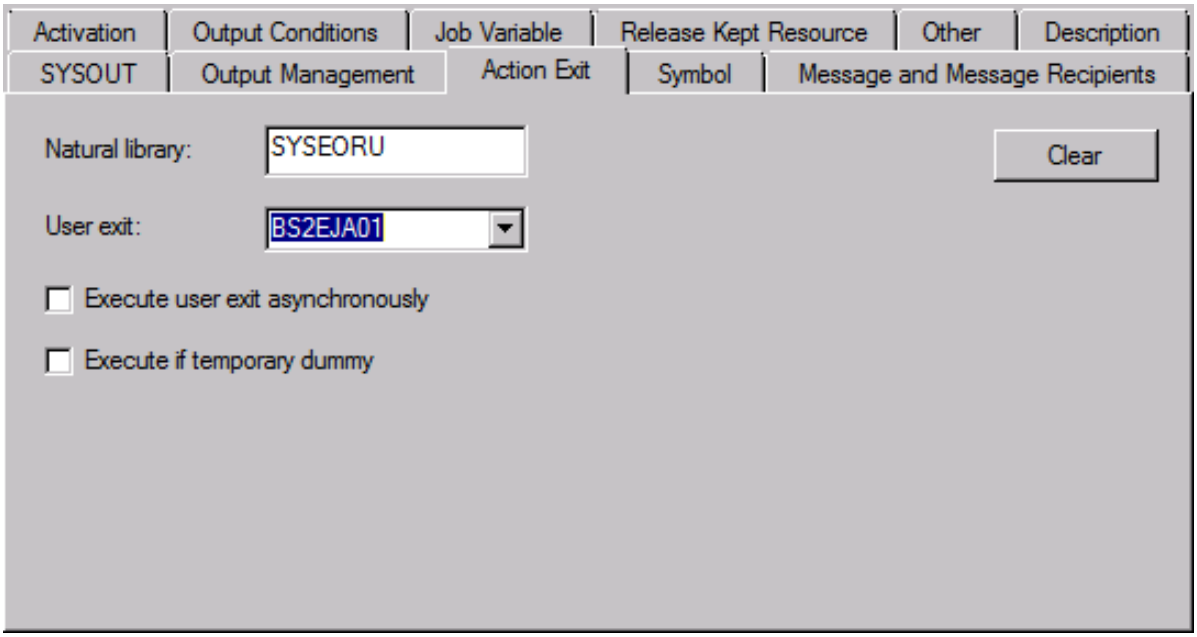
Or:

On the **EOJ Checking page**, select the event whose user exit definition you want to view or change and choose **Modify**.

The letter **X** in the **Action** column indicates whether an action user exit.

The **EOJ checking window** opens.

2 Open the tabbed page **Action Exit** like the example below:



3 In the input fields, enter the required values. They are described in *Fields: Action User Exit*.

If required, choose **Clear** to remove all current entries from the window.

You can also specify the name of a user exit that does not yet exist. Entire Operations saves the definition assuming that you will create the user exit at a later time.

For information on coding user exits, see *User Exits for End-of-Job Checking and Actions* (section *User Exits*) and *Editing Master JCL and Natural Sources*.

4 When you are finished, choose **OK** to save your entries.

To remove an action user exit definition

- Proceed as described in *Removing Actions*.

This section covers the following topics:

- Fields: Action User Exit

Fields: Action User Exit

The fields available on the **Action Exit page** correspond to the fields available for a checking user exit (event type **User exit** in the **Event type specific settings** section of the **EOJ checking window**.

Field	Description
Natural library	Name of the Natural library that contains the user exit. See also **in Natural Library/in NAT Library Natural library** in *Fields: EOJ Checking Window*.
User exit	Name of the user exit. See also **User exit** in *Fields: EOJ Checking Window*.
Execute user exit asynchronously	See **Execute user exit asynchronously** in *Fields: EOJ Checking Window*.
Execute if temporary dummy	See *End-of-Job Actions after Execution as a Temporary Dummy Job*.

Defining Other Actions: Deactivate Job Automatically

The **tabbed page Other** is used to deactivate a job automatically, even if it ended `not OK`.

Select **Accept** from the drop-down list box if you want to deactivate the active job after termination even when it terminates with `not OK`. If you select **Do not accept**, the job remains active until you have corrected or manually deactivated it.

Choose **Clear** if you want to reset the selection box to its default (blank) value **Do not accept**. See also *Removing Actions*.

Defining Network and Job Activation Actions

Related Topic:

- *Removing End-of-Job Actions*

The occurrence of a defined event during job execution can trigger the activation of a specified single job or a whole job network. You can define activation of a job or network for specific events or at the job level for **All checks ok** and **Any check not ok**.

Activation is only possible if the network belongs to the defining user or if the defining user is authorized to activate the network of another user. The user defining the network to be activated must at least have authorization to activate this network (see *Authorizing Other Users or Owners to Access a Network (Granting Access)* in the section *Network Maintenance*).

To add, view or change an activation definition

1 On the **EOJ Checking page**, select the event for which you want to define an activation action and choose **Add**.

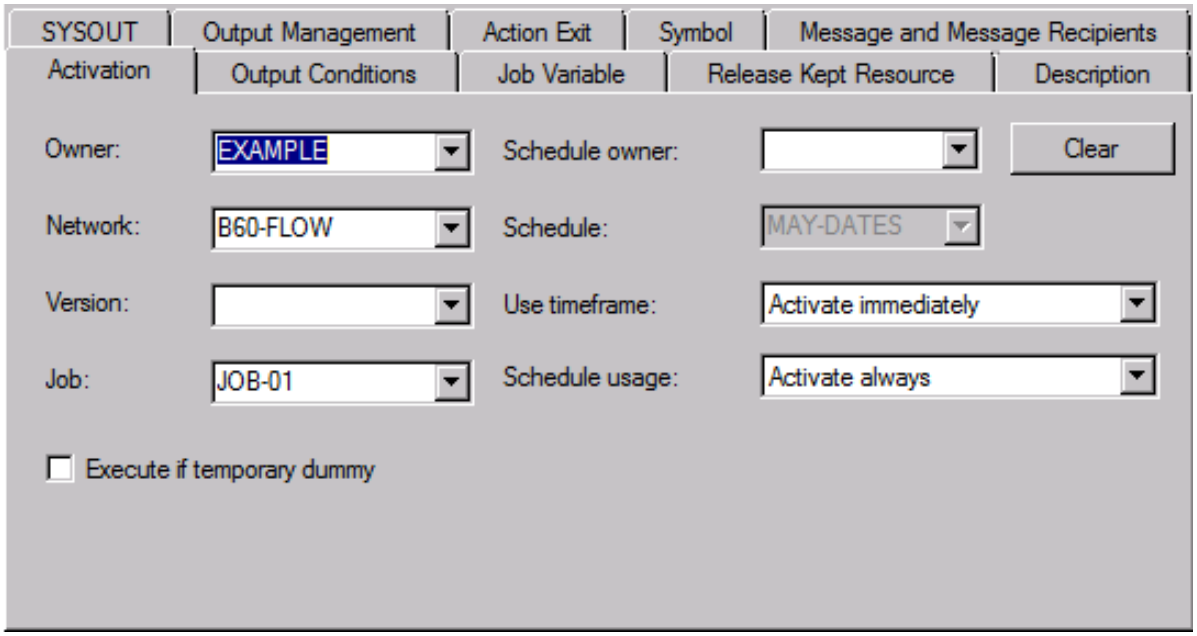
Or:

On the **EOJ Checking page**, select the event whose activation definition you want to view or change and choose **Modify**.

The letter **A** in the **Action** column indicates whether an activation action has already been defined for the event.

The **EOJ checking window** opens.

2 Open the tabbed page **Activation** like the example below:



3 Enter the required values. The input fields and options are explained in *Fields: Network and Job Activation*.

If required, choose **Clear** to remove all current entries.

4 Choose **OK** when you are finished.

Your action definition is saved and the window closes.

This section covers the following topics:

- Fields: Network and Job Activation

Fields: Network and Job Activation

Field	Description
Owner	Name of the owner of the network to be activated. For possible input values to list a specified name range from the drop-down list box (if available), see *Specifying Filter Criteria*.
Network	Name of the network to be activated. For possible input values to list a specified name range from the drop-down list box (if available), see *Specifying Filter Criteria*.
Version	Version of the network to be activated. The current version (empty field) is the default.

Field	Description	
	For possible input values to list a specified name range from the drop-down list box (if available), see *Specifying Filter Criteria*.	
Job	Job to be activated. If the field is empty (default), the whole network is activated. For possible input values to list a specified name range from the drop-down list box (if available), see *Specifying Filter Criteria*.	
Use timeframe	Valid selection options:	
	Use time frame of called network	Activate the End-of-Job action according to the time frame/schedule defined for the called network.
	Activate immediately	Activate the job immediately (default).
Schedule usage	Valid selection options:	
	Explicitly defined schedule	Activate the End-of-Job action only if the current day is defined in the network's resp. explicitly defined schedule.
	Activate always	Always activate the End-of-Job action (default).
Schedule owner	Name of the owner of the schedule to be used for activation. If this field is empty (default), the owner of the network to be activated is used.	
Schedule	Name of the schedule to be used for activation. If this field is empty (default), the schedule of the network to be activated is used.	
Execute if temporary dummy	See *End-of-Job Actions after Execution as a Temporary Dummy Job*.	

Defining Recovery Actions

You can define a recovery process to be started as the result of a defined event during job execution. You can also define recovery action at the job level according to **Any check not ok**.

Usually, recovery is used in case of job failure. For example, a recovery job can be a compression job started after the message `Library full` is received. The string `Library full` should be defined as a string event (event type **S**). Entire Operations starts the recovery job automatically when the event occurs.

The recovery process comprises the following tasks:

- Activation of a recovery network or a recovery job.
- Creation of an internal condition to provide a correct return to the calling network.

- Rescheduling of the job to be recovered.

  One recovery network can be used in multiple recovery definitions.

  It always returns to the calling network.

  It is possible to stop the calling network after the recovery.

The conventional way to trigger a recovery network would be to define an event which activates the recovery network. Recovery is then started only if the event occurs or if any check is `not OK`. The recovery network must be defined as a network master.

To add, view or change a recovery definition

1 On the **EOJ Checking page**, select the event for which you want to define a recovery definition and choose **Add**.

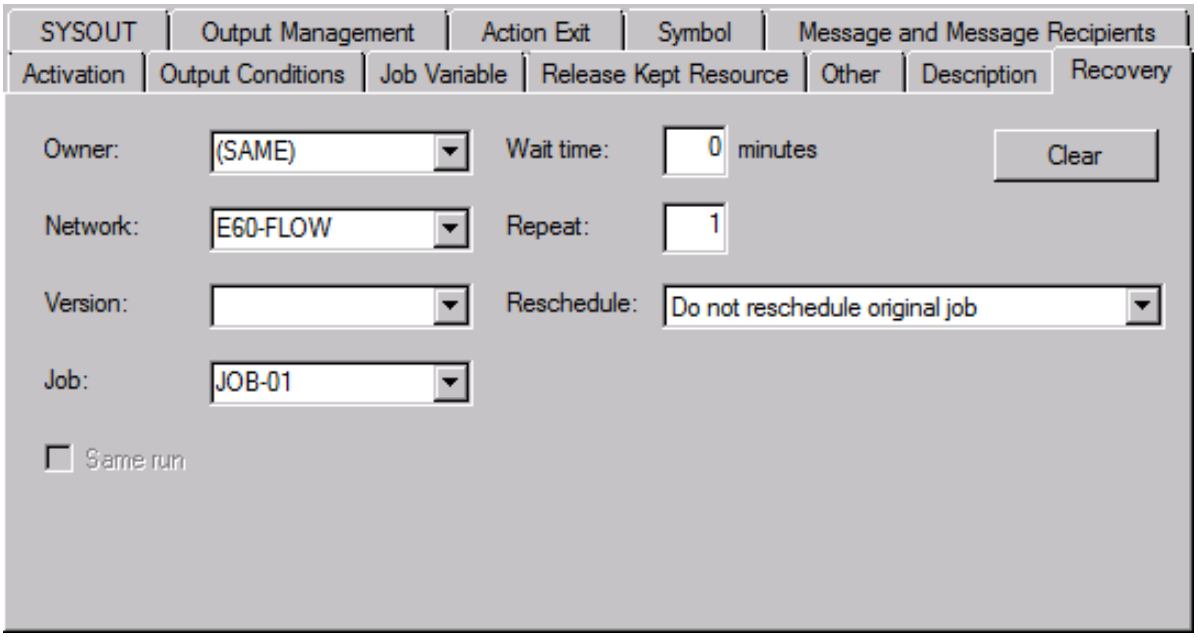
  Or:

  On the **EOJ Checking page**, select the event whose activation definition you want to view or change and choose **Modify**.

  The letter **R** in the **Action** column indicates whether an activation action has already been defined for the event.

  The **EOJ checking window** opens.

2 Open the tabbed page **Recovery** like the example below:



3 Specify the recovery network to be started. The input fields and valid values are explained in *Fields: Network and Job Activation*.

If required, choose **Clear** to remove current entries.

4 Choose **OK** when you are finished.

Your recovery definition is saved and the window closes.

To remove a recovery definition

- Proceed as described in *Removing Actions*.

This section covers the following topics:

- Fields: Recovery Definition
- System Symbols for Recovery Actions

Fields: Recovery Definition

The fields on the **Recovery page** are described in the following table:

Field	Description	
Owner	Owner of the network that contains the recovery job(s). selection options:	
	owner-name	Owner name other than the current owner. For possible input values to list a specified name range from the drop-down list box (if available), see *Specifying Filter Criteria*.
	(SAME)	Owner of the current network is used (default). The default value is replaced at execution time.
Network	Network that contains the recovery job(s). The recovery network must be defined in the Network Maintenance facility. selection options:	
	network-name	Name of the network to be recovered. For possible input values to list a specified name range from the drop-down list box (if available), see *Specifying Filter Criteria*.
	(SAME)	(Only applies to single recovery jobs.) Same name (default) as the current network that contains the job to be recovered. The default value is replaced at execution time.

Field	Description	
		When you copy a recovery job to another network, the default values (SAME) for owner, network, and network version remain unchanged, so that the recovery job can be used unchanged in the network to which it was copied.
Version	Version of the network that contains the recovery job(s). selection options:	
	network-version	Version of the network that contains the job(s) to be recovered. For possible input values to list a specified name range from the drop-down list box (if available), see ***Specifying Filter Criteria***.
	(SAME) or empty field	Same version (default) as the version of the network that contains the job(s) to be recovered. The default value is replaced at execution time. When you copy a recovery job to another network, the default values (SAME) for owner, network, and network version remain unchanged, so that the recovery job can be used unchanged in the network to which it was copied.
	(CURRENT)	Current version of the network that contains the job(s) to be recovered.
Job	Name of the last job in the recovery network to be executed upon job failure. This job must terminate successfully to start rescheduling. For possible input values to list a specified name range from the drop-down list box (if available), see ***Specifying Filter Criteria***.	
Same run	Single recovery jobs can be defined in the same network and can be executed under the same run number as the network to be recovered. All active objects of the network are then accessible for the recovery job with this run number. Possible check-box settings:	
	checked	Use same run number (default for single jobs).
	unchecked	Use a separate run number for the recovery (default for whole network).
Reschedule	Specifies whether the job is to be rescheduled for submission after recovery. Possible selection options:	
	Reschedule original job	Resubmit the job.
	Do not reschedule original job	Do not resubmit the job (default).
	Stop active network after recovery	Stop the original network after recovery.
Repeat	Maximum number of times the original job is to be rescheduled after a recovery. Possible values: 1 (default) to 99.	

Field	Description
	The Entire Operations Monitor sets the reserved condition *jobname*-MAX-RETRY if the retry limit was reached with unsuccessful rescheduling attempts. **Repeat** is meaningful only if **Reschedule** is set to **Reschedule original job**.
Wait time	Time to wait in minutes until the recovery is started, and between the recovery attempts.

System Symbols for Recovery Actions

When defining a recovery action, the following Entire Operations system symbols can be prefixed with the escape symbol for symbol prompting and used, for example, in a comment line in the JCL of the recovery job:

Field	Description
P-C-OWNER	Owner of the job making the call.
P-C-NETWORK	Network of the job making the call.
P-C-NETWORK-VERSION	Version of the network of the job making the call.
P-C-JOB	Name of job making the call.
P-C-RUN	Run number of the job making the call.
P-C-SUFFIX	Value of the suffix symbol of the job making the call.

They are substituted at JCL load with their current values. In the SYSOUT of the recovery job, you can then find the substitution of the parameters:

```
*** Symbol  : P-C-OWNER
  ***   Value : GFR
  *** Symbol  : P-C-NETWORK
  ***   Value : NET-1
  *** Symbol  : P-C-JOB
  ***   Value : JOB-6
  *** Symbol  : P-C-RUN
  ***   Value : 208
  *** ================================================
2 //STEP1    EXEC PGM=NOPCONTI
3 //STEPLIB  DD   DISP=SHR,DSN=NOP.DEV.LOAD
  *** CALLER OWNER   GFR
  *** CALLER NETWORK NET-1
  *** CALLER JOB     JOB-6
  *** CALLER RUN     208
  ***
```

In this way, you can find out the job making the call, that is, the job to be recovered by the recovery job.

Defining SYSOUT Actions

You can define SYSOUT actions to be performed at job completion if the associated event occurs.

You can define the job SYSOUT to be cancelled or printed as the result of a defined event or the End-of-Job status (**All checks ok** and **Any check not ok**).

End-of-Job user exits are available for more complex actions, for example for obtaining selected information from the job SYSOUT. See also ***User Exits for End-of-Job Checking and Actions*** in the section *User Exits*.

To add, view or change a SYSOUT action

1 On the **EOJ Checking page**, select the event for which you want to define a SYSOUT action and choose **Add**.

Or:

On the **EOJ Checking page**, select the event whose SYSOUT action you want to view or change and choose **Modify**.

The letter **S** in the **Action** column indicates whether an activation action has already been defined for the event.

The **EOJ checking window** opens.

2 Open the tabbed page **SYSOUT**.

The fields and options available in the window are explained in ***Fields: Network and Job Activation***.

3 Select the SYSOUT action that corresponds to the function you want to perform.

If required, choose **Clear** to remove current entries.

4 Choose **OK** when you are finished.

The specified SYSOUT action is saved and the window closes.

To remove a SYSOUT action

- Proceed as described in ***Removing Actions***.

This section covers the following topics:

- Fields and Options: SYSOUT Actions

Fields and Options: SYSOUT Actions

Field	Description	
SYSOUT action	Select one of the following options to perform the required function:	
	Delete	Delete SYSOUT on job completion.
	Print	Release SYSOUT for printing.
	Log	Log SYSOUT in the Entire Operations log file.
	Log, then delete	Log SYSOUT and then delete it.
	Log, then print	Log SYSOUT and then print it.
	Pass to Entire Output Management	**Pass SYSOUT to Entire Output Management** (NOM).
	Spool class	You can specify that the spool class of a job is to be modified after completion. The definition created here overrides the default setting. **Note:** 1. This field only appears if the executing operating system is z/OS. 2. The class defined here has priority over other definitions. 3. If no definition is made on the job level, the default for the node level is used: see *Other Definitions for a Node (Mainframe)* in the *Administration* documentation. 4. If no other definition is found, the system-wide default for the operating system is used (see *Defaults for z/OS* in the *Administration* documentation). 5. We recommend that you only define SYSOUT class changes under **All checks ok** or **Any check not ok**. 6. The spool class change is allowed only if the job is in the OUT queue.

Defining Notification Messages

You can define a notification message to be sent as the result of a defined event, or if the job terminated `OK` or `not OK`. This is especially useful to inform appropriate user of job failures.

A user-defined message comes along with job execution information generated by Entire Operations. This information is more comprehensive if e-mails are used for sending; mailboxes have less information because they are limited to a single line. For example:

Mailbox message:

```
This line is the user-defined text.
==> EXAMPLE / EXA-NET3 / 10 / EXA-JOB1
```

Corresponding e-mail message:

```
Owner EXAMPLE - Network EXA-NET3 - Run 10 - Job EXA-JOB1
... Execution Node 123 (DAEF-123)
Event: Additional Actions for Job-ok

Ended ok
This line is the user-defined text.
```

A message can be defined for any event or at the job level according to **All checks ok** and **Any check not ok**.

The Entire Operations Monitor continues to send a message until it has been successfully transmitted.

This section covers the following topics:

- Displaying, Modifying and Adding Notification Messages
- Deleting Notification Messages
- Fields and Columns: Message and Message Recipients
- Rules for Message Text
- Rules for E-Mail Attachments
- Available Functions: Message and Message Recipients
- E-Mails on z/OS
- E-Mails on BS2000
- E-Mails on UNIX and Windows

Related Topic:

- *Global Messages for Events* in the *Administration* documentation

Displaying, Modifying and Adding Notification Messages

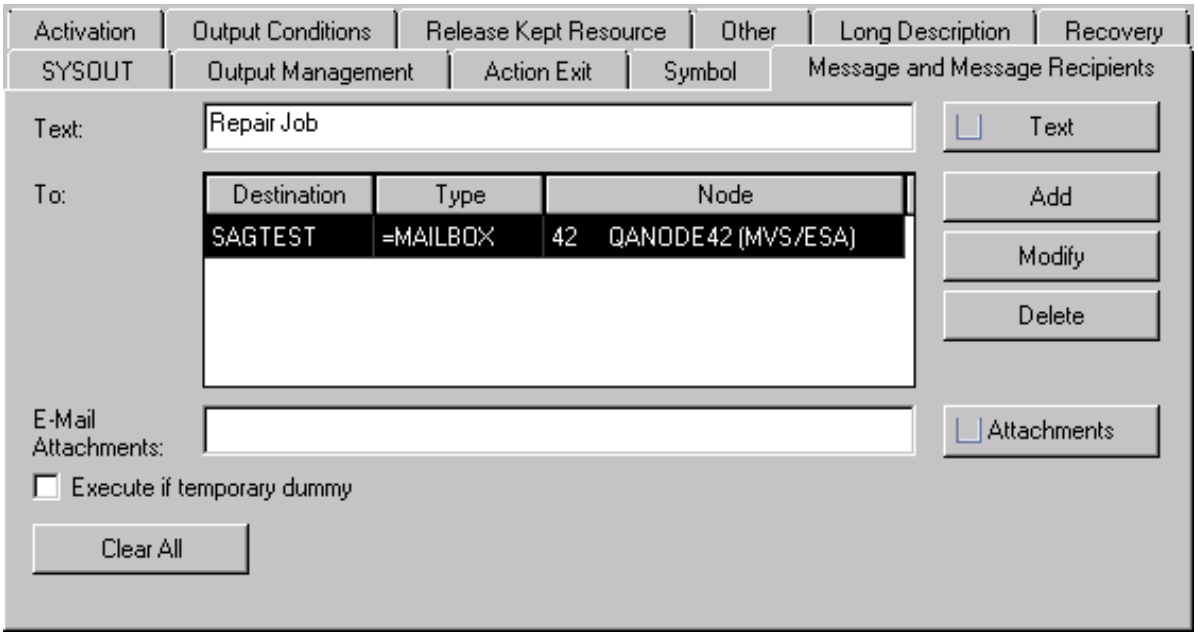
To view or change a notification message

1 On the **EOJ Checking page**, select the event whose message you want to view or change and choose **Modify**.

The letter **U** in the **Action** column indicates whether a message has already been defined for the event.

The **EOJ checking window** opens.

2 Open the tabbed page **Message and Message Recipients**:



The page shows the short message text and the recipient(s) defined for the message.

All fields contained on the page are described in *Fields and Columns: Message and Message Recipients*.

All functions provided on the page are described in *Available Functions: Message and Message Recipients*

3 **Message texts**

If required, write new message text or replace the current text in the **Text** field.

Or:

Choose **Text** to open a **Text** dialog to write or display text that exceeds the length of the **Text** field.

In the **Text** dialog, you can choose **Clear** to remove all current text entries.

See also *Rules for Message Text*.

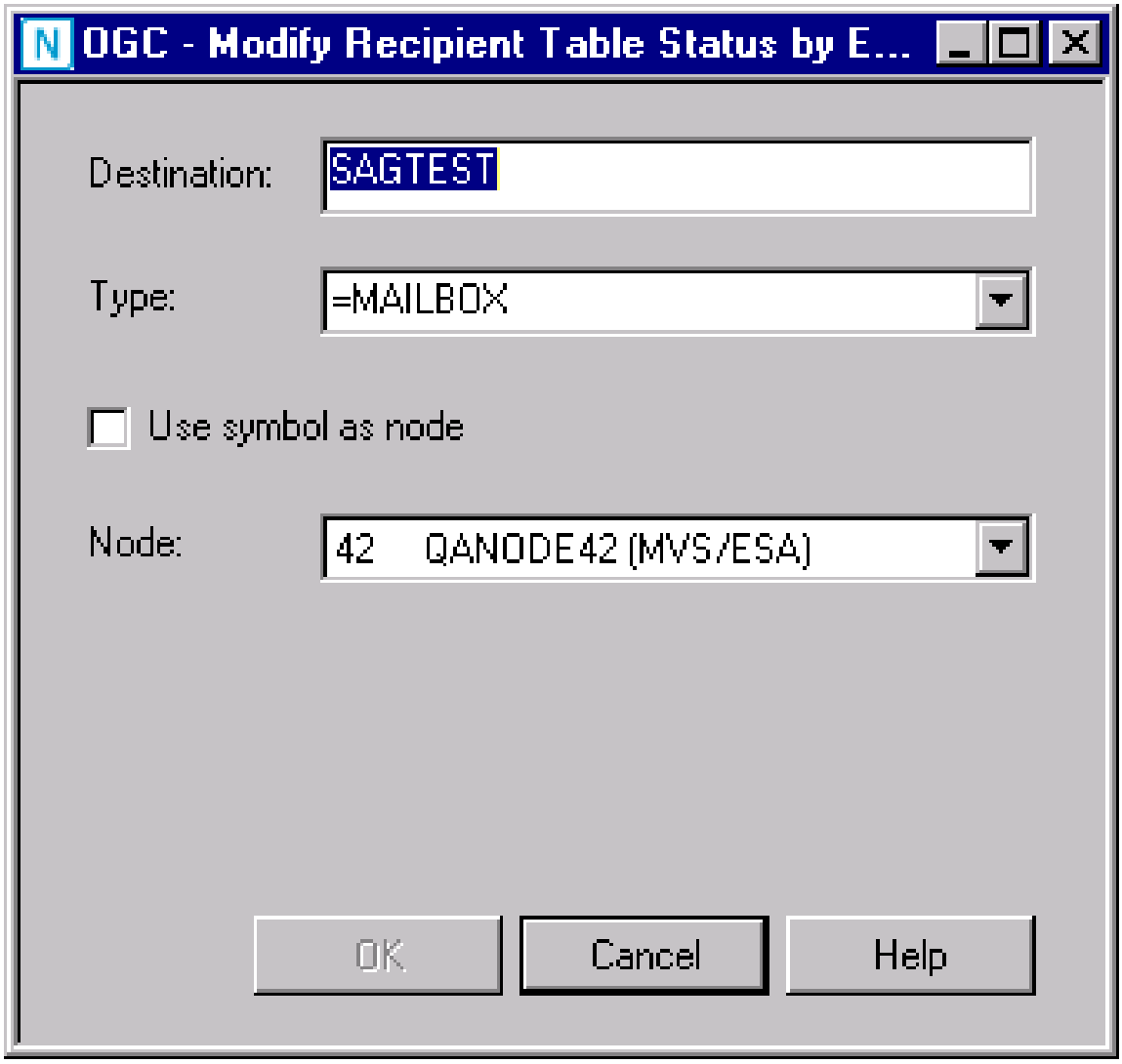
Recipients

If you want to add a recipient definition, choose **Add**.

Or:

If you want to change a recipient definition, in the **To:** section, select the table row that contains the required definition and choose **Modify**.

A **Recipient Table** window like the following opens:



Add or change the definition as required and choose **OK**.

The input fields are described in *Fields and Columns: Message and Message Recipients*.

Or:

If you want to delete a recipient definition, in the **To:** section, select the table row that contains the required recipient and choose **Delete**.

Caution: Use this function with care.

E-mail attachments

If you want to attach a file to the message, choose **Attachments**.

An **E-Mail Attachments** dialog opens where you can enter one or more addresses of files to be attached to the message.

See also *Rules for E-Mail Attachments*.

You can choose **Delete** to remove all current entries.

4 Choose **OK** when you are finished.

The specified notification message is saved and the window closes.

Or:

Choose **Clear All** if you want to remove all current text entries (message text and all recipient definitions).

To add a notification message

1 On the **EOJ Checking page**, select the event for which you want to define a message and choose **Add**.

The **EOJ checking window** opens.

2 Open the tabbed page **Message and Message Recipients**, enter message text in the **Text** field and choose **Add** to define message recipients.

All fields contained on the page are described in *Fields and Columns: Message and Message Recipients*.

All functions provided on the page are described in *Available Functions: Message and Message Recipients*

3 Proceed as described in **Steps 3 to 4** of *To view or change a notification message*.

Deleting Notification Messages

To remove a notification message

- Proceed as described in *Removing Actions*.

Fields and Columns: Message and Message Recipients

The fields and columns on the **Message and Message Recipients page** and in the **Recipient Table** window are described in the following table.



Note: The fields are mandatory unless indicated otherwise in the table. All fields are cleared if you leave a mandatory field empty and you cannot save the message definition.

Field/Column	Description	
Text	One-line text of the message to be sent if the associated event occurs. The text in the **Text** field is dimmed if additional text exists. See also *Rules for Message Text*.	
E-Mail Attachments	Shows the address of the file attached to an e-mail. The address in the **E-Mail Attachments** field is dimmed if more than one address exists. See also *Rules for E-Mail Attachments*.	
Destination	Message destination or recipient, respectively. In z/OS: enter the user ID of a TP system user. In BS2000: enter a terminal name and the related processor name in the **Type** field. In UNIX: enter a UNIX user ID valid in the UNIX environment of the addressed server. The message is sent with the UNIX mail function. Since the **Destination** field is usually too short for an e-mail address, it is recommended to put a symbol into this field, preceded by the **submission escape character** (in the previous example, `$EMAIL-ID`). The effective e-mail address must be defined as symbol value in the symbol table used by the job. If e-mail addresses are to be defined (in an EBCDIC character set) on the mainframe, the string `(a)` must be used instead of the `@` (commercial at) sign, for example: `user(a)any.host`	
Type	Message recipient (destination) type. Possible selection options:	
	`=EMAIL`	Sends the message by e-mail to any recipient in the intranet or internet. The e-mail is sent in plain text mode. As Windows does not have a built-in `sendmail` command, a command line tool must be specified in the node definition. See also *E-Mails on UNIX and Windows*.

Field/Column	Description	
	=EMH	Sends the message by e-mail to any recipient in the intranet or internet. The e-mail is sent in HTML format. See =EMAIL for more information on sending e-mail.
	=EXIT	Use the global message sending exit. See *Global Message Sending Exit* in section *Global User Exits* of the *Administration* documentation.
	=MAILBOX	Sends the message to the internal Entire Operations mailbox specified in **Destination**.
	=COMPLET	Sends the message explicitly to a Com-plete user (z/OS only).
	=TSO	Sends the message explicitly to a TSO user (z/OS only).
	=CONSOLE	Sends the message to the operator console (mainframes only). An entry in the **Destination** field is not required.
	processor-name	BS2000: The processor name related to the BS2000 terminal name specified in **Destination**.
Node	Message sending node. Enter the Entire System Server node through which the message is routed. You can define the message sending node as a symbol if the **Use symbol as node** option is selected. For details, see ***Symbols in Node Definitions*** in the section *Symbol Table and Symbol Maintenance*. Default is the execution node of the job.	
Use symbol as node	Select this option to define the message sending node as a symbol.	
Execute if temporary dummy	See ***End-of-Job Actions after Execution as a Temporary Dummy Job***.	

Rules for Message Text

The following rules apply when writing message text on the **Message and Message Recipients** page:

Message Text Field

- The **Text** field can contain up to 120 characters on a GUI client and 45 characters on a CUI client. Characters that exceed the CUI limit are truncated on the CUI screen.
- If the text contains an **activation escape character**, text replacement is performed from the active symbol table.

Message Text Window

- If the **Text** field contains text, the first input line of the **Text** window is filled with this text by default.

  If no text is entered in the **Text** field, the first input line of the **Text** window is copied into the **Text** field by default. Text that exceeds the 45-character limit of the **Text** field is truncated.

  Text can be entered on 10 input lines with a maximum of 120 characters per line on a GUI client and 76 characters on a CUI client. Characters that exceed the CUI limit are truncated on the CUI screen. Empty lines are removed from the text.
- If the text contains an **activation escape character**, text replacement is performed from the active symbol table.
- The text in the **Text** field is removed when the text in the **Text** dialog is deleted.



Note: If the text on the **Message and Message Recipients** page is not modifiable, the effective message text is automatically generated for the event by Entire Operations.

Rules for E-Mail Attachments

The following rules apply when attaching a file to an e-mail from the **Message and Message Recipients** page:

Send Limits

You can attach up to 5 files to a message for e-mails sent through UNIX or Windows execution nodes.

You can attach only one file to e-mails sent through mainframe nodes.

There is no limitation for the size of an attachment. However, consider that large attachments, for example, consume large database space and may need more time to send the message. This can have an impact on the Entire Operations Monitor task used for message sending.

Node Usage

Attachments are always taken from the execution node of the job, for which the message is sent.

For e-mails sent through UNIX nodes: Attachments from any execution node can be sent.

For e-mails sent through mainframe nodes: Attachments can be sent only if they are located on the e-mail sending node (recipient node).

If an attachment cannot be found, a notification is appended to the e-mail body, and a message is written to the **Entire Operations log**.

File Format

Attachments from z/OS can be sequential text files and PDS members only. They are sent with `Content-Transfer-Encoding: 8bit`.

Attachments from BS2000 can be sequential text files only. They are sent with `Content-Transfer-Encoding: 8bit`.

Attachments from UNIX and Windows can be files of any type. They are sent with `Content-Transfer-Encoding: base64`.

File Name

UNIX file names can contain '*' (asterisk). This will be treated as part of the file name and not as a wildcard character. Wildcard characters are not supported.

The file name can contain symbols.

Symbols preceded by the **activation escape character** are replaced at job activation time.

Symbols preceded by the **submission escape character** are replaced during message sending.

SYSOUT

The SYSOUT file of a job can be attached by using the **submission escape character** defined for the job or containing network. Example:

```
^P-SYSOUT
```

(assuming ^ is the submission escape character).

The attachment of a job's SYSOUT to an e-mail is available, if the job was executed on BS2000, UNIX, or Windows. It is not available for z/OS.

Note: You must use the submission escape character for P-SYSOUT. Depending on the **repetition count** of the job, different SYSOUT file names will be generated by Entire Operations, as part of the submission process.

File Address

The maximum file address is 120 characters on a GUI client and 76 characters on a CUI client. Characters that exceed the CUI limit are truncated on the CUI screen.

Each file name must begin on a new line.

Files must be specified with their full path name. Example:

```
/home/sag/work/NOP 551 Adabas Files.docx
```

A Windows path name can be specified with a slash (/) or a back slash (\).

For mainframes, the file address can be specified using the syntax *file-name*(*member-name*), where *member-name* is the name of a source file. Example:

```
PDS.SOURCE1(IEFBR12)
```

A mainframe file address is converted to upper case during job activation.

Available Functions: Message and Message Recipients

The following functions are provided on the **Message and Message Recipients page**:

Function	Shortcut	Description
Add	---	Add a recipient definition to the **To:** section of the page.
Modify	---	Modify the recipient definition selected in the **To:** section of the page.
Delete	---	Delete the recipient definition selected in the **To:** section of the page. **Caution:** Use this function with care.
Clear All	---	Clear all fields on the page.
Text	---	Add additional message text. See ***Rules for Message Text***.
Attachments	---	Attach one or more files to an e-mail message. See ***Rules for E-Mail Attachments***.

E-Mails on z/OS

1. The sender name contains `Entire Operations` for all sending platforms.
2. An additional line with the Entire Operations event text is added to the mail.

Usage Notes:

- Before using a z/OS Entire System Server node for sending e-mails, you must invoke the node definition in Entire Operations and enter the line command `O` (Other) for that node.

  For further information, see *Special Definitions for a Node (Mainframe)* in the section *Definition of Nodes* in the *Administration* documentation.
- You must define at least the mail destination and the SYSOUT class for the node through which the e-mail is to be routed.

- The e-mail send routine for z/OS performs an Entire System Server logon, if necessary. The user ID for the logon is the submit user ID for the active job. If no submit user ID is defined for the job, the default user ID for the node is used.
- To use a z/OS Entire System Server for sending e-mails, make sure that all users who send e-mail are correctly defined in the security systems (if applicable). It might be necessary to define user IDs as USS users.

  For more information, see the appropriate *Entire System Server* documentation for your operating and security system.

E-Mails on BS2000

E-mails can be sent via BS2000 nodes if Entire System Server nodes are used.

- Before using a BS2000 Entire System Server node for sending e-mails, you must invoke the node definition in Entire Operations and enter the line command `0` (Other) for that node.

  For further information, see *Special Definitions for a Node (Mainframe)* in the section *Definition of Nodes* in the *Administration* documentation.
- The e-mail send routine for BS2000 performs an Entire System Server logon, if necessary. The user ID for the logon is the submit user ID for the active job. If no submit user ID is defined for the job, the default user ID for the node is used.

E-Mails on UNIX and Windows

- UNIX: To be able to send e-mails on UNIX machines from Entire System Server, the following mail programs must be available there:

AIX, HP-UX, Linux, Sun Solaris:	sendmail
Others:	rmail

- Windows: To be able to send e-mails on Windows machines from Entire System Server, you must define a message command. This is a (third-party) command line tool to be used for sending e-mails via Entire System Server/Windows.
- Before using a UNIX or Windows node for sending e-mails, you must invoke the node definition in Entire Operations and enter the line command `0` (Other) for that node.

  The return address of the e-mail is set to the same value as **E-Mail Reply-To**.

  For further information, see *Special Definitions for a Node (UNIX and Windows)* in the section *Definition of Nodes* in the *Administration* documentation.

Defining Release Actions for Kept Resources

You can define an End-of-job action to explicitly release the resource held by another job of this network. The quantity released for the resource is equal to the quantity defined in the **prerequisite resource definition** of the respective job (that is, the quantity that was allocated).

It is possible to release a resource held by any other job of this network with the **Release Kept Resource** option.

To add, view or change a resource release definition

1 On the **EOJ Checking page**, select the event for which you want to define a resource release action and choose **Add**.

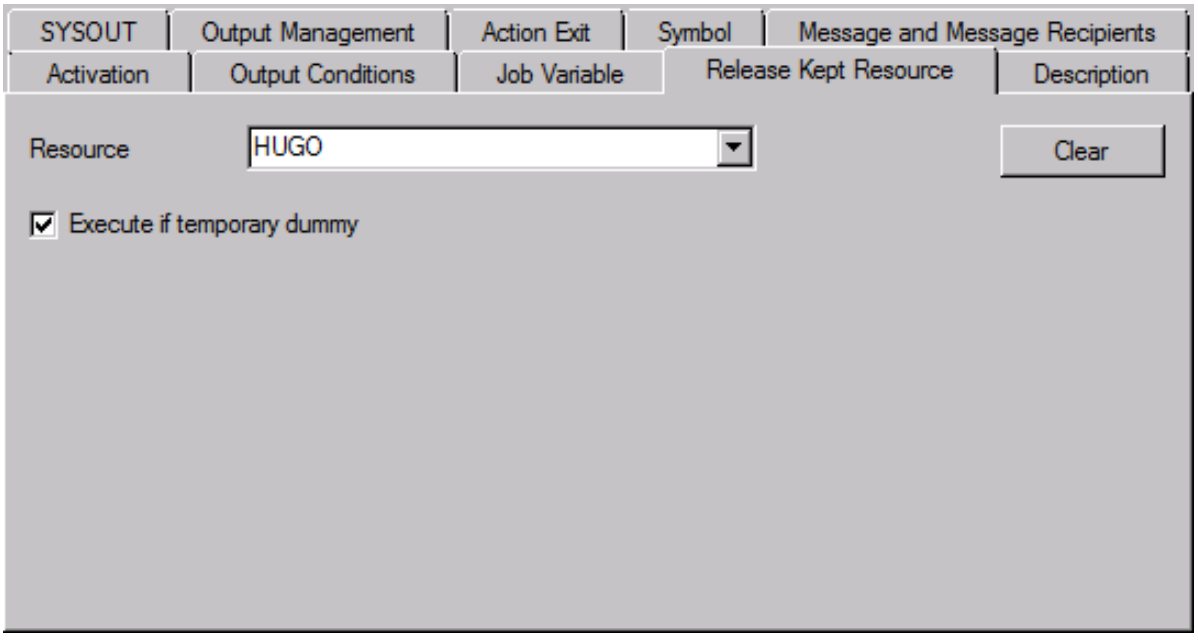
Or:

On the **EOJ Checking page**, select the event whose resource release definition you want to view or change and choose **Modify**.

The letter **L** in the **Action** column indicates whether a resource release definition already exists for a defined event.

The **EOJ checking window** opens.

2 Open the tabbed page **Release Kept Resource** shown in the following example:



3 Enter the required values. The fields are explained in *Fields: Release Kept Resource*.

If required, choose **Clear** to remove the current entries.

4 Choose **OK** when you are finished.

The resource release definition is saved.

To remove a release kept resource definition

- Proceed as described in *Removing Actions*.

This section covers the following topics:

- Fields: Release Kept Resource

Fields: Release Kept Resource

The fields in the **Release Kept Resource page** are explained in the following table.

Field	Description
Resource	Name of the resource to be released. You can select a name from the drop-down list box.
Execute if temporary dummy	See *End-of-Job Actions after Execution as a Temporary Dummy Job*.

35 Scheduling a Job

- Defining Scheduling Parameters for a Job 408
- Maintaining Schedule Dependencies for a Job 412
- Sending a Late Message to One or More Users 417

Defining Scheduling Parameters for a Job

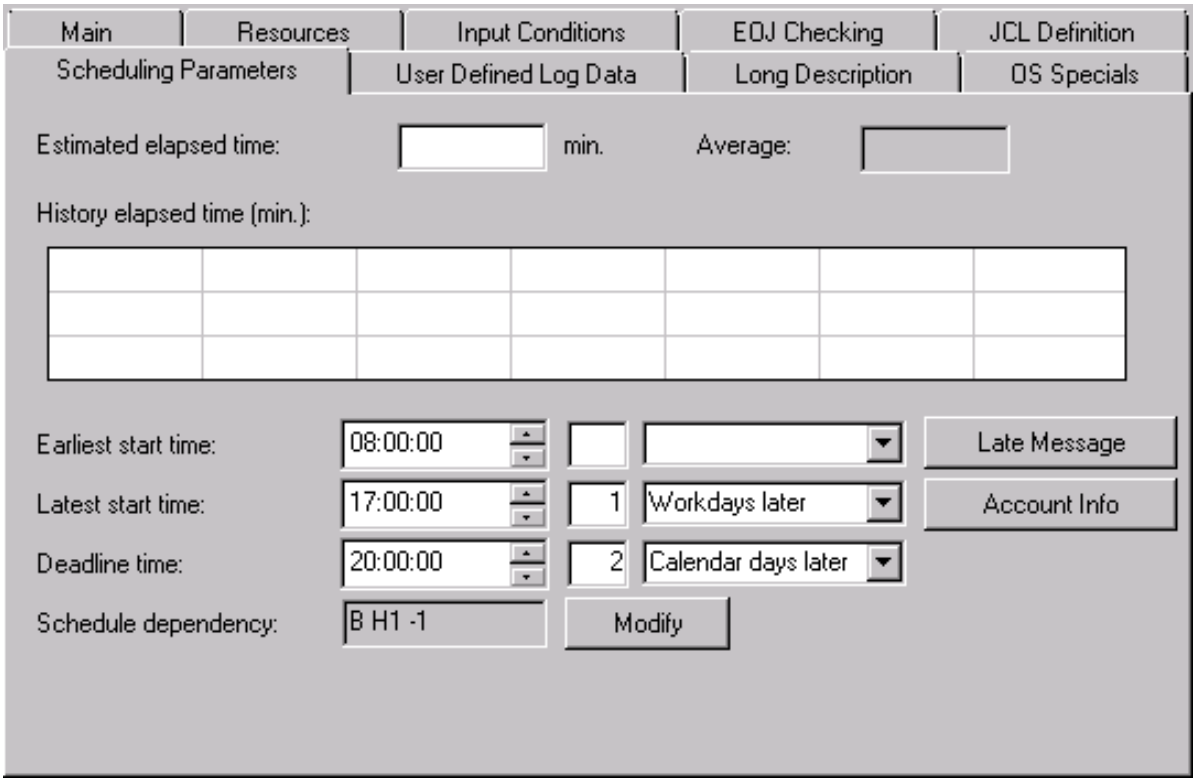
This feature allows you to define a schedule for a job master or an active job. You can define the time frame for the schedule, insert a message, assign account information and edit the schedule dependency information.

To define scheduling parameters for a job

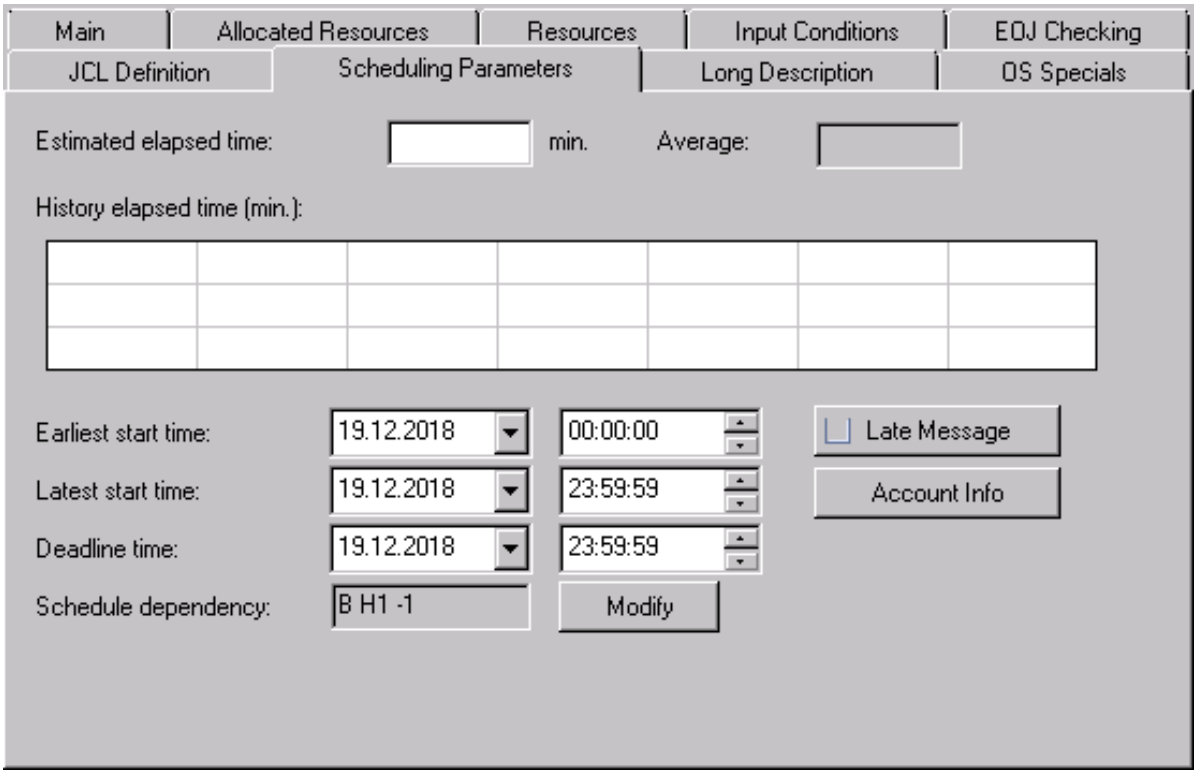
1 In the object workspace, open a **Job Master** instance.

2 In the **Maintenance Job Master** or **Maintenance Job Active** window, open the tabbed page **Scheduling Parameters**.

For a job master, the page looks like the following example:



For an active job, the corresponding page looks like the following example:



3 Make your definitions. The fields contained in the window are explained in *Fields: Scheduling Parameters*.

4 Choose **OK**.

This section covers the following topics:

- Fields: Scheduling Parameters

Fields: Scheduling Parameters

The fields of the **tabbed page Scheduling Parameters** are described in the following table.

Field	Description
Estimated Elapsed Time	The expected total running time (in minutes) for the job. ■ If not specified explicitly, this value is derived from the average running time of previous runs of the same job, displayed in the field directly below. ■ If this value is not equal to zero (0) for a permanent dummy job (**job type Dummy Job**) with a **special type** other than **Execute as dummy** (**dummy due to a condition**), this dummy job is executed for exactly the amount of time given here. This enables you to perform time simulations, etc. ■ Active scheduling parameters for a running dummy job:

Field	Description
	Modification of the estimated elapsed time causes the end time of the dummy job to be recalculated as sum of the start time and the new estimated elapsed time. If the new calculated end time is lower than the current time, the current time will be used instead. ■ Temporary dummy jobs with the job type **Dummy Job** and special type **Execute as dummy** (**dummy due to a condition**) execute for the amount of time defined as estimated elapsed time for the job. ■ For temporary dummy jobs with other **special types** (for example, **dummy due to a condition**), the defined estimated elapsed time is ignored. Temporary dummy jobs always have the execution time zero. Special value: For jobs of the job type **Subnetwork**, 999999 can be specified. In this case, all start times after that are set to the ending times in the subnetwork.
Average	Average running time (in minutes) computed from the last 20 runs shown in the **History Elapsed Time (min.)** table.
History Elapsed Time (min.)	Table containing the actual running times (in minutes) of the last 20 runs of the job. The running time of the last 20 runs with non-zero elapsed times is used to compute the average time (in the **Average** field) the job needs to finish. History elapsed times for multiple jobs (see also the **Suffix Symbol field** described in *Job Maintenance*): ■ For a job master, the table contains the elapsed times of all multiple active jobs, which were last activated by this job master. ■ For an active job, the table contains the elapsed times of all multiple active jobs whose names match the name of this active job.
Earliest Start Time	The job will not be started before this time. Use **days later** if the **Earliest Start Time** is more than one day later than the network start time. You can specify the number of workdays or calendar days to be used for later starts: see **Calendar days later** | **Workdays later** below. See also *Processing of Time Frame Definitions*. For an active job, the **Earliest Start** fields contain the time calculated for the respective job run according to the definitions made in the corresponding job master. Both fields can be modified for the given job run only.
Latest Start Time	The job will not be started after this time. If this time precedes the **Earliest Start Time** and **days later** is empty, **days later** is set to 1 so this time follows the **Earliest Start Time**. You can specify the number of workdays or calendar days to be used for later starts: see **Calendar days later** | **Workdays later** below.

Field	Description
	See also *Processing of Time Frame Definitions*. For an active job, the **Latest Start** fields contain the time calculated for the respective job run according to the definitions made in the corresponding job master. Both fields can be modified for the given job run only.
Deadline Time	Time by which the job must be finished. If the job cannot be finished before this time, it is not submitted. Dummy jobs with **Estimated Elapsed Time** not equal to zero (0) are finished at this time at the latest. You can specify the number of workdays or calendar days to be used for later starts: see **Calendar days later** | **Workdays later** below. See also *Processing of Time Frame Definitions*. For an active job, the **Deadline** fields contain the time calculated for the respective job run according to the definitions made in the corresponding job master. Both fields can be modified for the given job run only.
Input fields for number of days and day type: **Calendar days later** | **Workdays later**	(Applies to job masters only.) You can use the input fields next to **Earliest Start Time**, **Latest Start Time** and **Deadline Time** to define a time frame that exceeds 24 hours. Enter the number of days (maximum is 999) you want to postpone a start time or deadline. Use the drop-down list boxes to select the day type to be used for the specified start time or deadline: a calendar day or a workday as defined in the calendar linked to the schedule. See also *Processing of Time Frame Definitions*. **Note:** The Entire Operation default settings provide the option to define later start times in units of hours instead of integer days, for time ranges that are not a multiple of 24.
Schedule Dependency	A code shown in this read-only field indicates that this job has a schedule dependency. The field is empty if no schedule dependency is defined for the job. You can choose **Modify** to open a window where you can view, add or modify a schedule dependency. See also *Maintaining Schedule Dependencies for a Job*.
Cyclic Interval	This field is only available for cyclic jobs. Wait time (in minutes) between two executions of a cyclic job (**special type Cyclic execution**). Default: Monitor wait time.
Late Message	See *Sending a Late Message to One or More Users*.
Account Info	See *Viewing Job/Network Accounting Information* in the section *Job Maintenance*.

Maintaining Schedule Dependencies for a Job

You can define particular days of the week, month or year when a job (or all jobs of a network, see *Adding and Modifying a Time Frame Definition*) executes as a temporary dummy due to schedule condition (see also *Temporary Dummy Jobs*).

You can also define day dependencies for a job input condition. If defined, the job executes without performing an input condition check.

If a day dependency is not defined (default) at the job or network level, or for a job input condition, the job executes normally, that is, exactly as defined in the job master.

The date and time defined for job execution on the **Scheduling Parameters page** is then checked against the current schedule of the current network, or the current calendar. For jobs in a subnetwork, we recommend you to specify that the topmost calling network is checked instead of the current network by using **Check schedule of the topmost calling network** option.

Prerequisite for Concatenated Jobs

If you want to deactivate single or multiple jobs in a chain of dependent jobs, for all jobs in the chain, the option **End-of-Job Action** must be set to **No impact on the job result**. Otherwise, errors can occur during End-of-Job processing.

See also *Excluding a Job from Actual Execution*.

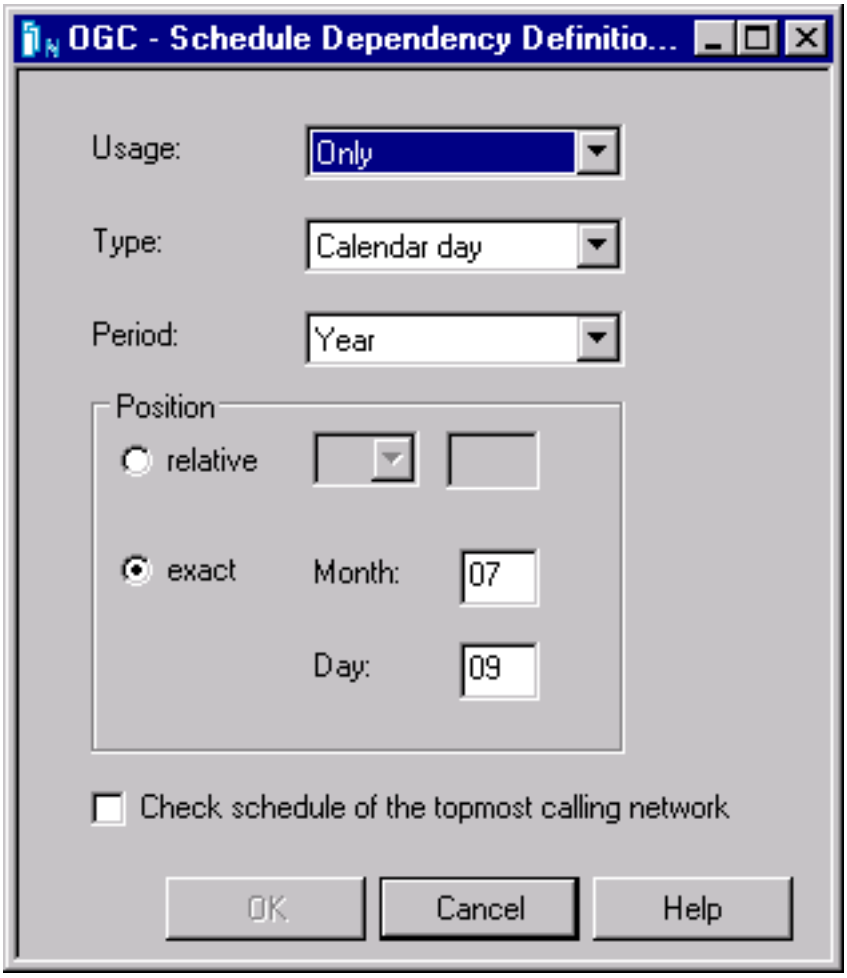


Note: The job execution date and time can be different (one or more days later) from the job activation date and time.

To define schedule dependencies at the job level

- On the tabbed page **Scheduling Parameters** of the **Maintenance Job Master** window, choose **Modify** next to **Schedule dependency**.

  A **Schedule Dependency Definition** window like the example below opens:



The input fields and options in the window are described in ***Fields: Schedule Dependency Definition***.

This section covers the following topics:

- Defining Schedule Dependencies for an Input Condition
- Fields: Schedule Dependency Definition
- Examples of Schedule Dependency Definitions

- Deleting Schedule Dependency Entries

Defining Schedule Dependencies for an Input Condition

You can define an input condition that is not used on all activation dates of a job network. This can be done for the following reasons:

- A predecessor job or network does not run on all schedule dates of the successor job or network.
- Some predecessor jobs are also schedule-dependent.

To define schedule dependencies for an input condition

1 From the table on the **tabbed page Input Conditions**, select an input condition and choose **Schedule Dep.**.

A **Schedule Dependency Definition window** opens.

The fields contained in the window correspond to the fields of the **Schedule Dependency Definition for Job** window. They are explained in *Maintaining Schedule Dependencies for a Job*.

2 Make your definitions.

3 Choose **OK**.

The schedule dependency is allocated to the input condition and the window closes.

The schedule dependency is now indicated in the **Sched.Dep.** column.

Fields: Schedule Dependency Definition

The input fields of the **Schedule Dependency Definition** window provided for a job, a job input condition and a network are described in the following table:

Field	Description	
Check schedule of the topmost calling network	Possible check box settings:	
	unchecked	Checks the schedule of the current network (default). This setting is recommended for jobs in main networks.
	checked	Checks the schedule of the topmost network that calls the job. This setting is recommended for jobs in subnetworks.
Usage	Possible selection options:	

Field	Description	
	Only	For a job or condition: The job or condition is activated only if the execution date satisfies the criteria entered in the **Type**, **Period** and **Position** fields.
		For a network: The object is valid only for schedule days matching the following definitions.
	After holiday too	Like **Only**, but the first workday after an otherwise matching holiday is valid, too.
	Before holiday too	Like **Only**, but the last workday before an otherwise matching holiday is valid, too.
	Except	For a job or condition: The job or condition is activated except when the execution date satisfies the criteria entered in the **Type**, **Period** and **Position** fields.
		For a network: The object is valid for all schedule days except if they match the following definitions.
Type	Possible selection options:	
	Is holiday	Any holiday (no period and/or position required). The relative position to the holiday can be specified.
	Is workday	Any workday (no period and/or position required). The relative position to the workday can be specified.
	Calendar day	Calendar day within the defined period. A relative position refers to the real calendar period.
	Workday	Workday within the defined period. A relative position refers to the workdays in the period.
	Schedule day	Schedule day within the defined period. A relative position refers to the schedule days in the period.
Period	Possible selection options:	
	Week	Current week.
	Month	Current month.
	Year	Current year, with relative or exact position.

Field	Description	
	Position	Schedule dependency position. The position within the defined period depending on the specified **type**. **Relative** position: ■ Positive values (+) are relative to the period begin, negative values (-) are relative to the period end. For the types **Is holiday** and **Is workday**, this is the day difference. Valid value ranges are: For a week: -7 to -1 and +1 to +7 For a month: -31 to -1 and +1 to +31 For a year: -366 to -1 and +1 to +366 **Note:** You can specify up to three days for the period week. Example: 24 is Tuesday and Thursday. **Exact** position: ■ Fixed month (1 - 12) and day (1 - 31) of the year. Only applies if **Calendar Day** is selected for **Type** and **Year** for **Period**. See also *Examples of Schedule Dependency Definitions*.

Examples of Schedule Dependency Definitions

The following are example combinations for day calculation:

Type	Period	Position or Month/Day	Calculated Day
Calendar day	Week	+2	Tuesday
		-1	Sunday
	Month	+1	01
		-1	28 ... 31
	Year	+1	January 01
		-3	December 29
		Month: 4, Day: 9	April 09 (yearly, including leap years)
Workday	Month	-1	Last workday of the month
		234	Second, third and fourth workday of the month
Is holiday	n/a	+1	Day after a holiday

Deleting Schedule Dependency Entries

To delete a schedule dependency entry for a job

1 On the tabbed page **Scheduling Parameters**, choose **Modify** next to the entry in the **Schedule dependency** field.

A **Schedule Dependency Definition** window opens.

2 Clear the **Usage** field by selecting a blank option from the drop-down list box.

All entries are removed from the window.

3 Choose **OK**.

The **Schedule Dependency Definition** window closes and the **Schedule dependency** entry is removed from the **Scheduling Parameters** page.

Sending a Late Message to One or More Users

You can define one or more users (recipients) to receive job-related standard messages that are sent when a job could not be executed in the defined time range.

To inform users that a job could not be executed

1 On the tabbed page **Scheduling Parameters**, choose **Late Message**.

A **Maintenance Message and Message Recipients window** opens.

2 Change, add or remove a message recipient as described under **Recipients** in Step 3 of *To view or change a notification message* in the section *Defining Notification Messages* (*Job Maintenance*).



Note: You cannot change the text of job-related standard messages.

36 Passing Files to Entire Output Management

- Basic Requirements for File Transfers to Entire Output Management 420
- Listing Files Defined for Entire Output Management 421
- Adding and Modifying File Definitions 423
- Deleting File Definitions 427
- Handing Over SYSOUT and Files to Entire Output Management 428

You can define and pass job SYSOUT and/or one or more other files (for example, output files of a job) to Entire Output Management (NOM) to further process the output in this environment.

Basic Requirements for File Transfers to Entire Output Management

This section describes the requirements for transferring files from Entire Operations to Entire Output Management.

1. An `NTLFILE` or `LFILE` definition for the Entire Output Management system file must be entered in the start parameters for System Automation Tools or Entire Operations. For further information, see the section *Installing Entire Operations on Mainframes and UNIX* in the *Installation and Setup* documentation.
2. If Entire Output Management (`LFILE 206`) is defined, and if the Entire Operations Monitor receives a NAT0082 message calling the Entire Output Management API, the message `NOM Trigger interface not active` will be written to log before Entire Operations task restart. Subsequent attempts to pass SYSOUT to Entire Output Management will be ignored with the message `Passing to NOM impossible`.
3. If SYSOUT or another file cannot be passed to the Entire Output Management API, the Entire Output Management API calls for a job to be **retried**. Check the Entire Operations log for more detailed error information.
4. In Entire Output Management, the report name is EOR-SOUT.
5. For UNIX and Windows, refer to ***Passing SYSOUT from UNIX and Windows Nodes to Entire Output Management***.

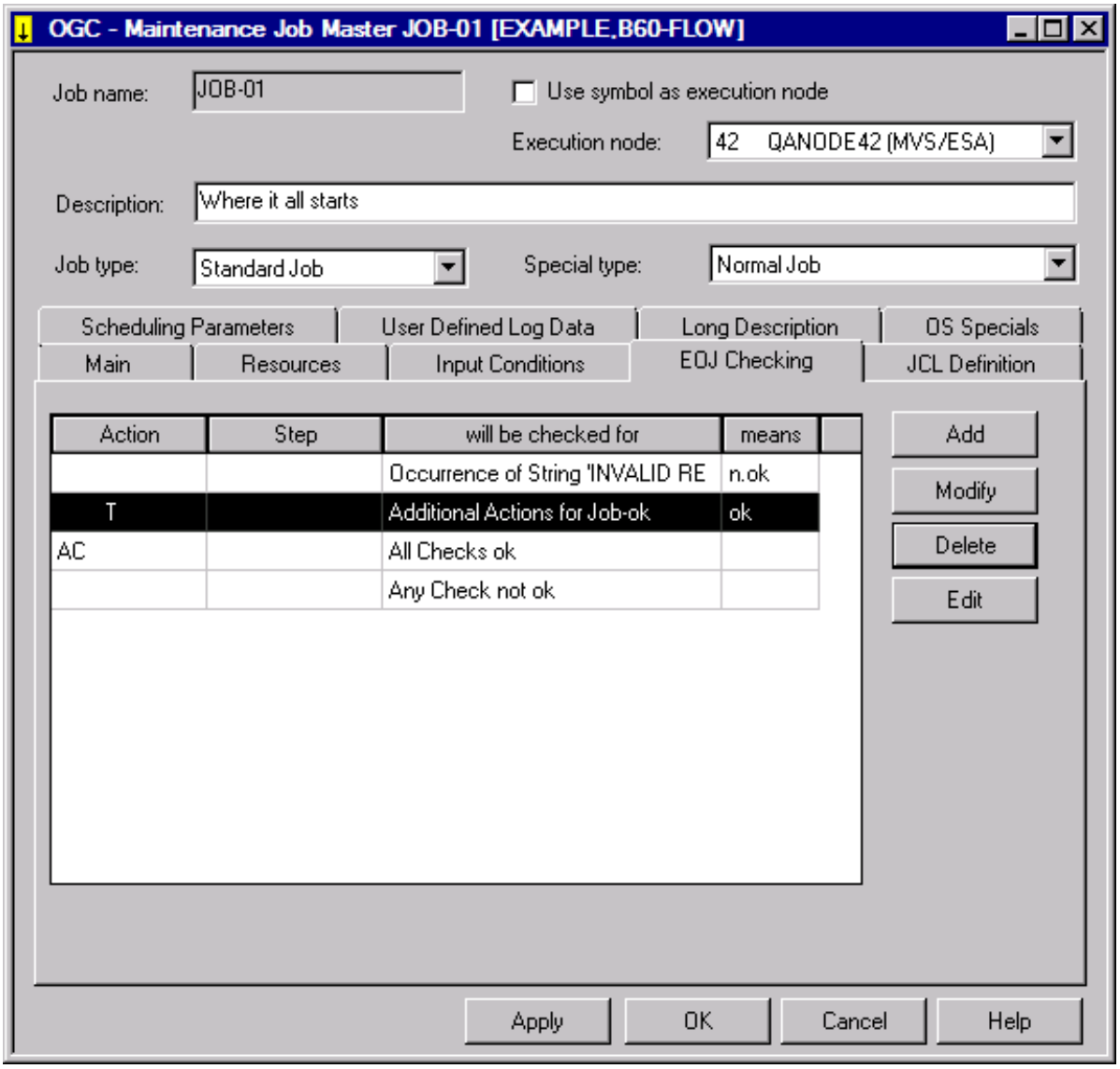
See also:

- ***Handing over SYSOUT and Files to Entire Output Management***.
- The Entire Operations default setting **NOM API Settings** described in *Defaults for Other Settings* in the *Administration* documentation.
- The Entire Operations default setting **Copy SYSOUT File before passing it to NOM** described in *Defaults for Other Settings* in the *Administration* documentation.

Listing Files Defined for Entire Output Management

To list files defined for processing with Entire Output Management

1 Select a job and open the **EOJ Checking** page shown in the following example:



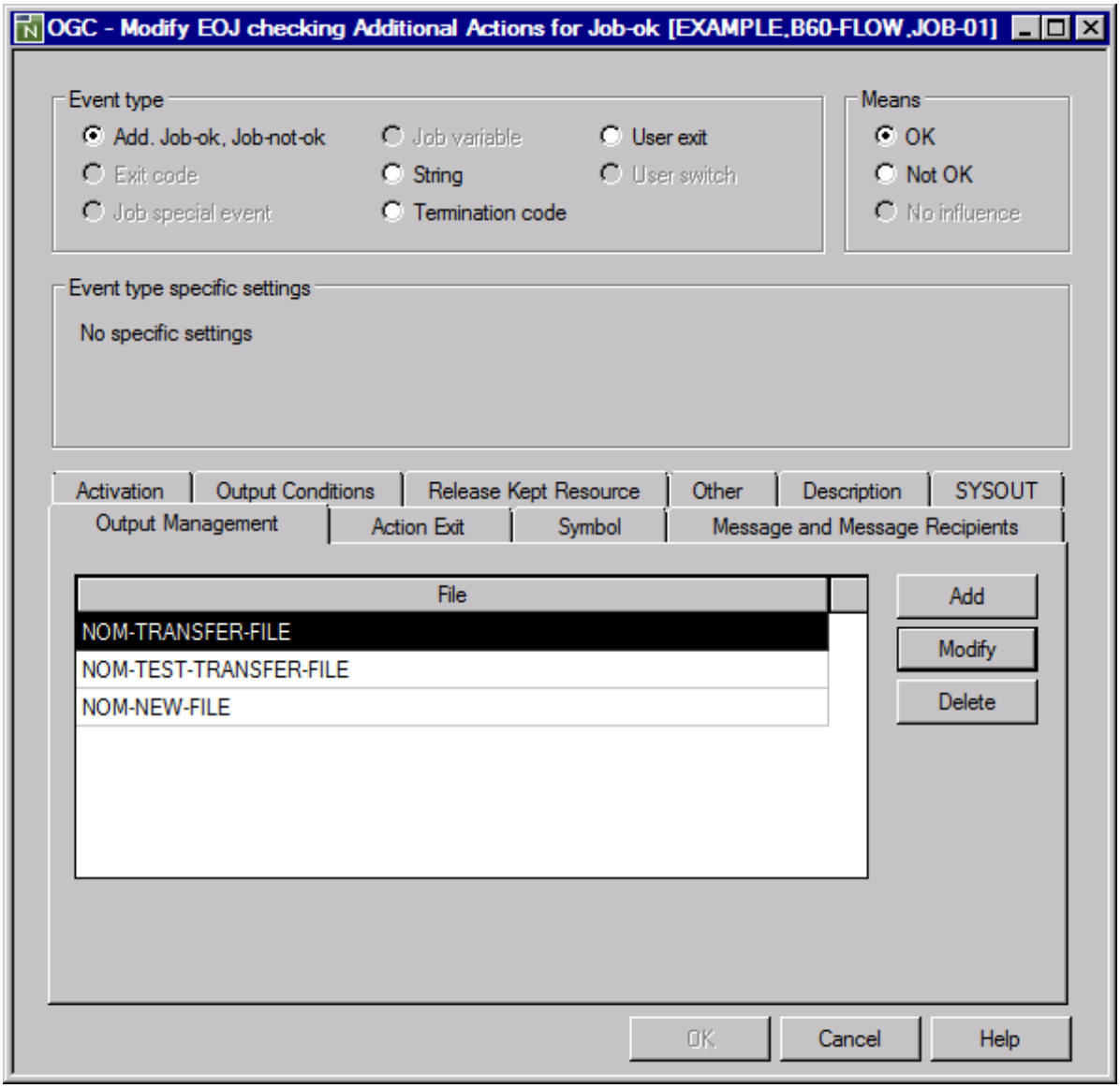
The letter **T** in the **Action** column indicates whether an EOJ (End-of-Job) action for Entire Output Management exists.

The **columns** and **functions** available on the page are described in the section *Defining and Managing End-of-Job (EOJ) Checking and Actions*.

2 Select the table row that contains the letter **T** and choose **Modify**.

An **EOJ checking** window like the example below opens.

3 Open the tabbed page **Output Management** shown in the following example:



The page contains a list of all files defined as EOJ action for transfer to Entire Output Management.

Adding and Modifying File Definitions

This section provides instructions for adding and modifying file definitions for processing in Entire Output Management.

Note: The number of files you can define per job event is limited to 10. If you want to specify more than 10 files for a job, define one or more additional events of the type **Add. Job-ok, Job-not-ok**, and define the extra files for these events.

To add or modify a file definition

1 In a **Maintenance Job Master** window, open the tabbed page **EOJ Checking**.

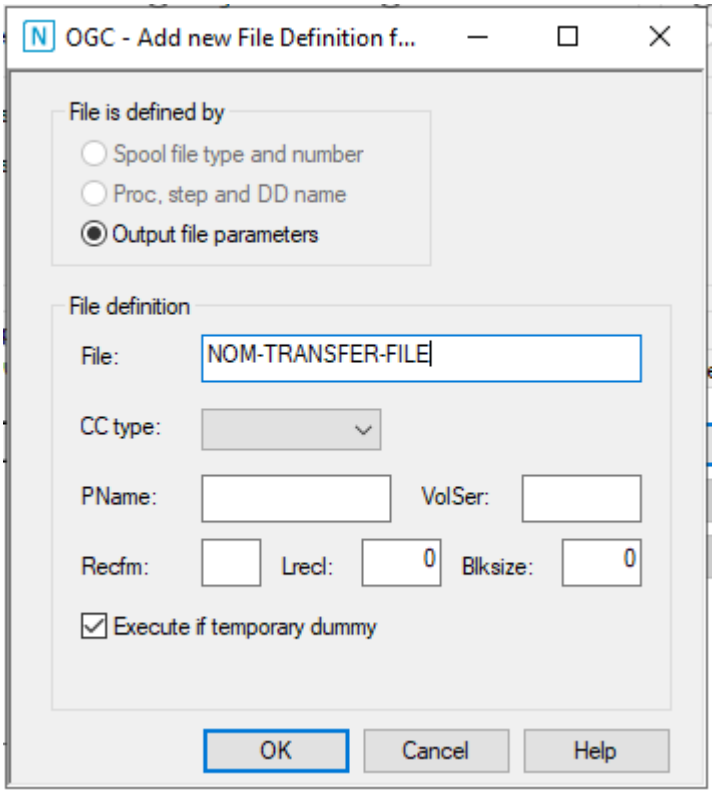
2 Choose **Add** for a new EOJ (End-of-Job) action, or select an EOJ action defined for Entire Output Management (indicated by the letter **T** in the **Action** column) from the window table and choose **Modify**.

An **EOJ checking window** opens.

3 Open the **Output Management page**.

4 Choose **Add** for a new file definition, or select an existing definition and choose **Modify** for a file definition listed in the window.

A **File Definition for Output Management** window like the example below opens:



The fields available in the window depend on the option selected in the **File is defined by** section:

- For **Spool file type and number** and **Proc. step and DD name**, the fields are described in *Fields: File Definition*.
- For **Output file parameters**, the fields are described in *Fields: Output File Parameter Definitions*.

5 Make your definitions and choose **OK**.

The new file is assigned to Entire Output Management.

- Fields: File Definition

- Fields: Output File Parameter Definitions

Fields: File Definition

The fields required to define a file of the job SYSOUT to be handled are provided using the **Spool file type and number** and **Proc. step and DD name** options of the **File Definition for Output Management** window:

Field | Option	Description
Spool file type and number:	
Spool file type	Spool file type, for example, SO for JES SYSOUT.
Spool file number	Spool file number.
Proc. step and DD name:	
Procname	PROCNAME that identifies the file.
Stepname	STEPNAME that identifies the file.
DD name	DD name that identifies the file.

For more information on spool file types and report identification, see *Attributes of a Report* in the section *Reports* of the Entire Output Management *User's Guide*.

Fields: Output File Parameter Definitions

The fields available to define an output file of the job to be handled are described in the following table.

The fields are provided if you select **Output file parameters** in the **File Definition for Output Management** window.

Field	Description
File	File to be passed to Entire Output Management. The definition may contain up to 54 bytes. If the file name contains the activation escape character, a symbol replacement is performed, during the job's activation. File names with a maximum resulting size of 54 bytes can be handled in this way only. If the file name contains the submission escape character, a symbol replacement is performed immediately before the file will be passed. File names with a maximum resulting size greater than 54 bytes must be handled in this way. The maximum resulting size is 250 bytes in this case. By using a wildcard in the file name, you may pass all files to Entire Output Management at once, which are matching the wildcard.

Field	Description
CC Type	Not applicable to z/OS. Carriage control type. If the file contains carriage control characters, you must specify their type: see *Carriage Control Type (CC Type)*.
PName	Applies to BS2000 only. If you want to identify the job by the PNAME option of the BS2000 print command (`/Print ...,PNAME=ADAREP`), enter the PNAME here.
VolSer	Information required to allocate the data set/file: Volume serial number (**Blksize**), record format (**Recfm**), record length (**Lrecl**) and the block size (**Blksize**) of the file/data set.
Recfm	
Lrecl	
Blksize	
Execute if temporary Dummy	If this check box is selected (this is the default), the End-of-Job action is performed even if the job executed as a temporary dummy. If this check box is not selected, the action is not performed.

Notes:

1. Entire System Server/UNIX performs wildcard file copying by invocation of a cp (UNIX) or copy (Windows) command, instead of performing the copy by own code.
2. All file copying is performed synchronously.
3. Make sure not to copy too many files at once, due to the limitation of the EntireX Broker timeout, etc.

Normally Entire Output Management identifies report definitions by spool attributes. However, if Entire Operations triggers processing of output files, no such spool attributes exist. Instead, Entire Operations attributes are passed to Entire Output Management as follows:

Entire Operations Attribute	Entire Output Management Identification	
	z/OS	**BS2000**
JOBNAME	Job name	PNAME
USER ID	Destination	User ID

Carriage Control Type (CC Type)

You must specify a carriage control (CC) type for the output file depending on your operating system and the catalog entries contained in the file catalog.

No specification is required in z/OS, where the carriage control type is taken from the file's catalog entry.

Operating System	CC Type Setting	Explanation
BS2000	*blank*	If the file was created with `RECFORM=(...,M)` or `=(...,A)`, CC type can be taken from the file's catalog entry; otherwise, it must be specified in the **CC Type** field.
	ASA code	ASA code
	Machine code	Machine code
	EBCDIC	BS2000 EBCDIC carriage control characters
z/OS	*blank*	CC type not required

Deleting File Definitions

You can delete single or multiple file definition for Entire Output Management by either deleting the EOJ (End-of-Job) action that corresponds to the file definition(s), or by deleting a single file defined for the EOJ action.

To delete an EOJ action

1 From a **Maintenance Job Master** window, open the tabbed page **EOJ Checking**.

2 Select an EOJ action defined for Entire Output Management (indicated by a **T** in the **Action** column) and choose **Delete**.

A confirmation window opens.

3 Choose **Yes** to confirm the deletion.

The EOJ action with all file definitions for Entire Output Management is deleted.

To delete single files defined for an EOJ action

1 From a **Maintenance Job Master** window, open the tabbed page **EOJ Checking**.

2 Select an EOJ action defined for Entire Output Management (indicated by a **T** in the **Action** column) and choose **Add** or **Modify**.

An **EOJ checking** window opens.

3 Open the tabbed page **Output Management**.

4 Select the file you want to delete and choose **Delete**.

A confirmation window opens.

5 Choose **Yes** to confirm the deletion.

The selected file is removed from the tabbed page **Output Management**.

Handing Over SYSOUT and Files to Entire Output Management

This section covers the following topics:

- Repetition if File is missing or open
- Copying Files before passing them to Entire Output Management
- File Copy to another Entire System Server Node
- Connection with Defined Recovery Action
- SYSOUT Line Limit
- Passing SYSOUT from UNIX and Windows Nodes to Entire Output Management
- Naming Rules for Files copied into the EOR_NOM Directory (UNIX and Windows)

Repetition if File is missing or open

If the file to be handed over did not (yet) exist or was (not) yet closed, the handover will be retried.

The following rules apply:

Case	Retries	Interval between retries
Output file cannot be found (does not apply to **spool files**)	3	5 minutes
All other cases	1000	30 minutes

The retry queue will not be deleted in case of a Monitor shutdown and restart.

Copying Files before passing them to Entire Output Management

In the default settings, you can define that the SYSOUT files are copied physically before handover to Entire Output Management.

The advantage of this is that exclusive use (as well as renaming and deleting) of the copy by Entire Output Management can still be looked at in Entire Operations. This, however, means that SYSOUT files need more space.

File Copy to another Entire System Server Node

Under the following conditions, the SYSOUT copy is created on another node:

- The symbol `SYSOUT-NODE-GLOBAL` exists in the symbol table SYSDBA/A and contains a valid node number.
- In the target environment, the same PubIDs and BS2000 user IDs exist as in the source environment.

Connection with Defined Recovery Action

Defined recovery actions for an erroneous job are only executed, after all activities for the handover of the SYSOUT to Entire Output Management have been concluded, or after the maximum number of handover tries has been reached.

In case of a recovery action, the SYSOUT of all job runs are handed over to Entire Output Management. For further information, see ***Defining Recovery Actions*** in the section *Defining and Managing End-of-Job (EOJ) Checking and Actions*.

SYSOUT Line Limit

If the SYSOUT file or spool data set exceeds a given line limit, the copied file will be truncated: see **SYSOUT line limit** in *Defaults for Network Options* in the *Administration* documentation.

Passing SYSOUT from UNIX and Windows Nodes to Entire Output Management

On UNIX and Windows, the environment variable `EOR_NOM` must be defined. It must contain a valid directory name. Entire Operations copies the SYSOUT files for Entire Output Management into this directory. If the directory does not exist, Entire System Server/UNIX (Entire System Server/Windows) creates it at the first usage attempt.

To define the copying of SYSOUT to Entire Output Management

- From the **EOJ Checking** page, open the tabbed page **SYSOUT** and **Pass to EOM** from the **SYSOUT action** list box. See also ***Defining SYSOUT Actions*** in the section *Defining and Managing End-of-Job (EOJ) Checking and Actions*.

This is a job-level definition that only applies to the selected job.

Entire Operations creates the SYSOUT file copies in this directory with unique names.

It is up to Entire Output Management to fetch the files from there.

Notes:

1. If copied SYSOUT files are not fetched by Entire Output Management within the retention period for active jobs, they are removed by Entire Operations cleanup after that time.
2. If the SYSOUT file or spool data set exceeds a given line limit, the copied file will be truncated: see **SYSOUT line limit** in *Defaults for Network Options* in the *Administration* documentation.

Naming Rules for Files copied into the EOR_NOM Directory (UNIX and Windows)

SYSOUT Files

The file names for SYSOUT files for the $EOR_NOM directory have the following syntax:

```
dbid-fnr.owner.network.run.job.txt
```

File Name	Meaning
dbid-fnr	Concatenated DBID and FNR, 10 digits.
owner	Owner of job.
network	Network of job.
run	Run number of job.
job	Job.

Other Files

The file names for other files for the $EOR_NOM directory have the following syntax:

Format A with the components:

```
yymmdd.hhiisst.rrrrr.filename.ext
```

File Name	Meaning
yymmdd	Year, month, day.
hhiisst	Time, incl. 1/10 sec.
rrrr	Entire Operations run number, with leading zeros.
filename	Unqualified part of original file name.
ext	Extension of original file name.

Format B with the components:

```
yymmdd.hhiisst.rrrrr.job.ix
```

File Name	Meaning
yymmdd	Year, month, day.
hhiisst	Time, incl. 1/10 sec.
rrrr	Entire Operations run number, with leading zeros.
job	Entire Operations job name.
ix	Index of file definition in Entire Operations (2 digits).

Note: Format B is used only if the length of *filename.ext* is longer than 20 characters, because of a size limitation in Entire Output Management.

37 Viewing Job/Network Accounting Information

- Displaying Accounting Data 434
- Fields: Range Specification for Accounting Data 436

You can view the start and end times, elapsed time and CPU time for previous runs of a job. The information is taken from the Entire Operations log file.

Note: You can also generate reports from accounting data (see *Example of Accounting Information* in the section *Reporting*).

Displaying Accounting Data

To view accounting data

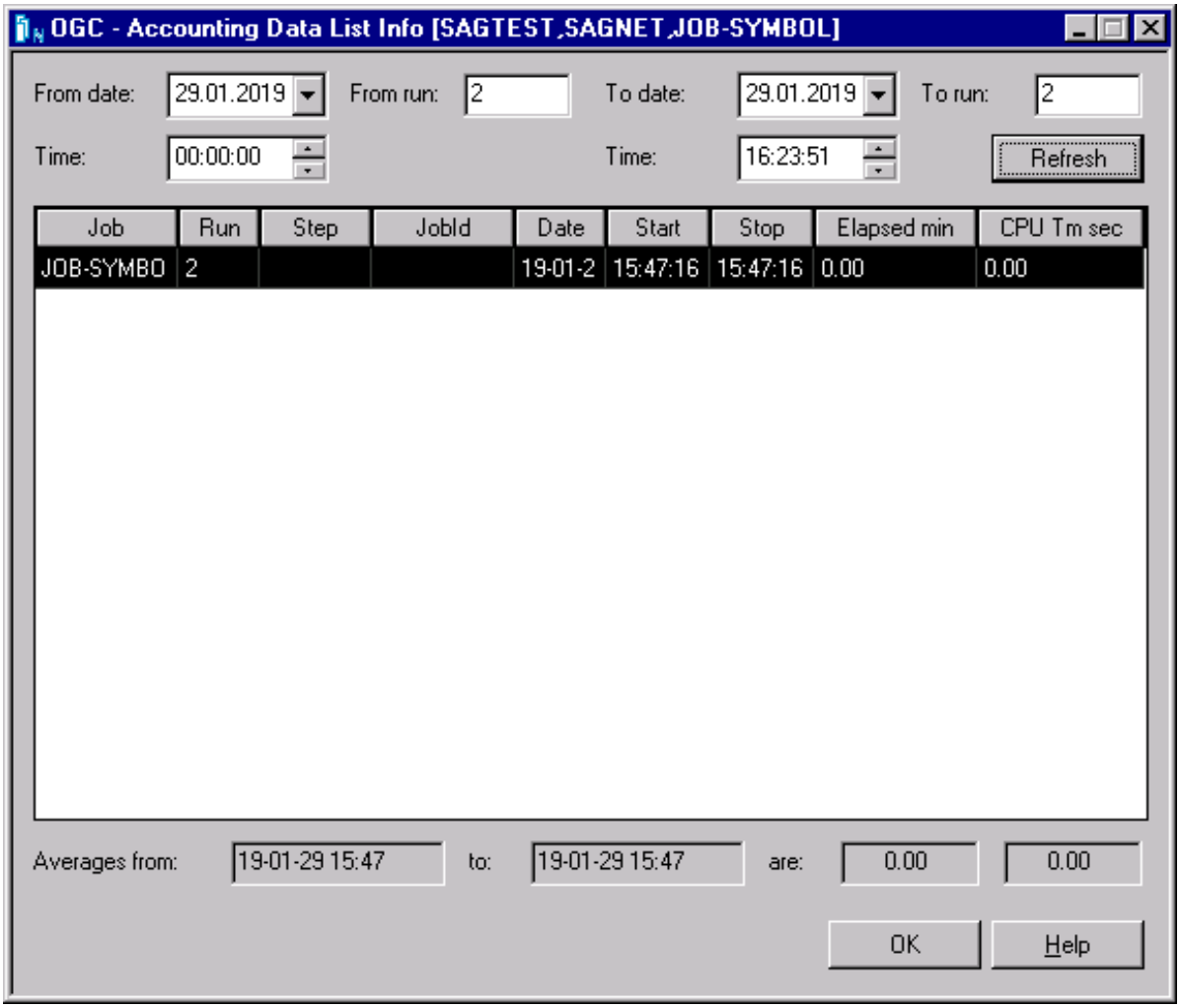
1 In a **Maintenance Job Master** or **Maintenance Job Active** window, open the tabbed page **Scheduling Parameters** and choose **Account Info**.

An **Accounting Data List Info** window like the following example opens.

2 If required, changes the field entries to specify the period and number range for which you want to view accounting data: see *Fields: Range Specification for Accounting Data*.

3 Choose **Refresh**.

The **Accounting Data List Info** window is refreshed and the accounting information is displayed as shown below:



The window contains job execution time information collected by the Entire Operations Monitor. The job elapsed time is in minutes and the CPU time is in seconds. Average values are available at the bottom of the window. For further information, see ***Accounting Information/Schedule of Jobs reports only*** under ***Fields and Columns: Reports*** in the section *Reporting*.

4 Choose **OK**.

Note: Step data is only displayed if the **Collect z/OS step accounting data** option is enabled in the Entire Operations default settings: see *Defaults for Operating System Specials* in the *Administration* documentation.

Fields: Range Specification for Accounting Data

The fields of the **Accounting Data List Info** window are described in the following table:

Field	Description
From Date/Time **To Date/Time**	Enter or select start (**From**) and end (**To**) dates and times to specify the time frame for which to display accounting data. Default is the current date, from midnight until the current time.
From Run/To Run	Enter or select start (**From**) and end (**To**) run numbers to specify the number range for which to display accounting data. Default is from 1 to 99999.
Averages from ... to ... are	Read-only field at the bottom of the window listing average consumption values which correspond to the values in the field **History Elapsed Time (min.)**.

VII Active Job Network and Active Job Maintenance

This section contains general information on schedule-controlled or manual action of job networks and jobs, deletion of work files, cleaning the active database and describes the functions available for activating job-networks and jobs and for performing ad-hoc actions on active jobs in the active database.

General

Meaning and Use of Active Job Networks and Active Jobs in Entire Operations
Activation of Networks or Jobs
Prerequisite Check before Job Submission
Messages in Active Jobs Lists

Active Job Networks

Maintaining Active Job Networks

- **Listing Active Job Networks**

Activating a Job Network Manually (in the section *Network Maintenance*)
Deactivating Active Job Networks
Viewing Long Descriptions of Active Jobs
Viewing Job/Network Accounting Information (in the section *Job Maintenance*)
Viewing the Execution History of an Active Network

Active Jobs

Maintaining Active Jobs
Listing Active Jobs
Adding a New Job to the Active Network
Displaying and Modifying an Active Job Definition
Deactivating a Job in an Active Network
Listing Active Conditions

Defining and Managing End-of-Job (EOJ) Checking and Actions
Viewing and Modifying Resources Used by Active Jobs
Viewing Job/Network Accounting Information (in the section *Job Maintenance*)
Cancelling, Holding and Releasing Active Jobs
Browsing Active Job SYSOUT
Resubmitting Active Jobs
Listing Jobs of an Active Subnetwork
Reactivating an Active Job
Viewing Long Descriptions of Active Jobs
Displaying Prerequisites for Active Jobs

Active Job Conditions

Maintaining Active Job Conditions

Active JCL

Maintaining Active JCL (Job Control Language)
Browsing Active JCL
Editing Active JCL
Release Edit Lock (for administrators only)
Exchanging Active JCL
Regenerating Active JCL

38 Meaning and Use of Active Job Networks and Active Jobs in Entire Operations

When Entire Operations activates a job or job network, the job or job network defined in the master database is copied to the active database and assigned a run number. Several copies of the same job or job network can be in the active database, each distinguished by their run numbers. All copies and run numbers are listed under the **Active Run** node of a **Job Master** or **Network Active** node.

Entire Operations allows extensive modification of job networks and jobs after activation. All current information pertaining to condition status, job status, active JCL and symbols is contained in and can be modified on the active database.

The following sections explain the maintenance functions you can perform on active jobs and networks.

For example, you can modify active copies of a job definition including associated logical conditions, resources and JCL. All modifications to active jobs take place on the active database and are valid for the current job run only, so that all original definitions on the master database remain unchanged.



Note: Your user profile must contain the appropriate authorization to access active networks. See the **Monitoring Functions** permissions your administrator can set to control access.

39 Activation of Networks or Jobs

- Terminology 442
- Manual Activation 443
- Automatic (Scheduled) Activation 444
- Start of Job Activation 445
- Run Number 445
- Cleanup of the Active Database 446
- Cleanup of the Active Database in Batch Mode 447

Activating a job network or job means preparing it for execution. On activation, the following is performed:

- The definitions of jobs, networks, logical conditions, symbol tables etc. are copied to the Entire Operations active database and assigned a unique **run number**;
- If necessary, symbol prompting is requested (see also ***Symbol Prompting during Network or Job Activation***). However, symbol prompting is not performed for any subnetworks.
- The global activation exit user exit is invoked, if defined in the Entire Operations defaults;
- The JCL defined for jobs within the network is copied to the active JCL storage on the active database;
- Variables (symbols) used in dynamically generated JCL are substituted by their current values. This does not apply to variables defined to be substituted at job submission time;
- The JCL definitions of active job networks, respectively of active jobs, can differ from the JCL definitions in the master definition. To allow this, the corresponding symbol tables must contain certain reserved symbols on activation. See also ***Predefined Symbols*** in the section ***Symbol Table and Symbol Maintenance***.
- If you use pre-generated JCL, symbol replacement is performed at the time of JCL generation.
- The Entire Operations Monitor recognizes the job network as active and checks time frames, input conditions and resources defined for the jobs. If all prerequisites for any jobs are fulfilled, these jobs are submitted.

Related Topics:

- ***Activating a Job Network Manually***
- ***Activating a Single Job Manually***
- ***Schedule Maintenance***
- ***Symbol Prompting during Network or Job Activation*** - ***Symbol Table and Symbol Maintenance***
- *Master Database and Active Database - Concepts and Facilities* documentation

Terminology

In this documentation and on the user interface, the terms *activation* and *network start/job start* are used.

- **Activation**
  denotes the process of creating an active copy of a network or job definition.

- **Network start/job start**
  denotes the actual execution start time of the activated/active job network or job.

Manual Activation

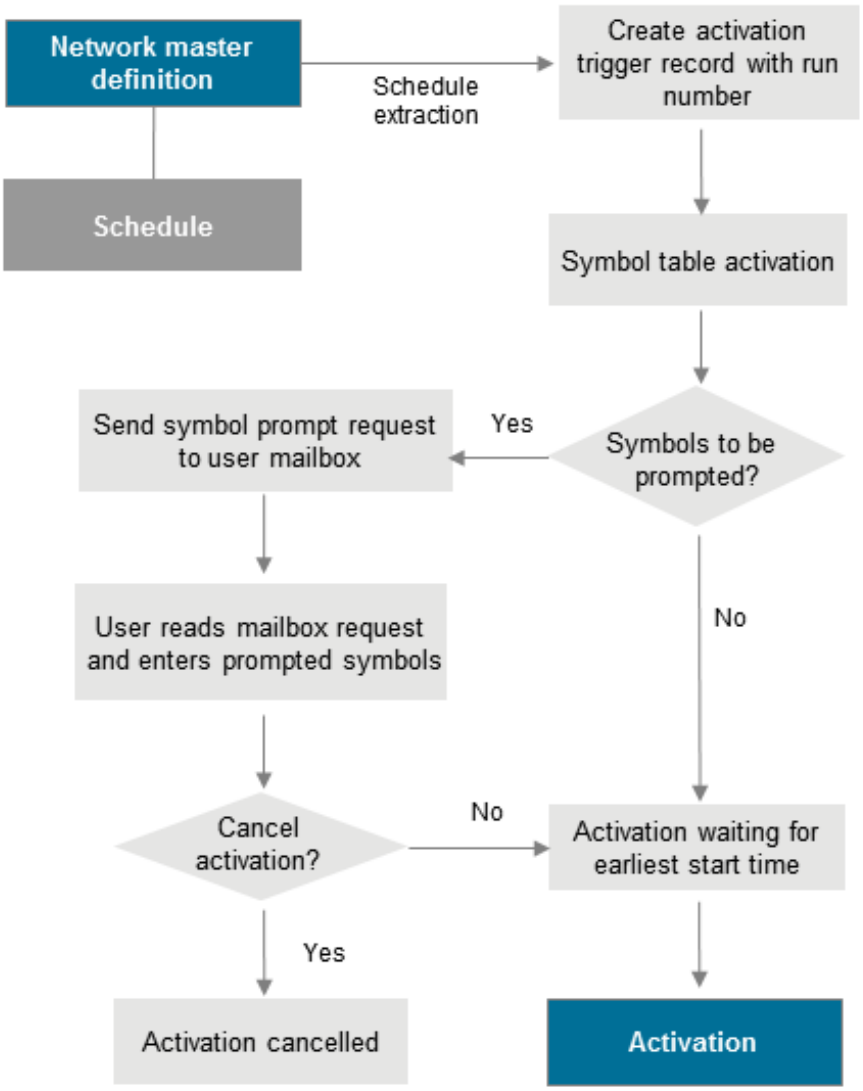
It is also possible to activate a job network manually irrespective of any defined schedule. This may become necessary for a number of reasons, for example:

- No schedule has been defined for the job network;
- To override defined activation date and time;
- The job network is not scheduled for the required date.

For details, see *Activating a Job Network Manually*.

Job network or job activation can also be triggered by any event within Entire Operations, for example by the termination of another job network or by the Entire Operations Application Programming Interface (API); see also *API Routines*. Like manual activation, this can be performed at any time.

Symbol prompting for active symbols is also performed, when a job or network is activated manually, if at least one symbol of a symbol table used by the job or network is appropriately marked.



Automatic (Scheduled) Activation

Job networks are activated automatically in two steps:

- At the beginning of a new day or during Monitor startup, all **schedules** are checked for job networks to be executed during that day. This process is called schedule extraction and the data extracted are called the activation trigger records.
- The activation trigger records force job network activation a short time before the earliest start of the network. This time span can be defined in the Entire Operations defaults: see the **Extraction**

of schedules before activation option described in *Defaults for Time Ranges* in the *Administration* documentation.

Notes:

1. If no earliest start time is defined on the network level, the network is activated immediately after schedule extraction.
2. The modification of a calendar or schedule always triggers a schedule extraction for the dependent job networks. For this reason, a job network could be activated even for the current day after such a modification.

Automatic Activation - Symbol Prompting

After the creation of an activation trigger record, active symbol tables are created for the specific network run. If there is at least one symbol marked as to be prompted within these active tables, a **symbol prompting** request is sent to the mailboxes of all users defined as message recipients for that network.

The network activation is kept in hold, until any user sees the request and enters or confirms the symbols to be prompted. For this reason, schedule extraction can be performed several days in advance. (See *Global Schedule Extraction* in the section *Special Functions* in the *Administration* documentation.)

Start of Job Activation

1. If the calculated latest start is after the calculated deadline, the last start will be set 1 minute before the deadline.
2. If the (new) latest start is before the earliest start, the job activation will be aborted with an error message.

Run Number

Entire Operations automatically assigns a run number to each active copy of a job network on the active database. This run number uniquely identifies the active copy of a job network and is automatically passed to its jobs, input conditions, etc.

The run number is assigned:

- During the creation of an activation trigger record;
- During a manual activation;

- If a network is activated by an API routine.

Run numbers are in the range 1 to 99999 by default and are unique on network level. When the maximum run number has been reached, assignment again starts from 1.

The upper limit for run numbers can be modified in the Entire Operations defaults as described in *Defaults for Network Options* in the *Administration* documentation.

The assignment of a run number to each activation of a job network allows multiple activations of a job network on the same date, and allows you to distinguish between multiple active copies of the same job network.



Note: There is no guarantee that subsequent activations will have ascending run numbers. They are as unpredictable as operating system job numbers. Entire Operations retains the last run number, even for deleted job networks. If you define a new job network of the same name, the new run numbers start from the deleted network's last run number incremented by 1.

When manually activating a job or a network, you can specify a preferred run number to be used for the next run (see also ***Activating a Job Network Manually*** and ***Activating a Single Job Manually***.

Related Topics:

- ***Maintaining Active Runs*** - *Active Job Networks*

Cleanup of the Active Database

The operative data of Entire Operations must be removed again from the active database after a certain time. Part of this process is the removal of work files as well, which Entire Operations has created in the file system for job control purposes.

- The retention periods for active objects can be defined (see *Administration*) documentation.
- The cleanup may be defined to be carried out automatically every day. If no time is defined for the cleanup, then it will be started at 00:00. A time for the daily cleanup start can be defined. For a more detailed description, see *Administration* documentation.
- The cleanup of the active database can also be started manually any time (see *Cleanup of the Active Database* in the *Administration* documentation).
- Furthermore, it is possible to run the cleanup of the active database in a Natural batch job (see ***Cleanup of the Active Database in Batch Mode***) exterior to the Entire Operations Monitor. The cleanup in batch mode can be executed with the Monitor running or shut down.

Note that the cleanup of the active database depending upon the data quantity to be processed affects the system. It is recommended to schedule the cleanup for silent times.

Cleanup runs can also be performed several times a day. This makes it possible to reduce the volumes to be processed per run.

Deleting Work Files

Entire Operations creates files in the operating system under BS2000, UNIX and Windows. Among other things, they contain the job SYSOUT or the JCL to be executed.

During the deactivation of active jobs, which have run in one of these operating systems, the assigned work files are deleted as well.

All definitions are created in the Entire Operations Defaults. They are described in the *Administration* documentation.

Cleanup of the Active Database in Batch Mode

- Parameter for CLEAN
- Example of CLEAN
- Cleanup of Mailbox Messages to SYSDBA
- Parameter for MX-DEL1P
- Example of MX-DEL1P

In addition to an automatic cleanup, or alternatively, you can also do a cleanup of the active database in batch mode outside of the Entire Operations Monitor.

Use a Natural batch job with the following commands:

```
LOGON SYSEOR
CLEAN function log-target
FIN
```

Parameter for CLEAN

Parameter	Description	
function	Possible values:	
	ALL	Cleanup, deactivation, deletion of work files.
	CLEAN	Cleanup only.
	DEACT	Deactivation, deletion of work files only.
log-target	Possible values:	
	NOP	The cleanup is logged in the **Entire Operations log**.

Parameter	Description	
	SO	The cleanup is logged in the SYSOUT of the batch job. In addition, the start and end messages of the cleanup are logged in the **Entire Operations log**.
	ALL	The cleanup is both logged in the **Entire Operations log** and the SYSOUT of the batch job.

The batch cleanup can take place with the Monitor running or not. It is recommended to do the cleanup at times of low Monitor utilization.

Like the automatic cleanup the batch cleanup should be performed at least once a day. Several runs a day are possible. This may be useful to reduce the data amount of each cleanup run.

The Entire Operations retention periods also apply to batch cleanup.

Example of CLEAN

```
CLEAN ALL SO
```

Complete cleanup and log into SYSOUT.

Cleanup of Mailbox Messages to SYSDBA

The cleanup of mailbox messages to SYSDBA can be executed in batch mode. To do this, use a Natural batch job including the following commands:

```
LOGON SYSEOR
MX-DEL1P yyyymmdd hhiiss
FIN
```

Parameter for MX-DEL1P

Parameter	Description
yyyymmdd	Day up to which cleanup is to take place.
hhiiss	Time up to which cleanup is to take place.

Example of MX-DEL1P

```
MX-DEL1P 20020201 100000
```

40 Prerequisite Check before Job Submission

- Order of Prerequisite Checking 452
- Passive Wait 453
- Prerequisite Check according to the Round-Robin Procedure 455

Each active job is checked for its prerequisites before it can be submitted. The job can only be started if all defined prerequisites are available at the same time. The prerequisite checking of an active job is repeated until all defined prerequisites are available, but only before its latest start time is reached.

The following prerequisites must be met before a job can start running:

- The start and end times defined for a job or network must be reached.
- The input conditions defined for the job must be fulfilled.
- The resources defined for job usage must be available.
- Operating-system specific objects defined for a job (for example, a BS2000 user switch) must be available.
- The execution node defined for the job or network must be available.

Entire Operations uses several procedures to reduce the effort involved for the prerequisite check. These procedures are transparent to the user. Nevertheless, they are explained in the following section.

For information on actions Entire Operations performs after job termination if specified events occur, see *Events and Actions: End-of-Job Checking* in the *Concepts and Facilities* documentation and ***Defining and Managing End-of-Job (EOJ) Checking and Actions***.

Related Topics:

- *Events and Actions: End-of-Job Checking - Concepts and Facilities* documentation
- ***Defining and Managing End-of-Job (EOJ) Checking and Actions***

Order of Prerequisite Checking

The sort order of prerequisite checking is:

1. Earliest start time;
2. Owner, network, run, job.

The sort is only applied to jobs, which reside within the prerequisite check input queue at the same time.

Passive Wait

Active jobs waiting for one or several input conditions, resources or for the availability of an operating system server (node) are placed into a particular queue, which removes them temporarily from the active check carried out by the Monitor.

Active jobs are woken up (released) from a passive wait state:

- During setup or deletion of active prerequisites at any location;
- During setup or deletion of resources at any location;
- After modification or deletion of definitions for input conditions and resources in active jobs;
- During Monitor start;
- When the date changes;
- By explicit request; see *Special Functions* in the *Administration* documentation.

After wake-up, an active check of the prerequisites, resources and operating system server is performed again. If the prerequisites required for job start are not met, then another passive wait can result out of this.

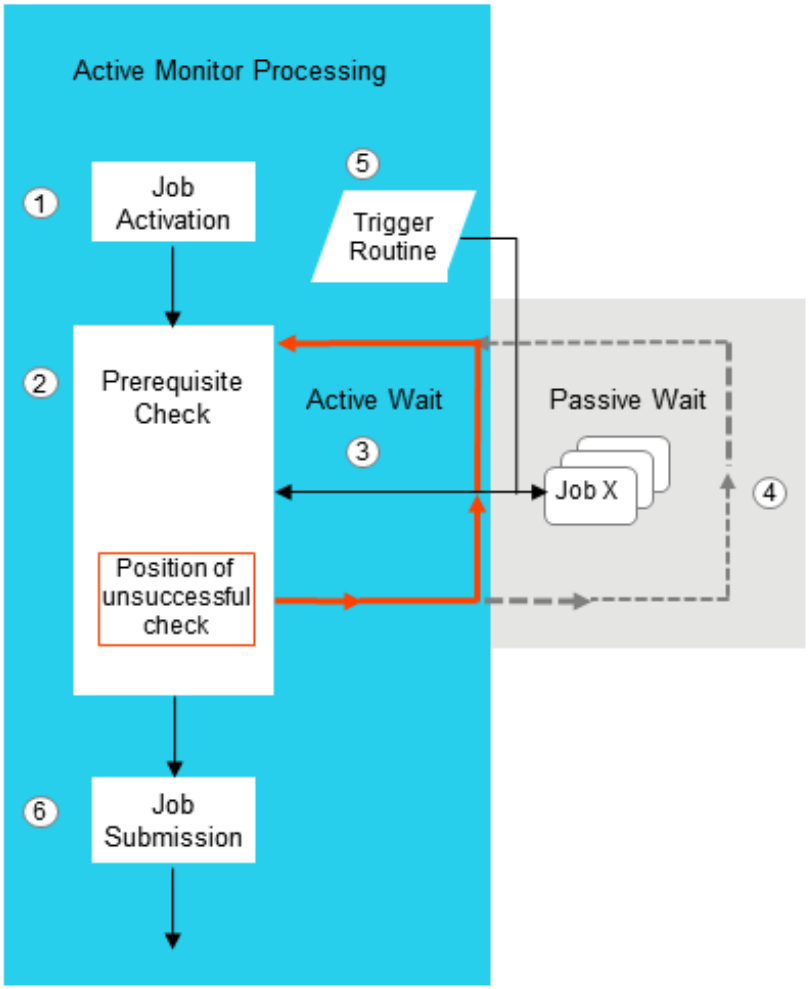


Note: The main passive wait release routine does not reactivate the waiting jobs at the same time. Instead, it performs the release in portions of 300 jobs. Between the portions, there is a wait of 30 seconds. This spreads the Monitor and database activity for the prerequisite check of a large number of jobs over a longer period of time.

- Course during Passive Wait for Prerequisites
- Exceptions from Passive Wait

Course during Passive Wait for Prerequisites

The following graphic shows the course during passive wait for prerequisites:



Legend

1. A network has been activated and job processing is controlled by the Monitor.
2. The prerequisites of a job are checked after job activation.

   If a prerequisite is not met (for example, the execution node defined for the job is not available), the prerequisite check stops at the position where it failed.
3. The job is placed into an active wait state waiting for the next check to meet the required prerequisite.

   The next check continues at the position where the previous check failed.
4. The Monitor determines how long to wait for the missing prerequisites before it places the job into a passive wait state.

5. A trigger routine reactivates the job if the criteria defined to reactivate the job are met (for example, the missing execution node is available now), and forces the job back to active checking.

   The check procedure (from active to passive wait and vice versa) can repeat several times.

6. If all prerequisites are met, the job is submitted for execution.

Note: Each time the Monitor is started, all jobs in the passive wait queue are reactivated for another prerequisite check.

Exceptions from Passive Wait

A job cannot be placed into a passive wait state in the following cases:

- Waiting for an input condition which depends upon the existence of a file;
- Waiting for an input condition which depends upon the result of a user exit.

In these cases, Entire Operations cannot acknowledge on its own when such a job is to be placed again into the active wait. Therefore, in such a case, an active job is not placed into the passive wait.

Nevertheless, at least for part of the wait, a passive wait can also be carried out for these jobs, if, in parallel to the above mentioned cases, they are waiting for a *normal* prerequisite, which is set up as shortly as possible before job submission.

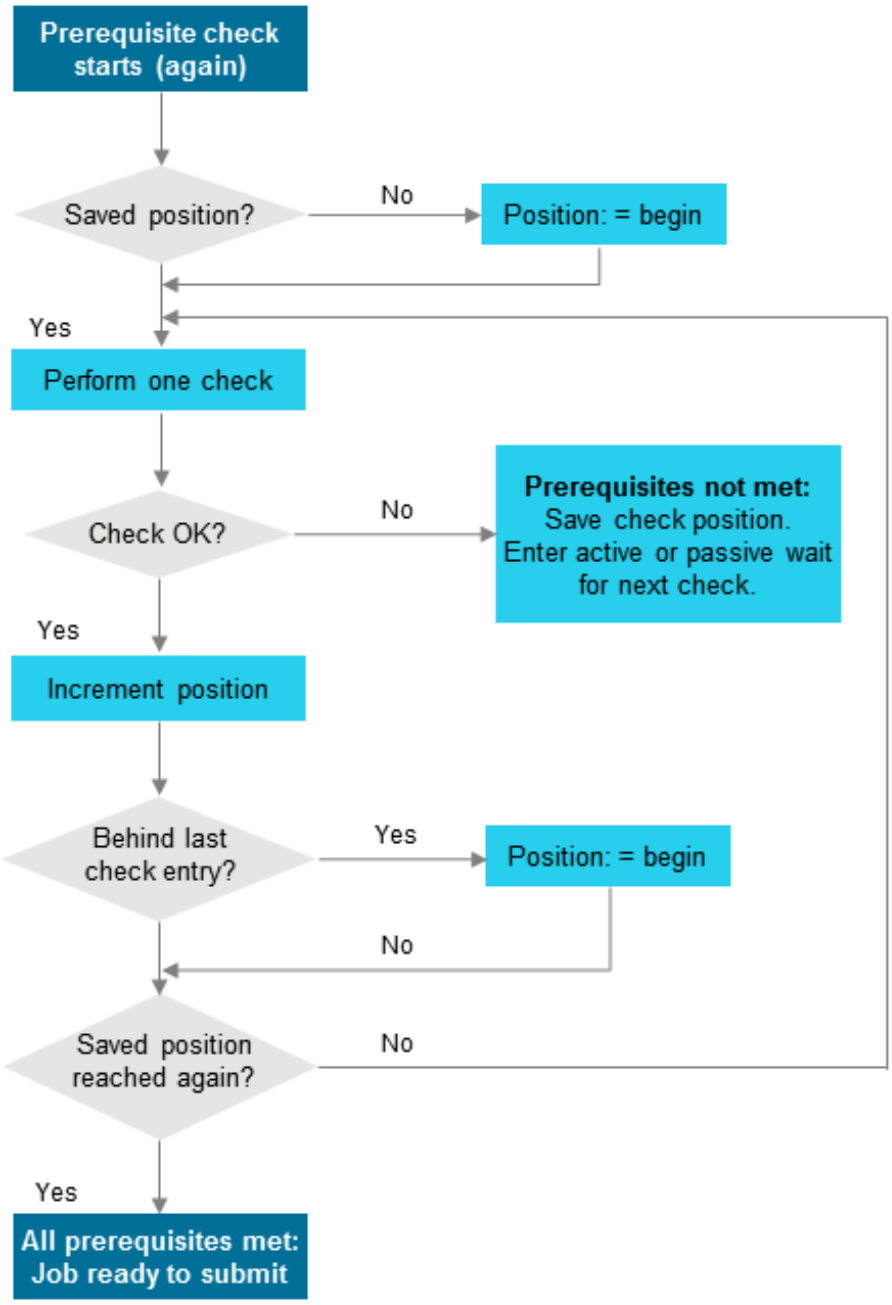
In other words: it is recommended to replace a wait for prerequisites with special dependencies by a wait for normal prerequisites.

Prerequisite Check according to the Round-Robin Procedure

If prerequisites and resources of an active job are actively checked, then the order of the job checks will be optimized dynamically.

For a follow-up check, the last unsuccessful check will be the starting point. This prevents successful checks from being redundantly repeated several times. It is guaranteed, however, that immediately before the job start release all input conditions and resources have been checked together at one point in time.

The following diagram shows the course of the Round-Robin Procedure for the check of prerequisites and resources:


Prerequisite check starts (again)
Saved position?
No
Position: = begin
Yes
Perform one check
Check OK?
No
Prerequisites not met: Save check position. Enter active or passive wait for next check.
Yes
Increment position
Behind last check entry?
Yes
Position: = begin
No
Saved position reached again?
No
Yes
All prerequisites met: Job ready to submit

41 Messages in Active Jobs Lists

This section lists the most important messages which may appear in the **Message** column of the **List Active Jobs** window (see the *User's Guide*).

Most messages speak for themselves, but for some messages, an explanation is provided below.

Job activated

Explanation:	The job has been loaded to the active queue.

Cyclic job executed

Ended not ok - <job step> <condition code>

Explanation:	Job ended not OK. See condition code and job step (z/OS only).

Ended ok

Explanation:	The job terminated OK.

HOLD in Spool Input Queue

Job disappeared from Spool Queue

Awaiting JCL Load

Job failed - JCL Error

Job not run - JCL Error

Job terminated - In Spool Output Queue

Explanation:	The job terminated but has not been examined by the Monitor yet.

Member Name missing

Waiting for submission

Job resubmitted

Explanation:	The job has been resubmitted but has not been recognized by the Monitor yet.

Start Time <date> <time>

Explanation:	The earliest start time for a job, unless its type is cyclic. Job waits.

Next Start Time <date> <time>

Explanation:	The earliest (next) start time for cyclic jobs. Job waits.

Latest Start <date> <time> exceeded

Submitted as job NNNN

Explanation:	The job was submitted successfully.

Job XXXX Waiting in Spool Input Queue

Cannot delete <in-condition>-<run-number>-<in-condition-reference> - in use

Explanation:	The specified in-condition is used by at least one job.

Condition <in-condition> not satisfied

<condition> - <run> - <date> not found

Explanation:	The job is waiting for one or more input conditions to be fulfilled.

User Rtn <user exit> called non-existent program

Recov. Network <network> not found

<Message from User Exit>

Explanation:	A user exit can return a user-defined message text for the active jobs display. The message is displayed only if the user exit returns a `not OK` status.

Job variables not supported for <operating system>

Explanation:	Job variables are only allowed in the BS2000 operating system.

JV <job variable> not matching

User Switch <BS2000 user ID / BS2000 user switch> is on

Explanation:	The BS2000 user switch is on, although it should not be on.

User Switches not supported for <operating system>

Explanation:	User switches are only allowed in the BS2000 operating system.

User Switch <BS2000 user ID / BS2000 user switch> not on

Node NNN undefined

Explanation:	The execution node cannot be found in the node table.

Node NNN not active

Explanation:	The execution node is not active.

Prerequ. Check: Node NNN Error XXXX

Explanation:	An error occurred during node check.

NNN tape units required

Explanation:	There are not enough tape units available.

Resource <resource> undefined

Explanation:	The indicated resource cannot be found within the resource definitions.

Res. <resource> - requ. NNNNNNNNNN.NN avail. NNNNNNNNNN.NN

Explanation:	The requested amount of a specified resource exceeds the available amount of it.

Reference <in-condition-reference> Format Error NNNN

Explanation:	A format error for an input condition reference has been detected.

Prq. File Check - Symbol missing

Explanation:	The symbol to be replaced in the file name cannot be found in the symbol table.

42 Maintaining Active Job Networks

- Available Functions: Network Active 462
- Listing Active Job Networks 462
- Deactivating Active Job Networks 464
- Viewing the Execution History of an Active Network 465
- Maintaining Active Runs 466

Available Functions: Network Active

To list all functions available for a Network Active node

- In the object workspace, select a **Network Active** node and open the context menu.

  The following functions are available:

Function	Shortcut	Description
List	F8	See *Listing Active Job Networks*.
Refresh	F5	See *Refreshing Object Lists*.
Filter	F3	See *Filtering Objects*.
Set Drag And Drop Function	---	See *Drag & Drop*.

To list all functions available for a Network Active instance

- Select a **Network Active** instance and open the context menu.

  The following functions are available:

Function	Shortcut	Description
History	CTRL+C	See *Viewing the Execution History of an Active Network*.
List Active Jobs	---	See *Listing Active Jobs*.
Browse Log	---	See *Displaying Logged Information - Browse Log Function*.
Set Drag And Drop Function	---	See *Drag & Drop*.

Listing Active Job Networks

To list all active job networks of an owner

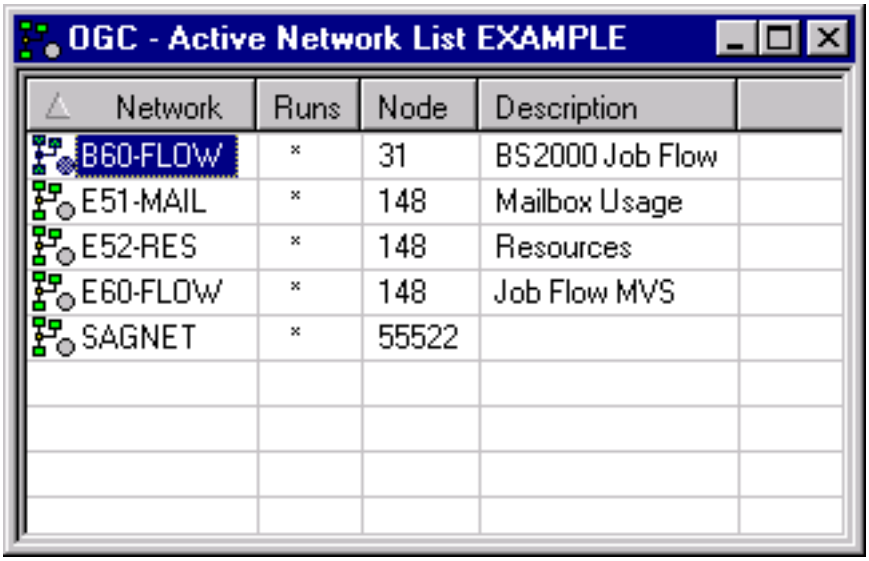
1 In the object workspace, select the **Network Active** node of an owner.

2 From the context menu, choose **List**.

  Or:

  Press F8.

An **Active Network List** window like the example below opens:



- Columns: Active Job Networks

Columns: Active Job Networks

The following table explains the column headings for the data listed in the **Active Network List window**:

Field	Description
Runs	Depending on your choice as described in *Selecting a Range of Networks to be Listed and the Number of Active Runs Display Mode* : The current number of active runs of this network. If there are more than 999 runs >999 is shown. or An asterisk (*) which in this column indicates that one or more active job runs exist for the network.
Network	Network name.
Node	Execution node specified for the network as the default for its jobs.
Description	Short description of network as defined in the job network definition on the master database.

Deactivating Active Job Networks

There are several ways of preventing or interrupting the execution of an active job network. These are described in the following section.

- Deactivation User Rights (Profile Settings)
- Deactivating Active Runs for a Network
- Deactivating all Networks
- Stopping a Running Network

Deactivation User Rights (Profile Settings)

Deactivation requires active job deletion user rights and either network master deletion rights or network activation rights. See *User Definitions and Profile Settings* in the *Administration* documentation.

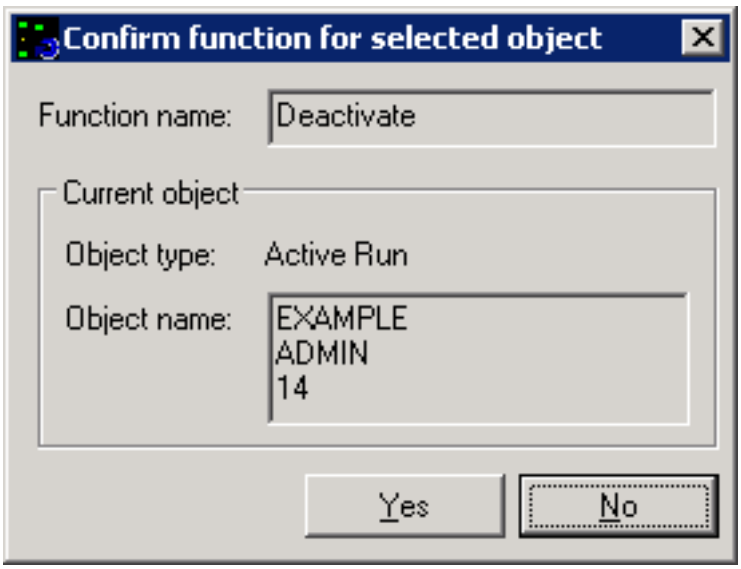
Deactivating Active Runs for a Network

If you wish to prevent Entire Operations from submitting active jobs for a network, you can deactivate a network before the first job run is performed.

To deactivate single or multiple active runs

1 In the object workspace, select an **Active Run** and choose **Deactivate** from the context menu.

A confirmation dialog like the example below opens:



2 Choose **Yes**.

The currently selected active run is deactivated.

To deactivate multiple active runs

1 In the object workspace, select an **Active Run** node and choose **List** from the context menu.

An **Active Run List** window opens with a table of all active runs.

2 Select multiple active runs and choose **Deactivate** from the table's context menu.

In the **confirmation dialog** that opens, choose **Yes** to deactivate one active run after the other, or choose **Yes to All** to delete all active runs at once.

Deactivating all Networks

The system administrator can stop the Entire Operations Monitor to halt the execution of all job networks. Jobs currently being executed continue until normal termination, unless they are explicitly cancelled from the operating system.

Stopping a Running Network

To interrupt a running network, add an input condition, e.g. `WAIT-HERE`, for the job before which the network is to stop running, and ensure that it will not be set. For more information, see *Adding an Active Condition*.

To restart the network at the point of interruption, you can manually set the input condition, or remove it from the active job definition (see *Displaying and Modifying an Active Job Definition*).

Viewing the Execution History of an Active Network

To view the history of a Network Active instance

1 In the object workspace, select a **Network Active** instance and choose **History** from the context menu.

A **History Network Active** window opens for the selected network. This window corresponds to the **History Network Master window** described in *Displaying the Network Execution History* in the section *Schedule Maintenance*.

2 In the calendar, click on the scheduled date for which you want to view the history data.

If history data exists for the selected network and date, execution information on all network starts is contained in the right-hand table columns of the window. The columns are explained in *Columns: Network History* in the section *Schedule Maintenance*.

Maintaining Active Runs

An active run is identified by owner, network and run number. Run numbers are unique per network. If a single job is activated, it will get a run number of the network to which it belongs.

Note: You can maintain active runs from either the **Active Run** node under **Network Active** or the **Active Run** node under **Network Master**. Both nodes control the same active runs. If you use both nodes to change active runs during a session, consider refreshing them manually to make sure that you view the latest status.

This section covers the following topics

- Available Functions: Active Run
- Using Filter Criteria to List Active Runs in a Network
- Viewing a Network Diagram
- Repeating an Active Run
- Modifying the Latest Start Time for an Active Run

Available Functions: Active Run

To list all functions available for an Active Run node

- In the object workspace, select an **Active Run** node and open the context menu.

  The following functions are available:

Function	Shortcut	Description
List	F8	List all active runs in a network: see *Listing Objects*.
Refresh	F5	See *Refreshing Object Lists*.
Filter	F3	See *Using Filter Criteria to list Active Runs in a Network*.
Set Drag And Drop Function	---	See *Drag & Drop*.

To list all functions available for an Active Run instance

- Select an **Active Run** instance and open the context menu.

  The following functions are available:

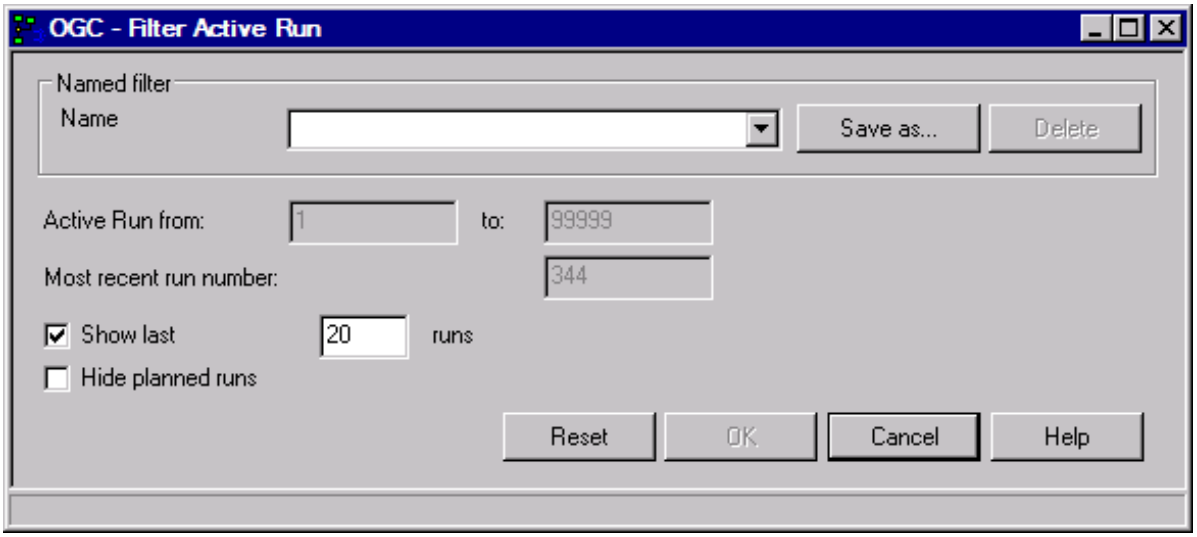
Function	Description
Diagram	See *Viewing a Network Diagram*.
Deactivate	See *Deactivating Active Runs for a Network*.
Release	Release job(s) from hold status for the selected active run. See *Releasing an Active Job*.
Repeat	See *Repeating an Active Run*.
Modify Latest Start	See *Modifying the Latest Start Time for an Active Run*.
List Active Jobs	See *Listing Active Jobs*.
Browse Log	See *Displaying Logged Information - Browse Log Function*.
Add to Workplan	See *Add to Workplan*.
Set Drag And Drop Function	See *Drag & Drop*.

Using Filter Criteria to List Active Runs in a Network

To filter active runs

1 Select an **Active Run** node and choose **Filter** from the context menu.

A **Filter Active Run** window like the example below opens:



2 In the **Show last** input field, enter the number of latest active runs you want to view.

Or:

Remove the mark from the **Show last** check box and enter the range of run numbers you want to view in the **Active Run from/to** input fields (default is `1` to `99999`). The most recent run number is indicated for orientation.

3 In addition, you can mark the **Hide planned runs** check box to hide planned runs from the display list.

You can set default values for an active run filter in your user profile by using the **Filter** option (see the *Administration* documentation).

For information on naming a filter, see ***Filtering Objects***.

Viewing a Network Diagram

Diagrams are provided for network masters and active networks. In case of active networks, they can be used to define an active run within a network and to monitor the progress within the network.

To view a network diagram

1 In the object workspace, select an instance of **Active Run** and choose **Diagram** from the context menu.

Or:

In the object workspace, select an instance of **Job Active** and choose **Network Diagram**.

2 Open the context menu and choose **Diagram** or **Network Diagram**, respectively.

A **Diagram Active Run window** with a network diagram opens like the example shown in the ***Network Maintenance*** section.

Here you can see all network components available for the currently selected **Active Run** in a graphical view.

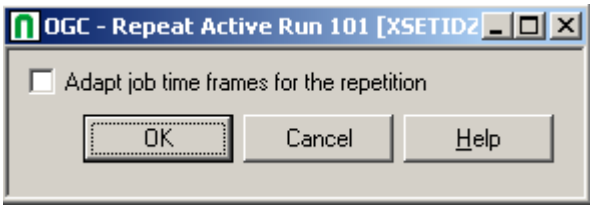
For explanations of the symbols used in the diagram and the functions provided, see ***Explanations of Diagram Symbols*** and ***Maintenance Functions for Diagrams*** in the section *Network Maintenance*.

Repeating an Active Run

To repeat an Active Run

1 In the object workspace, select an instance of **Active Run**.

2 Invoke the context menu and select **Repeat**.



3 You can adapt job time frames for the repetition.

If you select this option, then the time frames are adapted as follows:

Formula	Example
time delay = current time - old earliest start	13:00 - 9:00 = 4 hours
new earliest start = current time	13:00
new latest start = old latest start + time delay	17:00 + 4 hours = 21:00
new deadline = old deadline + time delay	18:00 + 4 hours = 22:00

4 Choose **OK**.

The active run is repeated immediately.

Modifying the Latest Start Time for an Active Run

You can change the latest start time for all jobs of an active run where the active job status permits it. The latest start time may be changed for an active job that is not started or is already terminated.

To modify the latest start for an active run

1 In the object workspace, select an instance of **Active Run**.

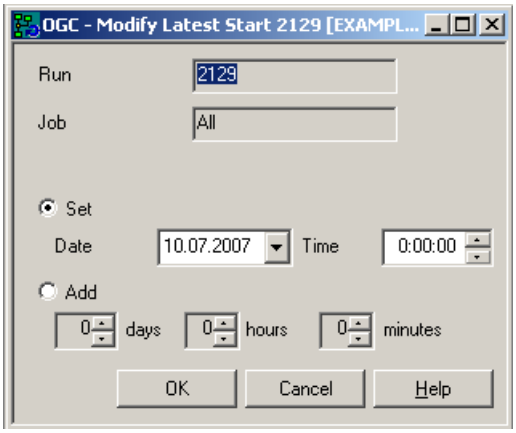
2 Invoke the context menu and select **Modify Latest Start**.

A window like the following opens:

You may choose only one of two methods to modify the latest start time:

- **Set**: Enter date and time values of new latest start;
- **Add**: Enter days, hours and minutes that shall be added to the latest start time value of active job.

The deadline value may get changed during this modification to ensure the following rule is honored: latest start time + estimated elapsed time <= Deadline.



All actions are logged in the Entire Operations log file.

43 Maintaining Active Jobs

- Available Functions: Job Active 472
- Listing Active Jobs 474
- Listing Jobs of an Active Subnetwork 478
- Adding a New Job to the Active Network 479
- Modifying the Latest Start Time for an Active Job 481
- Choosing the Job ID of a Job for Logging 482
- Cancelling, Holding and Releasing Active Jobs 483
- Resubmitting Active Jobs 485
- Deactivating a Job in an Active Network 488
- Reactivating an Active Job 490
- Displaying and Modifying an Active Job Definition 491
- Modifying EOJ Checking and Actions 493
- Viewing Long Descriptions of Active Jobs 494
- Displaying Prerequisites for Active Jobs: Waiting for 495
- Viewing and Modifying Resources Used by Active Jobs 496
- Browsing Active Job SYSOUT 498

Related Topics:

- *Activating a Single Job Manually*

Available Functions: Job Active

To list all functions available for a Job Active node

- In the object workspace, select a **Job Active** node and open the context menu.

  The following functions are available:

Function	Shortcut	Description
List	F8	See *Listing Active Jobs*.
New	CTRL+N	Creating a new **Job Active**: see *Adding a New Job to the Active Network*
Refresh	F5	See *Refreshing Object Lists*.
Filter	F3	Using selection criteria to list active jobs: see *Filtering Objects*.
Set Drag And Drop Function	---	See *Drag & Drop*.

To list all functions available for a single Job Active instance

- Select a **Job Active** instance and invoke the context menu.

  The following functions are available:

Function (or shortcut)	Description
Open (or CTRL+O)	Opens the **Maintenance Job Active** to modify all definitions of the specified job: see *Displaying and Modifying an Active Job Definition*.
Display (or CTRL+D)	Opens the **Display Job Active** window to view all definitions of the specified job. The read-only fields in the window correspond to the fields in the **Maintenance Job Active** window: see *Displaying and Modifying an Active Job Definition*.
Network Diagram	Opens the **Diagram Active Run** window displaying an overview of the active job runs within the network. See *Viewing a Network Diagram*.
Zoom Subnetwork	See *Listing Jobs of an Active Subnetwork*.

Function (or shortcut)	Description
Subnetwork Diagram	This function is used for navigation purposes. Using it for a job of type NET, the network diagram for the network that is linked to this object is opened. See *Viewing a Network Diagram*.
Usable Symbol Tables	Opens a list of symbol tables used by the active job. See also *Listing Usable Symbol Tables* in the section *Symbol Table and Symbol Maintenance*.
Deactivate	See *Deactivating a Job in an Active Network*
Hold	See *Holding an Active Job*.
Release	See *Releasing an Active Job*.
Cancel	See *Cancelling an Active Job*.
Resubmit	See *Resubmitting an Active Job*.
Reactivate	See *Reactivating an Active Job*.
Regenerate Active JCL	See *Regenerating Active JCL*.
Stop cyclic execution	Stops a currently executing cyclic job.
Modify Latest Start	See *Modifying the Latest Start Time for an Active Job*.
Browse Active JCL	See *Browsing Active JCL*.
Edit Active JCL	See *Editing Active JCL*.
Release edit lock	Removes a lock from active JCL held on the source by a user who previously edited it. The unlock action is archived in the Entire Operations log. **Important:** This function is for emergency use only. Data loss can occur when you remove the lock from a source while it is being edited by several concurrent users. See also *Locking of Natural Sources* in the section *Job Maintenance*.
Extended Log / Active JCL Changes	Opens a window that shows the extended log containing the changes to the JCL submitted for the job. See *Viewing the Extended Log for Active JCL Changes* for more information. See also *Choosing the Job ID of a Job for Logging*.
Browse SYSOUT	See *Viewing Job SYSOUT*.
Extended Log / JCL	Opens a window that shows the extended log containing the JCL submitted for the job. For information on defining extended logs, see *Defining Extended Log Information for a Job* in the section *Job Maintenance*. For more information on viewing extended logs, see *Displaying Extended Log Information* in the section *Log Information*. See also *Choosing the Job ID of a Job for Logging* and *JCL Log* in the section *Job Maintenance*.

Function (or shortcut)	Description
Extended Log / SYSOUT	Opens a window that shows the extended log containing the job's SYSOUT. For information on defining extended logs, see *Defining Extended Log Information for a Job* in the section *Job Maintenance*. For more information on viewing extended logs, see *Displaying Extended Log Information* in the section *Log Information*.
SYSOUT Messages	Opens a window that shows the extended log containing the SYSOUT messages of the job (z/OS only). For information on defining extended logs, see *Defining Extended Log Information for a Job* in the section *Job Maintenance*. For more information on viewing extended logs, see *Displaying Extended Log Information* in the section *Log Information*.
Waiting for	See *Displaying Prerequisites for Active Jobs: Waiting for*.
List Active Jobs	See *Listing Active Jobs*.
Browse Log	See *Displaying Logged Information - Browse Log Function*.
Add to Workplan	See *Add to Workplan*.
Set Drag And Drop Function	See *Drag & Drop*.

Listing Active Jobs

You can access active runs of a network to list and maintain active jobs from different nodes in the object workspace. The result list of active jobs depends on the node and the node instance selected.

User Restrictions for Active Jobs Lists

Users can be restricted to view jobs in the **List Active Jobs** window.

If the user is of type **Administrator** (A) or if the user has access to the owner SYSDBA, then all active jobs are displayed. In all other cases, the following active jobs are displayed:

- From networks of the current owner,
- From networks of owners to which the user has access,
- From networks to which the user has access.

See also *Granting Definition: Authorizing Other Users or Owners to Access a Network* in the section *Network Maintenance*.

To list active jobs

1 In the object workspace, select an instance from a **Network Active** or an **Active Run** node to list all active jobs of this network or active run, respectively.

Or:

In the object workspace, select an instance from an **Owner** or a **Network Master** node to list all active jobs of this owner or network, respectively.

Or:

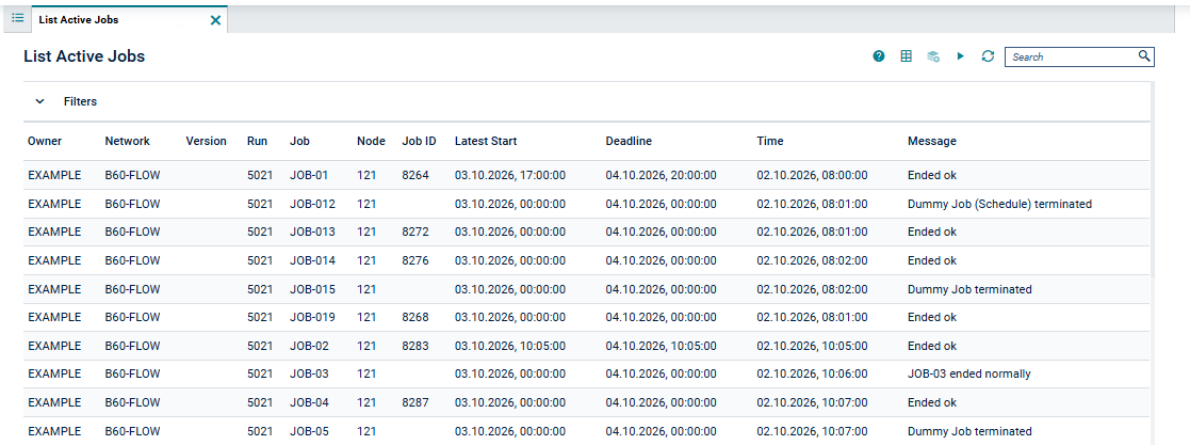
In the main menu, choose **General**.

2 Choose **List Active Jobs**.

Or:

Open the context menu and choose the **List Active Jobs** function.

A **List Active Jobs** dialog like the example below opens with a list of active jobs:



Owner	Network	Version	Run	Job	Node	Job ID	Latest Start	Deadline	Time	Message
EXAMPLE	B60-FLOW		5021	JOB-01	121	8264	03.10.2026, 17:00:00	04.10.2026, 20:00:00	02.10.2026, 08:00:00	Ended ok
EXAMPLE	B60-FLOW		5021	JOB-012	121		03.10.2026, 00:00:00	04.10.2026, 00:00:00	02.10.2026, 08:01:00	Dummy Job (Schedule) terminated
EXAMPLE	B60-FLOW		5021	JOB-013	121	8272	03.10.2026, 00:00:00	04.10.2026, 00:00:00	02.10.2026, 08:01:00	Ended ok
EXAMPLE	B60-FLOW		5021	JOB-014	121	8276	03.10.2026, 00:00:00	04.10.2026, 00:00:00	02.10.2026, 08:02:00	Ended ok
EXAMPLE	B60-FLOW		5021	JOB-015	121		03.10.2026, 00:00:00	04.10.2026, 00:00:00	02.10.2026, 08:02:00	Dummy Job terminated
EXAMPLE	B60-FLOW		5021	JOB-019	121	8268	03.10.2026, 00:00:00	04.10.2026, 00:00:00	02.10.2026, 08:01:00	Ended ok
EXAMPLE	B60-FLOW		5021	JOB-02	121	8283	03.10.2026, 10:05:00	04.10.2026, 10:05:00	02.10.2026, 10:05:00	Ended ok
EXAMPLE	B60-FLOW		5021	JOB-03	121		03.10.2026, 00:00:00	04.10.2026, 00:00:00	02.10.2026, 10:06:00	JOB-03 ended normally
EXAMPLE	B60-FLOW		5021	JOB-04	121	8287	03.10.2026, 00:00:00	04.10.2026, 00:00:00	02.10.2026, 10:07:00	Ended ok
EXAMPLE	B60-FLOW		5021	JOB-05	121		03.10.2026, 00:00:00	04.10.2026, 00:00:00	02.10.2026, 10:07:00	Dummy Job terminated

This list displays all active jobs available showing the owner, network, run, job node, job ID, time and a message.

The columns and input fields are explained in *Columns and Filter Criteria: List Active Jobs*.

Notes:

1. For performance reasons, the table view is filled dynamically with data if scrolled to the end of the table.
2. For a comprehensive overview of the job list, the **Latest Start**, **Deadline** and **Version** columns and the **Filter** section are hidden.

To filter the job list

1 Click on **Filters** to expand (or collapse) the **Filter** section.

The **Filter** section is expanded.

2 Specify or select the required filter criteria and choose **Apply** when finished.

The columns and input fields are explained in *Columns and Filter Criteria: List Active Jobs*.

To show or hide individual columns in the list:

1 Choose **Column** ( ).

The **Select Columns** dialog is displayed.

If the **Field** checkbox is ticked, all selectable column names are selected. This is the default setting.

The columns **Owner** and **Network** cannot be deselected.

2 Tick or untick the box to show or hide individual columns in the list.

3 Select **OK** to confirm your selection.

The changes will be saved and displayed when you click **Column** ( ) within the current function again.

To refresh the list contents

- Proceed as described *Refreshing Object Lists*.

This section covers the following topics:

- Columns and Filter Criteria: List Active Jobs
- Listing Active Jobs by Process Status

Columns and Filter Criteria: List Active Jobs

The columns and input fields available in the **List Active Jobs** window are explained in the following section.

Depending on the node or node instance for which the **List Active Jobs** function is used, the fields below Filter are either read-only or can be used to specify filter criteria to reduce the number of active jobs listed in the window.

Column/Field	Description
Owner	The name of the owner whose active jobs are shown. If available as input field, you can select an owner out of the drop-down-box to show only its active jobs.
Network	The name of the network for which the active jobs are shown. If available as input field, you can select a network out of the drop-down-box to show only its active jobs.
Version	The network version for which the active jobs are shown. If available as input field, you can select a version out of the drop-down list box to show only its active jobs (see also ***Maintaining Job Network Versions*** in the section *Network Maintenance*).
Run	The run number for which the active jobs are shown. The run number is automatically assigned at activation time. The job run number uniquely identifies an active copy of a job together with the job name. If available as input field, you can select the **run number** of the network to show only its active jobs.
Job	The job name as defined to Entire Operations. If available as input field, you can select a job out of the drop-down-box to show only its active jobs.
Node	Execution node of the machine designated for the job. If available as input field, you can select a node out of the drop-down-box to show only its active jobs.
JobId	The job identifier as assigned by the operating system or by the job entry subsystem.
Time	Last action or check date/time for the job.
Message	Last message issued for the job by Entire Operations. For a list of possible messages and their meaning, see *Messages in Active Jobs Lists* in the *Messages and Codes* documentation.
Submit user ID	Select a user ID out of the drop-down-box to show only its active jobs.
Show last *n* runs	Enter a number to restrict the list of active jobs to the last *n* runs where *n* is a number in the range from 1 to 999999999.
Show schedule	See ***To display the Latest Start and Deadline columns***.
Show version	See ***To display the Version column***.
Status filter	Select job status filters to additionally reduce the list of active jobs: see ***Listing Active Jobs by Process Status***.

Listing Active Jobs by Process Status

You can list active jobs according to their processing status by opening the **Filter** section of the **List Active Jobs Network Active** window and selecting one or more **Status filter** options:

Selection Option	Description
All active jobs	All active jobs (default).
Waiting for an event	All active jobs waiting for at least one event.
Jobs in hold	All jobs in hold.
In spool input queue	All jobs in the scheduling system input queue.
Ended not ok	All jobs with `job not ok` terminating status.
Ended ok	All jobs with `job ok` terminating status.
Submitted	All submitted jobs.
Terminated	All terminated jobs.
Waiting for activation	Planned network runs waiting for activation.
Executing	All jobs in execution.
Latest start time exceeded	All jobs that exceed the **Latest Start Time**: see *Modifying the Latest Start Time for an Active Job*.

Listing Jobs of an Active Subnetwork

This function is only applicable to jobs of type subnetwork (**NET**).

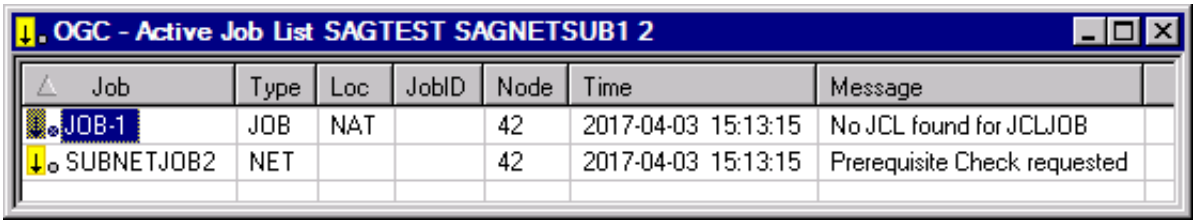
To list jobs of an active subnetwork

- From a **Job Active** node, select a job of the type **Subnetwork** (`NET`), and choose **Zoom Subnetwork** from the context menu.

  Or:

  From the table in an **Active Job List** window, select a job of the **Type** `NET`, and choose **Zoom Subnetwork** from the context menu.

  The jobs of the subnetwork defined for the selected `NET`-type job are listed in a (separate) **Active Job List** window, as shown in the following example:



Adding a New Job to the Active Network

To add a job to the active database for the current run of an active job network

- Select a **Job Active** node and choose **New** from the context menu.

  A **Create new Job Active** window (similar to **Maintenance Job Active**) opens where you can enter the required definitions as described for a job master in the section *Adding a Job Definition*.

Notes:

1. JCL is automatically loaded after the job has been added.
2. A job is not automatically activated after being added to an active network. The job is first put into hold status and additional definitions can be added, for example, active time frames. To activate the new job, select the new active job. On the **Job Active** dialog screen, choose the **Release** button.

Adding a job to an active job network involves temporarily changing the network structure and job flow. This is achieved with minimum effort as indicated in the following **example**.

This section covers the following topics:

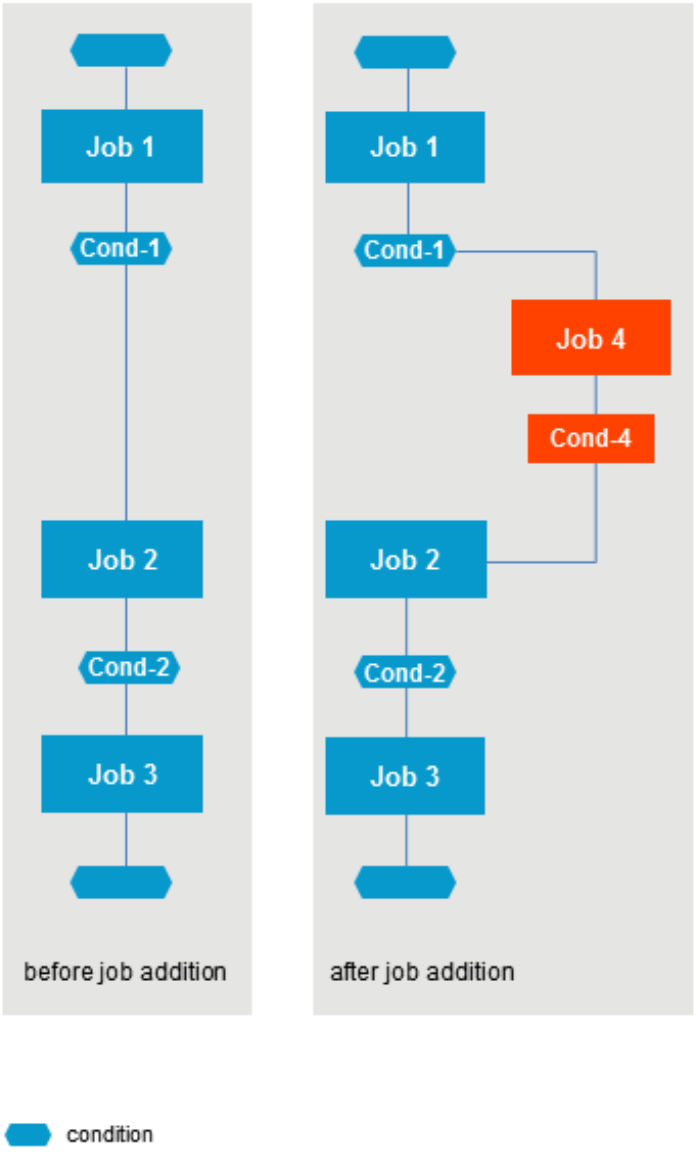
- Example

- Achieving a Sequential Flow

Example

You may wish to add a job to a network for one specific run.

This example illustrates the addition of a job to a network consisting of 3 sequential jobs. The new job (Job 4) is to be inserted between Job 1 and Job 2:



Achieving a Sequential Flow

To achieve a sequential job flow

1 Add the new active job using the **Create new Job Active** window;

2 Define Cond-1 as input condition for Job 4;

3 Define Cond-4 as output condition in End-of-Job handling for Job 4;

4 Change the input condition defined for Job 2 (Cond-1) to Cond-4.

Job 4 is executed for the current run of the job network only.

If you wish this change to be permanent, use this procedure on the master database.

Modifying the Latest Start Time for an Active Job

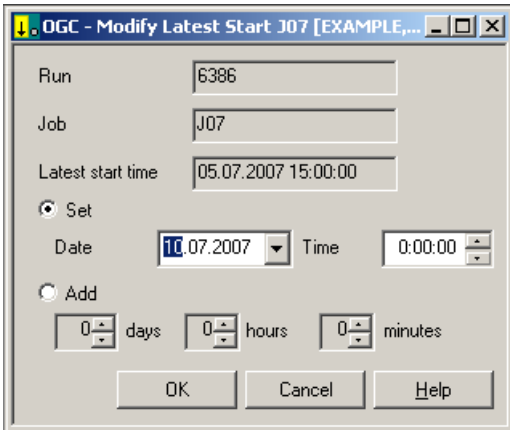
You can change the latest start time or an active job instance where the active job status permits it. The latest start time may be changed for an active job that is not started or is already terminated.

To modify the latest start time for an active job

1 In the object workspace, select an instance of **Job Active**.

2 Invoke the context menu and choose **Modify Latest Start**.

A window like the following opens:



The options available correspond to the methods described for an **active job run**.

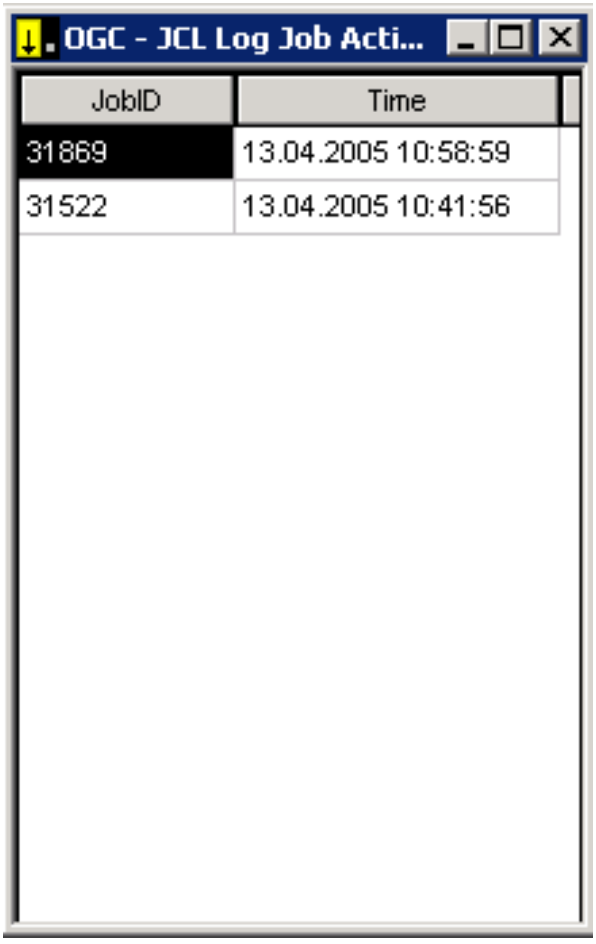
Choosing the Job ID of a Job for Logging

If you have several instances of one job running, for example, after resubmitting a job, you receive a selection window for selecting the job ID of the job whose log you want to view.

To choose the log

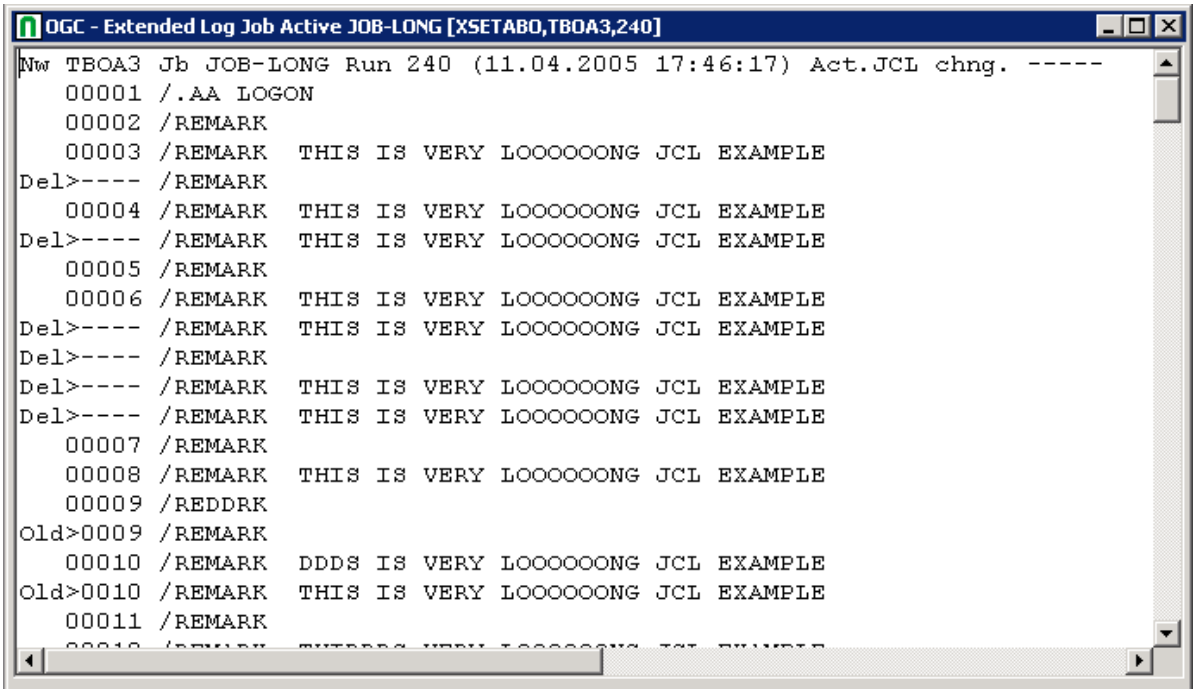
1 Select a job from the **Job Active** node and choose **Extended Log / Active JCL changes** or **Extended Log / JCL** from the context menu.

The following window opens:



2 Select a job by double-clicking on a job ID in the **JobID** column.

The selected log is displayed:



Cancelling, Holding and Releasing Active Jobs

- Operating System Jobs Handling
- Cancelling an Active Job
- Holding an Active Job
- Releasing an Active Job

Operating System Jobs Handling

Operating system jobs in the Entire Operations active database can be handled directly from a **Job Active** instance.

When Entire Operations submits jobs to the operating system, they are assigned a job number which appears, for example, in the **Maintenance Job Active** window. In z/OS systems, JES operator commands are issued.

Note: In BS2000, console commands are issued.

Cancelling an Active Job

You can cancel the planned activation of a job in a planned network run that is waiting for activation. The job is then not executed within the active network.

To cancel an active job from the operating system

- In the object workspace, select a **Job Active** instance and choose **Cancel** from the context menu.

  A **confirmation dialog** prompts you to choose **Yes** to cancel the job.

  The job is cancelled from the operating system.

Notes:

1. A cancelled job cannot be released for further processing.
2. For BS2000 jobs, you can only use the **Cancel** function if the following conditions are fulfilled:
   - The BS2000 user ID in effect can issue a cancel command for the Entire System Server node(s) in use. If required, specify the following in the Entire System Server parameter file and restart Entire System Server:

     ```
     SYSTEM-COMMAND-USER=TSOS,user-id[,...]
     ```

     For example:

     ```
     SYSTEM-COMMAND-USER=TSOS,SAG1,SAG2,SAG3
     ```

   - The active and passive Entire System Server console servers are available (see the Entire System Server *User's Guide*).
3. UNIX jobs are cancelled by the SIGTERM signal.

Holding an Active Job

You can stop an active job and place it in hold status.

To stop a job and place it in hold status

- In the object workspace, select a **Job Active** instance and choose **Hold** from the context menu.

  The message `Job set to Hold` appears in the **Message** column for the job

Notes:

1. If the job has already been submitted to the operating system, it will be held in the operating system.
2. Operating system hold does not work for Windows jobs.
3. For UNIX jobs, the whole process group belonging to the job ID will be held.

Releasing an Active Job

You can release a job from hold status.

To release a job from hold status and allow it to continue

- In the object workspace, select a **Job Active** instance and choose **Release** from the context menu.

  The message `Job released from Hold` appears in the **Message** column for the job.

Notes:

1. If the command is used for a job of the type `NET` (subnetwork): during a release, no new activation of the subnetwork will be performed.
2. The release from hold command is available for networks in the status `to be activated` too.

Resubmitting Active Jobs

After a job has terminated, you can modify and resubmit it while it is still in the active database. This function is useful after a job has failed.

For example, if a JCL error has occurred, you can modify the active JCL (see *Editing Active JCL*) and resubmit the job. The resubmitted job uses the old input conditions for submission, but can set different output conditions, depending on the End-of-Job analysis.

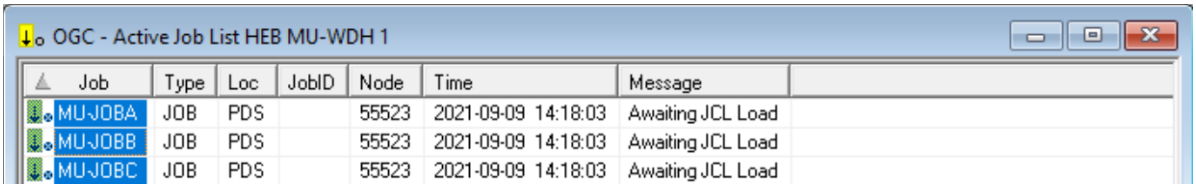
Any output conditions set by the job's first run can be reset by the second run if this feature is defined in the job's original End-of-Job handling.

To resubmit active jobs

1 In the **object workspace**, select the nodes **Owner** > *owner-name* > **Network Master** > *network-name* > **Active Run** > *run-number* > **Job Active** > *job-name*.

In the **object workspace**, you can select only one active job at a time.

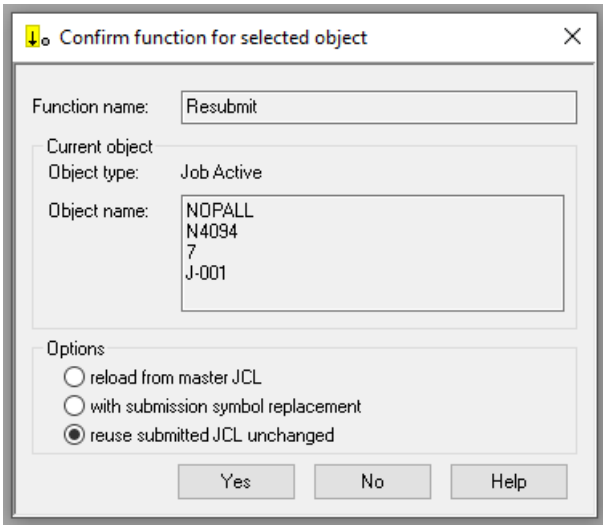
In the **content pane**, you can select multiple active jobs simultaneous, for example:



2 Invoke the context menu and choose the **Resubmit** function.

The window **Confirm function for selected jobs** opens, for example:

Function: Resubmit with multiple active jobs selected.



Select a **resubmission option** and choose **Yes** to confirm the resubmission.

If multiple active jobs were selected, choosing a button will result in the following:

Yes	The confirmation dialog will be displayed for each of the selected active jobs successively, so you can choose different options for each of them.
Yes to All	The confirmation dialog will not be repeated. Alle selected active jobs will be repeated with the chosen option.
No	The selected job being displayed will be skipped (not repeated).
Cancel	Return to active job selection. **Note:** Active jobs previously confirmed with **Yes** will **not** be repeated.

The options are explained in detail in *Options for Resubmission*.

This section covers the following topics:

- Options for Resubmission

- Resubmitting an Active Job of NET type (Subnetwork)
- No Reloading after Editing
- Job Resubmission under BS2000

Options for Resubmission

You have one of the following options to resubmit a job:

reload from master JCL	Reload the JCL from the master JCL definition and resubmit the job.
with submission symbol replacement	Reload the active JCL with symbol replacement (default) and resubmit the job. Symbols and symbol function values are updated depending on the setting of the Entire Operations default **Submit symbol/function recalculation at resubmit** in the **Defaults for Other Settings** window described in the *Administration*
reuse submitted JCL unchanged	Reload the active JCL from the previous job submission and resubmit the job exactly as before. Symbol replacement is not repeated. **Note:** Any modifications of the active JCL will be overwritten.

Note: Jobs of the types `NAT` (Natural Program) or `STC` (Started Task) are always resubmitted with **keep active JCL**; a different option selected for resubmission is ignored.

Resubmitting an Active Job of NET type (Subnetwork)

If the **Repeat** function is used for jobs with `NET` type (subnetwork), the subnetwork will be repeated with the same subnetwork run number. However, this is only allowed if the subnetwork has not started yet.

No Reloading after Editing

In case of a resubmission of the active job, the **pregenerated JCL** is not reloaded if the active JCL of the job has been edited in the meantime.

Job Resubmission under BS2000

The SYSOUT file copy attempts are limited if the SYSOUT file is locked. The limit is 10 times the task (Monitor) wait time. If the task wait time is smaller than 30 seconds, the limit is 10 * 30 = 300 seconds.

If the SYSOUT file is still locked when the limit is reached, the active job will be set to a permanent error status, with the message text `10 SYSOUT Renaming Attempts failed`.

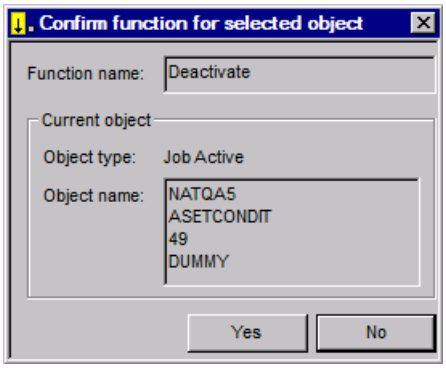
For each unsuccessful attempt, the message text will be set to `SYSOUT File Renaming - File in use`. In addition, the event will be logged with the SYSOUT file name.

Deactivating a Job in an Active Network

To prevent the execution of a certain job in an active network for the current run

1 From a **Job Active** node in the object workspace, select an active job and choose **Deactivate** from the context menu.

A confirmation dialog like the example below opens for the selected job:



The window indicates the name of the owner (here: `NATQA5`), the name of the active network (here: `ASETCONDIT`), the run number (here: `49`) and name of the active job (here: `DUMMY`).

2 Choose **Yes** to deactivate the job or **No** to cancel the action.

Deactivation is performed by the Entire Operations Monitor in the background. Deactivating a job from the active network involves deleting it from the active job network and restructuring the active network to ensure uninterrupted job flow. The following **example** illustrates job deactivation by bypassing a job in an active network consisting of three sequential jobs.

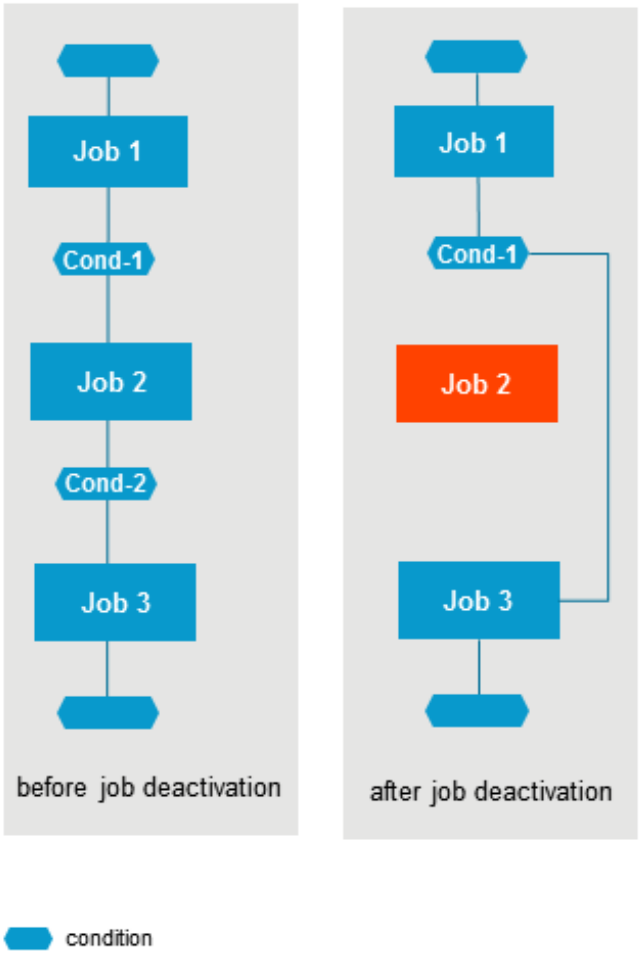
This section covers the following topics:

- Example: Deactivation of a Job

- Ensuring Uninterrupted Job Flow

Example: Deactivation of a Job

The network in the following figure is to run without Job 2:



Ensuring Uninterrupted Job Flow

To ensure uninterrupted job flow

1 To prevent Job 2 from running, it is sufficient to delete its input condition Cond-1. If you deactivate Job 2 with the **Deactivate** function, the input condition is automatically deleted.;

2 Replace the input condition for Job 3 (Cond-2) by the input condition Cond-1.

The current run of the job network skips Job 2.

If you wish to deactivate a job that is currently running (interrupt its execution), you must cancel it from the operating system before deactivation (see *Cancelling an Active Job*).

Reactivating an Active Job

This function is used to deactivate an active job and then reactivate it completely.

This function can be used to restore an inconsistent active job (for example, if a symbol was missing during the original activation).

To reactivate an active job

- In the object workspace, select a **Job Active** instance and choose **Reactivate** from the context menu.

  A **confirmation dialog** prompts you to choose **Yes** to reactivate the job.

Reactivation is performed in the background by the Entire Operations Monitor, whereby:

- The active job is deactivated and activated again.

  After deactivation, the job may temporarily disappear from the active jobs list or active diagram.
- All input and output definitions are recreated.
- The active JCL is newly generated.
- A prerequisite check is performed after reactivation.

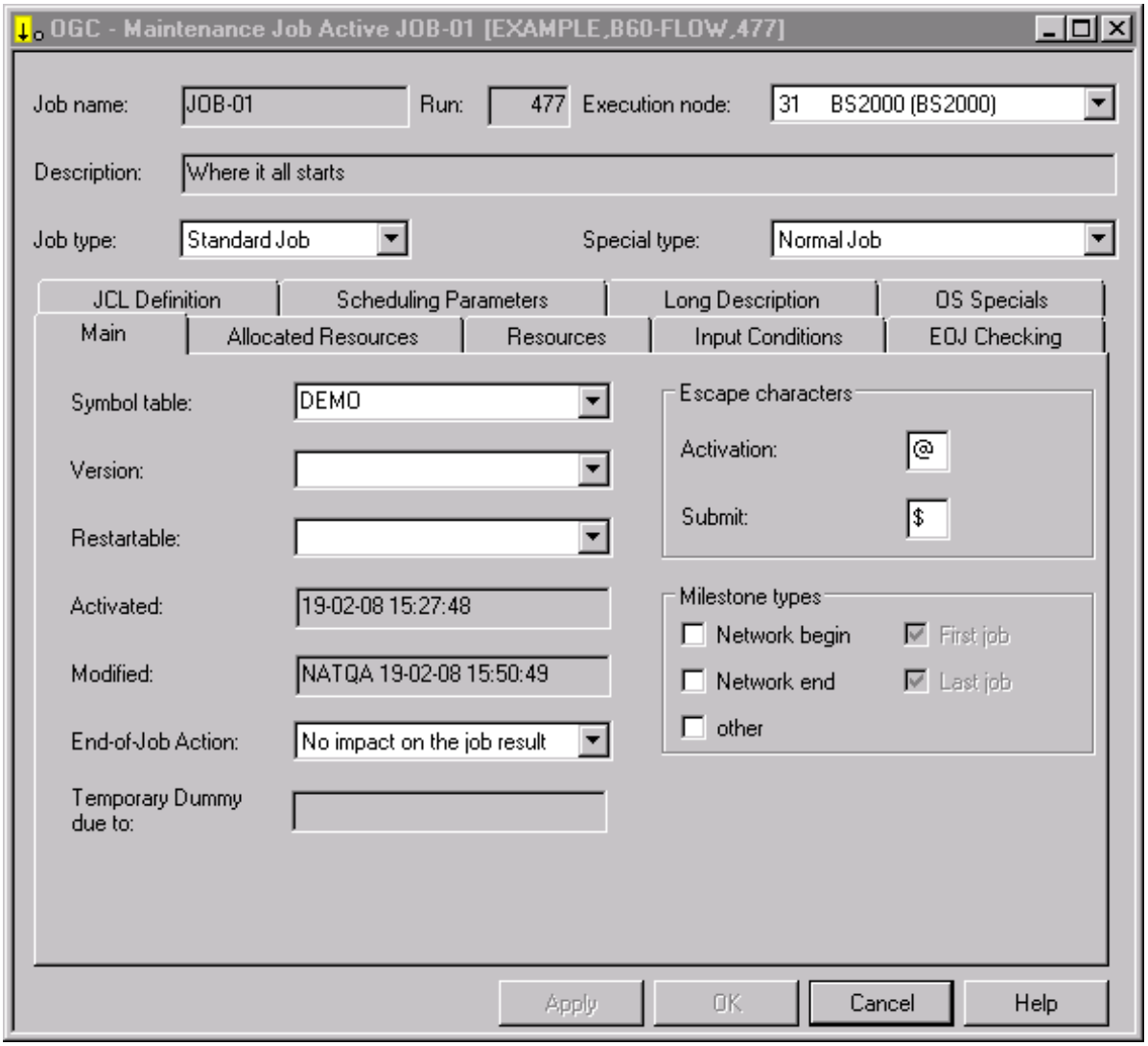
Displaying and Modifying an Active Job Definition

To view and modify an active job

1 In the object workspace, select a **Job Active** instance.

2 Invoke the context menu and choose **Open** or **Display**.

Open opens a **Maintenance Job Active** window like the example below:



Display opens a **Display Job Active** window which corresponds to the **Maintenance Job Active** window above but is write-protected.

The following applies when you modify a job definition in an active job network definition:

- The current user ID is stored as the last modifying user of the job for all definition and JCL modifications. This user ID is taken as the submit security user ID. If the Monitor does not use its own user ID, see *Monitor Defaults - Submit Security User Type* in the *Administration* documentation.
- It may be necessary to release the active job after the modification with the **Resubmit** function. This causes a new check of all prerequisites.

This section covers the following topics:

- Fields: Maintenance Job Active
- Special Job Type D - Execution as a Dummy

Fields: Maintenance Job Active

The input fields in the **Maintenance Job Active** window mostly have the same meaning as in the maintenance window of a job master definition; see ***Fields: Job Definition (Master)*** in the section *Job Maintenance*. However, the **Maintenance Job Active** window contains the following additional fields which are protected:

Field	Description	
Temporary Dummy due to	Indicates why a job executed as a temporary dummy. Possible flags/entries:	
	C **Condition**	Dummy due to condition.
	D **Definition**	Dummy due to definition.
	E **Empty JCL**	Empty JCL (no JCL statement generated on purpose).
	J **JCL check**	JCL check.
	K **Deactivation**	Job is deactivated.
	M **Multiple suffix**	Dummy due to multiple suffixes.
	R **Recovery**	Dummy due to recovery.
	S	Dummy due to schedule dependency.

Field	Description	
	Schedule dependency	
	T **Repetition**	Dummy due to repetition.
	Note: A job that executes as a temporary dummy can have an impact on the execution of End-of-Job actions defined for the job.	
Activated	Activation date and time of the network.	
Modified	User ID and timestamp of the last modification.	
Run	Run number of the current job run.	

Special Job Type D - Execution as a Dummy

In the **Maintenance Job Active** window, it is possible to modify the field **Special Type** to **Execute as dummy**.

- If **Execute as dummy** is not set, an automatic reloading of the JCL is performed. In case of active jobs of the type `NET`, the subnetwork is activated.
- If **Execute as dummy** is set, the job is executed as a dummy job. In this case, it does not play any role, whether already active JCL or an active subnetwork were loaded.

See also: *Job Execution as a Dummy Job*.

Modifying EOJ Checking and Actions

To modify the End-of-Job checking and actions for the current run of a specific job

- In the **Maintenance Job Active** window, open the **EOJ Checking** page.

  The current definitions are displayed on the page like the **example** shown in the section *Defining and Managing End-of-Job (EOJ) Checking and Actions*. You can modify event and action definitions in the same way as described in this section.

Note: Any modifications made to End-of-Job handling parameters of an active job are valid for the current job run only.

Viewing Long Descriptions of Active Jobs

To view the long text description of an active job

- In the **Maintenance Job Active** window, open the **Long Description** page.

  The **Long Description** page contains either descriptive text for the job (see the **example** in the section *Job Maintenance*) or an URL address as shown in the example below:



  If an URL address is displayed, you can choose **Open URL** to open the page associated with the URL.

The page contains text, provided a long description exists for the job master. The text is displayed in Editor format. No modification is possible. The text can only be modified on the master database by using the appropriate job maintenance function as described in *Writing and Viewing Online Documentation for a Job*.

Displaying Prerequisites for Active Jobs: Waiting for

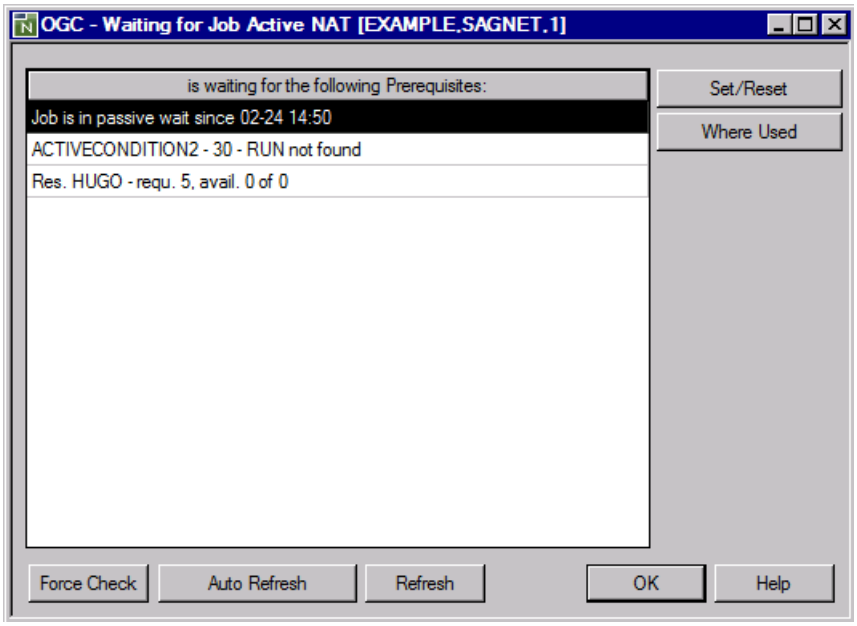
The **Waiting for** function is used to display all events pending for an active job.

To display all waiting prerequisites for an active Job

1 In the object workspace, select a **Job Active**.

2 Invoke the context menu and choose the **Waiting for** function.

A **Waiting for Job Active** window like the example below opens:



The window lists the events an active job is waiting to complete.

The listed messages indicate that the current run of the job (run number `1`) is waiting for the input condition `ACTIVECONDITION2` from the run number `30`, and the availability of the resource `HUGO`. A resource is indicated by a `Res.` prefix.

The text `Job is in passive wait since 02-24 14:50` shows since when the job has been in passive wait for the input condition and the resources.

In this case, the job is automatically returned to the active wait by setting the correct input condition and making sure that the resource can be allocated; that is, the Monitor will execute a prerequisite check.

For a complete list of possible messages and their meaning, see *Messages in Active Jobs Lists* in the *Messages and Codes* documentation.

All functions available in the **Waiting for** window are explained in *Functions: Prerequisites for an Active Job*.

Functions: Prerequisites for an Active Job

The following functions are available in the **Waiting for Job Active** window:

Function	Description
Force Check	Force an active prerequisite check. Only before the message `Job is in passive wait since ...`
Auto Refresh	See *Refreshing Object Lists*.
Refresh	
Set/Reset	**Set**: ■ Used for a condition wait: set the condition to true. **Reset**: ■ If used for a condition wait: reset the condition to false. ■ If used for a resource wait: delete the active prerequisite resource definition. The active job will then no longer wait for the resource.
Where Used	Where used: ■ If used for a condition wait: display the usage of the active condition. For further information, see *Displaying Conditions for an Active Job*. ■ If used for a resource wait: display the usage of the resource. For further information, see *Viewing and Modifying Resources Used by Active Jobs*.

Viewing and Modifying Resources Used by Active Jobs

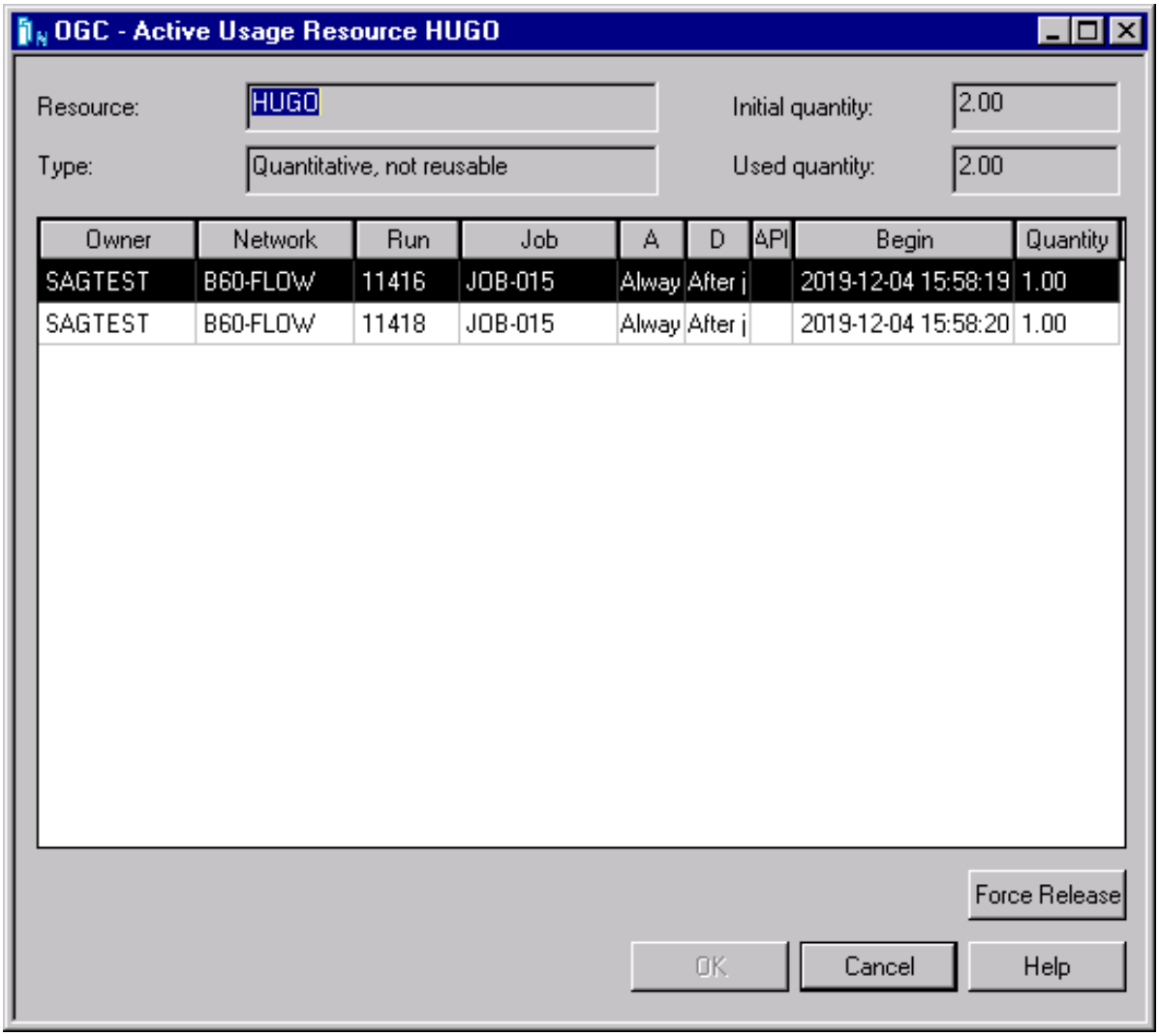
You can display the resources used by active jobs.

You can also modify resources for the current run of an active job without changing the original resource specification on the master database for future job runs. This can be useful if different job runs require different amounts of a specific resource, for example paper or tape drives.

To view resources used by an active job

■ In the **Waiting for Job Active** window, select the required resource and choose **Where Used**.

An **Active Usage Resource** window like the example below opens:



The window lists other active jobs in your active network environment which are currently using partial quantities of the resource for which your active job is waiting.

The fields and columns in the window are described in *Fields and Columns: Active Resource Usage* in the *Administration* documentation.

Browsing Active Job SYSOUT

You can view SYSOUT of an active job depending on the operating system installed at your site:

Operating System	SYSOUT Availability
z/OS	SYSOUT can be browsed immediately after job submission. It is readable until the current last line readable from the spooling system (for example, JES2).
BS2000	The SYSOUT file can be browsed only after job termination. This is because the SYSOUT file is open during execution and cannot be read.
UNIX/Windows	The SYSOUT file can be browsed immediately after job submission. It is readable up to its current last line.

This section covers the following topics:

- Viewing Job SYSOUT
- SYSOUT File Types on z/OS
- Header and Symbol Information in SYSOUT
- Available Functions: Browse SYSOUT

Viewing Job SYSOUT

To view job SYSOUT

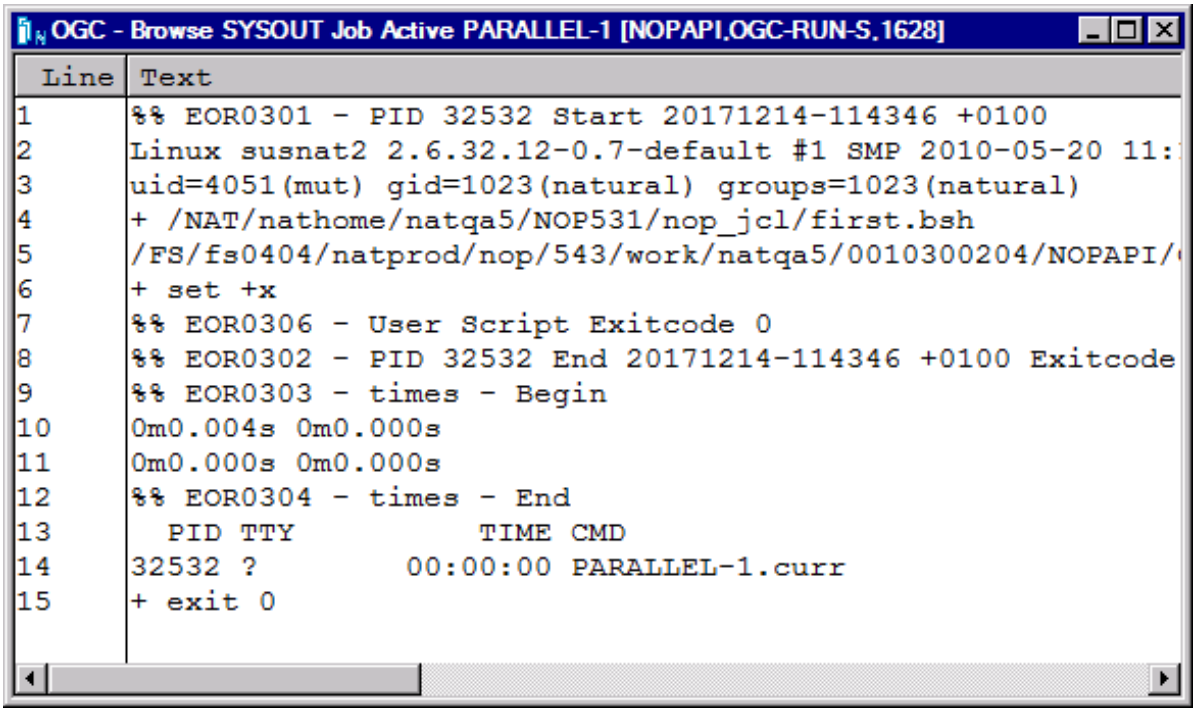
1 In the object workspace, select a **Job Active** instance and choose **Browse SYSOUT** from the context menu.

Depending on your access rights, a **Logon Node window** opens.

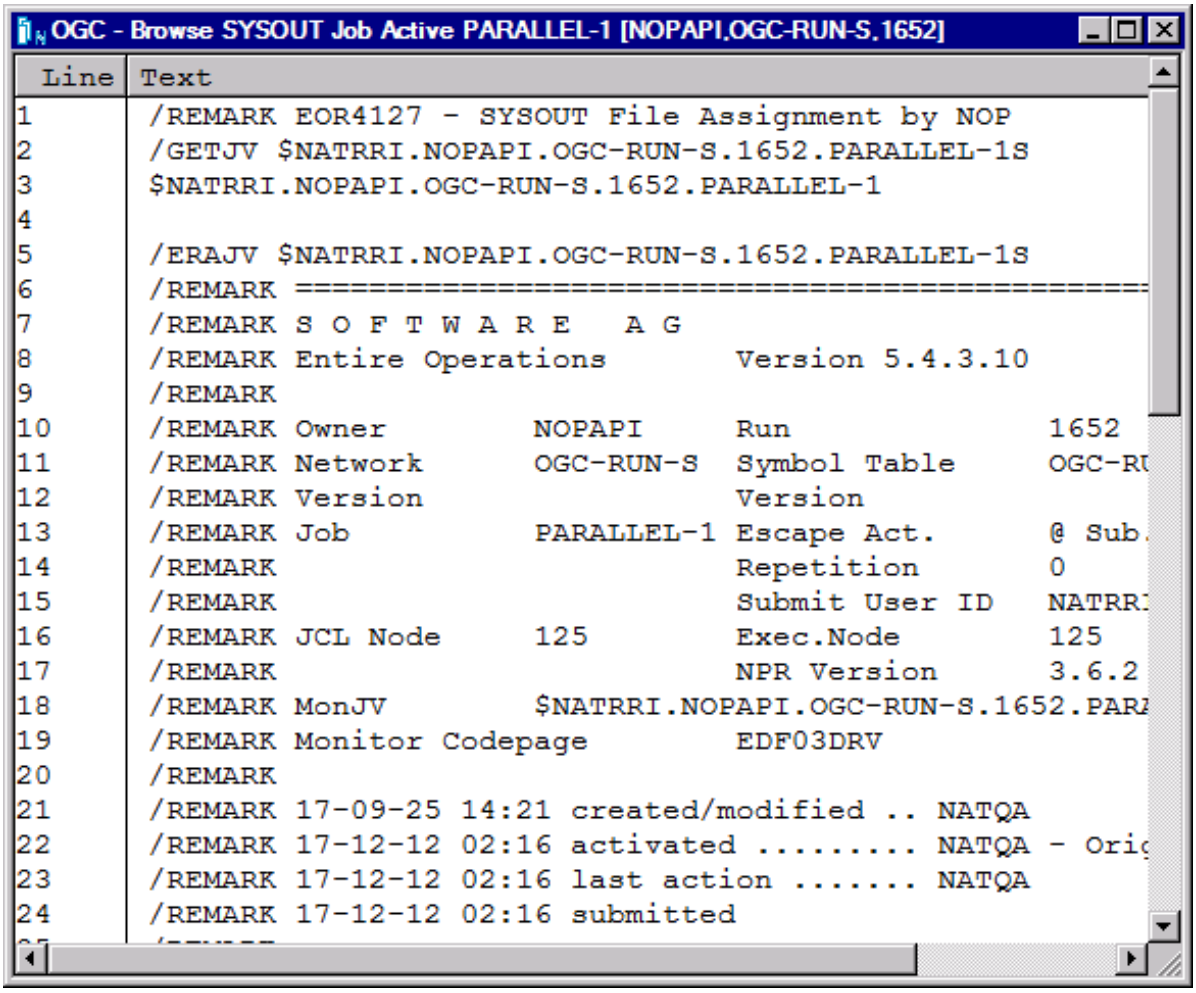
2 Enter your user ID and password and choose **OK**.

A **Browse** window like the example below opens:

On UNIX:

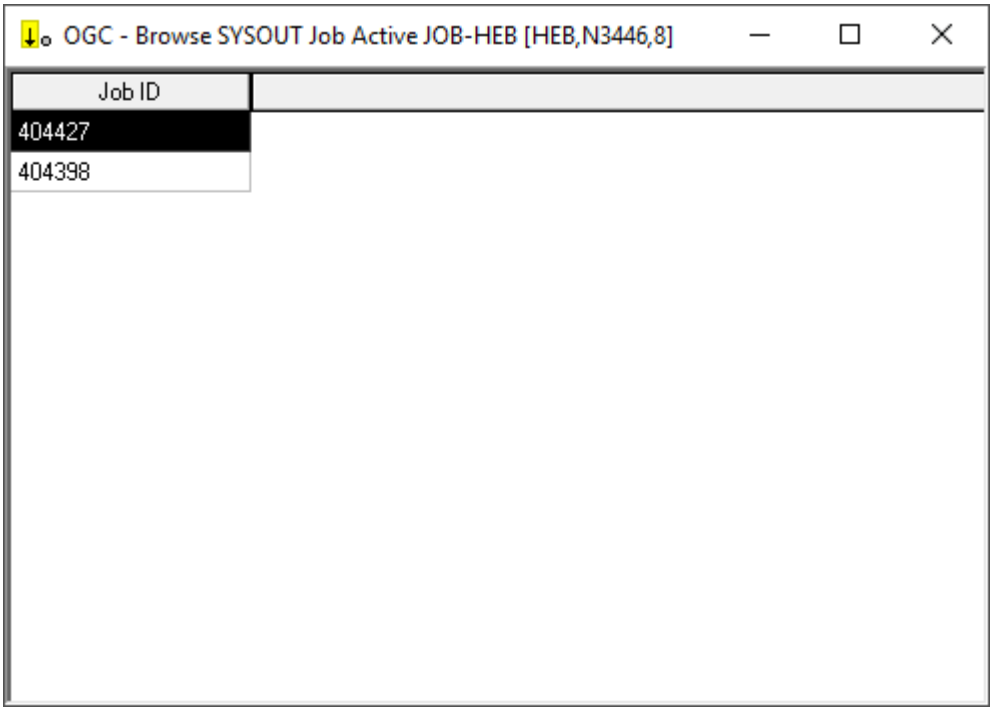

OGC - Browse SYSOUT Job Active PARALLEL-1 [NOPAPI.OGC-RUN-S.1628]
Line
Text
1 %% EOR0301 - PID 32532 Start 20171214-114346 +0100
2 Linux susnat2 2.6.32.12-0.7-default #1 SMP 2010-05-20 11:
3 uid=4051(mut) gid=1023(natural) groups=1023(natural)
4 + /NAT/nathome/natqa5/NOP531/nop_jcl/first.bsh
5 /FS/fs0404/natprod/nop/543/work/natqa5/0010300204/NOPAPI/
6 + set +x
7 %% EOR0306 - User Script Exitcode 0
8 %% EOR0302 - PID 32532 End 20171214-114346 +0100 Exitcode
9 %% EOR0303 - times - Begin
10 0m0.004s 0m0.000s
11 0m0.000s 0m0.000s
12 %% EOR0304 - times - End
13 PID TTY TIME CMD
14 32532 ? 00:00:00 PARALLEL-1.curr
15 + exit 0

On BS2000:

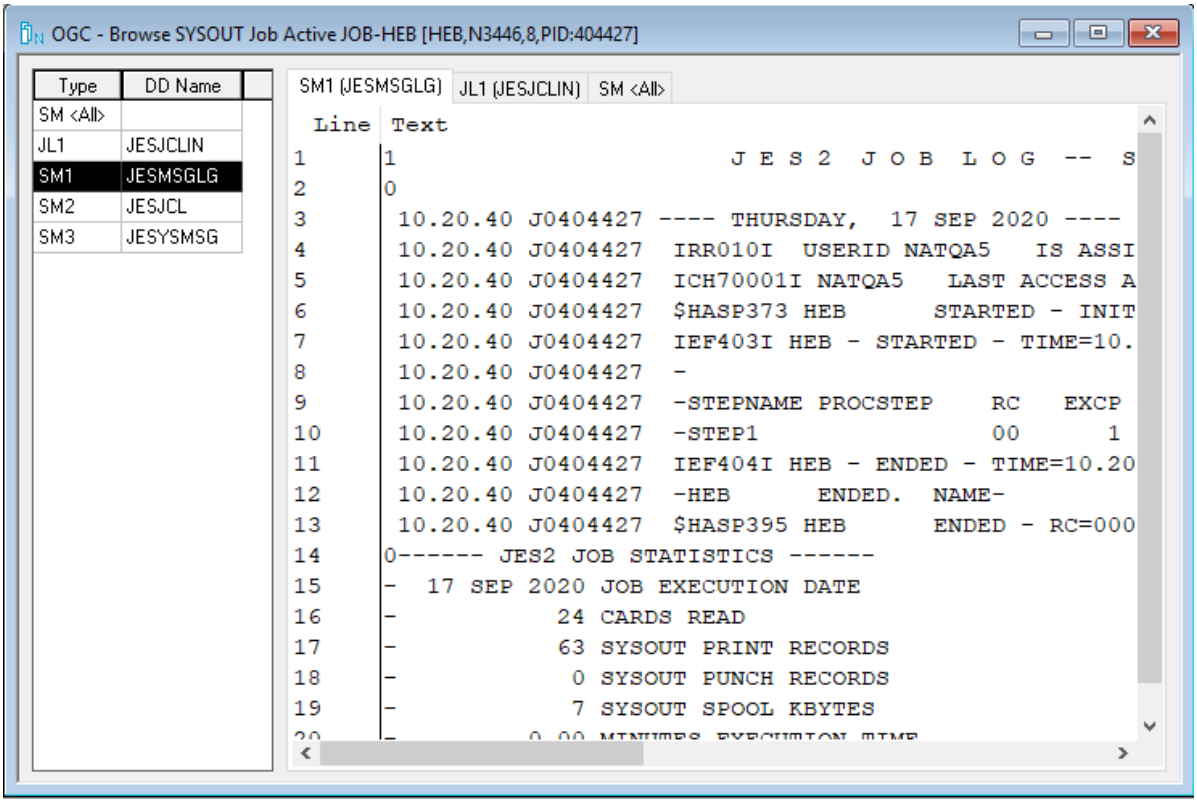


The window displays active SYSOUT produced for the DD statements processed by the selected job (here: `PARALLEL-1`, owner `NOPAPI`, network `OGC-RUN-S`, run number `1628` (UNIX) or `1652` (BS2000). SYSOUT output data is contained in numbered text lines sorted in ascending order.

If more than one SYSOUT file exists, a selection window like the example below opens:


OGC - Browse SYSOUT Job Active JOB-HEB [HEB,N3446,8]
Job ID
404427
404398

On z/OS:


OGC - Browse SYSOUT Job Active JOB-HEB [HEB,N3446,8,PID:404427]
Type
DD Name
SM <All>
JL1
JESJCLIN
SM1
JESMSGLG
SM2
JESJCL
SM3
JESYSMSG
SM1 (JESMSGLG)
JL1 (JESJCLIN)
SM <All>
Line
Text
1 1 J E S 2 J O B L O G -- S
2 0
3 10.20.40 J0404427 ---- THURSDAY, 17 SEP 2020 ----
4 10.20.40 J0404427 IRR010I USERID NATQA5 IS ASSI
5 10.20.40 J0404427 ICH70001I NATQA5 LAST ACCESS A
6 10.20.40 J0404427 $HASP373 HEB STARTED - INIT
7 10.20.40 J0404427 IEF403I HEB - STARTED - TIME=10.
8 10.20.40 J0404427 -
9 10.20.40 J0404427 -STEPNAME PROCSTEP RC EXCP
10 10.20.40 J0404427 -STEP1 00 1
11 10.20.40 J0404427 IEF404I HEB - ENDED - TIME=10.20
12 10.20.40 J0404427 -HEB ENDED. NAME-
13 10.20.40 J0404427 $HASP395 HEB ENDED - RC=000
14 0------ JES2 JOB STATISTICS ------
15 - 17 SEP 2020 JOB EXECUTION DATE
16 - 24 CARDS READ
17 - 63 SYSOUT PRINT RECORDS
18 - 0 SYSOUT PUNCH RECORDS
19 - 7 SYSOUT SPOOL KBYTES

A table in the left window section lists all DD statements produced by the job and the **type of the SYSOUT file**.

For each file you select in this table, a tabbed page opens in the right window section showing the SYSOUT of the corresponding DD statement. Several files of the same file type are numbered sequentially from one (`SM1`, `SM2`, and so on). Files of the same type are combined in an `<All>` collection file, for example, `SM <All>`.

The browse functions available are explained in ***Available Functions: Browse SYSOUT***.

Notes about SYSOUT File Display

The following applies when viewing SYSOUT:

- **Name of the SYSOUT file**
  In BS2000, UNIX and Windows, the SYSOUT file name is indicated in the first line of the SYSOUT text.
- **Automatic Logon to the Operating System Server**
  Before a SYSOUT file is displayed, Entire Operations checks whether the user is allowed to access the SYSOUT file. If necessary, an automatic logon is carried out with the user ID, which is entered for the user as a standard user ID for the operating system server (see *Defining Node Default User IDs* in the *Administration* documentation). If this automatic logon does not function, for BS2000 files, a further attempt will be made with the BS2000 user ID from the fully qualified file name. If this was successful, the current user ID is set accordingly for the operating system node. If the user does not (implicitly or explicitly) log on to the operating system server with the credentials required to access SYSOUT, the SYSOUT file cannot be displayed.
- **Line Length under UNIX and Windows**
  Lines longer than 250 characters will be abbreviated in the SYSOUT display. They will be marked with `>>>` in the columns 247 to 250 of the display.
- **Maximum SYSOUT Size**
  If a maximum SYSOUT size (in MB) is defined and no SYSOUT line limit is defined, the SYSOUT display is rejected with an error message. See **Maximum SYSOUT Size (in MB)** .
- **File Truncation**
  If the SYSOUT file or spool data set exceeds a given line limit, the display will be truncated: see **SYSOUT line limit** in *Defaults for Network Options* in the *Administration* documentation.
- **Number of Previous Executions**
  Entire Operations can display SYSOUT of up to 180 previous executions of the job (e.g., repetitions, cyclic executions).

SYSOUT File Types on z/OS

On z/OS, SYSOUT files are identified by the following output types to distinguish their contents:

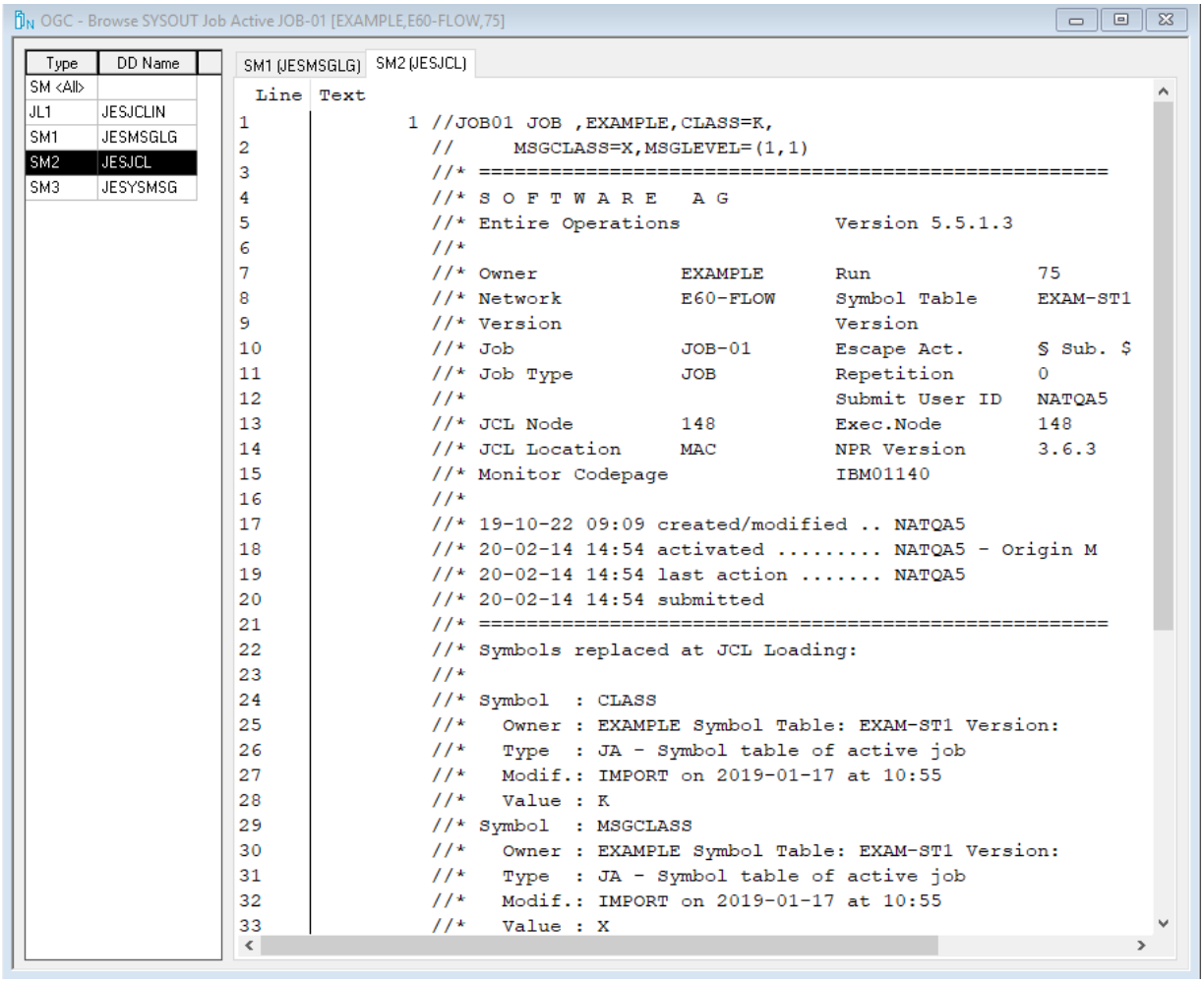
File Type	Purpose	Applies on
CC	Summary of job steps and condition codes.	
JL	JCL of selected job.	
SI	SYSIN data.	
SM	System messages.	
SO	SYSOUT data.	
<All>	Identifies a collection file that combines all files that belong to a certain file type, for example, SM <All>.	

Header and Symbol Information in SYSOUT

You can generate standard header information and symbol replacement information that is inserted as comments into job SYSOUT of submitted JCL if the appropriate Entire Operations default settings are activated:

- For header information, activate the **Generate Header in submitted JCL** option in the *Defaults for Other Settings* (see the *Administration* documentation).

  Header information in job SYSOUT looks like the following example:



Notes:

1. The `Submit User ID` is always shown.
2. The created/modified user ID is modified only if the job definition or JCL was modified. Activation or reactivation does not count as a modification.
3. In BS2000, the LOGON statement is checked. If nothing else was specified at job definition, the LOGON parameters, account-number, job-class, monitoring job variable (with password, if specified) are taken from here. Job priority, run priority and CPU time limit can also be given via LOGON statement.

- For symbol replacement information, activate the **Log Symbol Values in submitted JCL** option in the *Defaults: System/Log Files* (see the *Administration* documentation).

  Symbol replacement information shows replaced symbols and their current values if any symbols were replaced at job submission as shown in the previous example, underneath `Symbols replaced at JCL Loading`.

Available Functions: Browse SYSOUT

The following functions are available from the context menu of the **Browse SYSOUT** window:

Function	Shortcut	Description
Copy	CTRL+C	Copy the text lines selected in the SYSOUT file to the clipboard to be pasted into another file.
Reverse Order	CTRL+R	Display SYSOUT text in descending order of line numbers (last line first). The text is sorted in ascending order by default.
Refresh	F5	See *Refreshing Object Lists*.
Find	CTRL+F	Open the **Find** dialog to enter a search string and specify search options (case sensitive and/or whole word only) and the search direction backward (scrolls up in the source) or forward (scrolls down in the source).
Find Next	F3	Find the next occurrence of the search string specified in the **Find** dialog.
Go To	CTRL+G	Scan the SYSOUT file for a specified line number and highlight this line.
Print	CTRL+P	Output the SYSOUT file on a printer.

44 Maintaining Active Job Conditions

- Available Functions: Condition Active 508
- Listing Active Conditions 509
- Viewing an Active Condition and Changing Its Status 511
- Adding an Active Condition 511
- Using Global Active Conditions 512
- Displaying Conditions for an Active Job 516
- Viewing Job Usage of an Active Condition 516

Logical conditions are defined on the master database when linking jobs within a job network. When a job network is activated and running, Entire Operations sets logical conditions automatically (active conditions) according to the occurrence of system events. Alternatively, you can set active conditions manually. You can display and maintain active conditions.

An active condition reflects the current value of the condition for a given job network activation. It can have the value true (the condition exists) or false (the condition does not exist). The **run number** assigned to the job network at activation is automatically passed to the active conditions defined for the jobs in the network. An active condition is uniquely identified by owner, network, run number and condition name.

Notes:

1. You can define new input conditions for an active job, or delete or modify existing conditions. Any changes to active conditions for active jobs are valid for the current job run only.
2. You can maintain active input conditions in the same way as described for master input conditions in *Defining and Managing Job Conditions* in the section *Job Maintenance*.

Related Topics:

- *Logical Conditions - Concepts and Facilities* documentation

Available Functions: Condition Active

To list all available functions for a Condition Active

- In the object workspace, select a **Condition Active** node and open the context menu.

  The following functions are available:

Function	Shortcut	Description
List	F8	See *Listing Active Conditions*.
New	CTRL+N	See *Adding an Active Condition*.
Refresh	F5	See *Refreshing Object Lists*.
Filter	F3	Use filter criteria to list conditions active: see *Filtering Objects*
Export	---	Opens the **Export Objects** window to export all items of the metanode **Condition Active**: see *Exporting Objects* in the *Import/Export Functions* documentation.

To list all available functions for a Condition Active instance

- Select an active condition from a **Condition Active** node and open the context menu.

The following functions are available:

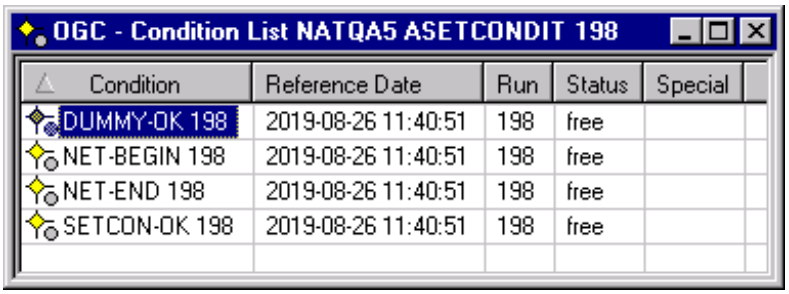
Function	Shortcut	Description
Open	CTRL+O	See *Viewing an Active Condition and Changing Its Status*.
Display	CTRL+D	Display condition active information: see *Viewing an Active Condition and Changing Its Status*.
Active Usage	---	See *Viewing Job Usage of an Active Condition*.
Delete	DELETE	Delete a condition active: see *Deleting Objects - Delete Function*.
Add to Workplan	---	See *Add to Workplan*.

Listing Active Conditions

To list active job conditions

- In the object workspace, select a **Condition Active** node from a **Network Active** node or an **Active Run** instance and choose **List** from the context menu, or press F8.

  A **Condition List** window like the example below opens:



  Depending on the **Condition Active** node selected, the window lists all active conditions of the selected network active or network active run, respectively.

  The example above lists all active conditions of the active job run `198` of the network `ASETCONDIT` which belongs to the owner `NATQA5`.

  The columns contained in the window are explained in *Columns and Fields: Active Conditions*.

To list global active conditions

- See *Listing Global Active Conditions*.

This section covers the following topics:

- Columns and Fields: Active Conditions

Columns and Fields: Active Conditions

The following table explains the columns of the **Condition List** window. The names of the columns correspond to the names of the fields available when adding or viewing an active condition in the **Condition Active or Global Active Condition** window, unless indicated otherwise in the **Field** column of the following table.

Column	Field	Description	
Condition	**Condition**	Active condition name as defined on the master database. **Note:** In the object workspace, an active condition that is being used for **JCL checking only** has the suffix `C` or *`run-number C`* (for example, `TESTCONDITION 2 C`). When adding or modifying a condition: ■ Pay attention to the **restrictions for condition names** described in the section *Job Maintenance*. ■ For global active conditions, use the functions provided for the **Global Active Condition** metanode.	
Reference Date	**Reference date**	Date and time when the condition was set or modified. The date and time serve as search criteria for active conditions, if no run number is specified. See also ***Date and Time Formats***.	
Run	**Run**	Run number of the job which sets this condition, or the value `abs.` (absolute) or `void`. `void` is displayed if a run number is not relevant for a job run. For information on `abs.`, see `ABS` in ***Possible References for Input Conditions***.	
Status	**Status**	Status of the condition. Possible selection options:	
		free	Can be used by any job.
		in use	Can be used by jobs which do not require exclusive usage.
		exclusive	Currently in use by a job; not usable by other jobs.
		destructive	Currently in use by a job; not usable by other jobs. The condition will be deleted after job termination.
Special	**Special Type**	Usage of the condition. Possible selection options for input fields:	
		Normal usage	Condition is not used for checking JCL.

Column	Field	Description	
		Use only for JCL check	Condition is **only used for checking JCL**. If this option is selected, the letter C is shown in this column.

Viewing an Active Condition and Changing Its Status

You can view an active condition and modify its status to change the use made of it by future runs of the job for which it is defined.

To view a condition active and change its status

- In the object workspace, select a **Condition Active** instance and choose **Display** (CTRL+D) or **Open** (CTRL+O) from the context menu.

  If you choose **Open**, you can change the **status** of the condition, if required.

  A **Condition Active** window opens.

  The fields in the window have the same meaning as the corresponding table columns in the **Condition List** window. They are explained in ***Columns and Fields: Active Conditions***.

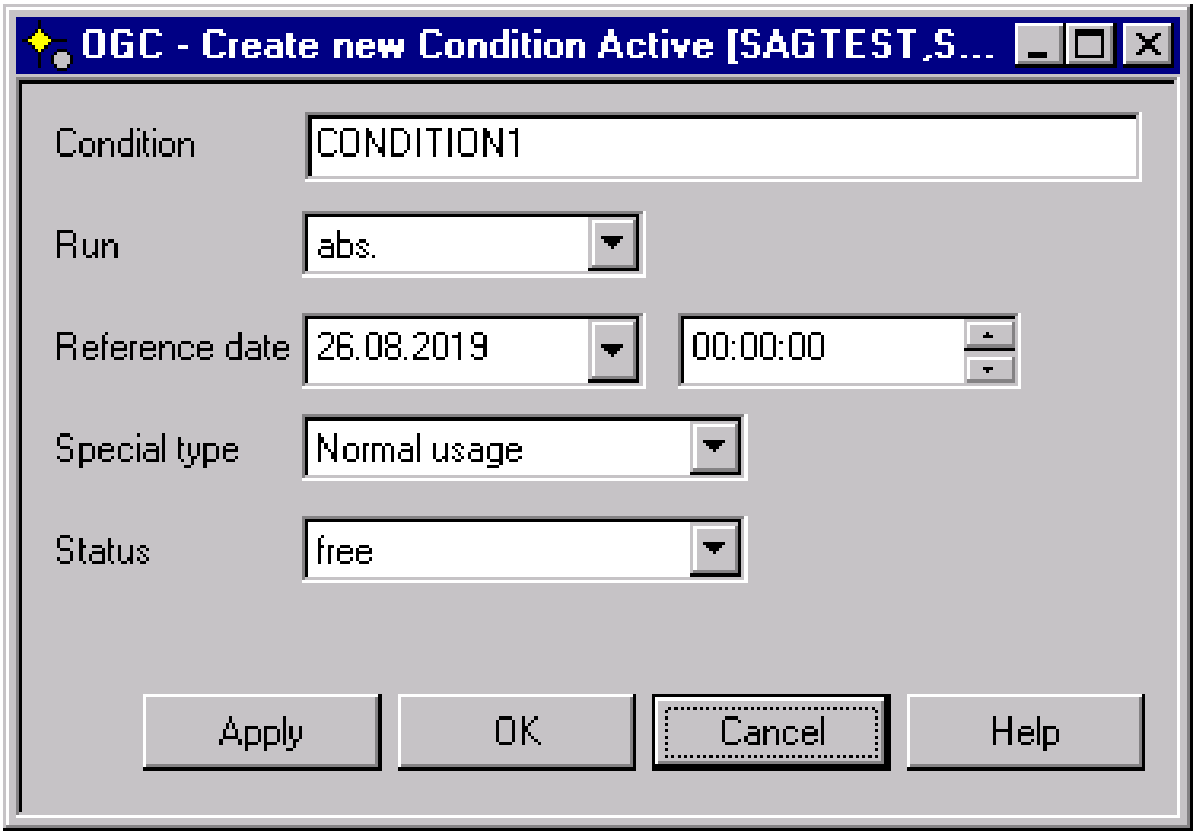
Adding an Active Condition

You can add an active condition to change the job flow by defining it for a job before submission.

To create a new condition active

1. In the object workspace, select a **Condition Active** node.
2. Open the context menu and choose **New**, or press CTRL+N.

   A **Condition Active** window like the example below opens:



The input fields have the same meaning as the corresponding columns in the **Active Conditions** window. They are explained in *Columns and Fields: Active Conditions*.

3 Make your definitions and choose **OK** when finished.

The condition is created.

Using Global Active Conditions

This section covers the following topics

- Available Functions: Global Active Conditions
- Listing Global Active Conditions
- Viewing a Global Active Condition and Modifying Its Status

- Adding a Global Active Condition

Available Functions: Global Active Conditions

To list all available functions for the Global Active Condition metanode

- In the object workspace, select the **Global Active Condition** metanode and open the context menu.

  The following functions are available:

Function	Shortcut	Description
List	F8	See *Listing Global Active Conditions*.
New	CTRL+N	See *Adding a Global Active Condition*.
Refresh	F5	See *Refreshing Object Lists*.
Filter	F3	Specifies selection criteria for listing global active conditions. See *Filtering Objects*.
Export	---	Exports global active conditions. See *Exporting Objects* in the *Import/Export Functions* documentation.
Set Drag And Drop Function	---	See *Drag & Drop*.

To list all available functions for a global active condition

- In the object workspace, select a **Global Active Condition** instance and open the context menu.

  The following functions are available:

Function	Shortcut	Description
Open	CTRL+O	View a global active condition and change its status. See *Viewing a Global Active Condition and Modifying Its Status*.
Display	CTRL+D	View a global active condition (no status change possible). See *Viewing a Global Active Condition and Modifying Its Status*.
Active Usage	---	See *Viewing Job Usage of an Active Condition*.

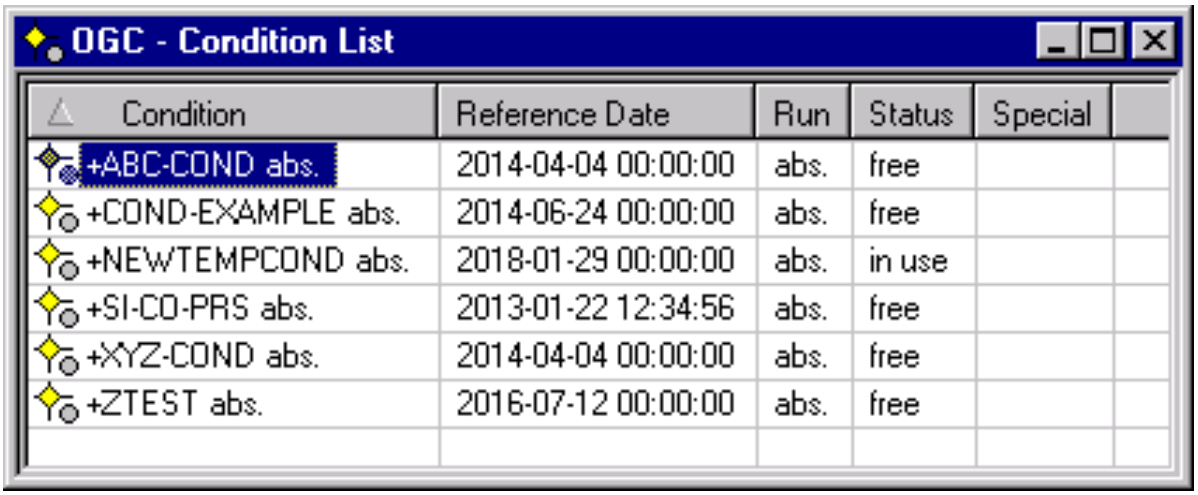
Function	Shortcut	Description
Delete	DELETE	Delete a global active condition. See also *Deleting Objects*.
Add to Workplan	---	See *Add to Workplan*.
Set Drag And Drop Function	---	See *Drag & Drop*.

Listing Global Active Conditions

To list global active job conditions

- In the object workspace, select the **Global Active Condition** metanode and choose **List** from the context menu, or press F8.

  A **Condition List** window like the example below opens:



  This window lists all global active conditions defined in your environment.

  The fields contained in the window correspond to the fields in the **Condition List** window for a single owner. They are explained in *Columns and Fields: Active Conditions*.

Viewing a Global Active Condition and Modifying Its Status

To view a global active condition and change its status

1 In the object workspace, select a **Global Active Condition** instance

2 From the context menu, choose **Display** (CTRL+D) or **Open** (CTRL+O).

If you choose **Open**, you can change the **status** of the condition, if required.

A **Global Active Condition** window opens.

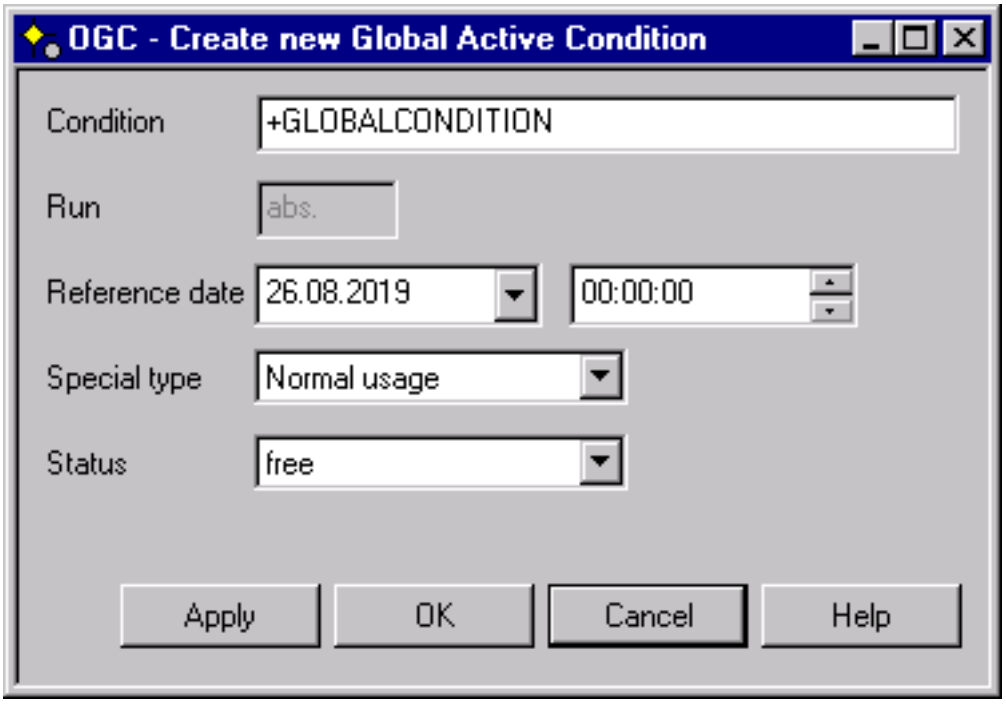
The fields in the window have the same meaning as the corresponding table columns in the **Condition List** window. They are explained in *Columns and Fields: Active Conditions*.

Adding a Global Active Condition

To add a global condition

1 In the object workspace, select the **Global Active Condition** metanode and choose **New** from the context menu, or press CTRL+N.

A **Global Active Condition** window like the example below opens:



The fields contained in the window correspond to the fields in a **Condition Active** window. They are explained in *Columns and Fields: Active Conditions*.

For the rules and restrictions that apply to global active conditions, see *Global Conditions* in the section *Job Maintenance*.

2 Change or add the required values and choose **OK** when you are finished.

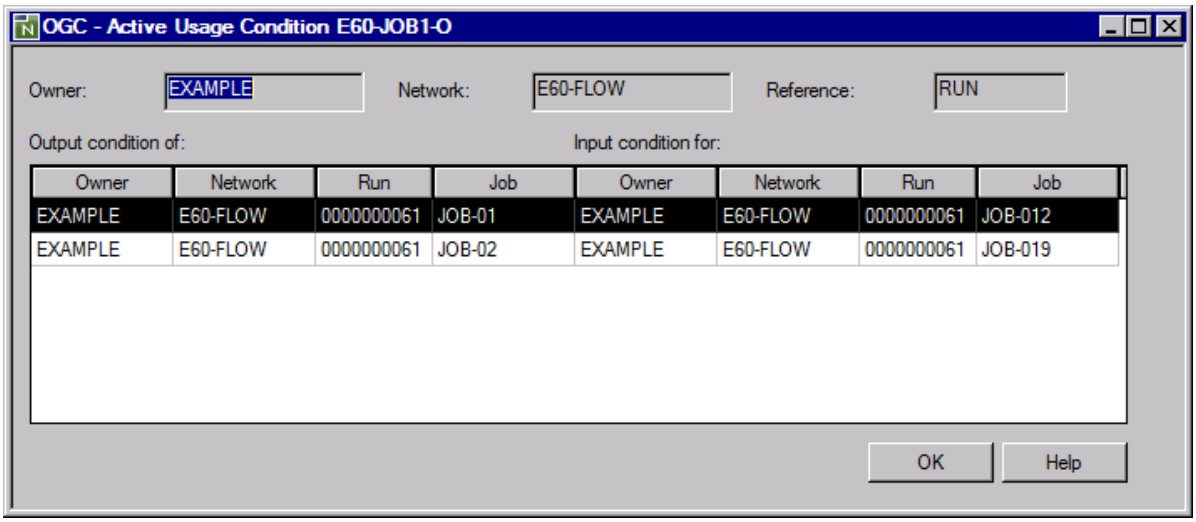
The global condition is created and can be used in further networks.

Displaying Conditions for an Active Job

To view condition usage of an active job

- From the list of items on the **Input Conditions** page of the **Maintenance Job Active** window, select the required condition and choose **Where Used**.

An **Active Usage Condition** window like following example opens:



The columns contained in the window correspond to the columns in the **Where used Condition** window described in *Input Condition Maintenance*.

Viewing Job Usage of an Active Condition

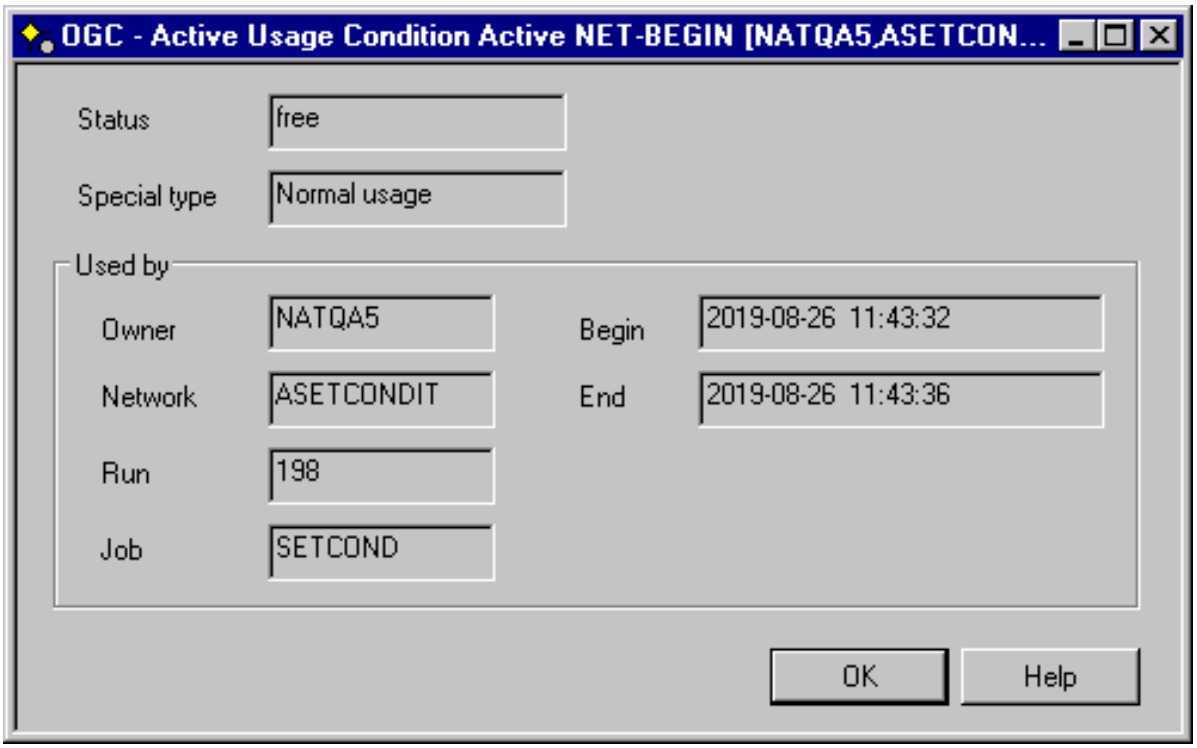
This function is used to display the job that uses or last used an active condition.

To view the job that uses or last used an active condition

1 In the object workspace, select a **Condition Active** instance.

2 Open the context menu and choose **Active Usage**.

An **Active Usage** window like the example below opens:



This window displays the job that currently uses the active condition, or the job that used the condition most recently. If the field **End** is empty, the condition is currently in use. If a date and time is entered, it indicates when the job stopped using the condition.

The **Begin** field shows the date and time since when the job started using the condition.

If several jobs use the condition simultaneously, only one job is displayed.

For explanations of other fields contained in the window, see *Columns and Fields: Active Conditions*.

45 Maintaining Active JCL (Job Control Language)

- Browsing Active JCL 520
- Editing Active JCL 521
- Release Edit Lock 522
- Viewing the Extended Log for Active JCL Changes 523
- Exchanging Active JCL 524
- Regenerating Active JCL 524

Related Topics:

See also the following in the section *Job Maintenance*:

- *Pregenerating Active JCL*
- *Defining and Managing JCL for a Job*
- *Editing Master JCL and Natural Sources*

Browsing Active JCL

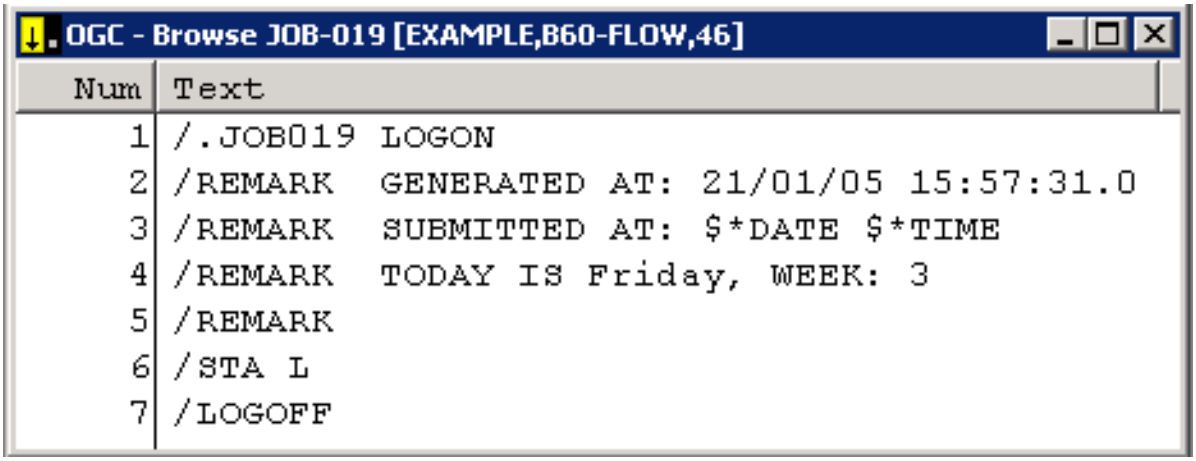
You can view the actual JCL submitted to the operating system for execution. It is produced from the master JCL when the job or network is activated. The symbols are replaced with values from the active symbol table. If it is dynamic JCL, the generation is performed at this time. The active JCL is stored in the active Entire Operations database.

To browse active JCL

1 In the object workspace, select a **Job Active** instance.

2 Invoke the context menu and choose the **Browse Active JCL** function.

The following window appears:



3 Here, you can see the actual JCL submitted to the operating system for execution. It is produced from the Master JCL when the job or network is activated. The symbols are replaced with values from the active symbol table. If it is dynamic JCL, the generation is performed at this time. The active JCL is stored in the active Entire Operations database.

Editing Active JCL

For the current job run, you can edit the active JCL of an active job. Editing active JCL is useful, for example, for correcting JCL errors after job failure.

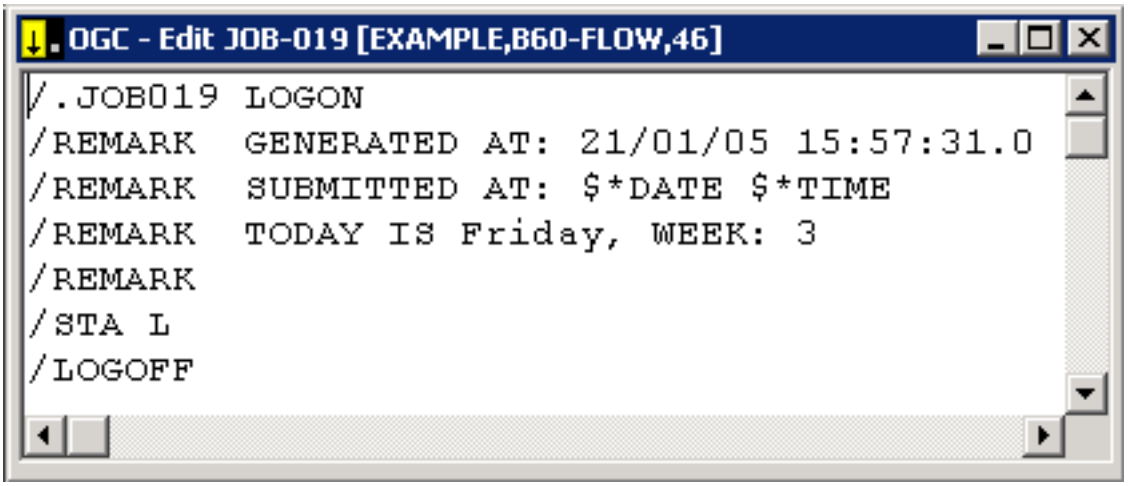
The following applies when editing active JCL:

- While editing active JCL, an implicit logon is triggered to establish the connection to the required node. The logon is not prompted if you have already logged on to the node within this session. For further information on logging on a node, see ***Logon Function*** in the section *Using Entire Operations GUI Client*.
- The source changes for the current job run only. The master JCL definition remains unchanged. If you want to edit the JCL for all runs, you must edit the master JCL.
- For jobs of the type `NAT` (Natural program) with JCL location `NAT` (Natural source), the original Natural source object is edited.
- Any changes to JCL made from the **Job Active** node affect the current job run only.
- Browsing or editing of active JCL is prohibited, while **JCL regeneration** for the same active job is in progress.
- If you modify active JCL before the job is submitted, Entire Operations submits the job automatically according to its defined prerequisites.
- If you modify the JCL after the job terminated and want to resubmit the job with the new JCL, use the **Resubmit** function from the context menu (see ***Resubmitting an Active Job***).

To edit active JCL

1 From a **Job Active** node in the object workspace, select an active job and choose **Edit Active JCL** from the context menu.

An editor window like the example below opens:



The window contains the source of an active JCL.

2 Modify the source as required and confirm the changes when closing the window. See also *Editing Master JCL and Natural Sources* in the section *Job Maintenance*.

Release Edit Lock

This function can only be executed by administrators.

This function removes a lock from active JCL held on the source by a user who previously edited it. The unlock action is archived in the Entire Operations log.

Important: This function is for emergency use only. Data loss can occur when you remove the lock from a source while it is being edited by several concurrent users.

To remove the edit lock

- Select a Job Active instance, invoke the context menue and choose **Release edit lock**.

  A window opens and you are prompted to confirm the release action.

See also *Locking of Natural Sources* in the section Job Maintenance.

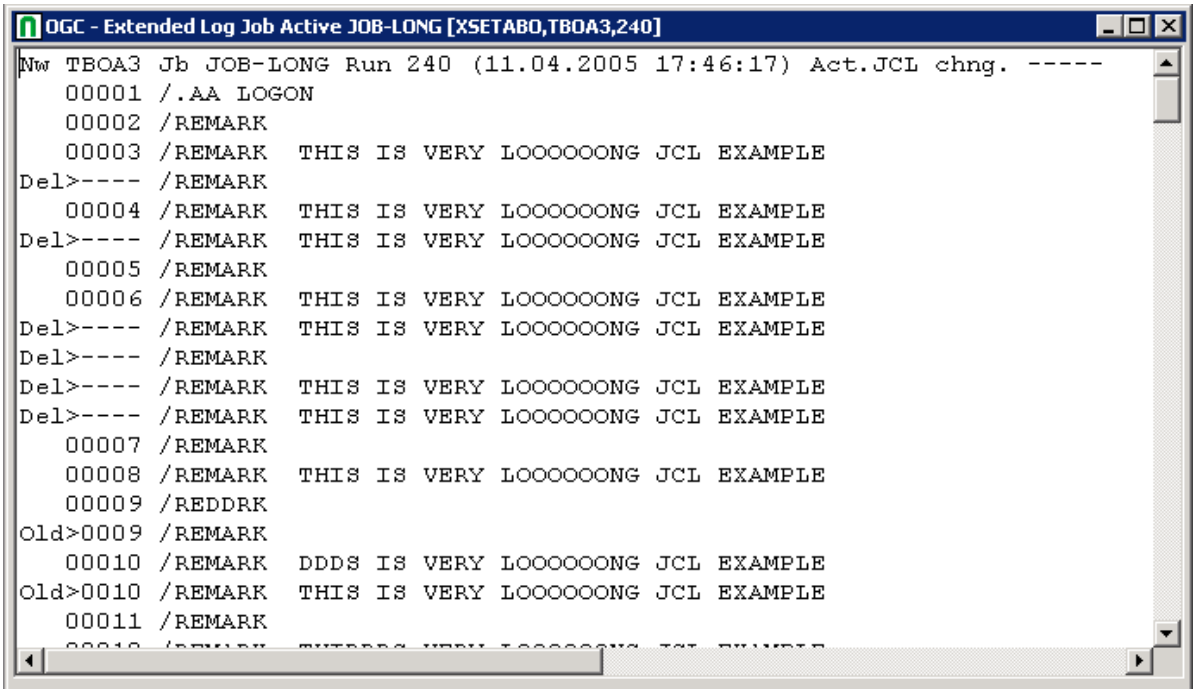
Viewing the Extended Log for Active JCL Changes

To view JCL changes logged for an active job run

1 In the object workspace, select a **Job Active** instance.

2 Invoke the context menu and choose the **Extended Log / Active JCL Changes** function.

The following window appears:



All JCL changes which were made since the last editing process are listed.

For more information on viewing extended logs, see *Displaying Extended Log Information* in the section *Log Information*.

For information on defining extended logs, see *Defining Extended Log Information for a Job* in the section *Job Maintenance*.

See also *Choosing the Job ID of a Job for Logging*.

For information on source editing, see *Editing Master JCL and Natural Sources* in the section *Job Maintenance*.

Exchanging Active JCL

When modifying an active job, you can specify another JCL member and library by overtyping the current values in the corresponding input fields. This allows you to run a different job in place of the old one using the same input conditions, End-of-Job handling, etc. The replacement is valid for the current network run only.

When you have specified a different JCL member and/or library, Entire Operations replaces the old JCL with the new and notifies you of the replacement with a message. The new member can use dynamically generated JCL as described in *Dynamic JCL Generation (JCL Location MAC)*.

Note: After exchanging the JCL, the job must be restarted with the **Resubmit** function. For further information, see *Resubmitting an Active Job*.

Regenerating Active JCL

The JCL for operating system jobs is generated (copied to the active database) when Entire Operations activates a job network. Entire Operations allows you to regenerate the JCL while the job is in the active database. This is useful when you wish to refresh the variables in dynamically generated JCL in MAC-type jobs or when you wish to (re)submit active jobs with their original JCL after editing their JCL for a specific job run.

If **pregenerated JCL** exists, this command uses it for the regeneration. Symbol values are taken in their current status from the active symbol table. If necessary, the active symbol table should be checked beforehand.

This section covers the following topics:

- Symbol Prompting During JCL Regeneration
- Regenerating Active JCL
- Resubmitting the Active Job after JCL Regeneration

Symbol Prompting During JCL Regeneration

If a symbol entry is defined for at least one symbol of the symbol table of the job, the symbol is prompted now and a **Symbol Prompting window** opens. If you choose **Cancel Activation**, prompting is cancelled, and the JCL is not regenerated.

Symbol prompting during the JCL regeneration can be switched on or off by a global switch in the Entire Operations defaults. See the corresponding option setting described in *Defaults for Other Settings* in the *Administration* documentation.

The (re)generation of active JCL is always being performed online, i.e., not by the Entire Operations Monitor. This also applies to MAC-type jobs.

See also ***Symbol Prompting during Network or Job Activation*** in the section *Symbol Table and Symbol Maintenance*.

Regenerating Active JCL

To regenerate active JCL

1 In the object workspace, select a Job Active instance.

2 Invoke the context menu and choose the **Regenerate active JCL** function.

The job is regenerated and ready to execute.

Resubmitting the Active Job after JCL Regeneration

After regeneration of the JCL with the **Re-generate Active JCL** function, the active job is *not* automatically restarted. You must resubmit the job.

In addition, regenerating JCL for a NET type job (re)activates the subnetwork. However, it does not automatically start a subnetwork as a consequence. The subnetwork receives a new run number because of this action.

To start a job (and a subnetwork) after regeneration of the JCL

- Select the job with the regenerated JCL and choose **Resubmit** from the context menu.

See also ***Resubmitting an Active Job of NET type (Subnetwork)***.

VIII Schedule Maintenance

Schedules contain the planned execution dates of job networks. They can contain periodic and/or explicit schedule dates. You can define an unlimited number of schedules, and one schedule can be referenced in different job networks.

This chapter describes general aspects of system-wide schedules and the schedule management functions available.

General Scheduling Considerations

Maintaining a Schedule Master

Related Topics:

- *Calendar Maintenance*
- *Scheduling a Network*
- *Scheduling a Job*
- *Deleting a Scheduling Definition for a Single Network*

46 General Scheduling Considerations

- Use of Schedules 530
- Possible Schedule Definitions 530
- Schedule Extraction Times 532
- Manual and Automatic Activations on the Same Day 532
- Multiple Network Activations 532
- Influence of Deactivations on Schedules 532
- Imported Schedules 533
- Schedule Dependencies across the Turn of the Year 533
- Using Calendars 533

Use of Schedules

A schedule is a predefined time table according to which a job network is activated. Entire Operations monitors schedules to determine which job networks are to be activated.

You can define activation dates in a schedule as explicit dates and/or periodic dates (days of the week, days of the month or a combination of days and months).

Entire Operations can optionally account for holidays (non-working days) in a schedule. For example, if you schedule a job network to run on every first day of a month and the schedule table is based on a calendar in which Saturdays and Sundays are defined as holidays, then Entire Operations does not start the job network if the first day of the month is a Saturday or Sunday. Activation can be postponed until the following workday (working day), Monday. In other words, Entire Operations can automatically interpret the first day of a month as the first workday of a month.

A schedule can be based on a predefined calendar which distinguishes between workdays and holidays (see the section ***Calendar Maintenance***). If a schedule does not depend on a predefined calendar, a calendar of workdays only is assumed.

You can inspect the defined schedule in calendar format, irrespective of whether activation dates are defined as explicit or relative dates: Entire Operations automatically translates relative dates into explicit dates.

You can make the execution of single jobs in a network dependent on their position in the schedule (for example, first schedule day of the week) or in the calendar (for example, last workday of the year).

Possible Schedule Definitions

Schedule definitions are optional. You need not define a schedule for a network. If no schedule is defined and all schedule definitions are left empty, the network is never activated automatically by the Entire Operations Monitor.

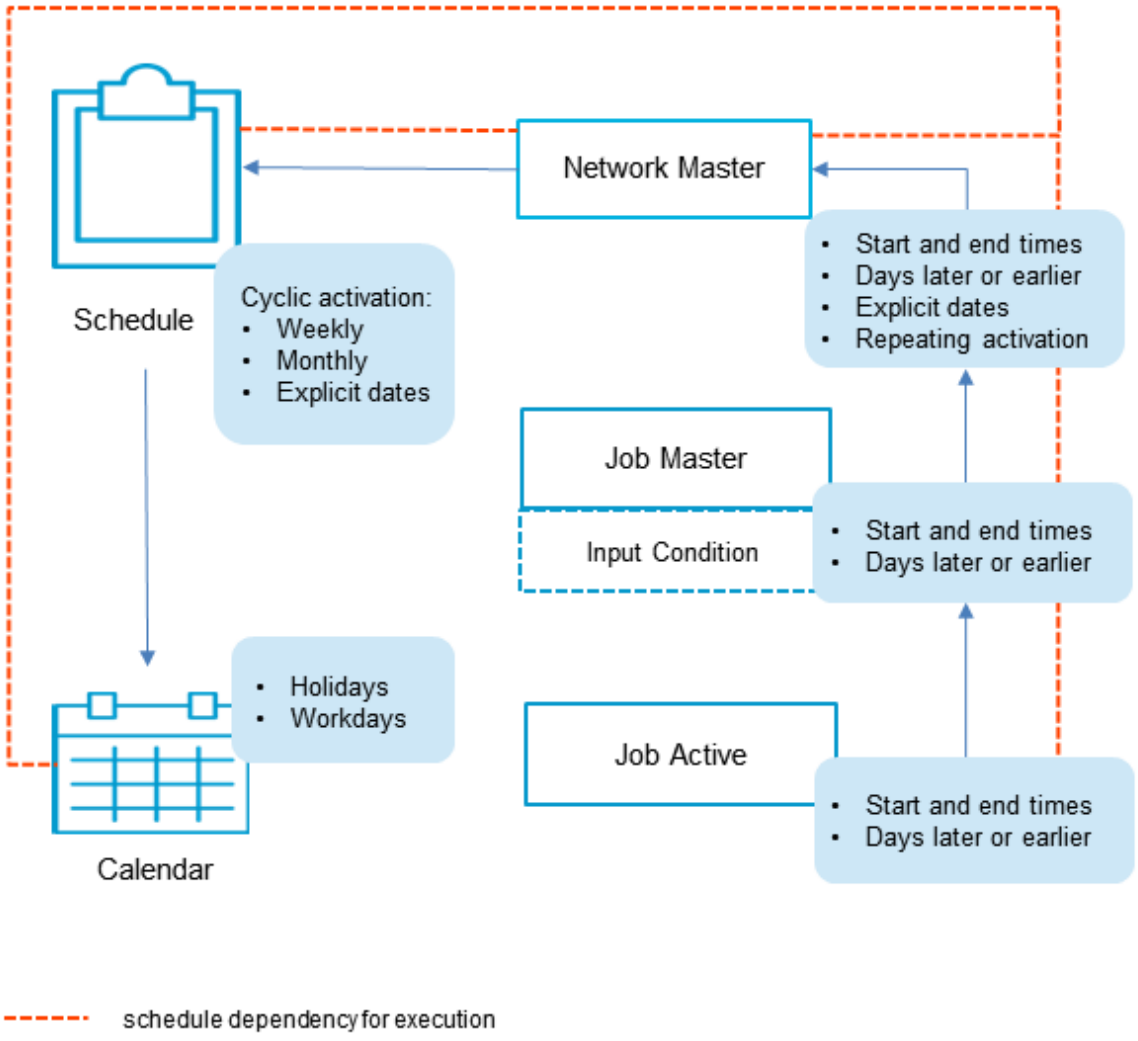
You can use a schedule master that defines daily, weekly or monthly activation cycles. This schedule can be linked to all networks that belong to the schedule owner.

In addition, you can specify activation days and cycles for individual jobs, and time frames with start and end time definitions for individual jobs and networks.

Network schedules can be combined with a schedule master.

In general, job schedules override network schedules, and network schedules override schedule master definitions.

The following graphic is an overview of scheduling options:



All options indicated in the overview are explained in the remainder of this chapter.

Schedule Extraction Times

All network schedules are checked by the Monitor at least once a day, usually at midnight or after the first start of the Monitor on a new day. The networks with a schedule entry for the current day are activated. If no **Earliest Start Time** is defined at the network or job level, execution starts immediately after activation. New or modified networks or jobs with a schedule entry for the current day are activated also if the activation time is not reached already.

The extraction date is always used for the job time frame calculation, even if the network is activated some time later than planned, because the Monitor was down.

For more information, see *Processing of Time Frame Definitions*.

Manual and Automatic Activations on the Same Day

A scheduled activation will not be rejected if a manual or API activation of the same network has already been made for the same day.

Multiple Network Activations

You can define multiple network activations in a network schedule.

For detailed information, see *Defining Multiple Network Activations*.

Influence of Deactivations on Schedules

The day of the activation is removed during a deactivation from the schedule only if the planned activation was originated by a schedule extraction by the Monitor. The day of the activation is not removed from the schedule if a manual activation is deactivated.

Imported Schedules

If a schedule is imported by using import/export functions, a check for the current day in the schedule will be performed. If the current day is a schedule day, it will be excluded explicitly from the schedule automatically. If you want make the schedule active for the current day, you must remove this explicit exclusion definition from the schedule.

Reason: The exclusion of the current day during import is done to prevent an unwanted automatic start of an imported job network by the Entire Operations Monitor.

Schedule Dependencies across the Turn of the Year

Schedule dependencies can also be defined across the turn of the year. This applies in particular to the week, which begins in the old year and ends in the new year.

This functions only if the schedules and calendars used are defined for both years.

Using Calendars

You need not use calendars when defining a schedule. If no calendar is specified, all days are treated as workdays.

You may use calendars, which belong to the owner of the network, and calendars of other owners.

If a calendar is specified, a schedule definition date is used only if it is a calendar workday. Networks are not activated on holidays.

For more information on calendars, see the section *Calendar Maintenance*.

Notes:

1. The Monitor searches for the calendar under the network owner. If it does not find the calendar, the Monitor searches for it system-wide under SYSDBA.
2. Calendars are year-dependent. If a calendar cannot be found for the current year, the network is not scheduled and an error message is issued. Be sure to define calendars for the coming year before they are needed. From the month of November of the past year onwards, the Entire Operations Monitor issues warning messages to the log.
3. Calendars are linked to schedules, not to networks.

4. A calendar modification causes the automatic recalculation of all linked schedules, and a recalculation of the current schedules of all networks linked to these schedules. This automatic recalculation is performed by the Entire Operations Monitor in the background. Be aware that this may have influence on many networks, and may take some time. To check which networks were affected, check the Entire Operations log.

47 Maintaining a Schedule Master

- Listing Schedules 536
- Displaying, Adding or Modifying a Schedule 537
- Deleting a Schedule 548

Listing Schedules

To list all schedule definitions of an owner

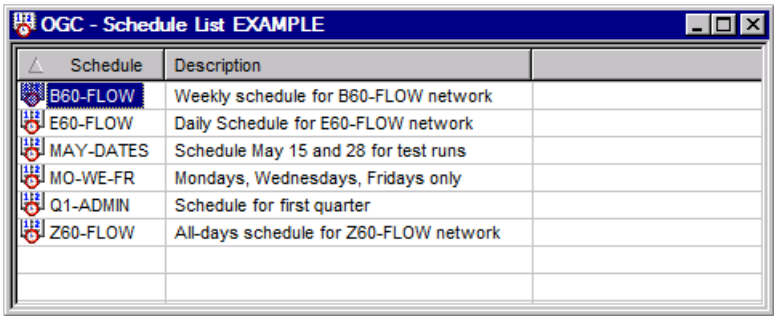
1 In the object workspace, select the **Schedule** node of an owner.

2 From the context menu, choose **List**.

Or:

Press F8.

All schedules defined for the selected owner (here: EXAMPLE) are listed in the **Schedule List** window as shown in the following example:



If one or more job schedules are defined for the owner, they are listed in this window.

This section covers the following topics:

- Fields: Schedule List Window

Fields: Schedule List Window

The following fields are available in the **Schedule List window**:

Fields	Description
Schedule	User-defined schedule name.
Description	Short description of the schedule.

Displaying, Adding or Modifying a Schedule

This section covers the following topics:

- Available Functions: Schedule Maintenance
- Viewing a Schedule Definition
- Viewing a Schedule Definition as a Calendar
- Adding a Schedule Definition
- Modifying a Schedule Definition
- Fields: Schedule Definition
- Available Functions: Schedule Definition
- Using Day Lists

Available Functions: Schedule Maintenance

To list all functions available for a schedule node

- In the object workspace, select the **Schedule** node from an **Owner** instance and open the context menu.

  The following functions are available for a selected **Schedule** node:

Function	Description
List	List all schedules for an owner in the **Schedule List window**.
New	Add a schedule definition: see *Adding a Schedule Definition*.
Refresh	Refresh the list of schedules: see *Refreshing Object Lists*.
Filter	Use selection criteria to list schedules: see *Filtering Objects*.
Paste data	Paste a schedule: see *Pasting Objects*.
Export	Open the **Export Objects** window to export all items of the **Schedule** metanode: see *Exporting Objects* in the *Import/Export Functions* documentation.
Set Drag and Drop Function	Drag nodes and drop them in the empty working area: see *Drag & Drop*.

To list all functions available for a schedule

- From the **Schedule** node, select an instance and open the context menu.

  The following functions are available for a selected schedule:

Function	Description
Open	Modify the schedule: see *Modifying a Schedule Definition*.
Display	Display the schedule definition: see *Viewing a Schedule Definition*.
Display as a Calendar	Display the calendar view of the schedule definition. See *Viewing a Schedule Definition as a Calendar*.
Where used	List using networks.
Delete	Delete the schedule. **Note:** You cannot delete a schedule, which is defined (used) in at least one network. To find out which networks are using the schedule, choose the **Where used** function.
Copy data	Copy a schedule definition: see *Copying Objects*.
Export	Open the **Export Objects** window to export the schedule: see *Exporting Objects* in the *Import/Export Functions* documentation.
Add to Workplan	Store a reference to the schedule in a list of activities to be done: see *Add to Workplan*.
Set Drag And Drop Function	See *Drag & Drop*.

Viewing a Schedule Definition

To view a schedule definition

- In the object workspace, select a **Schedule** instance and choose **Display** from the context menu, or press CTRL+D.

  A **Display Schedule** window opens showing the current date settings of the selected schedule.

  The protected fields and tabbed pages in the window correspond to the input fields and tabbed pages of the **Maintenance Schedule window**. They are explained in *Fields: Schedule Definition*.

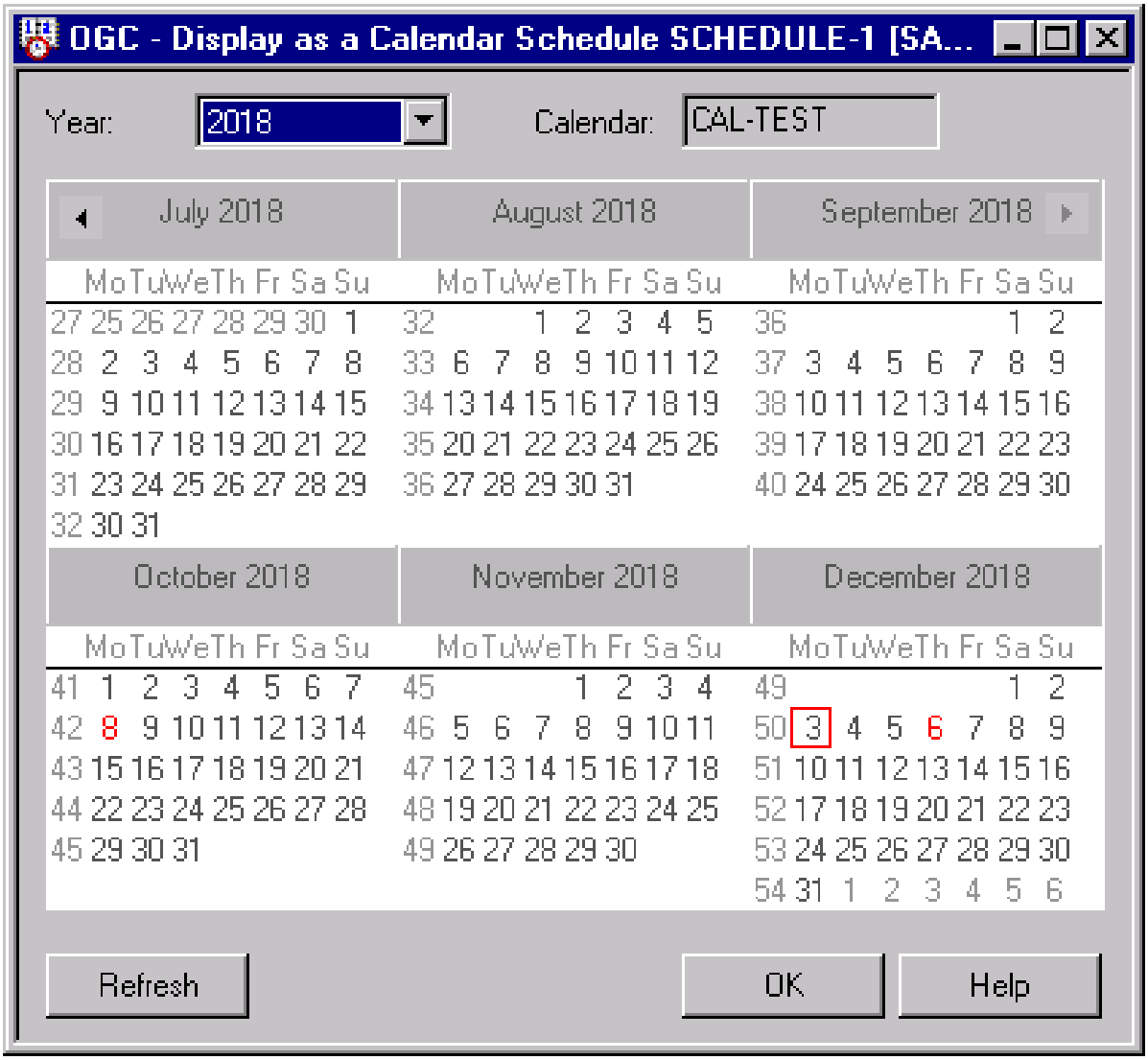
Viewing a Schedule Definition as a Calendar

If a calendar is assigned with a schedule, you can view the schedule definition as a calendar.

To display the calendar view of a schedule

1 In the object workspace, select a **Schedule** instance and choose **Display as a Calendar** from the context menu.

  A **Display as a Calendar** window like the example below opens:



The current six months defined for the schedule are shown by default. You can view additional months or all month of the year by drawing the window borders to resize the window as required.

The current date is indicated by a red square, defined dates are highlighted in red.

You can choose **Refresh** to update the calendar view.

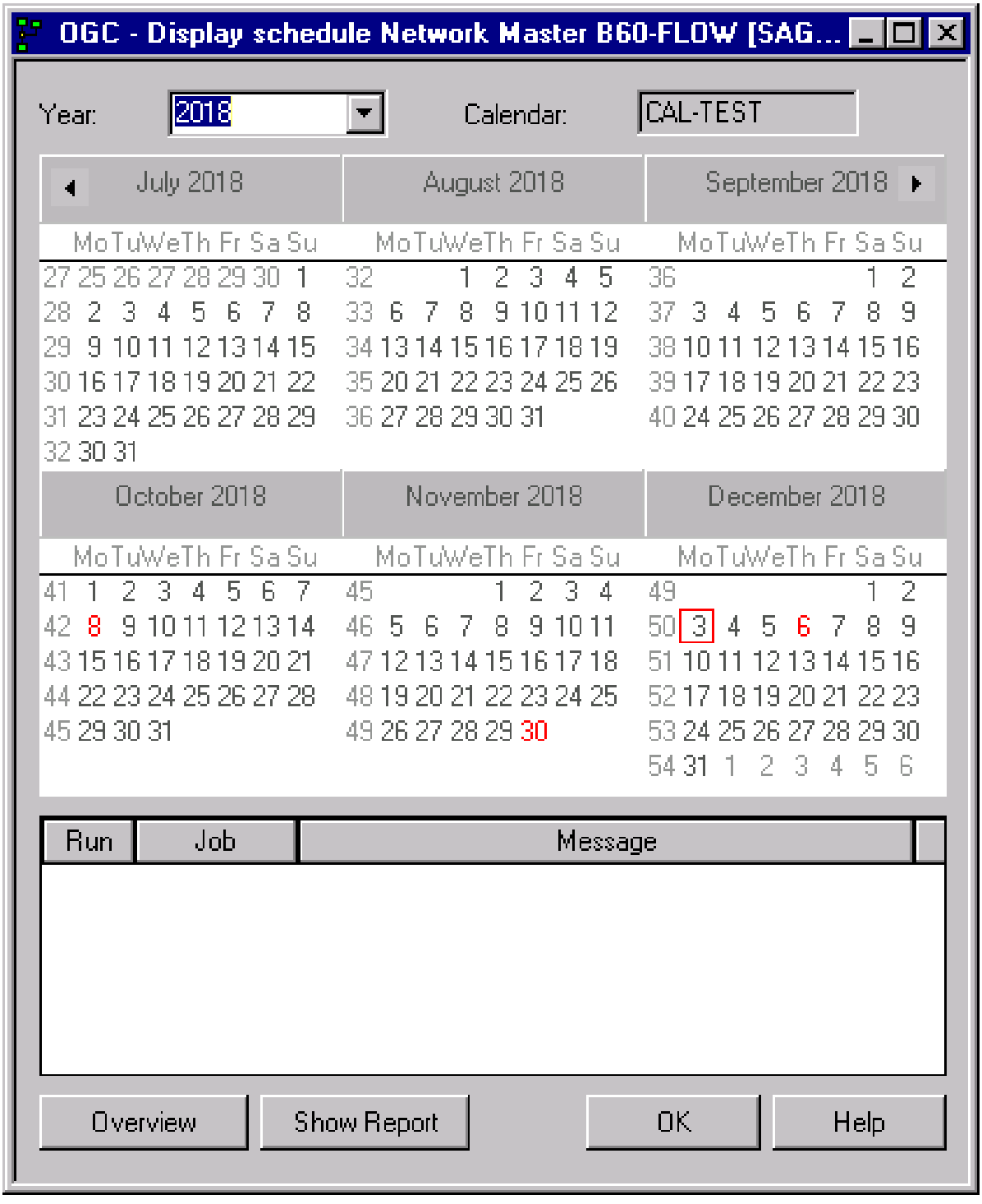
2 If you select a year in the **Year** box that is not defined in the referenced calendar, an EOR3014 error can occur with a corresponding message.

To display the calendar view of a network schedule

1 In the object workspace, select a **Network Master** instance.

2 Invoke the context menu and choose the **Display Schedule** function.

A **Display Schedule** window like the example below opens:



3 Select the year you want to display.

Dates that are marked in red indicate that a schedule is provided.

You can view an overview of network starts by proceeding as described in *Producing a Network Start Summary*.

Adding a Schedule Definition

To add a schedule definition

1 In the object workspace, select a **Schedule** node.

2 From the context menu, choose **New**, or press CTRL+N.

A **Create new Schedule** window opens.

The fields and tabbed pages in the window correspond to the fields and tabbed pages of the **Maintenance Schedule window**.

3 Fill in the fields you want to define. The fields and commands available in the window are explained in *Fields: Schedule Definition* and *Available Functions: Create/Maintenance Schedule*, respectively.

4 Choose **OK** when you are finished.

The schedule is saved and you can use it for networks and jobs.

Modifying a Schedule Definition

To modify a schedule

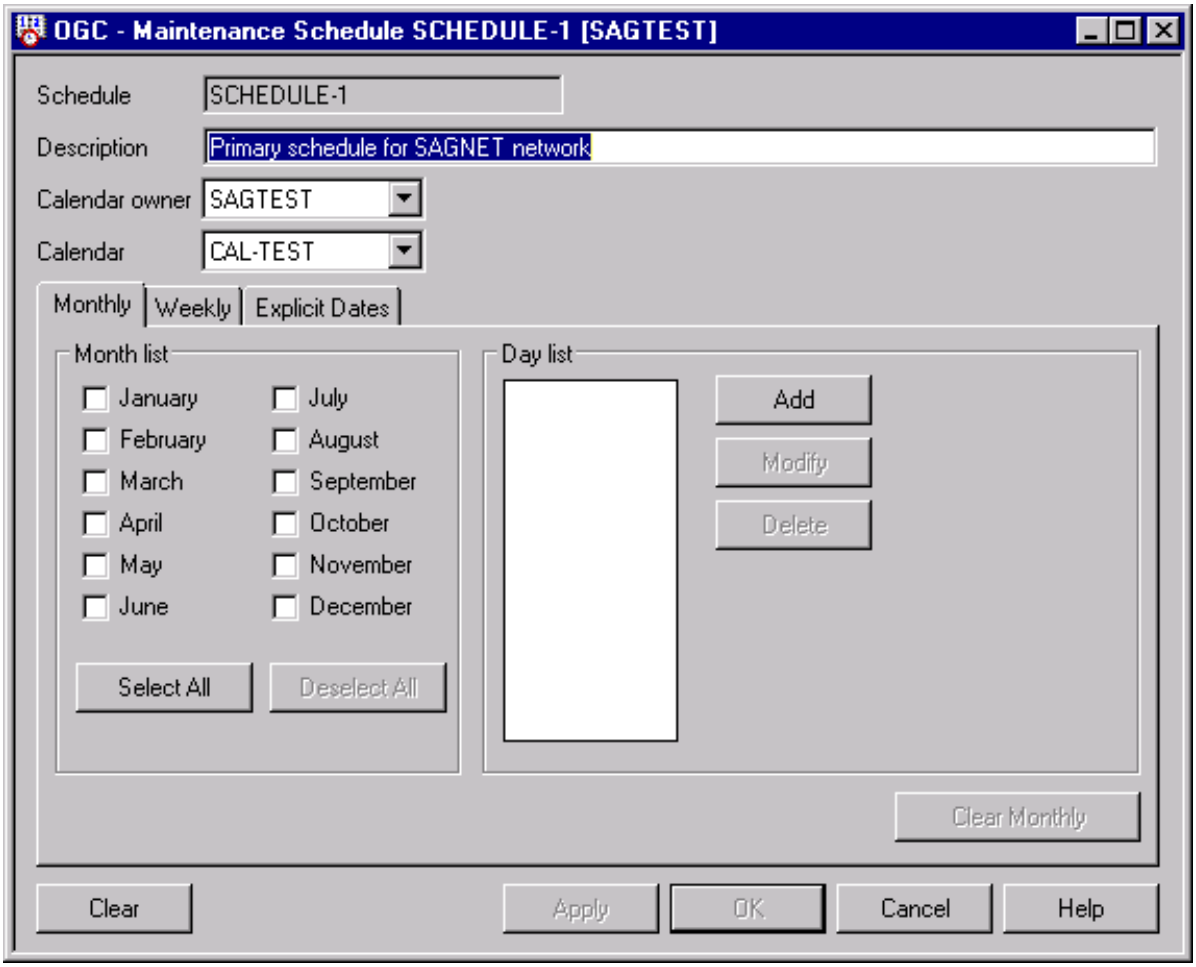
1 In the object workspace, select a schedule from the **Schedule** metanode.

2 From the context menu, choose **Open**.

Or:

Press CTRL+O.

A **Maintenance Schedule** window like the example below opens:



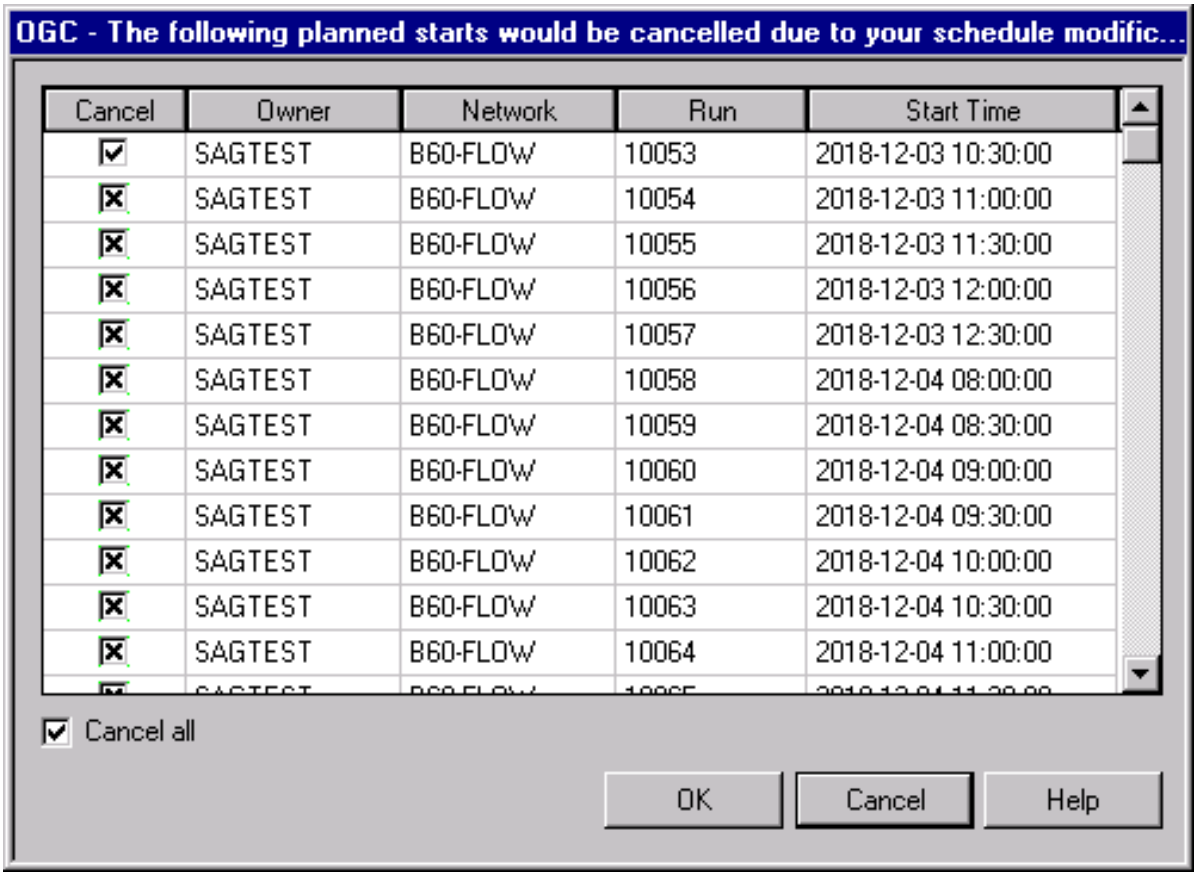
3 Make your definitions. The fields and commands available in the window are explained in *Fields: Schedule Definition* and *Available Functions: Create/Maintenance Schedule*, respectively.

4 Choose **OK** to save the definitions.

Or:

Choose **Cancel** if you want to keep the previous definitions.

5 If your modification of a schedule affects any planned starts, a window prompts you to cancel the starts or to keep them active.



6 Select the **Cancel all** check box if you want to cancel all starts.

Or:

Select the **Cancel** check box next to the listed start(s) you want to cancel.

Or:

Choose **Cancel** to keep all listed starts active, regardless of your schedule modifications.

Fields: Schedule Definition

The fields and tabbed pages provided in the **Create new Schedule** or **Display/Maintenance Schedule window** are explained in the following table. Each tab within this window opens a separate schedule definition time frame (monthly, weekly and explicit dates) to make your schedule the most precise.

Field/Tabbed Page	Description
Owner	Owner of the schedule in the window title
Schedule	Name of the schedule
Description	Description of the schedule
Calendar owner	Owner of the assigned calendar Default: the schedule owner You can select a name from the drop-down list box.
Calendar	Assigned calendar You can select a name from the drop-down list box.
Monthly / Month list	List of months for the schedule They always correlate with **Day list**.
Weekly / Month list	A list of months for the schedule. They always correlate with **Day list**.
Day list	Define the days to be used for the monthly and/or weekly schedule. The days specified in this list correlate with the list of months selected on the tabbed pages **Weekly** and/or **Monthly**. See ***Using Day Lists*** for the options provided to define days.
Explicit Dates	Define single or multiple dates for executing or not executing a network. **Note:** 1. Explicit dates of a schedule list have a higher priority than all periodic definitions. 2. There is another network-specific list of explicit dates, which overrides all definitions made on the schedule level. See also ***Defining Dates for Explicit Network Execution***.

Available Functions: Schedule Definition

The functions provided in the **Create new Schedule** or **Maintenance Schedule** window are explained in the following table.

Function	Description
Select All	Select all months in the current month list.
Deselect All	Deselect all months marked in the current month list.
Add/Modify (Day list)	Open the **Maintenance Monthly** or **Maintenance Weekly** window where you can select a day definition for the **Day list**.

Function	Description
Add/Modify (Explicit Dates)	Add a date or modifies a selected date. See also *Defining Dates for Explicit Network Execution*.
Delete (Day list)	Delete all entries in the **Day list**.
Delete (Explicit Dates)	Delete the selected date. See also *Defining Dates for Explicit Network Execution*.
Clear Weekly	Clear the **Month list** on the tabbed page **Weekly**.
Clear Monthly	Clear the **Month list** on the tabbed page **Monthly**.
Clear	Clear all existing definitions.
Apply	Apply changes and save the schedule.
OK	Save the schedule.
Cancel	Cancel all pending changes.

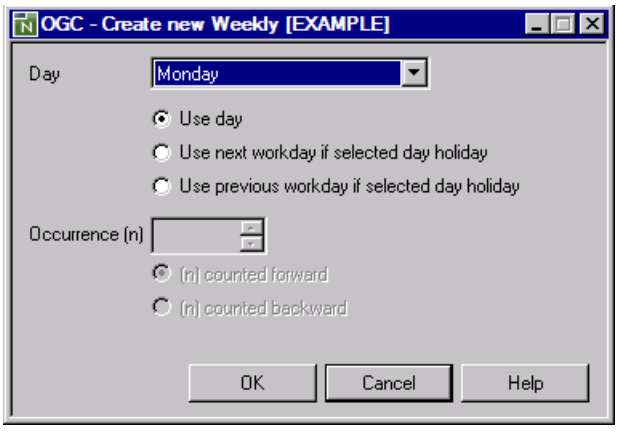
Using Day Lists

The **Day list** is used to define days for a monthly and/or a weekly schedule.

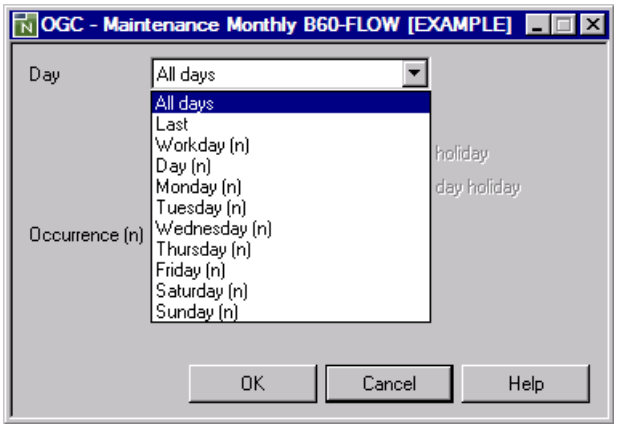
To add or modify a day definition

1 In the **Day list** section of the **Create new Schedule** or **Maintenance Schedule** window, choose **Add** or **Modify**.

A **Monthly** or **Weekly** window like the example below opens:



2 Open the drop-down list box to select the required definition:



The day definitions available for selection depend on whether you use the **Day list** on the tabbed page **Weekly** or **Monthly**. For details, see *Possible Day Definitions*.

3 Select a radio button if you want to define special conditions for a day:

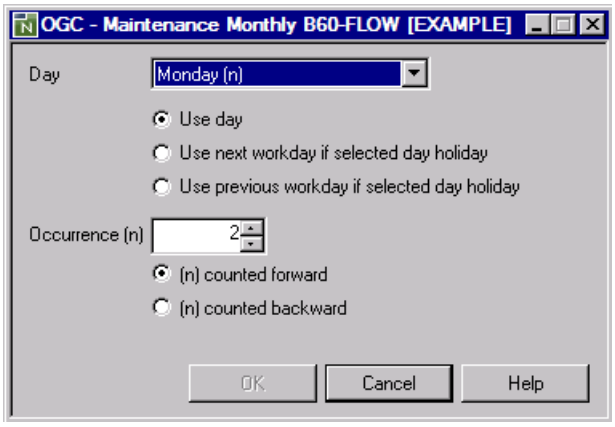
For a selected day (in the **Weekly example**: **Monday**) you can specify whether to use the next or previous workday if this day is a holiday.

Or:

For a monthly schedule, you can also select a day **(n)** to determine which occurrence of the weekday to use; enter a valid number in the **Occurrence (n)** box:

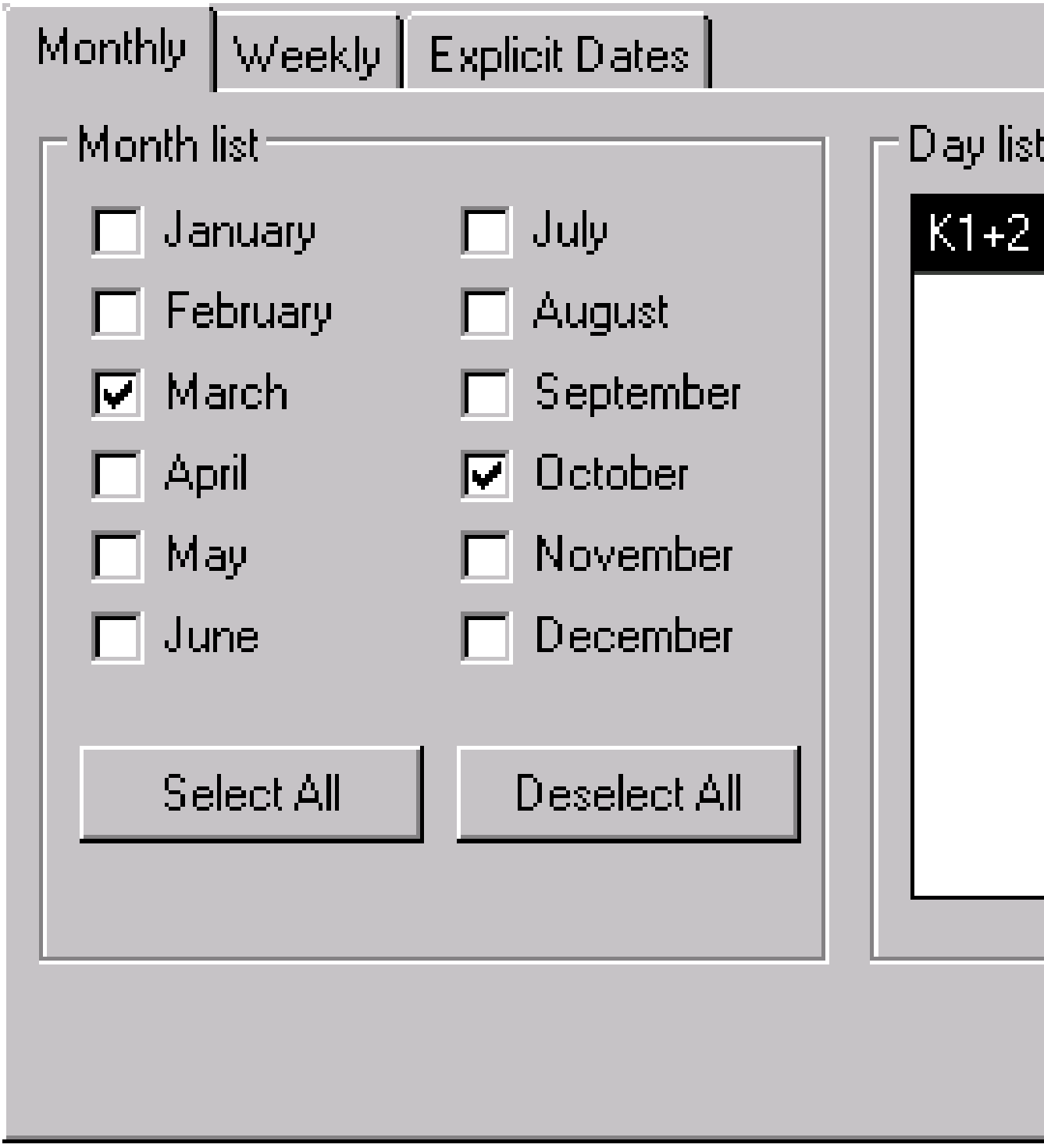
- **(n) counted forward**: counts *n* times forward in the given time range.
- **(n) counted backward**: counts *n* times backward in the given time range.

In the example below, 2 determines that the second Monday of the month is used:



4 Select the required definition and choose **OK**.

The **Day list** now contains the code Entire Operations automatically generates for the day definition you made:



In the example above, the **K1+2** code indicates that you defined the second Monday for the months of March and October.

Possible Day Definitions

The day definitions available for selection depend on whether you use the **Day list** on the tabbed page **Weekly** or **Monthly**. Remember that the month(s) selected on these pages always correlate with **Day list**. Each combination of month and day is a schedule date.

The following table lists possible selection options for day definitions and provides examples of codes Entire Operations generates for the defined day into the **Day list** of a **Weekly** or **Monthly** schedule:

Day	Example Weekly	Example Monthly
Weekday (Monday through Sunday)	**1** is Monday and **2** is Tuesday.	**K1** is each Monday, **K2** each Tuesday of the month.
Workday (day defined in the calendar):		
Next workday if the specified day is a holiday	**1A** is the next workday after Monday.	**K1A** is the next workday after a Monday in a month.
Previous workday if the specified day is a holiday	**1B** is the workday before Monday.	**K1B** is the workday before a Monday in the month.
Workday (*n*), counted forward (default setting) in the given time range	**1W** is the first workday of the week.	**01W** is the first workday of the month.
Workday (*n*), counted backward in the given time range	**1V** is the last workday of the week.	**01V** is the last workday of the month.
All days of the month	n/a	**AL** is all days of the month.
Day (*n*)	n/a	**01** is the first day of the month.
Last day of the month	n/a	**LD** is the last day of the month.

For further information on the codes generated for a day definition and weekday dependency within a month, refer to the corresponding *Schedule Maintenance* section for mainframes and UNIX.

Deleting a Schedule

To delete a schedule definition

1 From the **Schedule** node in the object workspace, select the schedule you want to delete and choose **Delete** from the context menu.

A confirmation window opens.

2 Choose **Yes** to confirm the deletion (**No** cancels the action).

The schedule definition is deleted.



Note: You cannot delete a schedule, which is defined (used) in at least one network. Choose **Where used** from the context menu of the selected schedule to find out which networks are using the schedule. See also *Available Functions: Schedule Maintenance*.

IX Calendar Maintenance

48 Calendar Maintenance

- General Rules and Restrictions 554
- Available Functions: Calendar 555
- Listing Calendars 556
- Displaying, Modifying or Adding a Calendar Definition 557
- Defining Workdays and Holidays 559
- Where Used - Listing Schedules Using a Calendar 560
- Deleting a Calendar Definition 561

Calendars can form the basis for schedules defined for jobs and job networks. An Entire Operations calendar distinguishes between workdays (working days) and holidays (non-working days) as defined by the user (weekends, national holidays, personal vacations).

If a schedule depends on a calendar, calendar-dependent definitions can be made in the schedule. If a schedule is not linked to a calendar, the schedule assumes that all days are workdays.

Calendars are identified by owner, name and year, and can belong to an owner or be used system-wide. You can specify a system calendar or a calendar belonging to your owner for a schedule table, but you can only modify calendars belonging to your owner. System calendars can be modified by authorized users only.

Any number of calendars can be defined to the system.

Calendars can be modified to change, include or delete holidays (system-wide calendars can only be modified by the system administrator). Modifications to calendars can affect the associated job network schedule(s).

General Rules and Restrictions

This section covers the following topics:

- How Calendars Work
- Calendar Types
- Calendar Names

How Calendars Work

Entire Operations calendars include workdays and holidays (non-workdays).

Entire Operations does not activate any networks on a day defined as a holiday.

For example, a job network is scheduled to run on every first day of a month and the schedule is based on a calendar in which Saturdays and Sundays are defined as non-workdays:

If the first day of a month is a Saturday or Sunday, Entire Operations does not start the network on these days, but either delays activation until the next workday following the holidays or brings activation forward to the last workday previous to the holidays, depending on the schedule definition.

Calendar Types

There are two types of calendars in Entire Operations:

- **System-wide calendars**
  System calendars can be modified by authorized users only.
- **Owner-specific calendars**
  When you define a calendar, it is automatically associated with your owner. You can modify only those calendars that belong to your owner.

You can specify a calendar belonging to your owner for a schedule, but you can also select a system calendar.

Calendar Names

A calendar can be valid for several years, so you are not forced to change the name of the calendar at the end of the year. Just define the calendar for the next year. For this reason, you should use calendar names independent of the year.

You must define a calendar separately for each year you want to use it. Networks cannot be started if a calendar is missing for the year for which it is needed. To be notified about such cases, select the Calendar undefined for year option in Global Messages for Events (see the *Administration* documentation). In November and December, the Entire Operations Monitor writes warning messages to the log if a calendar currently used is not defined for the next year.

Reserved Calendar Names

Calendar names must not begin with the prefix `=EOR=`.

This prefix is reserved for **special usage in symbol functions** (see the section *Symbol Table and Symbol Maintenance*).

Available Functions: Calendar

To list all functions available for a Calendar node

- In the object workspace, select a **Calendar** node and open the context menu.

  The following functions are available:

Function	Shortcut	Description
List	F8	See *Listing Calendars*.
New	CTRL+N	See *Displaying, Modifying or Adding a Calendar Definition*.
Refresh	F5	See *Refreshing Object Lists*.
Filter	F3	See *Filtering Objects*.
Paste data	CTRL+V	See *Pasting Objects*.
Export	---	See *Exporting Objects* in the *Import/Export Functions* documentation.
Set Drag And Drop Function	---	See *Drag & Drop*.

To list all available functions for a Calendar instance

- In the object workspace, select a **Calendar** instance and open the context menu.

  The following functions are available:

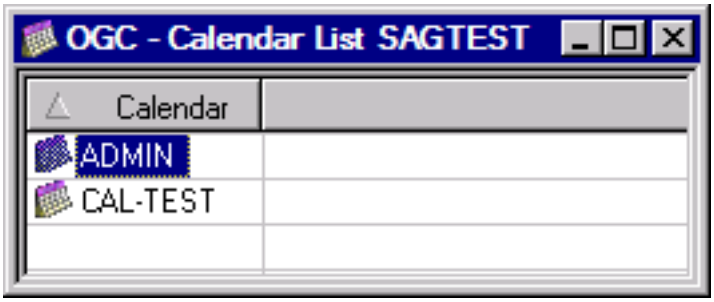
Function	Shortcut	Description
Open	CTRL+O	See *Displaying, Modifying or Adding*.
Display	CTRL+D	See *Displaying, Modifying or Adding*.
Where Used	---	Displaying schedules using a calendar. See *Where Used - Listing Schedules Using a Calendar*.
Delete	DELETE	See *Deleting a Calendar Definition*.
Copy data	CTRL+C	See *Copying Objects*.
Export	---	See *Exporting Objects* in the *Import/Export Functions* documentation.
Add to Workplan	---	See *Add to Workplan*.
Set Drag And Drop Function	---	See *Drag & Drop*.

Listing Calendars

To list calendars

1. In the object workspace, select the **Calendar** node of the required owner
2. From the context menu, choose **List**, or press F8.

   All calendars defined for the selected owner are listed in the **Calendar List** window as shown in the following example:



The list of calendars contains system-wide calendars and calendars belonging to the selected owner (here: SAGTEST).

A calendar is identified uniquely by its owner name and year.

This section covers the following topics:

Displaying, Modifying or Adding a Calendar Definition

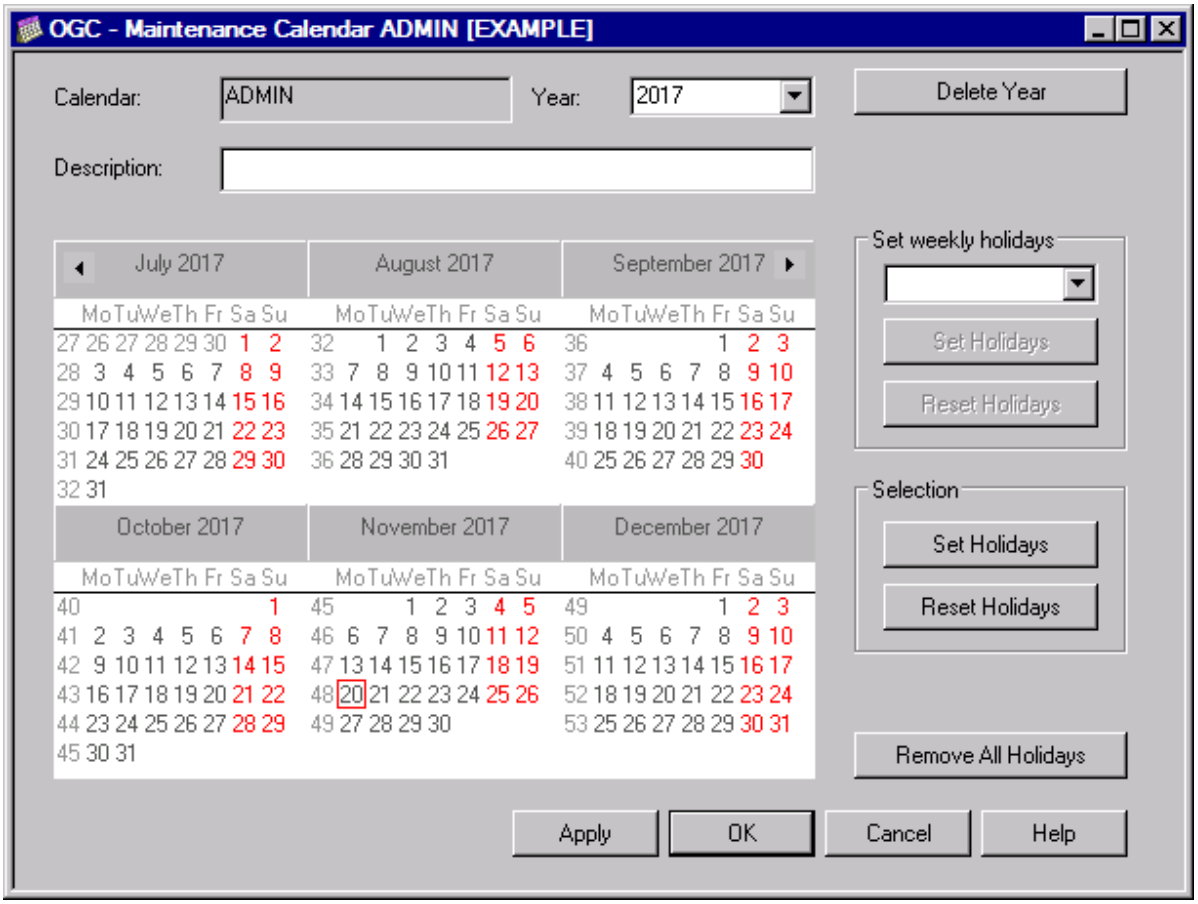
To display, modify or add a calendar definition

1 From the **Calendar** node in the object workspace, select the calendar you want to view or modify and choose **Display** or **Open**, respectively, from the context menu. Alternatively, you can press CRTL+D or CRTL+O, respectively.

Or:

Select a **Calendar** node and choose **New** from the context menu, or press CRTL+N.

Depending on the function performed, a **Display Calendar**, **Maintenance Calendar** or **Create new Calendar** window opens:



If the calendar is for the current year, the window displays six months from the current month of the specified calendar year by default.

For other years, the first six months are displayed initially.

Click on the right arrow button to scroll right in the calendar and view the next months, or click on the left arrow button to scroll left and view the previous months.

The current date is indicated by a red square, holidays are highlighted in red.

The fields in the upper section of the window are explained in *Fields: Calendar Definition*.

2 If defined for the calendar, you can select another year from the **Year** drop-down list box.

When modifying or adding a calendar, you add another year by replacing the current entry in the **Year** field.

3 When modifying or adding a calendar, define the required holidays as described in *Defining Workdays and Holidays*.

4 Choose **OK** to save your calendar definition.

The calendar is saved, and you can use it for scheduling network and jobs.

This section covers the following topics:

- Fields: Calendar Definition

Fields: Calendar Definition

The fields contained in the upper section of the **Display Calendar**, **Maintenance Calendar** and **Create new Calendar** window are described in the following table:

Field	Description
Calendar	Calendar name.
Year	Calendar year. The current year is entered by default.
Description	A short description of the calendar.

Defining Workdays and Holidays

Defining a calendar consists of defining workdays and holidays (non-workdays). This is done in two steps:

1. Define weekly holidays. These are the non-workdays in every week, such as weekends.
2. Define special holidays or personal vacations.



Note: A network schedule can override workdays marked on a calendar, because a network runs on a day specified in a calendar, only if this day is also specified in the schedule.

To define or remove weekly holidays

1 In the **Set weekly holidays** section of the Maintenance Calendar or **Create new Calendar** window, open the drop-down list box, select the weekday you want to define as a holiday and choose **Set Holidays**.

Or:

In the **Set weekly holidays** section of the Maintenance Calendar or **Create new Calendar** window, open the drop-down list window and select the defined holiday you want to change to a weekday and choose **Reset Holidays**.

Or:

Choose **Remove All Holidays**.

Caution: This function removes both all weekly and all special holidays for the year currently selected and shown in the **Maintenance Calendar** window.

2 Choose **OK** to save your calendar definition.

Weekdays defined as holidays are now highlighted in red.

To define or remove special holidays

1 In the **Selection** section of the **Maintenance Calendar** or **Create new Calendar** window, click on single or multiple (SHIFT-click) dates and choose **Set Holidays**.

The selected dates are highlighted in red.

Or:

In the **Selection** section of the **Maintenance Calendar** or **Create new Calendar** window, click on single or multiple (SHIFT-click) highlighted (defined) dates and choose **Reset Holidays**.

Or:

Choose **Remove All Holidays**.

Caution: This function removes both all weekly and all special holidays for the year currently selected and shown in the **Maintenance Calendar** window.

2 Choose **OK** to save your calendar definition.

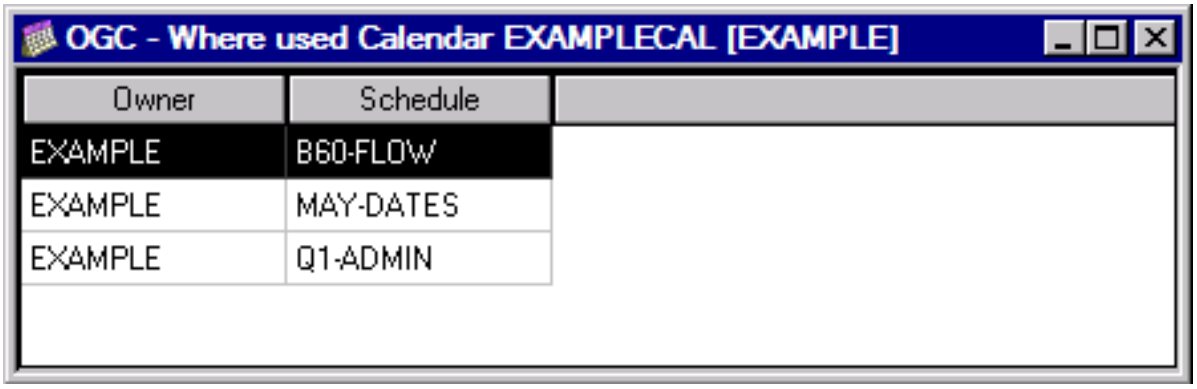
Weekdays defined as holidays are now highlighted in red.

Where Used - Listing Schedules Using a Calendar

To list schedules using a calendar

1 In the object workspace, select the required calendar from the **Calendar** node and choose **Where used** from the context menu.

A **Where used** window like the example below opens:



The window indicates the name of the selected calendar (here: EXAMPLECAL) and its owner (here: EXAMPLE) and lists all schedules (and their owners) associated with the calendar.

2 You can modify a listed schedule by selecting the required table row and choosing **Open** from the context menu.

A **Maintenance Schedule window** opens where you can change or remove the name of the calendar entered for the schedule.

Deleting a Calendar Definition

To delete a calendar definition

1 In the object workspace, select the calendar you want to delete from the **Calendar** node.

2 From the context menu, choose **Delete**.

Or:

Press DELETE.

A dialog opens where you must confirm the deletion.

You cannot delete a calendar if it is specified for at least one schedule. If you wish to delete a calendar which is still used by a defined schedule, Entire Operations notifies you with an appropriate message. Use the **Where used** function to find out which schedule(s) are associated with the calendar (see *Where Used - Listing Schedules Using a Calendar*). In this case, you cannot delete the calendar until you have done either of the following:

- Specified another calendar for the relevant network schedule(s).

Or:

- Deleted the network for which the calendar is specified.

You can delete only the calendars you are permitted to delete.

X Mailboxes

49 Mailboxes

- Concept of Single or Multiple Mailbox Users 566
- Listing and Maintaining Mailboxes 567
- Viewing Mailbox Messages 567
- Handling Messages and Requests 570

Mailboxes are used for sending messages to Entire Operations users. The messages are triggered by either system-detected or user-defined events.

If a message requires a reply, it is prompted in the **Show Messages and Requests window**. You can then take appropriate steps and manually set the conditions necessary for the job to continue.

Several **message destination types** can be defined. Among them is the sending of **e-mails on z/OS**, **BS2000** and **UNIX and Windows** systems.

Optionally, you can use a global exit for message sending. This exit detects all messages that must be sent for various reasons by the Entire Operations Monitor. The exit can store the message content in files and forward the message to other applications, etc.

Related Topics:

- *Defining Notification Messages - Job Maintenance*
- *Mailbox Definition - Administration* documentation
- *Mailboxes, Message Sending - Concepts and Facilities* documentation (general information including example scenarios)

Concept of Single or Multiple Mailbox Users

A mailbox can be available to one or more users which are linked to the mailbox (see also the **Mailboxes** field described in *User Definition and Profile Settings* in the *Administration* documentation).

All users linked to a mailbox have access to the same range of messages and requests. If a message or request of this mailbox is handled by any of these users, the Entire Operations Monitor triggers all jobs waiting for actions and the message handled by the user disappears from the message list for all other users linked to the mailbox.

SYSDBA Mailbox

All generated messages and requests, for which no user is defined, are sent to the SYSDBA mailbox, which is accessible to the owner SYSDBA.

For an example illustration of single or multiple mailbox use, see *Example Scenario - Concept of Single and Multiple Mailbox Users* in the *Concepts and Facilities* documentation.

Listing and Maintaining Mailboxes

This section covers the following topics:

- Listing Mailboxes
- Creating, Modifying and Deleting Mailboxes
- Clearing of Mailboxes

Listing Mailboxes

You can list all mailboxes defined in your environment and view their contents depending on the access rights set in the **Mailbox Definitions** of your user profile (see *Administration Functions* in the *Administration* documentation).

To list all mailboxes available in your environment

- Proceed as described in *Listing Mailboxes defined to Entire Operations* in the *Administration* documentation.

Creating, Modifying and Deleting Mailboxes

You can only create, change and delete a mailbox if you have appropriate access rights set in the **Mailbox Definitions** of your user profile (see *Administration Functions* in the *Administration* documentation).

For details on defining mailboxes, see *Mailbox Definition* in the *Administration* documentation.

Clearing of Mailboxes

Mailbox entries are cleared in accordance with the maximum length of time specified in the Entire Operations default settings for long-term logging. For further information, see the **Long-term log** option described in *Defaults for Time Ranges* in the *Administration* documentation.

Viewing Mailbox Messages

To view the messages for a mailbox

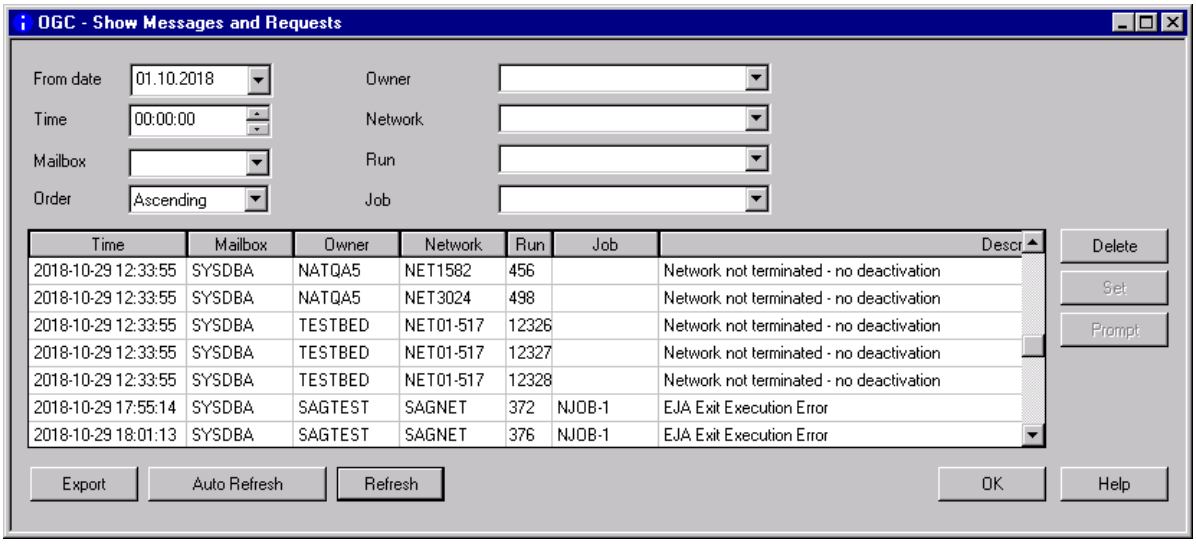
- In the object workspace, select a **Mailbox Definition** instance and choose **Show Messages** from the context menu.

  Or:

In the object workspace, select the **General** metanode and choose **Show Messages** from the context menu.

Alternatively, you can use the direct command `MAIL` as described in the *Direct Commands* documentation.

A **Show Messages and Requests** window like the example below opens:



The table view is filled dynamically with data if scrolled to the end of the table.

All messages and requests of all mailboxes to which the current Entire Operations user (**Owner at Logon**) is linked are listed in the table depending on where you executed the **Show Messages** function.

You choose **Export** to export the current mailbox settings. For details, see *Exporting Current Settings* in the *Import/Export Functions* documentation.

You can choose **Auto Refresh** and **Refresh** to refresh the message list. For details, see ***Refreshing Object Lists***.

The window columns and input fields provided to change the display and filter the messages are described in ***Fields and Columns: Messages and Requests***.

The messages and requests listed in the window and the functions you can perform on them are described in ***Handling Mailbox Messages and Requests***.

This section covers the following topics:

- Fields and Columns: Messages and Requests

Fields and Columns: Messages and Requests

The fields and columns provided in the **Show Messages and Requests** window are described in the following table:



Note: Preselection settings of the **Show Messages and Requests** window can be saved within a session. For this purpose, check the **Save mailbox message list settings within a session** option in the user profile: see the **GUI General** page of the **Maintenance User** window described in *Defining GUI-Specific Attributes* in the *Administration* documentation.

Column/Field	Description	
From Date/Time	Input fields: Specify the start date and time for displaying the messages.	
	Date and time when the message was sent.	
Mailbox	Mailbox to which the message was sent. Drop-down list box: Select the mailbox for which to list the messages. If there is no specification, all messages of all mailboxes from the user are displayed.	
Order	Sort order of the message list. Select either option from the drop-down list box:	
	Ascending	Ascending order. Earliest messages come first.
	Descending	Descending order. Latest messages come first.
Owner	Name of the owner. Drop-down list box: Select the owner whose messages you want to list.	
Network	Name of the network. Drop-down list box: Select the network for which to list the messages.	
Run	Number of the run. Drop-down list box: Select the run number for which to show the messages.	
Job	Name of the job. Drop-down list box: Select the run number for which to show the messages.	
Description	Message text. For possible messages, see the **message types** described in *Handling Mailbox Messages and Requests*.	

Handling Messages and Requests

You can perform message-specific actions on the messages and requests listed in the **Show Messages and Requests window**, such as symbol prompting, setting a condition or removing a message.

The action (delete, set or prompt operations) you can perform on a selected message depends on the type of message you select from the message list and the permissions you are granted in your Entire Operations user profile. The types of messages that can be shown are listed in the following table. The numbers in the example messages denote the job run numbers.

Messages on which actions are performed are removed for all users linked to the respective mailbox.

To perform an action on a message or request

- In the **Show Messages and Requests window**, select the required message from the list and choose **Delete**, **Set** or **Prompt**, whichever is available for the type of message selected (otherwise, the functions are dimmed).

Message Type	Possible Action	Example Message
System Information This is internal processing information and error messages and global messages defined for events (see *Global Messages for Events* in the *Administration* documentation).	**Delete** You can remove a message of this type from the list. A dialog prompts you to confirm the deletion.	NETWORK DEMO-NET ACTIVATED ==> SAG / DEMO-NET / 195 / MESSAGE EJA Exit Execution Error ==> SAGTEST / SAGNET / 564 / JOB-1
User Notification This is a notification defined as an action for and End-of-Job event (see *Defining Notification Messages*).		Remove job next month - no longer required ==> SAGTEST / SAGNET / 577 / NJOB-1
Confirmation Request Only available for jobs requesting a condition. A message of this type means that a scheduled network or job is waiting for a condition to be set.	**Set** You are prompted to set the selected condition to true. Choose **YES** (true) to confirm or **NO** (false) to reject the condition.	NETWORK-START-OK - 549 - RUN not found MAIL-ADMIN - 7 - RUN not found ==> SAG / SAGNET / 7 / DATA-PROCESSING-OK
Symbol Prompting Only available for networks or jobs requesting symbol prompting.	**Prompt** Proceed as described in *Performing Symbol Prompting*.	Symbol Prompting E60-FLOW Run 530 (OVER START TIME)

Message Type	Possible Action	Example Message
A message of this type means that a scheduled network or job is waiting for symbols to be entered or modified. See also *Symbol Prompting during Network or Job Activation*.		

- Performing Symbol Prompting

Performing Symbol Prompting

Note: Symbol prompting via mailbox is only used for automatic jobs started by schedule and cannot be used for jobs started interactively. Definition of the recipient mailbox for symbol prompting messages is described in *Specifying Recipients for Network Messages* in the section *Network Maintenance*.

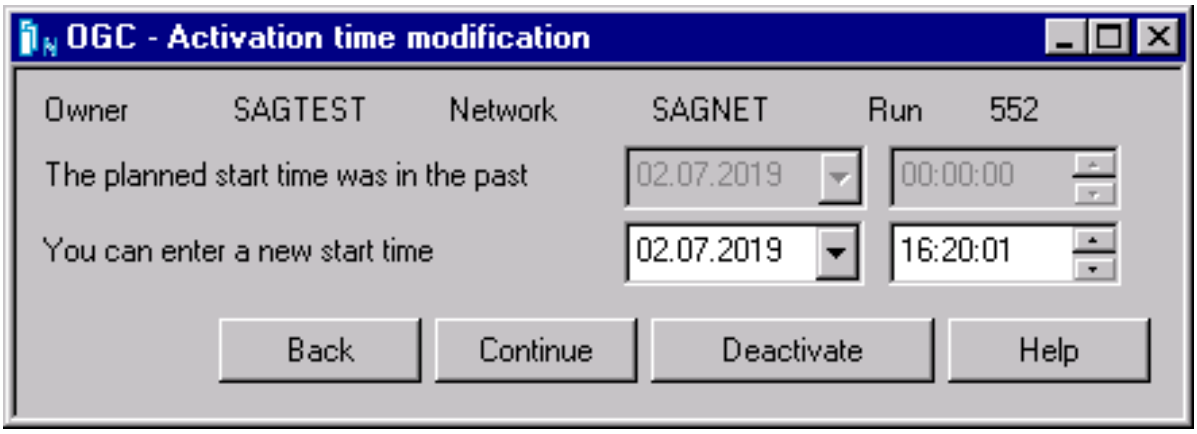
To perform symbol prompting

1 From the **Show Messages and Requests window**, select a message requesting symbol prompting.

A **Symbol Prompting window** opens where you can view, add, modify or deactivate the symbol(s) defined for the network indicated in the message.

Proceed as described in *Symbol Prompting during Network or Job Activation* in the section *Symbol Table and Symbol Maintenance*.

2 A message requesting symbol prompting usually displays the time until the next network start (in hours). However, if the network has exceeded its start time, the message says `OVER START TIME`.



3 Enter a new start time and choose **Continue**

Any symbol changes are changed and the window closes.

The new start time is entered as the earliest start time in all active network jobs.

Back closes the **Symbol Prompting** window without any deactivate or activate action.

Or:

Choose **Back** to close the window without any deactivate or activate action.

Or:

Choose **Deactivate** to cancel the active run of the specified network.

A dialog prompts you to confirm the deactivation.

XI Symbol Table and Symbol Maintenance

Purpose and Use of Symbol Tables and Symbols

Maintaining and Finding Symbol Tables

Defining Symbols and Symbol Values

Symbol Replacement

Functions for Symbol Replacement

50 Purpose and Use of Symbol Tables and Symbols

- Symbol Tables 576
- Symbols 577
- Handling Active Symbol Tables and Active Symbols 579
- Subnetworks and Recovery Jobs 580
- User Exits for User-Specific Symbol Maintenance Tasks 580

Related Topic:

- *Symbol Tables and Symbols* in the *Concepts and Facilities* documentation

Symbol Tables

A symbol table is used to maintain common parameter definitions that apply to all jobs of a network. A symbol table contains a list of defined symbols with values that can be used for parameter substitution during JCL generation.

You can substitute symbols when activating a job network or a job, that is, when the active JCL is loaded to the active database. You can also substitute symbols when submitting a job.

Each network activation initiates its own active copy (**active symbol table**) of the linked symbol table(s). This allows you to schedule networks with different parameter sets, even a long time in advance. Any occurrence of a symbol name in the JCL or in a script is replaced by its current value.

You can maintain symbol tables by using either Entire Operations maintenance functions or user-written programs, user exits and application programming interfaces (APIs) supplied by Entire Operations. This allows modification of symbol tables and symbol values during job network processing.

Symbol tables are associated with owners. Each owner can have several symbol tables. You can only maintain the symbol tables that belong to your owner.

The symbol table to be referenced by a job and/or network must be specified in the job or network master definition or in the active queue for a single job run. Exceptions are **global symbol tables**.

This section covers the following topics:

- Global Symbol Tables
- Active Symbol Tables
- Reserved Symbol Table Name
- Symbol Table Versions and Version Usage

Global Symbol Tables

A symbol table master defined by the owner SYSDBA is considered a global symbol table because it can be references by multiple jobs and networks without explicit specifications in the job and network definitions. A global symbol table with the name `A` is supplied for the owner SYSDBA by default. It can be accessed and referenced by all owners.

All other owners can also define a global symbol table with the name `A` for their networks. Their global symbol table `A` then takes priority over the global symbol table `A` defined by SYSDBA. See also ***Symbol Table Types and Symbol Search Order***.

Active Symbol Tables

During job or network activation, an active copy of each symbol table used by a job or network is created by copying it to the active database. This is then called the active symbol table. Symbols to be replaced are taken from the active symbol table. This enables different values of the same symbols to be generated for different runs of the network or job.

See also ***Handling Active Symbol Tables and Active Symbols***.

Reserved Symbol Table Name

The symbol table name with the prefix `=EOR=` is reserved for internal control purposes. This prefix must not be used for any user-defined table names.

Symbol Table Versions and Version Usage

You can maintain several versions of a symbol table master as described in ***Versioning of Symbol Tables***. Exceptions are **global symbol tables** for which versioning is not allowed.

If you have several symbol table versions, you can define a validity period for each version. See the section ***Maintaining the Usage of Symbol Table Versions***.

Symbols

A symbol is a variable definition that contains the format, prompting type, prompting text (optional) and the value(s) of the symbol.

Symbols can be defined by using Entire Operations maintenance functions, or they can be set and modified by any program, which invokes the Entire Operations symbol API **NOPUSY7N** (see the section *API Routines*) to set, reset, or modify a symbol.

For detailed information on defining symbols in JCL (including text objects) or Natural macro sources, see ***Editing Master JCL and Natural Sources*** and ***Inserting Text Objects into the JCL*** in the section *Job Maintenance*.

This section covers the following topics:

- Global Symbols
- Predefined Symbols
- Symbols for Replacement

- Symbol Prompting

Global Symbols

A symbol master contained in a global symbol table is considered a global symbol because it can be used by multiple jobs and networks without extra definitions for each job and/or network that references the symbol. See also *Global Symbol Tables*.

Predefined Symbols

Predefined symbols provided by Entire Operations are described in the section *Predefined Symbols*.

Symbols for Replacement

Symbols are usually replaced during the activation of a job network or job, that is, during the loading of the active JCL to the active database. In certain cases, symbol replacement can even be used in file names, messages texts, etc.

A symbol to be replaced is identified by an **escape character** defined in the Entire Operations default and network settings, or used as a prefix in JCL and Natural macro sources. You specify the symbols in JCL or macro sources as described in *Defining Escape Characters*.

If a symbol is not found in the symbol table specified for a job or network, the symbol is searched for in the **global symbol table(s)**. The first match is used to replace the symbol. A symbol is not replaced if it cannot be found in any table; an appropriate error message is then issued instead. See also *Symbol Table Types and Symbol Search Order*.

For further information, see the section *Symbol Replacement*.

Symbol Prompting

You can define whether a symbol is prompted for symbol modification during or before job network activation.

- During a manual network or job activation, the prompting is performed immediately. See the section *Symbol Prompting during Network or Job Activation*.
- For scheduled (automatic) activations, prompting requests are sent to a mailbox linked to the network. The network activation does not continue until the prompting is performed by a user. See also *Specifying Recipients for Network Messages* in the section *Network Maintenance*.

Handling Active Symbol Tables and Active Symbols

Active symbol tables are used to provide symbols for a selected job run. You can add, modify or delete the symbol of an active symbol table without affecting the symbol definitions in the symbol table master.

By default, modifications to an active symbol apply to the current run only. However, if the check box next to **Prompting** (see *Fields: Symbol*) is activated in a symbol master or active symbol definition window, modifications or additions apply to both the active symbol of the current run and the symbol master.

A symbol table is usually activated when a network or job is activated (see also *Activation of Networks or Jobs*), or when **symbol prompting** is performed.

For scheduled network activation, you can determine the time to activate the symbol tables defined in the network and job definitions. This can be either of the following:

- Directly after schedule extraction; this allows enough time for manual symbol replacement.
- During network activation. In this case, however, manual symbol replacement is not possible.

Active symbol tables are accessed through the **Symbol Table Active** node of a **Network Active** or **Network Master** metanode.

This section covers the following topics:

- Performing Maintenance Functions

Performing Maintenance Functions

You can only modify, add/copy or delete symbols if you are granted permission in your user profile or if the containing symbol table belongs to an owner to whom you are linked and who has permission to perform these operations. See also your **Symbol Tables** profile setting for *Network Maintenance Functions* described in the *Administration* documentation.

Active symbols are maintained analogously to symbol masters. In general, all maintenance functions, fields and columns provided for symbol masters are also provided for active symbols. Functional differences are documented where relevant.

Subnetworks and Recovery Jobs

Subnetworks and recovery jobs defined for a network require **predefined symbols** to pass information to the jobs of the network. These predefined symbols have names that begin with `P-C-`. For detailed information, see ***Predefined Symbols for Subnetworks and Recovery Jobs***.

User Exits for User-Specific Symbol Maintenance Tasks

You can use user exit routines that perform user-specific symbol maintenance tasks and symbol replacement functions. For detailed information, refer to the following sections:

***User Exits*:**

- ***User Exits for Validation Checks of Symbol Values***
- ***User Exits for Symbol Functions***

***Global User Exits* in the *Administration* documentation:**

- *Global Symbol Modification Exit*
- *Global Symbol Not Found Exit*

51 Maintaining and Finding Symbol Tables

- Available Functions: Symbol Table 582
- Listing Symbol Table Masters Defined for an Owner 584
- Listing Active Symbol Tables 585
- Listing Usable Symbol Tables 585
- Listing Jobs and Networks that Use a Symbol Table 587
- Determination and Activation of Necessary Symbol Tables 590
- Displaying and Modifying a Symbol Table 591
- Adding a Symbol Table Master 594
- Versioning of Symbol Tables 595
- Maintaining the Usage of Symbol Table Versions 599
- Saving Symbol Tables as Files 603
- Deleting a Symbol Table Master 604

Symbol tables are defined in a **Symbol Master** or **Symbol Active window**. Maintenance functions for symbol tables are provided from the context menu of a symbol table selected from a **Symbol Table Master** or **Symbol Table Active** node.

The fields and functions provided in a **Symbol Master** or **Symbol Active** window are explained in *Fields: Symbol*. The functions provided for symbol and symbol table maintenance are explained in *Available Functions: Symbol*.

Available Functions: Symbol Table

To list all functions available for a symbol table node

- For a symbol table master: In the object workspace, select the **Symbol Table Master** node from an **Owner** instance and open the context menu.

  Or:

  For an active symbol table: In the object workspace, select a **Symbol Table Active** node from an **Active Run** and open the context menu.

  The following functions are available for a list of symbol tables:

Function	Shortcut	Description
List	F8	Lists symbol tables: see *Listing Symbol Table Masters Defined for an Owner* and *Listing Active Symbol Tables*.
New	CTRL+N	**Adds a symbol table master**. This function is not available for active symbol tables.
Refresh	F5	See *Refreshing Object Lists*.
Filter	F3	See *Filtering Objects*.
Paste data	CTRL+V	See *Pasting Objects*. This function is not available for active symbol tables.
Export	---	See *Exporting Objects* in the *Import/Export Functions* documentation. This function is not available for active symbol tables.
Set Drag And Drop Function	---	See *Drag & Drop*.

To list all functions available for a symbol table

- For a symbol table master: In the object workspace, select an instance from a **Symbol Table Master** node and open the context menu.

  Or:

  For an active symbol table: In the object workspace, select an instance of a **Symbol Table Active** node and open the context menu.

  The following functions are available:

Function	Shortcut	Description
Open	CTRL+O	**Modifies the symbol table**.
Display	CTRL+D	**Displays the symbol table**.
Where used	---	**Lists jobs and networks that use the symbol table**. This function is not available for active symbol tables.
Active Usage		**Lists active jobs using an active symbol table**. This function is not available for symbol table masters.
Delete	DELETE	**Deletes a symbol table master**. This function is not available for active symbol tables.
Copy data	CTRL+C	See *Copying Objects*. This function is not available for active symbol tables.
Version Usage	---	**Maintains the usage of symbol table versions**. This function is not available for active symbol tables.
Export	---	See *Exporting Objects* in the *Import/Export Functions* documentation. This function is not available for active symbol tables.
Add to Workplan	---	See *Add to Workplan*.
Save as File	---	See *Saving Symbol Tables as Files*.
Set Drag And Drop Function	---	See *Drag & Drop*.

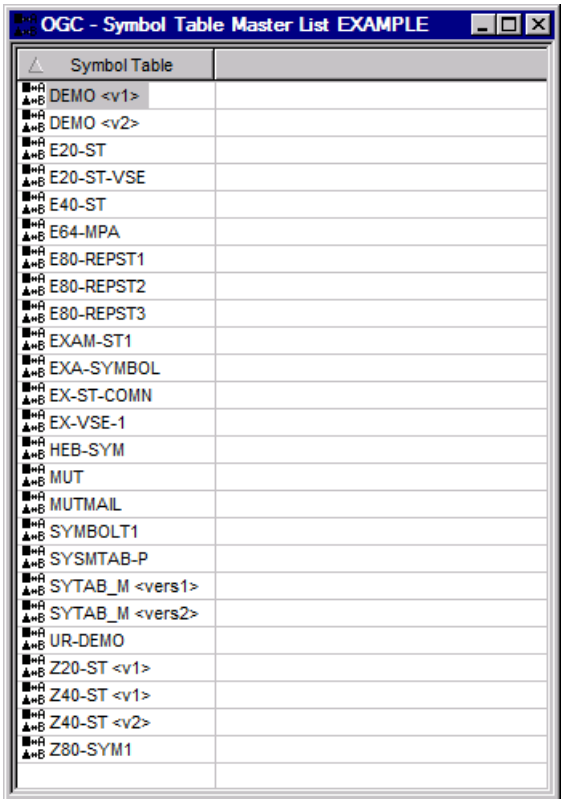
Listing Symbol Table Masters Defined for an Owner

This function applies to symbol table masters only.

To list all symbol table masters of an owner

- In the object workspace, select the **Symbol Table Master** node of an owner and choose **List** from the context menu, or press F8.

  A **Symbol Table Master List** window like the example below opens:



  This window lists symbol table masters already defined to Entire Operations. You see only the symbol tables that belong to your owner (here: `EXAMPLE`). If no symbol table masters exist, an appropriate message is returned.

  The functions available for a symbol master list are explained in *Available Functions: Symbol Table*.

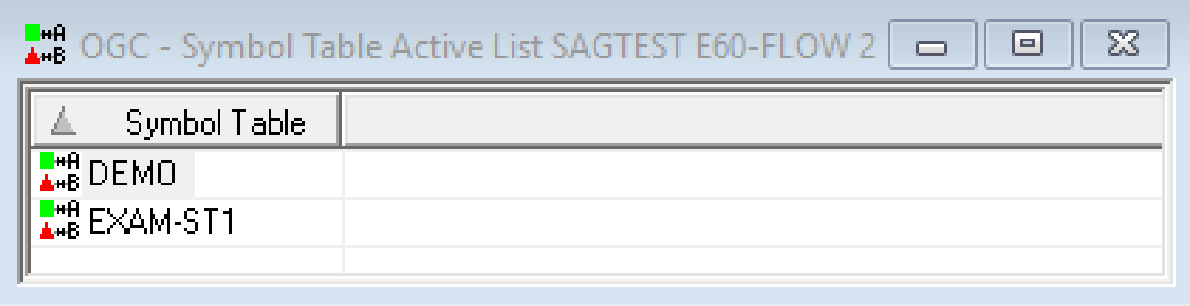
This section covers the following:

Listing Active Symbol Tables

To list symbol tables active during a job run

- In the object workspace, select the **Symbol Table Active** node of an **Active Run** and choose **List** from the context menu, or press F8.

  A **Symbol Table Active List** window like the example below opens:



  This window lists all active symbol tables for the selected job run (here: 2) of a network (here: E60-FLOW). If active symbol tables are not available, an appropriate message is returned.

  The functions available for a list of active symbol tables are explained in *Available Functions: Symbol Table*.

Listing Usable Symbol Tables

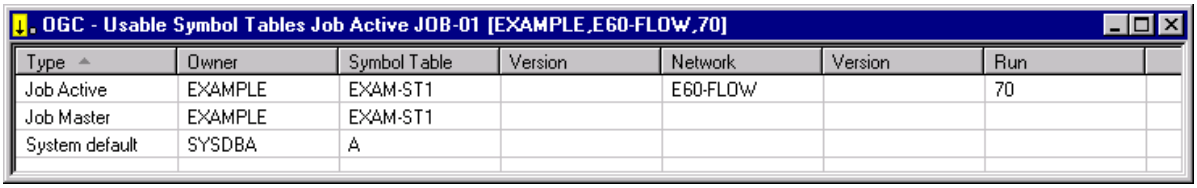
You can list all symbol tables that can be used by a job and/or network and network version. For an active network, usable symbol tables are active symbol tables required and actually used for symbol replacement during the current network run. You can list and access all symbols contained in listed symbol tables.

To list symbol tables available for a job or network

1 In the object workspace, select an instance of a job or network (master or active).

2 Open the context menu and choose **Usable Symbol Tables**.

  A **Usable Symbol Tables** window like the example of an active job below opens:



This window lists all symbol tables which can be used by the current object (network, job master, active job). The example above lists the symbol tables that can be used by the active job `JOB-01`.

The columns in the window are explained in *Columns: Usable Symbol Tables*.

The symbol tables are listed in descending hierarchical order, that is, the topmost symbol table is used first during symbol replacement.

This section covers the following topics:

- Columns: Usable Symbol Tables

Columns: Usable Symbol Tables

The columns contained in the **Usable Symbol Tables window** are described in the following table:

Column	Description
Type	Type of the symbol table: see *Symbol Table Types and Symbol Search Order*.
Owner	Owner of the symbol table.
Symbol Table	Name of the symbol table.
Version	Version of the symbol table.
Network	For active job networks only. Active network which uses the symbol table.
Version	For active job networks only Version of the active network which uses the symbol table.
Run	For active job networks only. Run number of the active job network.

Listing Jobs and Networks that Use a Symbol Table

You can list all jobs and networks that use a symbol table master or an active symbol table. This is required, for example, if you cannot delete a symbol table because it still uses one or more jobs. You then need to find these jobs and change their symbol table settings or delete the jobs.

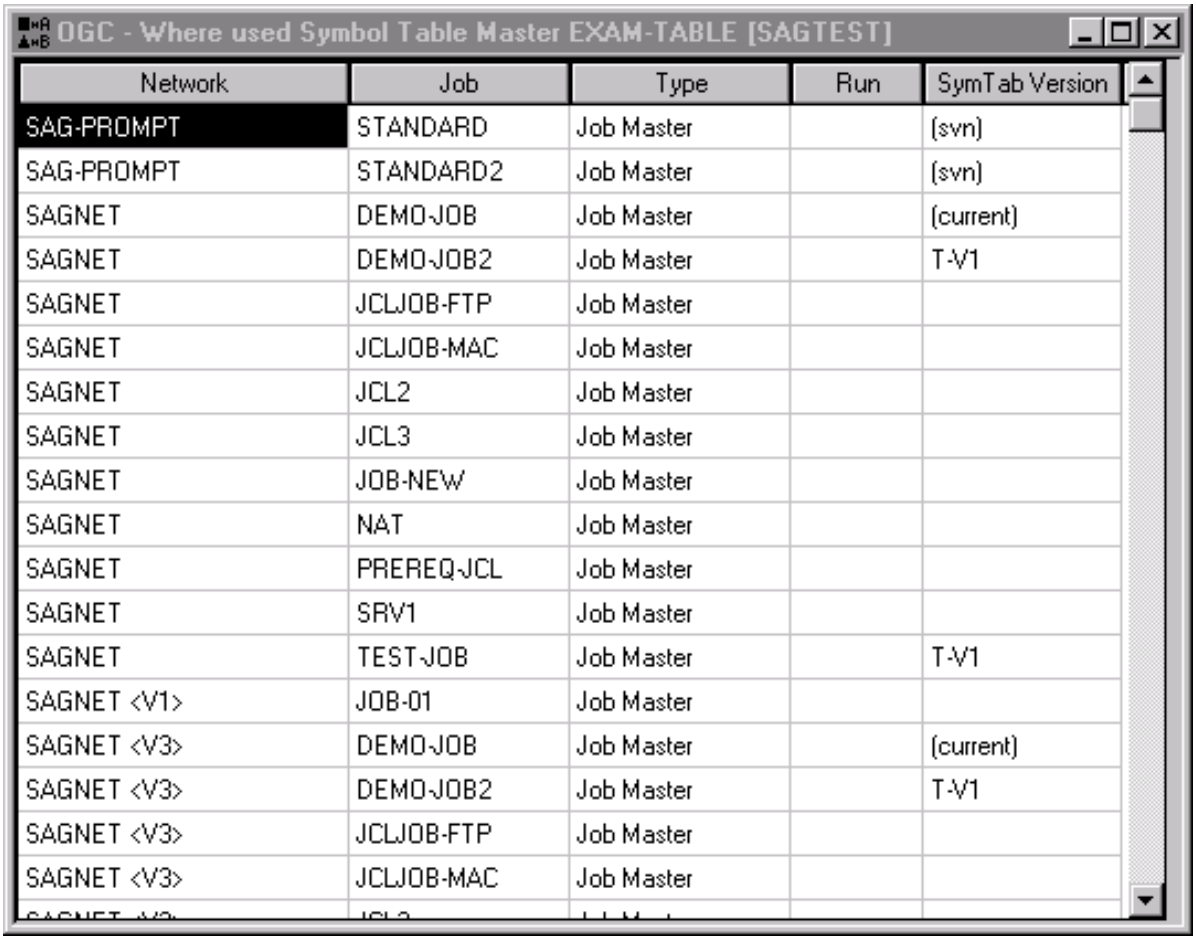


Note: This function only shows the references of a network or job to a symbol table master if it has been explicitly defined. You can use the cross-reference function **Symbol Table Usage** (see *Generating Cross-Reference Reports Online*) to get a list of all references to a symbol table master including the references contained in input conditions and End-of-Job actions.

To list jobs and networks that use a symbol table master

- In the object workspace, select a **Symbol Table Master** instance and choose **Where used** from the context menu.

  A **Where used Symbol Table Master** window like the example below opens:



Network	Job	Type	Run	SymTab Version
SAG-PROMPT	STANDARD	Job Master		(svn)
SAG-PROMPT	STANDARD2	Job Master		(svn)
SAGNET	DEMO-JOB	Job Master		(current)
SAGNET	DEMO-JOB2	Job Master		T-V1
SAGNET	JCLJOB-FTP	Job Master		
SAGNET	JCLJOB-MAC	Job Master		
SAGNET	JCL2	Job Master		
SAGNET	JCL3	Job Master		
SAGNET	JOB-NEW	Job Master		
SAGNET	NAT	Job Master		
SAGNET	PREREQ-JCL	Job Master		
SAGNET	SRV1	Job Master		
SAGNET	TEST-JOB	Job Master		T-V1
SAGNET <V1>	JOB-01	Job Master		
SAGNET <V3>	DEMO-JOB	Job Master		(current)
SAGNET <V3>	DEMO-JOB2	Job Master		T-V1
SAGNET <V3>	JCLJOB-FTP	Job Master		
SAGNET <V3>	JCLJOB-MAC	Job Master		

The window lists all networks and jobs that use the selected symbol table master (here: `EXAM-TABLE`) of the specified owner (here: `SAGTEST`).

All symbol table versions defined for the selected symbol table are listed.

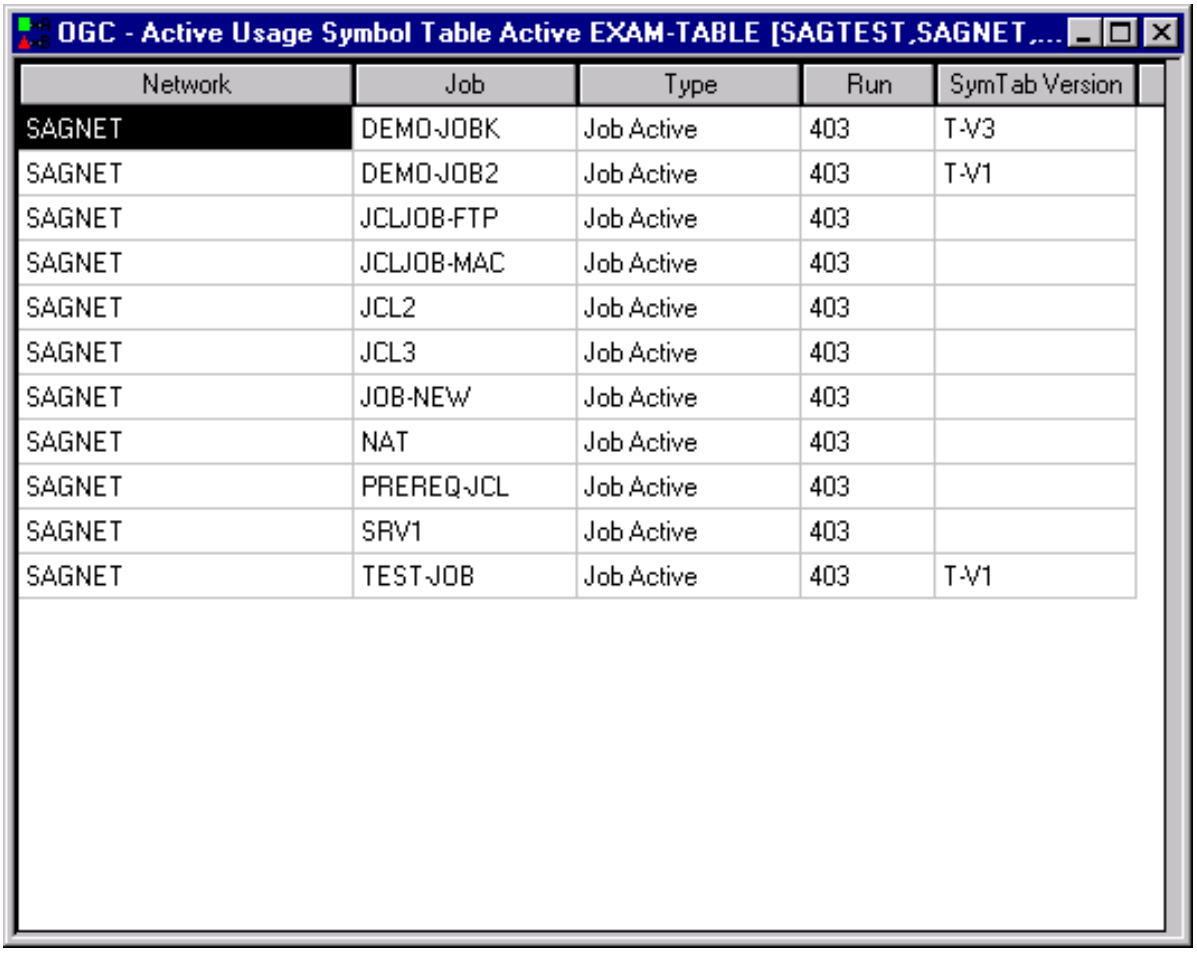
The columns in the window are explained in *Fields and Columns: Symbol Table Usage*.

To list all active jobs using an active symbol table

1 In the object workspace, select a **Symbol Table Active** instance.

2 Open the context menu and choose **Active Usage**.

An **Active Usage Symbol Table Active** window like the example below opens:



Network	Job	Type	Run	SymTab Version
SAGNET	DEMO-JOBK	Job Active	403	T-V3
SAGNET	DEMO-JOB2	Job Active	403	T-V1
SAGNET	JCLJOB-FTP	Job Active	403	
SAGNET	JCLJOB-MAC	Job Active	403	
SAGNET	JCL2	Job Active	403	
SAGNET	JCL3	Job Active	403	
SAGNET	JOB-NEW	Job Active	403	
SAGNET	NAT	Job Active	403	
SAGNET	PREREQ-JCL	Job Active	403	
SAGNET	SRV1	Job Active	403	
SAGNET	TEST-JOB	Job Active	403	T-V1

The window lists all active jobs used by the active symbol table. The columns contained in the window correspond to the columns in the **Where used Symbol Table Master**. They are explained in ***Columns: Symbol Table Usage***.

This section covers the following topics:

- Columns: Symbol Table Usage

Columns: Symbol Table Usage

The columns contained in the **Where used Symbol Table Master** or **Active Usage Symbol Table Active window** are explained in the following table:

Column	Description
Network	Name and version of the job network.
Job	Name of the job that uses the symbol table. A dash (-) or a blank indicates that the symbol table is specified in the network definition.
Type	Indicates whether a job master or job active, or a network master/version or network active uses the symbol table. For possible column entries, see the *Symbol Table Type* table column described in *Symbol Table Types and Symbol Search Order*.
Run	Displays the job's run number if the symbol table is used by an active job.
SymTab Version	Named version (if defined) or reserved name of the symbol table. For possible column entries, see *Reserved Version Names for Symbol Tables*.

Determination and Activation of Necessary Symbol Tables

During network activation or single job activation, the list of the required (active) **symbol tables** is determined by Entire Operations. The result of the determination is written to the Entire Operations log. It may look like this:

```
List of active Symbol Tables created
Determined Symbol Table Versions for 17.01.14
... Ob  Job         St  SymTab      defined         determined
... NV              00  N1649T00    (current)   ->  v002
... JM  J001        00  N1649T00    (unnamed)   ->  (unnamed)
... JM  J003        ED  N1649T00    (current)   ->  v002
... JM  J004        ED  N1649T00    (nv)        ->  (unnamed)
... JM  J005        ED  N1649T00    (svn)       ->  v002
```

The **St** column shows the status of the symbol table to be activated:

ED	Evaluation duplicate. This status is set if a previous determination (evaluation) resulted in the same symbol table with the same version. In this case, the symbol table (version) is activated only once.
EJ	Indicates End-of-Job checking or End-of-Job actions.
00	Undefined status.
10	Indicates that prompting is not required.
20	Indicates that prompting is required.
30	Indicates that prompting is performed.

The **Ob** column shows the symbol table type, for example, JM for a symbol table referenced by a job master. See also *Symbol Table Types and Symbol Search Order*.

The determined symbol table versions (see *Versioning of Symbol Tables* in the section *Symbol Table and Symbol Maintenance*) are used for the subsequent symbol table activation.

In case of any determination error, the network activation or job activation is aborted.

Displaying and Modifying a Symbol Table

Displaying and modifying a symbol table master or an active symbol table consists of viewing, adding, modifying or deleting symbols contained in the symbol table. You cannot change the name or version of a symbol table.

To display or modify symbols contained in a symbol table

1 For a symbol table master: In the object workspace, select a **Symbol Table Master** instance.

Or:

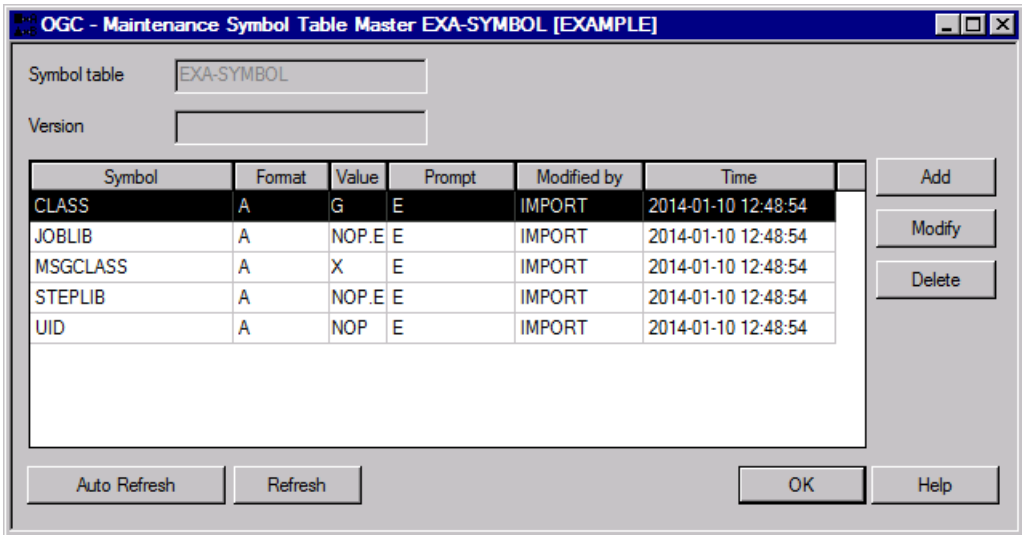
For an active symbol table: In the object workspace, select a **Symbol Table Active** instance.

2 To display symbols: Choose **Display** from the context menu, or press CTRL+D.

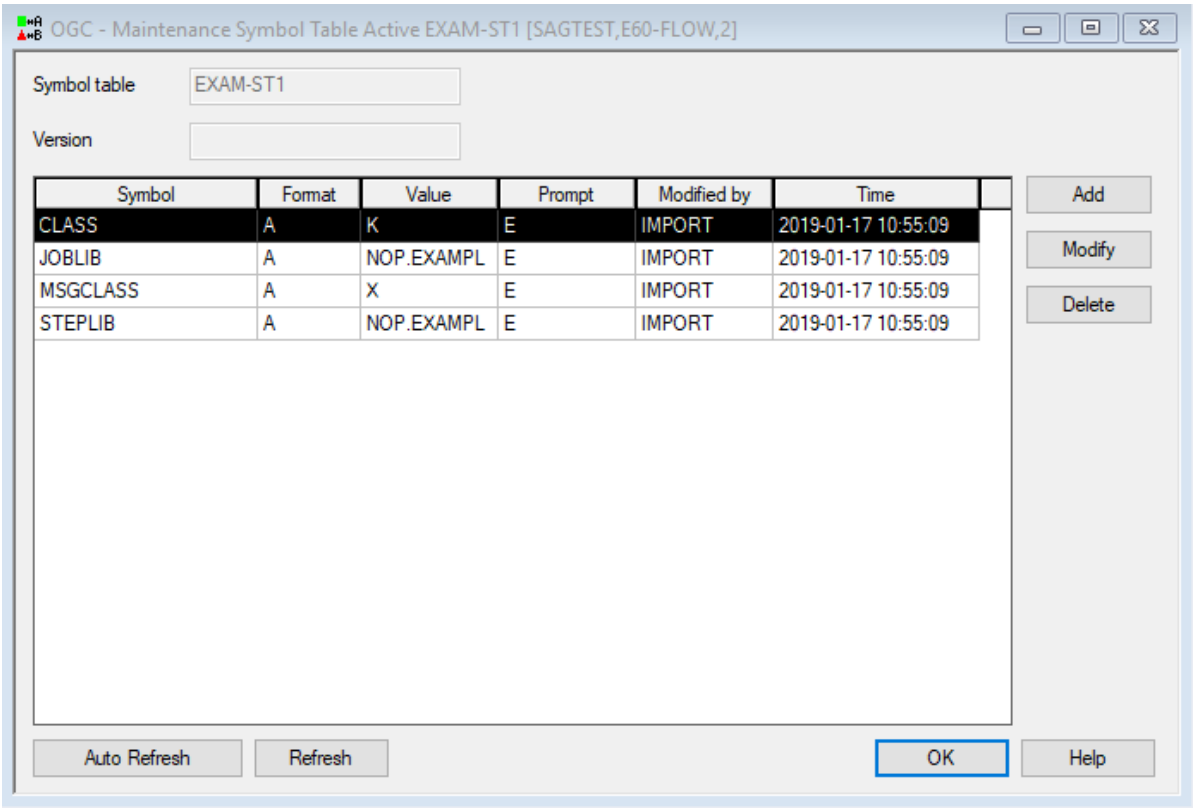
Or:

To modify symbols: Open the context menu and choose **Open**, or press CTRL+O.

For a symbol table master, a **Display Symbol Table Master** or **Maintenance Symbol Table Master** window like the example below opens:



For an active symbol table, a **Display Symbol Table Active** or **Maintenance Symbol Table Active** like the example below opens:

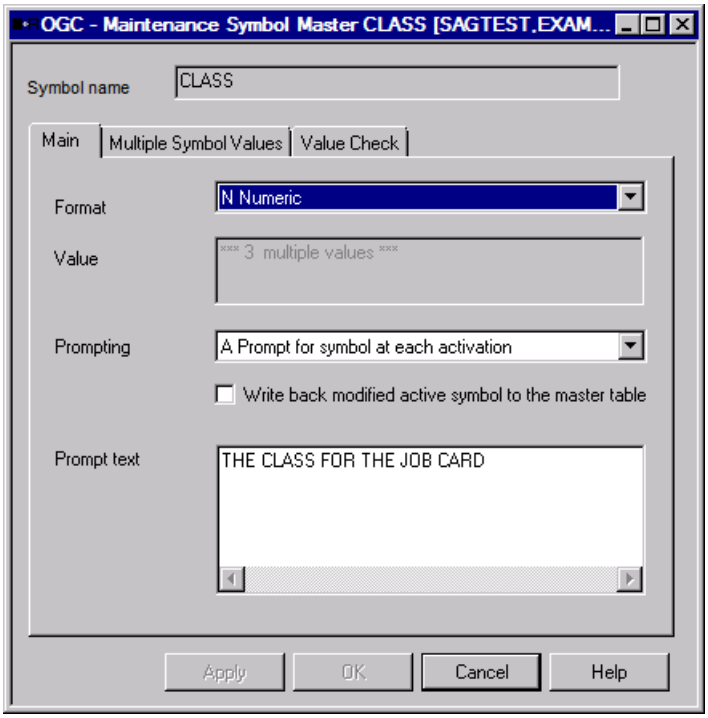


You can modify a selected symbol (here: `CLASS`) by choosing **Modify**, add a new symbol by choosing **Add**, or delete a selected symbol by choosing **Delete**.

The fields and columns contained in the window are explained in *Fields and Columns: Symbol Table*.

3 If you choose **Delete**, a window prompts you to delete the selected symbol.

4 If you choose **Modify** or **Add**, a **Maintenance Symbol Master** window like the example below opens:



5 Make the required definitions in the fields of the tabbed pages **Main**, **Multiple Symbol Values** and **Value Check**. They are explained in *Fields: Symbol*.

6 Choose **OK** when you are finished.

The symbol table master is changed and now ready to use in the network.

This section covers the following topics:

- Fields and Columns: Symbol Table

Fields and Columns: Symbol Table

The fields and columns in a **Symbol Table Master** or **Symbol Table Active** window are explained in the following table:

Field/Column	Description	
Symbol	Name of the symbol used as a variable.	
Format	Format of the variable. Possible values:	
	A	Alphanumeric (including special characters).
	D	Date.
	L	Alphanumeric; conversion to lowercase letters.

Field/Column	Description	
	N	Numeric.
	U	Alphanumeric; conversion to uppercase letters.
Prompt	Prompting. The user is prompted for this symbol during manual activation of the job network. Possible values:	
	A	Prompted for symbol during each activation.
	E	Prompted only if no value is specified in the symbol table.
	N	Never prompted for symbol.
Value	Current value of the variable, i.e., the string or number substituted in dynamically generated JCL. Possible entries:	
	empty column	The value of the symbol is an empty string.
	*** *n* multiple values ***	One or more values (*n*) are defined for the variable. Select the **Multiple Symbol Values** tab to view or change the values.
Modified by	User who last modified the symbol.	
Time	Date and time when a user last modified the symbol.	

Adding a Symbol Table Master

This function applies to symbol table masters only.

You can add a symbol table master by either copying an existing table (see the **Copy data**/**Paste data** functions for a symbol table instance of a master node) or creating a new symbol table with the **New** function. This requires that you define at least one symbol master along with the table.

This section provides instructions for using the **New** function.

To add a symbol table master

1 In the object workspace, select the **Symbol Table Master** node.

2 From the context menu choose **New**, or press CTRL+N.

A **Create new Symbol Table Master** window opens.

The fields and columns contained in the window correspond to the field and columns in the **Maintenance Symbol Table Master** window. They are explained in *Fields and Columns: Symbol Table*.

3 Enter a name for the symbol table and symbol table version and choose **Add**.

A **Create new Symbol Master** window opens.

The fields and tabbed pages contained in the window correspond to the fields and tabbed pages in the **Maintenance Symbol Master** window. They are explained in *Fields: Symbol*.

4 If required, enter a different symbol table name and version and make your definitions on the tabbed pages **Main**, **Multiple Symbol Values** and **Value Check**.

5 Choose **OK** when you are finished.

The symbol table master is created and now ready to use in the active network.

This section covers the following topics:

Versioning of Symbol Tables

This function applies to symbol table masters only.

This section describes how to maintain and use different versions of symbol tables.

- Version Names
- Version Names Exit
- Reserved Version Names for Symbol Tables
- Creating Symbol Table Versions by Copying Symbol Tables
- Copying Single Symbols
- Deleting Symbol Table Versions
- Deleting Symbol Table Versions or Single Symbols via API
- Using Symbol Table Versions for Schedule Activation
- Definition of Symbol Table Versions
- Versioning for Active Symbol Tables
- Symbol Prompting
- Search Order for Symbols
- Symbol Tables at System and Owner Level
- Logging
- Cross-References (XRef)
- Reporting
- Import/Export
- Exit Functionality (Symbol Table)

- Maximum Number of Versions per Symbol Table

Version Names

The following applies when creating version names:

- The name can contain up to 10 alphanumeric characters and uppercase or lowercase letters.
- Space characters and the following special characters are not allowed:

  ?<>*,()_
- To avoid problems when porting an Entire Operations environment to another platform, do not use special characters and umlauts.
- Do not use a **reserved version name** (see the relevant section).

Version Names Exit

You can force a user-specific version name syntax by using a global version name exit. For detailed information, see *Global Exit for Version Names* in the *Administration* documentation.

Reserved Version Names for Symbol Tables

You can specify a reserved version name to determine the symbol table to be used for an active network or job run.

A reserved version name is evaluated during network or job activation and is replaced by a defined version name if available for the symbol table at that time.

You can use the following reserved version names:

Blank value or `(unnamed)`
: A blank value or `(unnamed)` is replaced by the unnamed version of a symbol table.

  An unnamed symbol table network version is the only symbol table version that exists after migration from an earlier Entire Operations version that does not support symbol table versioning.

  In parameter listings (for example, for reporting functions such as **Compare Symbol Tables**) you can also use a hyphen (`-`).

`(current)`
: `(current)` is replaced by the symbol table version that is in effect during version evaluation. `(current)` can also be used in input fields that reference symbol table versions.

`(nv)`
: `(nv)` is replaced by the network version of the active network.

`(svn)`
`(svn)` is replaced by the symbol table version used by the active network. `(svn)` can also be used in symbol tables referenced in a subnetwork.

`(svj)`
`(svj)` is replaced by the symbol table version used by the active job. `(svj)` can also be used in input and output conditions defined for the job. For example:

- Requested prerequisite dependent on the symbol value,
- Requested prerequisite dependent on multiple-value symbols,
- End-of-Job action: set symbol.

Creating Symbol Table Versions by Copying Symbol Tables

You can copy symbol tables to add a new version for an existing symbol table. This is a common way to build new symbol table versions.

Copying Single Symbols

You can copy a single symbol from any version of the originating symbol table.

Deleting Symbol Table Versions

- If multiple versions of a symbol table exist, you must select the version to be deleted.
- A symbol table version cannot be deleted if listed in an activated time schedule as a standard version. A defined date in the past is not relevant.

Deleting Symbol Table Versions or Single Symbols via API

You can delete single symbol table versions by using the API **NOPUSY7N**.

Using Symbol Table Versions for Schedule Activation

You can maintain version usage in symbol table administration by using the functions described in *Maintaining the Usage of Symbol Table Versions*.

Definition of Symbol Table Versions

Symbol table versions can be defined in the following:

- Network version definition,
- Job definition.

Versioning for Active Symbol Tables

- The activation of symbol tables is a component of network and job activations.
- A symbol table can only be activated in a clearly identified version. The identification of the requested symbol table version is part of the activation process.
- Active symbol tables loose their version nomenclature `(current)` or `(nv)`. They are detached during activation.
- Active symbol tables can only have the version nomenclature `(none)` or a defined version name.
- If a requested symbol table version is missing or if the version cannot be defined, then the activation process will be aborted with an error message.

Symbol Prompting

Before symbol prompting (during manual activation and before executing the symbol prompting exit in the Monitor), the symbol table versions to be used will be clearly determined. See also *Symbol Prompting during Network or Job Activation*.

If at least one symbol table version cannot be identified, the activation process will be aborted with an error message.

Search Order for Symbols

The order in which symbols are searched for in the symbol tables defined in your environment depends on the hierarchy levels at which the symbol tables defined in you environment can be accessed: see *Symbol Table Types and Symbol Search Order*.

Symbol Tables at System and Owner Level

Global symbol tables are not version-controlled.

The symbol tables are:

```
SYSDBA / A
owner / A
```

Logging

Logging of symbol actions include the version of the table where the symbol was loaded from.

The generated comments in the Entire Operations JCL header contain the symbol table version of all used symbols.

Cross-References (XRef)

Symbol table versions will be considered.

Reporting

Symbol table versions will be considered.

Import/Export

Symbol table versions will be considered.

Exit Functionality (Symbol Table)

Entire Operations provides user exits and APIs that are related to symbols and support network versioning, for example, API `NOPUSY7N`.

Maximum Number of Versions per Symbol Table

The maximum number of symbol table versions can be limited system wide as described in *Defaults for Network Options* in the *Administration* documentation.

Maintaining the Usage of Symbol Table Versions

This function applies to symbol table masters only.

You can use the **Version Usage** function to determine a period (from/to date range) during which a defined symbol table version is used as the current symbol table for all scheduled job activations.

Note: You cannot use the **Version Usage** function to add a symbol table version. You can define a symbol table version when you create or copy a symbol table master.

This section covers the following topics:

- Listing, Defining and Deleting Version Usage
- Fields and Columns: Version Usage Definition

Listing, Defining and Deleting Version Usage

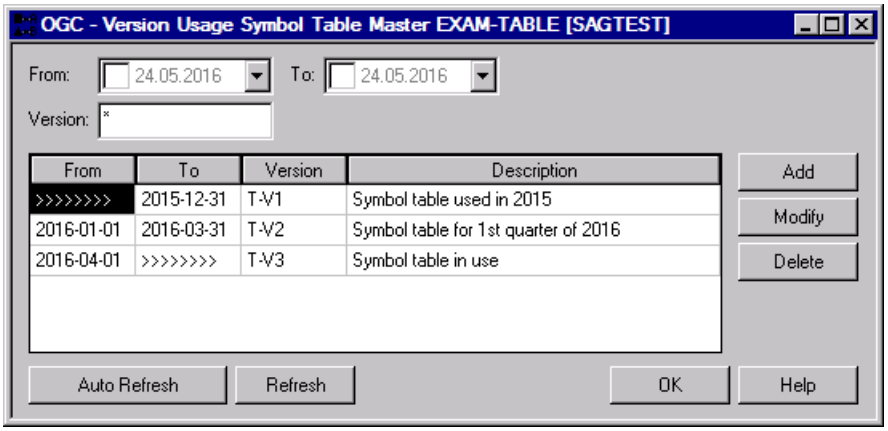
This section provides instructions for listing, defining and deleting version usage definitions for symbol table versions.

To list version usage definitions

1 In the object workspace, select a symbol table master.

2 Open the context menu and choose **Version Usage**.

A **Version Usage Symbol Table Master** window like the example below opens:



The window lists the version usage defined for the selected symbol table master version (the table is empty if no definition exists).

If you enter an asterisk (*) in the **Version** field, all version usages defined for all versions of the symbol table master are listed in the window as shown in the example above. The fields and columns are explained in *Fields and Columns: Version Usage Definition*.

The following functions are available:

Function	Description
Add	Adds a version usage definition: see *To add or modify a version usage definition*.
Modify	Modifies the usage definition selected in the table row. See *To add or modify a version usage definition*.

Function	Description
Delete	Deletes the usage definition selected in the table row. A window prompts you to confirm the deletion with **Yes** or cancel the action with **No**.
Auto Refresh/Refresh	See *Refreshing Object Lists*.

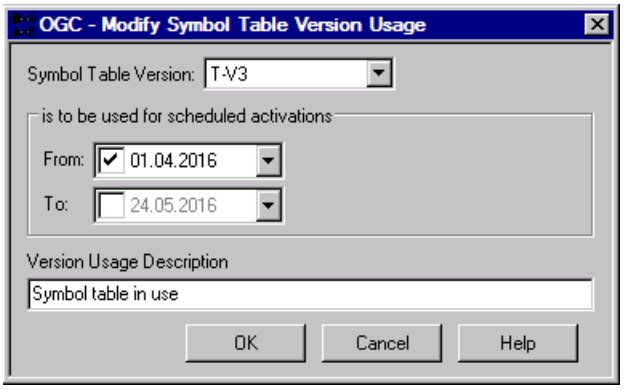
To add or modify a version usage definition

1 In the **Version Usage Symbol Table Master window**, choose **Add** to create a new definition.

Or:

In the **Version Usage Symbol Table Master window**, select the table row that contains the definition you want to modify, and choose **Modify**.

The input fields that appear in the window are shown in the following example:



2 Enter the required values or select a value from the drop-down list boxes. Activate or deactivate a **From** or **To** field by selecting the check box next to it. Deactivated fields are ignored and not processed. For additional information, see *Fields and Columns: Version Usage Definition*.

3 Choose **OK** when you are finished.

Fields and Columns: Version Usage Definition

Field/Column	Description
From (column or input field)	Start date of a default usage period of a network version. >>>>>>>> denotes that no start date has been defined. Possible value for an input field: A start date or no value (check box of field not selected) for no start date. You can open the drop-down list box to select a date from a calendar. Default: A start date is not defined. **Note:** Date ranges must not overlap.
To (column or input field)	End date of a default usage period of a network version. >>>>>>>> denotes that no end date has been defined; the usage is unlimited (infinite). Possible value for an input field: An end date or no date (check box of field not selected) for an unlimited time. You can open the drop-down list box to select a date from a calendar. **Note:** Date ranges must not overlap. Default: An end date is not defined.
Version (column)	Symbol table version for which the usage is defined.
Description (column)	Description of the version usage.
Symbol Table Version	Input field for the symbol table version to be used as the current version during the period specified in the **From/To** fields. Enter the required version or select a version from the drop-down list box.
Version Usage Description	Input field for a description of the version usage. Maximum input: 70 characters.

Saving Symbol Tables as Files

To save a symbol table master as a file

1 In the object workspace, select a **Symbol Table Master** instance and choose **Save as File** from the context menu.

2 Specify the file format and type as described in *Report Output Options* and choose **OK**.

An output file like the example below is created:

Master Symbol Table

Owner: SAGTEST Symbol Table: EXAM-TABLE

Symbol	MV	Symbol Value	Modified by	Time
CLASS	1	22.2	NATQA5	2016-05-12 14:18:10
CLASS	2	10.2	NATQA5	2016-05-12 14:18:10
CLASS	3	7	NATQA5	2016-05-12 14:18:10
JOBLIB		NOP.EXAMPLE.LOAD	IMPORT	2014-01-10 12:48:54
MSGCLASS		X	IMPORT	2014-01-10 12:48:54
STEPLIB		NOP.EXAMPLE.LOAD	IMPORT	2014-01-10 12:48:54
TESTSYMBOL		*** empty ***	NATQA5	2016-05-04 18:51:16

This output displays the selected symbol table master and all related information.

To save an active symbol table as a file

1 In a Symbol Prompting window, select **Save symbol table active as file** and choose **Continue**.

2 Specify the file format and type as described in *Report Output Options* and choose **OK**.

An output file like the example below is created:

Symbol Table Active

Owner: SAGTEST Network: SAG-PROMPT Run: 26 Symbol Table: EXAM-TABLET-V3

Symbol	MV	Symbol Value	Modified by	Time
CLASS		K	IMPORT	2014-01-10 12:48:54
JOBLIB		NOP.EXAMPLE.LOAD	IMPORT	2014-01-10 12:48:54
MSGCLASS		X	IMPORT	2014-01-10 12:48:54
STEPLIB		NOP.EXAMPLE.LOAD	IMPORT	2014-01-10 12:48:54
TESTSYMBOL		a	NATQA	2016-05-12 15:28:03

This output displays the symbol table active and all related information.

Deleting a Symbol Table Master

This function applies to symbol table masters only.

Restrictions:

- You cannot delete a symbol table master that is still used by one or more jobs. In this case, the error message `EOR1859 - Symbol Table ... used in Definitions` occurs. You can only delete the symbol table if you either specify another symbol table for these jobs or delete them. Proceed as described in *Listing Jobs and Networks that Use a Symbol Table Master* to find the jobs that use the symbol table.
- You cannot delete a symbol table master with a symbol table version that is still defined in the version usage of the symbol table. In this case, the error message `EOR2691 - Version ... - current usage found` occurs. You then need to delete the usage defined for the version as described in *Maintaining the Usage of Symbol Table Versions.*

To delete a symbol table master

1 In the object workspace, select a **Symbol Table Master** instance.

2 Choose **Delete** from the context menu, or press DELETE.

A confirmation window opens.

3 Choose **Yes** to confirm the deletion or **No** to cancel the action.

52 Defining Symbols and Symbol Values

- Available Functions: Symbol 606
- Listing Symbols of a Symbol Table 607
- Displaying a Symbol 607
- Modifying a Symbol 608
- Adding a Symbol 612
- Defining Multiple Symbol Values 613
- Specifying a Range Check for Numeric Symbol Values 615
- Reserved Symbols 616
- Predefined Symbols 620
- Symbols in Node Definitions 627
- Validating Symbol Values with a User Exit 628
- Global Symbol Modification Exit 629
- Symbol Setting triggered by the SYSOUT of a Job 629
- Deleting a Symbol 630

Available Functions: Symbol

To list all functions available for a symbol node

- For a symbol master: In the object workspace, select a **Symbol Master** node and open the context menu.

  Or:

  For an active symbol: In the object workspace, select a **Symbol Active** node and open the context menu.

  The following functions are available:

Function	Shortcut	Description
List	F8	**Lists symbols** contained in the symbol table.
New	CTRL+N	**Adds a symbol** to the symbol table.
Refresh	F5	See *Refreshing Object Lists*.
Filter	F3	See *Filtering Objects*.
Paste data	CTRL-V	See *Pasting Objects*.
Set Drag And Drop Function	---	See *Drag & Drop*.

To list all functions available for a symbol

- For a symbol master: In the object workspace, select an instance from the **Symbol Master** node and open the context menu.

  Or:

  For an active symbol: In the object workspace, select an instance from the **Symbol Active** node and open the context menu.

  The following functions are available:

Function	Shortcut	Description
Open	CTRL+O	**Modifies a symbol** definition.
Display	CTRL+D	**Displays a symbol** definition.
Delete	DELETE	**Deletes a symbol** definition.
Copy data	CTRL+C	Copies a symbol definition: see *Copying Objects*.
Set Drag And Drop Function	---	See *Drag & Drop*.

Listing Symbols of a Symbol Table

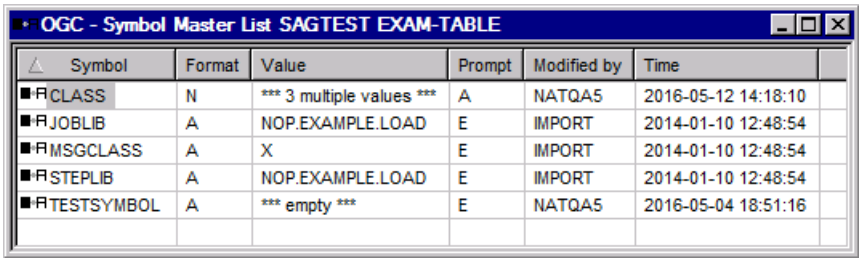
To list symbols for a selected symbol table

- For a symbol master: In the object workspace, select a **Symbol Master** node and choose **List** from the context menu, or press F8.

  Or:

  For an active symbol: In the object workspace, select a **Symbol Active** node and choose **List** from the context menu, or press F8.

  A **Symbol Master List** window like the example below opens:



Symbol	Format	Value	Prompt	Modified by	Time
CLASS	N	*** 3 multiple values ***	A	NATQA5	2016-05-12 14:18:10
JOBLIB	A	NOP.EXAMPLE.LOAD	E	IMPORT	2014-01-10 12:48:54
MSGCLASS	A	X	E	IMPORT	2014-01-10 12:48:54
STEPLIB	A	NOP.EXAMPLE.LOAD	E	IMPORT	2014-01-10 12:48:54
TESTSYMBOL	A	*** empty ***	E	NATQA5	2016-05-04 18:51:16

  The table contained in the window lists all symbol masters defined for the selected symbol table master (here: `EXAM-TABLE`).

  The columns in the window are explained in *Fields and Columns: Symbol Table*.

Displaying a Symbol

To display a symbol definition

1. For a symbol master: In the object workspace, select a **Symbol Master** instance.

   Or:

   For an active symbol: In the object workspace, select a **Symbol Active** instance.
2. Open the context menu and choose **Display**, or press CTRL+D.

   A **Display Symbol** window opens with the current symbol definition.

The read-only fields on the tabbed pages **Main**, **Multiple Symbol Values** and **Value Check** correspond to the fields and pages in the **Maintenance Symbol window**. They are explained in ***Fields: Symbol***.

Modifying a Symbol

To modify a symbol

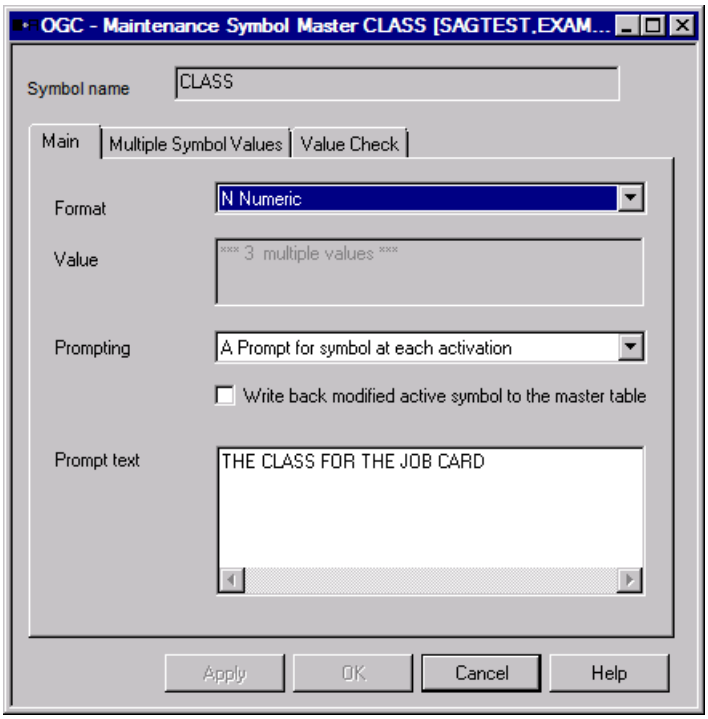
1 For a symbol master: In the object workspace, select a **Symbol Master** instance.

Or:

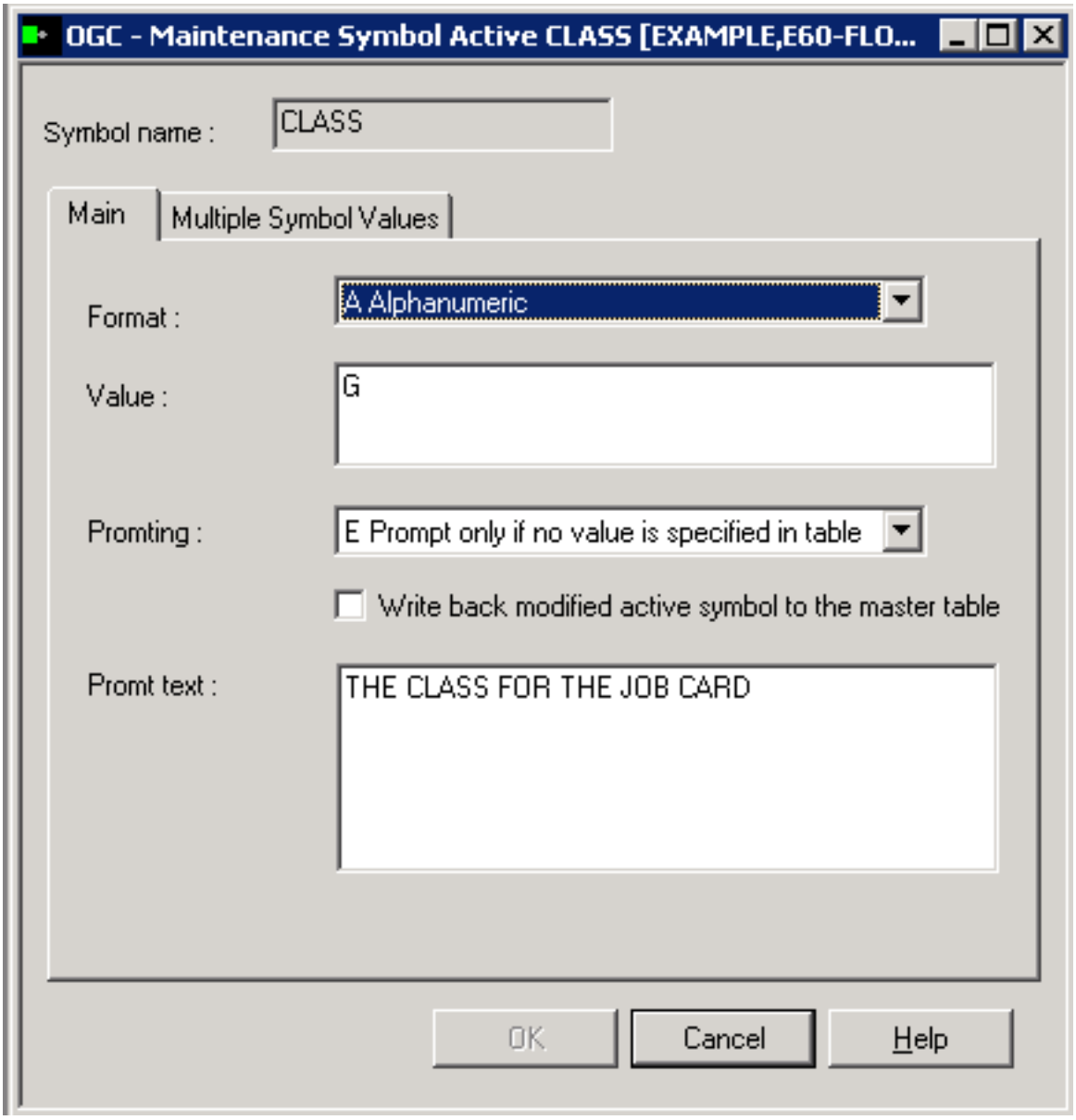
For an active symbol: In the object workspace, select a **Symbol Active** instance.

2 Open the context menu and choose **Open**, or press CTRL+O.

For a symbol master, a **Maintenance Symbol Master** window like the example below opens:



For an active symbol, a **Maintenance Symbol Active** window like the example below opens:



3 Make your changes. The fields on the tabbed pages **Main**, **Multiple Symbol Values** and **Value Check** are explained in *Fields: Symbol*.

4 Choose **OK** when you are finished.

The symbol is modified and saved.

This section covers the following topics:

- Fields: Symbol

Fields: Symbol

The fields on the tabbed pages **Main**, **Multiple Symbol Values** and **Value Check** (symbol masters only) of a **Symbol Master** or **Symbol Active** window are explained in the following table.

The fields are either write-protected or available for input depending on whether you view or modify the symbols. The table indicates when a field applies to active symbols or symbol masters only.

Caution: Be careful with insertions and deletions, since the lines of **Value** are concatenated to one symbol value.

Main Page:		
Symbol	Name of the symbol.	
Format	Format of the symbol (variable). Possible values:	
	A	Alphanumeric (including special characters).
	D	Date in the format YYYYMMDD (see also *Date and Time Formats*).
	H	Alphanumeric; hidden. ■ The symbol value is not visible in the **Value** field. ■ In lists and logs, the symbol value is displayed as `*** hidden ***`.
	L	Alphanumeric; lower case.
	N	Numeric. See **Value** for the numeric field format.
	U	Alphanumeric; upper case.
Value	Value of the symbol to be substituted. To enter a fixed length or an empty string, include the value in quotes. Use two apostrophes to get a quote within a string. Example: `'Fixed Length '` **Note:** 1. If a value is longer than 40 characters, continue typing in the second **Value** line. The maximum value is 120 characters. Be careful with insertions and deletions. The lines are concatenated to one symbol value.	

<table>
<tr><td></td><td colspan="2">2. The maximum length of a numeric value is N10.4 (10 digits before the decimal point and 4 digits after the decimal point). A decimal point can appear at any position.
3. A date must use the format YYYYMMDD.
4. The value of a hidden symbol (format H) can be entered, but it will be invisible.

See also Defining Multiple Symbol Values.</td></tr>
<tr><td>Prompting

(list box)</td><td colspan="2">Specifies whether the user is to be prompted for this symbol during manual activation of the job network.

Possible selection options:</td></tr>
<tr><td></td><td>A</td><td>Prompt for symbol during each activation.</td></tr>
<tr><td></td><td>E</td><td>Prompt only if no value is specified in the table.</td></tr>
<tr><td></td><td>N</td><td>Never prompt for symbol.</td></tr>
<tr><td></td><td colspan="2">See also Symbol Prompting during Network or Job Activation.</td></tr>
<tr><td>Prompting

(check box)</td><td colspan="2">If the check box next to Prompting is selected, changes to an active symbol are written back to the symbol master as well.

This applies when modifying an active symbol and if you are permitted to modify a symbol table master. See also Performing Maintenance Functions.

This option does not apply to add, copy or delete operations; symbols in the symbol table master then remain as they are.

Note: The symbol is also written back if the Rewrite prompted symbols to master symbol table option is activated in the Entire Operations default settings: see Defaults for Other Settings in the Administration documention.</td></tr>
<tr><td>Prompt Text</td><td colspan="2">Optional text to be displayed when symbol prompting is performed, to help the user specify the correct value.

See also Symbol Prompting during Network or Job Activation.</td></tr>
<tr><td colspan="3">Multiple Symbol Values Page:</td></tr>
<tr><td>Values 1 to 150</td><td colspan="2">Assign multiple values to a symbol.

These can be used in subsequent replacements of the same symbol. See also Defining Multiple Symbol Values.

You can enter up to 150 values here. The maximum value length is 120. These values are used for subsequent substitutions.</td></tr>
<tr><td colspan="3">Value Check Page:</td></tr>
<tr><td colspan="3">This option applies to symbol masters only.

Used to define a number range for numeric symbol values to be checked for valid input.

See also Specifying a Range Check for Numeric Symbol Values.</td></tr>
</table>

None	Select this button (default) if you neither use **Range check** nor **User exit**.
Range check	Select this button if you want to define and/or activate a range of numbers to be checked during input of numeric symbol values.
Value from	Start number (**Value from**) and end number (**Value from**) of the range to be checked. Valid input values are positive integer numbers in the range from `1` to `9999999999` (format N10).
Value to	
User exit	Select this button if you want to specify and/or activate a user exit.
Library	Name of the library that contains the user exit. **Note:** You cannot select the libraries SYSLIB and SYSLIBS. They are reserved for internal use.
User Exit	Name of the user exit. If the name entered in **Library** is valid, you can select a user exit from the drop-down list box. If the specified exit does not (yet) exist, you will be warned with an appropriate message.

Adding a Symbol

You can add a symbol by either copying an existing symbol (see the **Copy data**/**Paste data** functions for a symbol instance from a symbol node) or creating a new symbol with the **New** function.

This section provides instructions for using the **New** function.

To add a symbol master

1 In the object workspace, select the **Symbol Master** node.

2 From the context menu, choose **New**, or press CTRL+N.

A **Create new Symbol** window opens.

The input fields on the tabbed pages **Main**, **Multiple Symbol Values** and **Value Check** correspond to the fields and pages in the **Maintenance Symbol Master window**. They are explained in ***Fields: Symbol***.

3 Enter a name for the symbol and make the required definitions.

4 Choose **OK** when you are finished.

The symbol master is created and now ready to use in the network.

To add an active symbol

1 In the object workspace, select a **Symbol Active** node.

2 Open the context menu and choose **New**, or press CTRL+N.

A **Create new Symbol Active** window opens.

The fields on the tabbed pages **Main** and **Multiple Symbol Values** of the window correspond to the fields of the **Maintenance Symbol Active window**. They are explained in ***Fields: Symbol***.

Fill in the fields as required.

3 When you are finished, choose **OK**.

The active symbol is created and now ready to use in the active run.

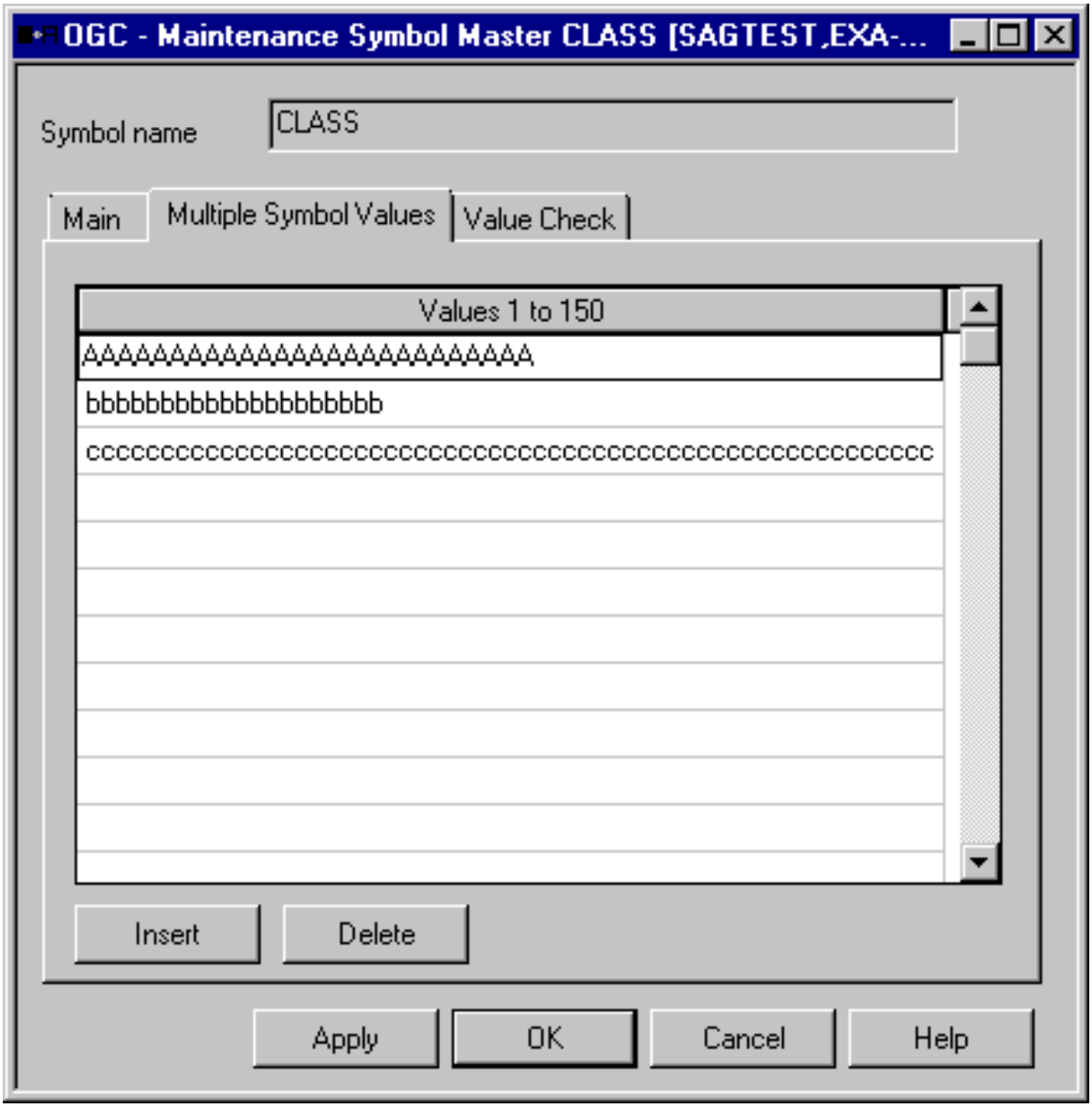
Defining Multiple Symbol Values

You can give a symbol several values which can be used in subsequent substitutions of the same symbol. These values can be used for the multiple parallel activation of a job. The number of parallel jobs is the same as the number of multiple values defined here. The maximum number of multiple values is 150.

To view and define multiple symbol values

1 In a **Symbol Master** or **Symbol Active** window, open the tabbed page **Multiple Symbol Values**.

A **Multiple Symbol Values** page like the example below opens:



All values currently defined for the selected symbol are listed in a table. The table is empty if no values exist.

2 You can add a new value in an empty table row at the end of the table. If you want to insert an empty row, select the row above which you want to place it and choose **Insert**.

Note: Incorrect value definitions (wrong format/length) or values that are not within a specified value range appear in red. For valid input values, see the description of the **Value** field and the **Value Check** page in the section *Fields: Symbol*.

If you want to delete an entry, select the appropriate row and choose **Delete**.

This section covers the following topics:

- Symbol Replacement

Symbol Replacement

You can use the **symbol function** `!MV` **or** `?MV` (see *Functions for Symbol Replacement*) to get single occurrences of multiple-value symbols for replacement.

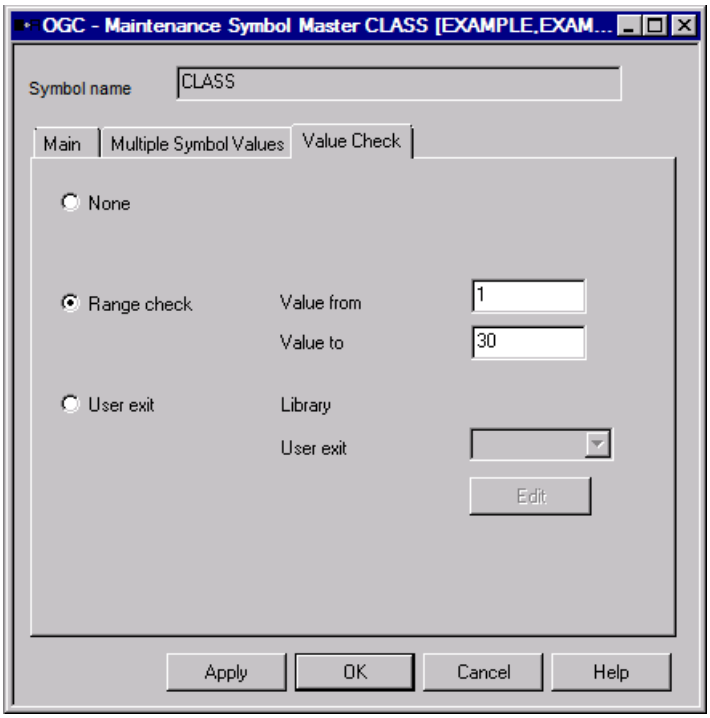
Specifying a Range Check for Numeric Symbol Values

You can guarantee that a numeric symbol value is within a specified number range.

To specify a range check for numeric symbol values

1 Open the **Value Check** page of a **Symbol Master window**.

2 Select the **Range check** button (default is **None** for no check) and enter a start number in the **Value from** field and an end number in the **Value to** field as shown in the following example of a **Value Check** page:

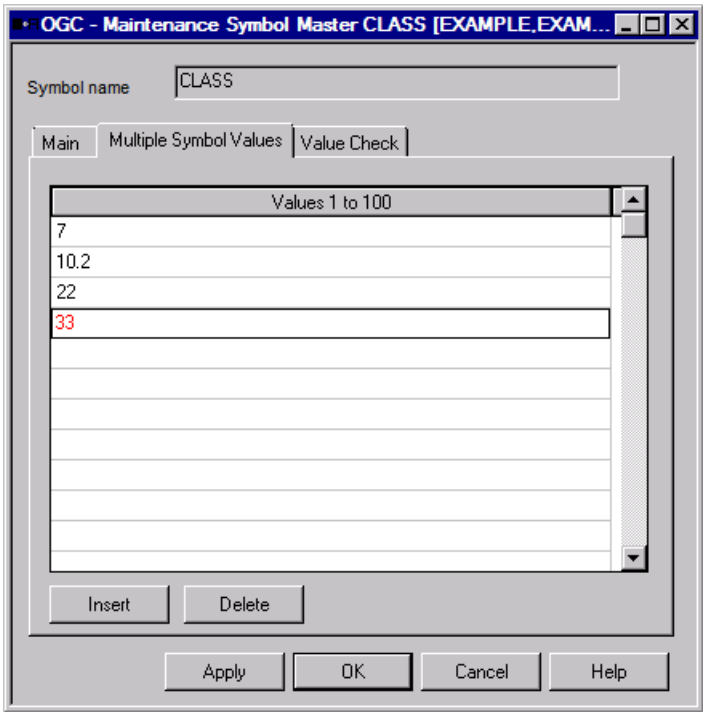


For valid input values and further information on range checks, see **Value Check** in the section ***Fields: Symbol***.

3 Choose **OK** to save your entries.

You can deactivate the check by selecting the **None** button.

4 If you open the **Multiple Symbol Values** page and add a new value that is not within the specified range (here: `1` to `30`) the new (incorrect) value (here: `33`) appears in red:



Reserved Symbols

Reserved symbols perform user-specified tasks. They can be changed as required if symbol table maintenance is allowed for the user (see *Performing Maintenance Functions*).

Reserved symbols can be added, modified and deleted in symbol tables like all other symbols.

This section covers the following topics:

- Reserved Symbols for Command Line Parameters
- Reserved Symbols for Mainframe Platforms
- Reserved Symbols for Profile Scripts (UNIX)

- Reserved Symbol for Epilog Scripts (UNIX and Windows)

Reserved Symbols for Command Line Parameters

Symbol	Format	Description
CMDLINE-*job*	A80	Command line parameter(s) for the following **job types** and/or **JCL locations**: ■ Started tasks (job type STC) on z/OS, ■ UNIX shell scripts (job type JOB, ■ Windows BAT files (job type JOB) and PowerShell scripts (job type WPS), ■ Windows services (job type SRV), and ■ Programs which are to be executed directly under UNIX or Windows (JCL location EXE).

Passing Command Line Parameters

If you want to pass command line parameters for a job, use the reserved symbol CMDLINE-*job* in the job's symbol table, where *job* is the job name.

Several parameters, separated by blanks, can be contained in this symbol.

Example:

Job name:	JOB1
Member name in JCL definition:	STC001
Content of symbol CMDLINE-JOB1:	PARM1=Y,PARM2=N

The start command generated and issued by the Entire Operations Monitor is:

```
S STC001,PARM1=Y,PARM2=N
```

Command Line Passing to Shell Scripts in Entire Operations

- If a job is to be activated, Entire Operations checks whether a CMDLINE-*job* symbol exists.

  If such a symbol exists, its content is appended to the invocation of the *job*.bat script.
- Nested symbol replacement:

  The CMDLINE-*job* symbol can contain nested symbols.

  These can be prefixed by the **activation escape character** for activation time replacement, or by the **submission escape character** for submission time replacement.
- The *job*.sh resp. *job*.bat script (the user's script, with some Entire Operations header files), is not invoked directly, but out of the wrapping script *job*.*nnnnnnnn*.frame.sh resp. *job*.*nnnnnnnn*.frame.bat.

nnnnnnnn denotes an 8-digit repeat count. For the first job submission, this is `00000000`.

- The *job*`.nnnnnnnn.frame.sh` resp. *job*`.nnnnnnnn.frame.bat` script is generated by Entire Operations at submission time. On UNIX, this is always a Bourne shell script (starting with the shebang `#!/bin/sh`), regardless of the shell of the user's script.
- UNIX: Command line parameters must be passed in Bourne shell syntax. This means quoting and masking of special characters according to Bourne shell rules. Note that the Bourne shell script quotes from the parameters quoted in the invoked shell script.
- Use *c:\\work\\test.txt* to pass a windows file name, for example, *c:\work\test.txt*.
- The maximum total length of all parameters is 2000 bytes (including separating blanks) after symbol replacement.
- The maximum length of a single parameter is 240 bytes.

Reserved Symbols for Mainframe Platforms

Symbol	Format	Description
`SYSOUT-NODE-GLOBAL`	N5	(Only in the table `SYSDBA/A`.) If this SYSOUT exists and contains a valid Entire System Server node number and if there is no other applicable SYSOUT node definition available, SYSOUT file copies for Entire Output Management are created on this node.
`JCL-NODE`	N5	If this symbol is found during job activation, this node is used as a JCL node for the active job.
`SUBMIT-NODE`	N5	If this symbol is found during job activation, this node is used as a job start node for the active job.
`SYSOUT-NODE`	N5	If this symbol is found during job activation, this node is used as a SYSOUT node for the active job. This symbol supersedes `SYSOUT-NODE-GLOBAL`.

The symbols `JCL-NODE`, `SUBMIT-NODE` and `SYSOUT-NODE` are searched for in the symbols tables as described in ***Symbol Table Types and Symbol Search Order***.

Reserved Symbols for Profile Scripts (UNIX)

This section describes the reserved symbols used when generating UNIX frame scripts (*job*.current.frame.sh) that source profile scripts as described in ***UNIX: Usage of Profile Scripts*** in the section *Job Maintenance*.

The reserved symbols are defined with **Format** A.

Symbol	Description	
ETC-PROFILE	Determines the profile scripts to be sourced. Possible symbol **values** are:	
	Y	Source the profile scripts contained in */etc/profile* and */etc/profile.local*, if available.
	N	Do not source any of the profile scripts contained in */etc/profile* and */etc/profile.local*. This setting has the same effect as if the ETC-PROFILE was not defined at all. The symbol is then not found in the symbol table hierarchy. If you do not want to source profile scripts, make sure that ETC-PROFILE set to Y is not defined at a higher level in the symbol table hierarchy (for example, SYSDBA/A). In this case, you must explicitly set ETC-PROFILE to N at a lower hierarchy level. See also ***Symbol Table Types and Symbol Search Order***. N is the default when no value is specified for ETC-PROFILE.
ENV	*script-name*	Source the user-defined profile script defined for this symbol. Example: $HOME/.profile

Reserved Symbol for Epilog Scripts (UNIX and Windows)

This section describes the reserved symbol used for executing epilog scripts for UNIX or Windows jobs.

Symbol	Format	Description
NOP-EPILOG-*execution_node*	A120	Fully-qualified name of an epilog script file. Example: NOP-EPILOG-517 Path to the shell file: /home/sag/jcl/epilog.node517.sh

Symbol Table Search for Epilog Scripts

Entire Operations searches for NOP-EPILOG-*execution_node* in the symbol table hierarchy each time a job is submitted on a UNIX or Windows execution node.

The epilog script is not executed if NOP-EPILOG-*execution_node* cannot be found in any symbol table or if the value supplied with the reserved symbol is empty, contains a dash (-) or is set to NO.

If the reserved symbol is contained in the topmost symbol table (`SYSDBA/A`), the respective epilog script is executed for all UNIX and Windows jobs, except for those that reference a reserved symbol contained in a lower-ranking symbol table.

Using Epilog Scripts

The shell script defined with `NOP-EPILOG-`*`execution_node`* must have executable rights.

The epilog script is invoked by the defined JCL frame script and starts to execute when the main script terminates.

If an epilog script is defined but cannot be found, the exit code of the job is set to 102 so that the whole job ends in error.

The epilog script does not accept any parameters. The epilog script can obtain information from environment variables set in the JCL frame script by using reserved symbols (see ***Reserved Symbols for UNIX and Windows Environment Variables***).

Symbol replacement is not performed for the epilog script. The epilog script is directly executed by the frame script, without intermediate storage in Entire Operations.

A Windows epilog script must not end with an `exit` statement. Set an exit code as follows:

```
set errorlevel=exit_code
```

If the exit code defined in the epilog script is higher than the exit code of the frame script, the exit code of the main script is set to the exit code of the epilog script.

Predefined Symbols

Predefined symbols perform special-purpose tasks for example, within JCL files or scripts. They are defined by Entire Operations and cannot be modified.

All symbol names starting with an asterisk (`*`) or `P-` are reserved for predefined read-only symbols. These prefixes must not be used for user-defined symbols.

Predefined symbols must not be defined in symbol tables.

This section covers the following topics:

- Table of Predefined Symbols
- Predefined Symbols for Multiple and Parallel Activations

- Predefined Symbols for Subnetworks and Recovery Jobs

Table of Predefined Symbols

The following symbols are read-only. They can be used without defining them in a symbol table.

For further explanations of date and time formats, see also ***Date and Time Formats***.

Symbol	Format	Description
*DATD	A8	Date in the format DD.MM.YY.
*DATE	A8	Date in the format DD/MM/YY.
*DATG	A15	Date in the format DD*monthname*YYYY (Gregorian date).
*DATI	A8	Date in the format YY/MM/DD.
*DATJ	A5	Date in the format YYDDD (Julian).
*DATN	N8	Date in the format YYYYMMDD.
*DATU	A8	Date in the format MM/DD/YY.
*DAT4J	A7	Date in the format YYYYDDD (Julian).
*TIME	A10	Time of day in the format HH:II:SS.T.
*TIME8	A8	Time of day in the format HH:II:SS.
*TIMN	N7	Time of day in the format HHMMSST. See also the symbol function !TIMN. This function allows the usage of constant time values. For single-digit hours (if before 10:00), the leading zero (0) will be contained.
*TIMN6	N6	Time of day in the format HHMMSS. For single-digit hours (if before 10:00), the leading zero (0) will be contained.
*TIMA6	A6	Time of day in the format HHMMSS. For single-digit hours (if before 10:00), the leading zero (0) will be contained.
*TIMA7	A7	Time of day in the format HHMMSST. For single-digit hours (if before 10:00), the leading zero (0) will be contained.
P-ACT-USER	A8	For manual activations: user ID of the user who activated the network or job. For subnetwork activations: activation user ID of the calling (parent) job.

Symbol	Format	Description	
		For scheduled activations: default user ID of the Entire Operations Monitor.	
P-ACT-ORIGIN	A1	Contains a letter that indicates the activation origin:	
		A	Activated by API.
		E	Activated by EOJ action.
		M	Activated manually.
		R	Recovery.
		S	Activated by schedule extraction.
		U	Activated by API, as a subnetwork.
P-DATE	A8	Execution date in the format YYYYMMDD. The date of the previous day is transmitted if the end time of the previous production day has not yet been reached.	
P-ESC-ACT	A1	Activation escape character (see *Symbol Escape Characters*), as defined for the current object (for example, a job). **Note:** 1. The value of P-ESC-ACT is not changed by the #EOR-ESC-ACT meta statement if defined in the JCL (see *Definitions in JCL and Macro Sources*). 2. P-ESC-ACT can also be used as a predefined variable in meta statements defined in the macro JCL.	
P-ESC-SUB	A1	Submission escape character (see *Symbol Escape Characters*), as defined for the current object (for example, a job). **Note:** 1. The value of P-ESC-SUB is not changed by the #EOR-ESC-SUB meta statement if defined in the JCL (see *Definitions in JCL and Macro Sources*). 2. P-ESC-SUB can also be used as a predefined variable in meta statements defined in the macro JCL.	
P-EXECUTION-NODE	N5	Execution node.	
P-JCL-FILE	A250	JCL file. If not applicable, an empty string is returned. P-JCL-FILE is always resolved to a fully-qualified file name. **Nested replacement** of symbols is performed.	

Symbol	Format	Description

		If the JCL node is a Windows node, backslashes (\) in the file name can be returned by slashes.
		
		For further usage in Windows BAT and Powershell JCL, the string +F+&P-JCL-FILE is to be used (with & as the escape character).
		
		Windows example, assuming the ampersand (&) is the escape character:
		
		`P-JCL-FILE = e:/sag/nop/jcl/job1.bat`
		
		`type +F+&P-JCL-FILE`
		
		is resolved to
		
		`e:\sag\nop\jcl\job1.bat`

P-JCL-MEMBER		JCL member. If not applicable, an empty string is returned.
P-JCL-NODE	N5	JCL node.
P-JOB	A10	Job.
P-JOB-ID	A10	Job ID (job number, BS2000 TSN, process ID).
P-JI	A10	Job ID (job number, BS2000 TSN, process ID).
P-MUL	A8	User who made last change (in lower case).
P-MUU	A8	User who made last change (in upper case).
P-NADIR	A250	Directory for temporary files for this network (UNIX and Windows only). **Note:** 1. On Windows nodes, the directory name contains backslashes (\). For the usage on mainframe (EBCDIC) environments, these backslashes will be returned **encoded in trigraphs** (see the relevant section in *Submission of Jobs by Entire Operations*). 2. The content of P-NADIR can be subject to change in different Entire Operations versions.
P-NETWORK	A10	Network.
P-NETWORK-VERSION	A10	Network version.
P-NODE	N5	Execution node.
P-OWNER	A10	Owner of the network.

Symbol	Format	Description	
P-REPEAT or P-REPEAT-COUNT	N10	Repeat count. Contains the number of the current repetition of the job (for example, if the job was resubmitted). **Note:** These symbols will be replaced with a correct value only if they are preceded by the **submission escape character**.	
P-RUN	N5	Run number of the network.	
P-RUN5	N5	Run number of the network, always with 5 digits (with leading zeros). For example: The run number 7 will be returned as 00007. This symbol is also available in the macro JCL.	
P-SGL	A50	UNIX group or Windows domain (in lower case) of the job.	
P-SGU	A50	UNIX group or Windows domain (as defined) of the job.	
P-SUL	A50	Job start user ID (in lower case).	
P-SUU	A50	Job start user ID (in upper case).	
P-SUBMIT-ID	A50	Job start user ID (as defined).	
P-SUBMIT-GRP	A50	UNIX group or Windows domain (as defined) of the job.	
P-SYSF1	A10	Concatenated DBID (database ID) and FNR (file number) of Entire Operations System File 1. Example: Entire Operations System File 1 has the LFILE setting LFILE=(216,9,17). P-SYSF1 will be set to 0000900017.	
P-SYSOUT	A250	Name of the SYSOUT file. This variable and its effective (usable) size depend on the operating system in which the job is executed:	
		BS2000	Available any time after activation. The effective size is 54.
		z/OS: Not available. The value is set to three consecutive periods (. . .).	

Symbol	Format	Description	
		UNIX	Available any time after activation.
		Windows	Available any time after activation. The file name is returned in UNIX syntax.
P-SYMTAB	A10	Symbol table.	
P-SYMBOL-TABLE	A10	Symbol table.	
P-SYMTAB-VERSION	A10	Symbol table version.	

Predefined Symbols for Multiple and Parallel Activations

Symbol	Format	Description
PMPA	A5	Content of the suffix symbol of a **multiple-value symbol**. This symbol can be used in a master JCL node, an execution node and a SYSOUT node definition, prefixed with the **activation escape character**. For an execution node, you can also use the **submission escape character** as a prefix. This allows the execution of multiple parallel jobs on different nodes. The multiple-value symbol must be an array of valid node numbers or mnemonic node short names. See also P-MPA.

P-MPA	A50	Full current value of the symbol containing the job name suffix if multiple jobs are activated in parallel.
		
		To redefine this field, code:
		
		`§ 1 P-MPA (A50)`
		`§ 1 REDEFINE P-MPA ...`
		`#GET-SYMBOL P-MPA`
		
		See also PMPA.

P-MPI	N3	Numeric index of the multiple-value symbol used for multiple jobs activated in parallel.

Symbol	Format	Description
P-SUFFIX or P-X	A10	Job name suffix if multiple and parallel job activation is in use. You can find detailed information about multiple and parallel job activation in the section *Job Maintenance*. P-X is an abbreviation for P-SUFFIX.

Predefined Symbols for Subnetworks and Recovery Jobs

The table below lists and describes the symbols a subnetwork or recovery job requires to communicate with its invoking network or the calling job, respectively.

These symbols have the value of a blank, unless they are used in a subnetwork or a recovery job.

Symbol	Format	Description	Rec. Job	Subnet
P-C-EXEC-NODE	N5	Execution node of the calling job.	X	n/a
P-C-JCL-NODE	N5	JCL node of the calling job.	X	n/a
P-C-JOB	A10	Name of the calling job.	X	X

P-C-MPA	A50	The full current value of the symbol, containing the suffix of the job which invoked the subnetwork if multiple and parallel job activation is in use in the calling network. To redefine this field, code:
		
		`§ 1 P-C-MPA (A50)`
		`§ 1 REDEFINE P-C-MPA ...`
		`#GET-SYMBOL P-C-MPA`
			X	n/a

P-C-NETWORK	A10	Network of the calling job or invoked subnetwork.	X	X
P-C-NETWORK-VERSION	A10	Network version of the calling job or invoked subnetwork.	X	X
P-C-OWNER	A10	Owner of the calling job or invoking network.	X	X
P-C-RUN	N5	Run number of the calling job or invoking network.	X	X
P-C-RUN5	N5	5-digit run number (with leading zeros) of the calling job or invoking network. For example: Run number 7 will be returned as 00007. This symbol is also available in the macro JCL.	X	X

Symbol	Format	Description	Rec. Job	Subnet
P-C-SUFFIX	A10	Suffix value of the job which invoked the subnetwork where the symbol is used.	X	X
P-C-SUG	A20	Submit group of the calling job. It is possible to define this symbol as the submit group in the jobs of the subnetwork, preceded by the activation escape character. (The subnetwork jobs must have a symbol table definition.)	X	n/a
P-C-SUU	A20	Submit user ID of the calling job. It is possible to define this symbol as the submit user ID in the jobs of the subnetwork, preceded by the activation escape character. (The subnetwork jobs must have a symbol table definition.)	X	n/a
P-C-SYMTAB	A10	Symbol table of the calling job or invoking network.	X	X
P-C-SYMTAB-VERSION	A10	Symbol table version of the calling job or invoking network.	X	X

Symbols in Node Definitions

In general, you can define all nodes used in network and job definitions as symbols. Extra rules and restrictions for nodes are described in the following section.

You can define a node as a symbol of up to 4 characters that is preceded by the **activation escape character**, for example, §NODE. Make sure that the **Use symbol as** option is selected in the respective node definition.

You can also use the predefined symbol PMPA preceded by the **activation escape character**, for example, §PMPA. See also ***Predefined Symbols*** in the section *Symbol Table and Symbol Maintenance*.

This section describes restrictions and special considerations that apply when using a symbol for a node:

- Using Symbols in Message Sending Nodes

Using Symbols in Message Sending Nodes

Symbol use for message sending nodes is currently only possible on the Entire Operations CUI client.

Caution: Using symbols for message nodes on Entire Operations GUI client can produce severe errors.

When using the activation escape character as a symbol prefix, the symbol is evaluated immediately, and the message can be sent at job activation time.

Validating Symbol Values with a User Exit

You can define a user exit to check the validity (plausibility) of symbol values during symbol modification or prompting.

For detailed information on writing a user exit, see ***User Exits for Validation Checks of Symbol Values*** in the section *User Exits*.

To check symbol values with a user exit

1 Open the **Value Check** page of a **Symbol Master window**.

2 Select the **User exit** button (default is **None** for no check) and enter the name of a library and a user exit in the fields **Library** and **User exit**, respectively. The input fields are described in ***Fields: Symbol***.

Choose **Edit** if you want to modify the source of the user exit. Remember that you need to recatalog (`STOW` command) the source after modification.

3 Choose **OK** to save your entries.

4 Select the **None** button if you can deactivate the user exit.

Global Symbol Modification Exit

For active symbol modification, a global symbol modification exit (described in the *Administration* documentation) can be defined. This global exit is only used if there is no specific exit defined on the network level. The global exit is called with the same parameter list as the exit on the network level.

Symbol Setting triggered by the SYSOUT of a Job

This option applies to UNIX and Windows only.

It is possible to set symbols during a job's End-of-Job checking, triggered by a meta command in the job's SYSOUT. The meta command is `EOR-SYMBOL`.

With this meta command, it is possible to set an Entire Operations symbol during the End-of-Job checking of a job.

The meta command is available for jobs running on UNIX or Windows.

The symbol will be set into the active job's active symbol table or symbol table master. If the active job has no active symbol table, the job result is set to `not ok`.

Syntax:

EOR-SYMBOL SA *symbol=value*

Sets an active symbol.

EOR-SYMBOL SM *symbol=value*

Sets a symbol master.

The meta command can be generated, for example, with an echo command:

```
echo "EOR-SYMBOL SA SYMBOL1=$HOSTNAME"
```

With `HOSTNAME=pcsn` this produces the SYSOUT line:

```
EOR-SYMBOL SA SYMBOL1=pcsn
```

During End-of-Job checking, the symbol `SYMBOL1` in the job's active symbol table will be set to `pcsn`.

The echo command can be generated dynamically by using symbol replacement. Therefore, this kind of symbol setting is very flexible.

Example:

```
echo "EOR-SYMBOL SA PID.&*TIMN=$$"
```

creates the SYSOUT line:

```
EOR-SYMBOL SA PID.1131185=4837
```

Deleting a Symbol

The options provided to delete a symbol master or an active symbol from a symbol table are described in the following section.

To delete a symbol

1 For a symbol master: In the object workspace, select a **Symbol Master** instance.

Choose **Delete** from the context menu, or press DELETE.

Or:

For an active symbol: In the object workspace, select a **Symbol Active** instance.

2 Choose **Delete** from the context menu, or press DELETE.

A confirmation window opens.

3 Choose **Yes** to confirm the deletion or **No** to cancel the action.

Note: You can delete individual symbols from a symbol table even if the table is specified for one or more MAC-type jobs. If the symbol table is specified for any job, the deleted symbol cannot be substituted when the JCL is dynamically generated. The Monitor records this event by writing a message to the log.

53 Symbol Replacement

- Symbol Escape Characters 632
- Symbol Prompting during Network or Job Activation 635
- Displaying Descriptions of Prompted Symbols 638
- Repeating Symbol Prompting 639
- Specifying User Exits for Symbol Modification 639
- Rules and Restrictions for Symbol Replacement 642

Symbols are replaced during network or job activation or during job submission:

- Symbols starting with the **activation escape character** are resolved during JCL loading, as part of the job activation.
- Symbols starting with the **submission escape character** are resolved during job submission.



Note: If a symbol replacement error occurs at submission time, the event JNR `Job not run - JCL error` will be set to `occurred`. If the event JNR is not defined, it will be added to the active job.

Note for BS2000:

BS2000 SYSJV references (for example, `&($SYSJV.TSN)`) are no longer treated as Entire Operations symbols, regardless of whether they are prefixed with an ampersand (`&`) denoting an activation escape or submission escape character.

Symbol Escape Characters

In JCL and Natural macro sources, the symbols to be substituted must be preceded by an escape character.

You can use different escape characters to determine whether symbol replacement is performed during JCL generation or job submission.

The following escape characters can be used in parallel in your Entire Operations environment:

- **Activation Escape Character**

  Symbols preceded by the activation escape character are replaced at activation of a network or job.

  If a JCL file contains the activation escape character, a symbol replacement is performed:

  - During JCL editing (temporary, using the symbol table master).
  - During JCL loading (permanently, using the active symbol table).
- **Submission Escape Character**

  Symbols preceded by the submission escape character (also called submit escape character) are replaced during submission of a job. They can be used to pass symbol values that cannot be specified earlier when the job is activated.

Both activation escape and submission escape characters can be defined as system-wide defaults in the Entire Operations default settings (see *Defaults for Network Options* in the *Administration* documentation). However, different escape characters can be used for each job, if necessary.

You can also specify escape characters in your JCL or macro source. See also *Defining Escape Characters*.

Caution: If you change escape character definitions for existing JCL, symbols to be replaced can no longer be detected. In this case, consider defining escape characters in your JCL or macro source.

This section covers the following topics:

- Recommended Escape Characters
- Defining Escape Characters

Recommended Escape Characters

We recommend that you use the following escape characters:

Operating System	Escape Characters
z/OS	§ (section sign) and $ (dollar sign).
BS2000	^ (circumflex accent) and ` (grave accent). Do not use $ (dollar sign), § (section sign) and @ (commerical at sign) because they have a special meaning in BS2000.
UNIX, Windows	No recommendation, except for UNIX: Do not use $ (dollar sign) because it has a special meaning in UNIX.

Caution: We strongly recommend that you do not define escape characters already used by other (operating system specific) JCL features, for example, parameters for DO procedures and SUBDTA characters in BS2000.

Defining Escape Characters

This section describes the options provided to define your default activation and submission escape characters.

- **Default Settings**

  The **Network Options** page of the **Entire Operations Defaults** function (see the *Administration* documentation) is used to define the default for all networks in your Entire Operations environment.

  On the **Operating System Specials** page, you can also define the default escape characters for each operating system.

- **Network Definition**

  Defines the default escape characters to be used by all jobs of a network.

- **Job Definition**

  Defines the default escape characters to be used by an individual job.

- **Definitions in JCL and Macro Sources**

  In JCL and Natural macro sources, the symbols to be substituted must be preceded by an escape character.

  You can define default escape characters in Entire Operations JCL or macro sources by coding one or both of the following meta statements as the first source line(s):

```
#EOR-ESC-ACT = activation-character
#EOR-ESC-SUB = submit-character
```

  For example:

```
#EOR-ESC-ACT = §
#EOR-ESC-SUB = ^
```

  The `#EOR-ESC-ACT = activation-character` meta statement is evaluated when the JCL containing this statement is loaded. *activation-character* replaces the current activation escape character in the current file. This replacement influences symbol replacement only, not a meta statement defined in a macro (Natural statement). The `#EOR-ESC-ACT` meta statement is removed from the active JCL after loading.

  The `#EOR-ESC-SUB = submission-character` meta statement is evaluated during job submission. *submission-character* replaces the current submission escape character in the current file. This replacement influences symbol replacement only, not a meta statement defined in a macro (Natural statement). The `#EOR-ESC-SUB` meta statement is kept in the active JCL after loading.

  Both meta statements are logged when they are evaluated during JCL loading or job submission.

  If you add a meta statement to a Natural macro source, you must recatalog the source with the `MACRO` command.

  Note about Trigraphs:

  Trigraph encoding uses question marks (`?`) as escape characters in the JCL definition. You can avoid decoding conflicts by **switching trigraph encoding off and on** in the JCL as described in *Trigraph Encoding for JCL Submission on UNIX and Windows*.

Symbol Prompting during Network or Job Activation

Entire Operations provides a standard method for symbol prompting to replace symbols during network activation. Standard symbol prompting is not used in jobs of the type `NET` (subnetworks).

If any of the activated jobs are jobs with JCL location `MAC` or jobs of the type `JOB` whose JCL contains symbols for variable values, these symbols are replaced by their current values during the manual activation, except those that are defined to be replaced at submission time.

In a job network definition and in the subsidiary jobs, several symbol tables can be defined. During a network or job activation, all symbol tables are prompted sequentially, which contain at least one symbol to be prompted.



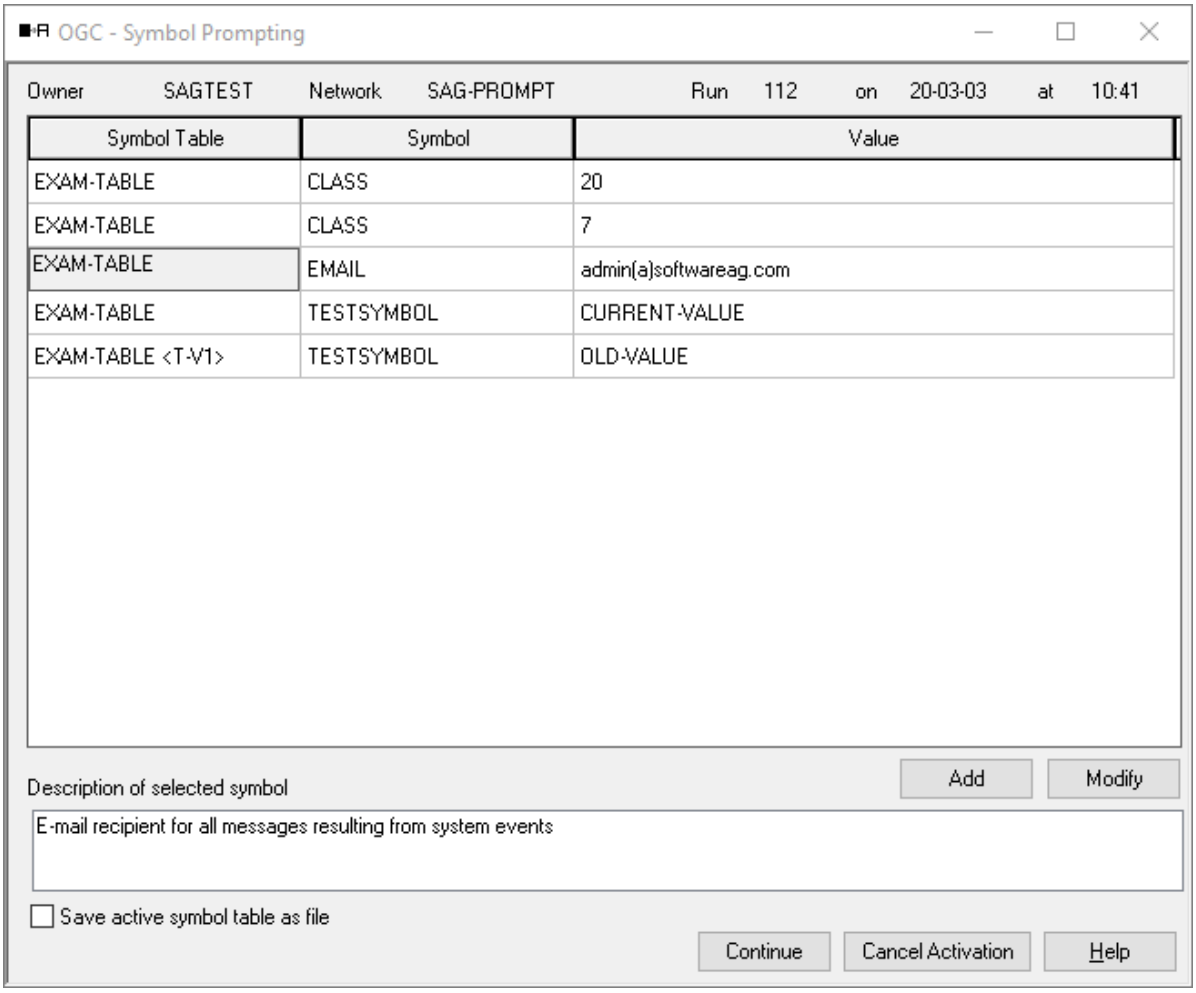
Notes:

1. Up to 100 different symbol tables can be prompted during a network activation.
2. The message `EOR1830 - More than :1: Symbol Tables in Network` will be shown as a warning if the network uses more symbol tables. In this case, some symbol table might not be prompted. If symbol table versions are used, the determination of the current symbol table version might not work properly.
3. If the activation is manual and if the JCL of a job contains symbols that are defined to be prompted, you are prompted for the values to be used in this activation. You can add new symbols and modify existing symbols of the active symbol table used during the network activation.

To modify and confirm symbols and save the active symbol table as a file

1 Activate a network with at least one symbol defined for prompting.

A **Symbol Prompting** window like the example below opens:



The window lists all symbols defined for prompting during network activation and the symbol table master that contains the symbols.

In the example above, different symbol tables are used for prompting: `EXAM-TABLE` is defined for the `SAG-PROMPT` network and `EXAM-TABLE <T-V1>` is defined for a job contained in the network. The `CLASS` symbol from `EXAM-TABLE` is listed twice because two multiple values (`20` and `7`) are defined for this symbol. The description defined for the `EMAIL` symbol is shown in the **Description of selected symbol** section. In this example, the `SAG-PROMPT` network is activated for `Run 112`.

2 You can add new symbols, modify existing symbol values or leave them unchanged.

If you want to change symbol parameters not listed in the **Symbol Prompting** window (for example, the format of a symbol), select the required symbol from the table and choose **Modify** to open a **Maintenance Symbol Active window** for this symbol. The fields in this window are explained in *Fields: Symbol*.

The columns contained in the window are explained in *Columns: Symbol Prompting*, all options and functions provided in the window are explained in *Options and Functions: Symbol Prompting*.

Note: If a **validation check** is defined for the symbol, you can only enter a valid symbol value. Incorrect values are rejected, and a user-defined error message is issued.

3 Choose **Continue** when you are finished.

If **Save active symbol table as file** and **Symbol printing after prompting** is selected in your user profile (see in the *Administration* documentation), a window opens for entering file specification parameters: see *To save an active symbol table* in the section *Saving Symbol Table Information as a File*.

If a workplan is defined for the network and **Show workplan after activation** is selected in your user profile (see *GUI General* in the *Administration* documentation), a **Workplan window** opens.

Network activation starts after the last window closes.

The new symbol values remain in the symbol table until the symbol table is modified specifically or until the next prompting after manual activation of a job network that uses the symbol table.

This section covers the following topics:

- Columns: Symbol Prompting
- Options and Functions: Symbol Prompting

Columns: Symbol Prompting

The columns of the **Symbol Prompting window** are explained in the following table:

Column	Description
Symbol Table	Name and version (if defined) of the symbol table master that contains the symbol.
Symbol	Name of the symbol defined for prompting.
Value	Value of the symbol defined for prompting.

Options and Functions: Symbol Prompting

The options and functions provided in the **Symbol Prompting window** are explained in the following table:

Option/Function	Description
Add	Add a new symbol for the symbol table selected in the window. A **Create new Symbol Active** window opens for the selected symbol table. The fields and tabbed pages in this window are explained in *Fields: Symbol*.
Modify	Change symbol parameters not listed in the **Symbol Prompting** window (for example, the format). A **Maintenance Symbol Active window** opens for the selected symbol. The fields and tabbed pages in this window are explained in *Fields: Symbol*.
Description of selected symbol	Shows the descriptive text of the symbol currently selected in the **Symbol Prompting** window. See also *Displaying Descriptions of Prompted Symbols*.
Save active symbol table as file	Select this option if you want to save the active symbol table as a file after network activation.

Displaying Descriptions of Prompted Symbols

To display the description defined for a prompted symbol

- In the **Symbol Prompting window** select the table row that contains the symbol whose description you want to display.

  The descriptive text is then displayed in the **Description of selected symbol** section of the **Symbol Prompting** window. In the **example of the window**, the description that belongs to the `EMAIL` symbol is shown.

  Text is only available if it has been entered in the **Prompt Text** field of a symbol definition (see *Fields: Symbol*).

Repeating Symbol Prompting

To repeat symbol prompting from the beginning

1 Choose **Cancel** or **Cancel Activation** to cancel the current network or job activation.

2 Start the activation again.

Specifying User Exits for Symbol Modification

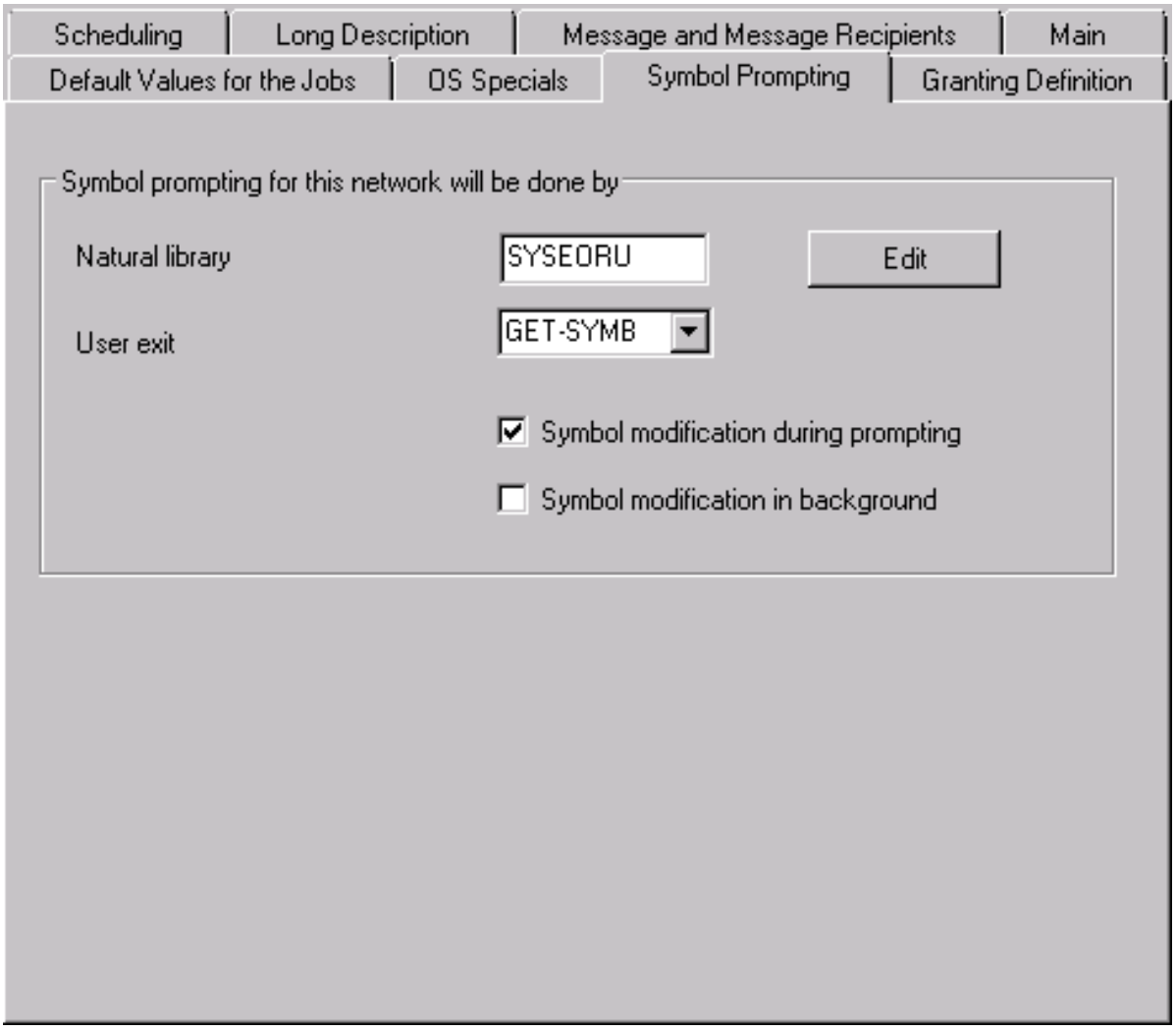
You can specify a customized user exit routine that defines your own validation checks and help information for symbol prompting. This can be used to modify active symbols depending on the values of other symbols, or of anything else.

You can then adapt a symbol table during each job run with or without using a window that prompts you for input. If you do not use a prompt window, the symbol table is adapted automatically in the background.

For details on defining user exit routines, see ***User Exits for Validation Checks of Symbol Values*** in the section *User Exits*.

To specify a user exit for symbol modification

1 In a **Network Master** window, open the tabbed page **Symbol Prompting**:



2 Specify the type of symbol prompting by defining the Natural library and the respective user exit routine. The fields are explained in ***Fields: Symbol Prompting User Exit***.

3 Choose **OK** when you are finished.

This section covers the following topics:

- Fields: Symbol Prompting User Exit

- Example of an Active Symbol Table List

Fields: Symbol Prompting User Exit

The input fields of the **Symbol Prompting page** are described in the following table:

Field	Description	
Natural Library	Name of the Natural library where the symbol prompting user exit resides.	
User Exit	Name of the symbol prompting user exit.	
Symbol Modification	Symbol modification option(s). Possible check box settings:	
	during prompting *(checked)*	A prompt window opens (default). You are asked to modify or confirm symbols defined for prompting for the active symbol table. See also ***Example of an Active Symbol Table List***.
	in Background *(checked)*	No prompt window opens. The symbol modification is managed by the user exit routine which runs in the background.
	(both boxes checked)	Performs both modification options: A prompt window opens first and then symbol modification is managed by the specified user exit. This allows you to set other symbols which are dependent on prompting.

Example of an Active Symbol Table List

The following is an example of an output file generated for an active symbol table using Entire Operations GUI Client:

Symbol Table Active

Owner: SAGTEST Network: SAG-PROMPT Run: 26 Symbol Table: EXAM-TABLET-V3

Symbol	MV	Symbol Value	Modified by	Time
CLASS		K	IMPORT	2014-01-10 12:48:54
JOBLIB		NOP.EXAMPLE.LOAD	IMPORT	2014-01-10 12:48:54
MSGCLASS		X	IMPORT	2014-01-10 12:48:54
STEPLIB		NOP.EXAMPLE.LOAD	IMPORT	2014-01-10 12:48:54
TESTSYMBOL		a	NATQA	2016-05-12 15:28:03

The file lists the current status of the symbols defined for the symbol table `EXAM-TABLET-V3` which is prompted when the network `SAG-PROMPT` is activated.

Rules and Restrictions for Symbol Replacement



Notes:

1. Symbol replacement is always attempted if at least one of the activation or submission escape characters currently in effect is found in a source line (see the section ***Symbol Escape Characters***). It is not necessary for a symbol table to be defined on the job or network level. Symbols can also be replaced via **global symbol tables**. **Predefined symbols** can be replaced without having to define a symbol table.
2. If an escape character is detected and no symbol table is defined, the error message `no symbol table defined` is issued.

This section covers the following topics:

- Symbol Table Types and Symbol Search Order
- Actions Triggered for Missing Symbols
- Nested Replacement
- Symbol Replacement in JCL
- Import/Export Restrictions
- Escape Character
- Indicator for End of a Symbol Name
- Symbol Replacement Algorithm
- Fixed Positions within a JCL Line
- Symbol Replacement with Multiple Symbol Values
- Symbol Replacement in Subnetworks

Related Topic:

- ***Predefined Symbols***

Symbol Table Types and Symbol Search Order

The types of symbol tables available and the search order used to find and replace symbols in symbol tables are described in the following section.

Entire Operations first searches for a symbol in the active user-specific symbol table. After this, it searches in the user-defined symbol table master, in the network default symbol table, and finally in the system-wide symbol table. If it searches for the symbol in a subnetwork, the symbol tables of the calling job of the **type** `NET` as well as the symbol tables of the superior job networks are searched (in ascending order) if necessary.

Symbol Table Type	Description	Search Order
JA (Job active)	Active symbol table referenced by the active job.	1
NA (Network, active)	Active symbol table referenced by the network.	2
	NO (Network, active, other symbol tables)	Other active symbol tables of the active network. These may have been created by symbol setting by API into a symbol table which is not defined on job or network level. 3
CA (Calling job)	Active symbol table referenced by the calling job. This only applies if the active job is within a **subnetwork**. Depending on the subnetwork nesting level, several symbol tables of calling jobs can be searched. The search operation can repeat in ascending order.	4
CN (Calling network)	Active symbol table referenced by the calling network. Depending on the **subnetwork** nesting level, several symbol tables of calling networks can be searched. The search operation can repeat in ascending order.	5
JM (Job master)	Symbol table referenced by the job master.	6
NV (Network version)	Symbol table referenced in the network version.	7
OD (Owner default)	Symbol table master *owner*/A. *owner* is the current owner of the active job. The symbol table *owner*/A is searched even if there is no symbol table definition on the job level and/or network level.	8
SD (System default)	Symbol table master SYSDBA/A. The symbol table SYSDBA/A is searched even if there is no symbol table definition on the job level and/or network level.	9

Note: If you invoke the link to the symbol table(s) in the object network, job master or active job, you will get a list of the **usable symbol tables** for this object in the hierarchical order described above.

Actions Triggered for Missing Symbols

The following actions are triggered as soon as a symbol cannot be found in any symbol table:

A message is written to the log
A message is written to the system log file (see *Log Information*), for example:

```
Symbol XYZ not found
... Symbol Table EXAMPLE/SYMTAB1
```

The logged symbol table (here: `SYMTAB1`) is the first symbol table searched upward in the symbol table hierarchy. Example: If no symbol table has been defined at the job level, the symbol table defined for the network is logged.

A global message is sent
A `symbol not found` message containing the name of the searched symbol table is sent if the Global Message Sending Exit (see the *Administration* documentation) is enabled in your environment.

A global message is saved in the event store
An appropriate message including the name of the searched symbol table is saved in the event store if the **Symbol not found** and **Event Store** options of the **Global Messages and Exits** function (see the *Administration* documentation) are enabled in your environment.

The problem is handled by a global user exit
A user exit takes action for the missing symbol to resolve the problem and continue processing if the Global Symbol Not Found Exit (see the *Administration* documentation) is defined and enabled in your environment.

Current processes are aborted
Current operations (for example, JCL loading) are terminated if a problem with missing symbols cannot be resolved.

Nested Replacement

Symbols can be used within other symbols (nested).

For nested replacement of symbols, the following rules apply:

- On the top level, only the escape character defined for the job or network is checked and replaced.
- If the activation escape character is used on a nested level, only activation escape characters are replaced on deeper levels. This is necessary to prevent too early replacements of submission escape characters.

- If the submission escape character is used on a nested level, submission and activation escape characters are replaced on deeper levels.

Example of Nested Replacement

Symbol	Value
PREFIX	EOR
STEPLIB	§PREFIX..EXAMPLE.LOAD

§STEPLIB is replaced by EOR.EXAMPLE.LOAD.

Note: If a point marks the end of a symbol, it is deleted on replacement. With nested replacement, the deletion takes place replacement by replacement. Thus, a sufficient number of points must be coded. If a point is to remain at the end of the symbol, two points must be coded.

Example of Nested Symbol Replacement in JCL

```
/* IN §JJ. SOFTWARE AG
```

changes to the following when J = 20 and JJ = §J.01:

```
/* IN 2001 SOFTWARE AG
```

The advantage of this method is that symbol replacement is performed within the fixed positions.

Symbol Replacement in JCL

For symbol replacement in JCL submitted on mainframes (z/OS and BS2000), the following particularities apply:

Column 72 contains one blank and the columns 73 to 80 contain digits.	Columns 71 to 80 of the line are saved before the first replacement, and they are written back after the last replacement.
Column 72 contains a hyphen (-) and the columns 73 to 80 contain blanks.	Columns 71 to 80 of the line are saved before the first replacement, and they are written back after the last replacement.
Column 71 contains a comma (,) and the columns 72 to 80 contain blanks.	Columns 71 to 80 of the line are saved before the first replacement, and they are written back after the last replacement.

Intermediate results	Intermediate results can even be longer than 71 characters. They are only overwritten by possibly saved strings at the end of the replacement as a whole.

Import/Export Restrictions

Avoid the equal sign (=) within a symbol value. This causes problems during import or export operations of symbol tables.

Escape Character

Symbols to be replaced are detected by a preceding escape character anywhere in the JCL. See *Symbol Escape Characters* for details and *Example of Nested Symbol Replacement in JCL*.

Indicator for End of a Symbol Name

Symbol names end with one of the following delimiter characters:

blank ( ), comma (,), semicolon (;),
period (.), apostrophe ('), commercial at (@),
percent sign (%), logical not (¬), ampersand (&),
parenthesis ((or)), or equal sign (=).

Example: `§SYMBOL-1;`.

A single period (.) after a symbol is removed during replacement. But two consecutive periods (. .) are converted to one period (.). This is important for the dynamic creation of file names.

Symbol Replacement Algorithm

The following algorithm is used for symbol replacement:

1. Search for the symbol farthest to the left.

   If none (or no more) available -> end.
2. Replacement

   In case of error -> end.
3. Write updated line.

   Continue at (1).

   This makes **nested symbol replacement** (symbols within symbols) possible.

Fixed Positions within a JCL Line

This section applies to JCL submitted on mainframes only.

Multiple symbol replacement can cause a shift within a line. However, in the following cases, parts of the line are kept in a fixed position:

1. Symbols do not follow one another directly.
2. The columns 72 to 80 always remain fixed. If the column 70 contains a blank and column 71 a comma, then these will also remain fixed.
3. If a symbol is followed by at least two blank spaces, the rest of the line is kept in its original position, unless the symbol value would overlap with it. For example:

   ```
   /FILE ABC.§VAR              COMMENT
   ```

   changes to the following when `VAR` = `REPLACE`:

   ```
   /FILE ABC.REPLACE           COMMENT
   ```

4. If a symbol is followed directly by continuous text, then by at least two blank spaces, then by an exclamation mark (`!`) and continuous text: the symbol is replaced and the exclamation mark (`!`) is replaced by blank spaces. For example:

   ```
   /FILE ABC.§VAR..XYZ        !COMMENT
   ```

   changes to the following when `VAR` = `REPLACE`:

   ```
   /FILE ABC.REPLACE..XYZ     COMMENT
   ```

Symbol Replacement with Multiple Symbol Values

If you want to replace a symbol with multiple values when a job is activated numerous times concurrently, you must use `P-SUFFIX` to identify the variable to be replaced. This is the variable you entered in the **Suffix Symbol** field in the **Job Master Definition** window (see the field description in the section *Job Maintenance*).

For information on defining multiple symbol values, see the symbol functions **`!MV` or `?MV` and `!MM` or `?MM`** described in the section *Functions for Symbol Replacement*.



Note: If **Suffix Symbol** contains no values, then the multiple job is activated as a temporary dummy job.

You can obtain a specific occurrence of a multiple-value symbol by using the **symbol function** `!MV` **or** `?MV`.

Symbol Replacement in Subnetworks

If you want to enter symbols in subnetworks as well, you can attach a symbol table to the main network; this table contains all symbols to be queried in the jobs of the subnetwork(s). At the same time, the symbol table can be linked to the corresponding jobs of the subnetwork(s) by means of symbols whose values are not to be retrieved. After symbol entry, this creates active symbol tables attached to the main network.

On the job level, Entire Operations tries to load the symbols from the level(s) above if a symbol with the corresponding symbol table is not found, for example from the active symbol table of the main network. Entire Operations does not create active symbol tables on subnetwork job levels as the run numbers are not yet known at the time of the main network's activation and as symbol prompting is activated first.

54 Functions for Symbol Replacement

- Results Returned by a Symbol Function 650
- Function !D or ?D - Date Calculation and Date Formatting 651
- Function !E or ?E - Date from Date Calculation 658
- Functions !MV or ?MV and !MM or ?MM - Access to Multiple-Value Symbols 659
- Function !TIMN or ?TIMN - Constant Time Values 661

A symbol function is used to set a symbol value depending on the parameters provided for the function.

A symbol function is identified by either an exclamation mark (!) or a question mark (?) that is used as a start value. You can choose either character depending on the current code page used in your Natural environment. The exclamation mark (!) can conflict with code page conversion.

You can also perform your own symbol functions with a user exit routine. For details, see ***User Exits for Symbol Functions*** in the section *User Exits*.

The predefined symbol functions provided are explained in this section.

Related Topic:

- ***User Exits for Symbol Functions***

Results Returned by a Symbol Function

Results of symbol functions are written back to the active job's symbol table after the first evaluation. The following invocations of the same symbol function with the same parameters do not invoke the symbol function again, but just return the function value.

Symbol functions with **activation escape character** are written to the active symbol table without escape character.

Symbol functions with **submission escape character** are written to the active symbol table with the submission escape character.



Notes:

1. The same symbol or symbol function can be used with both the activation escape character and the submission escape character. Both are handled separately and occupy different active symbol table entries. This is especially important if you use any time-dependent symbols or symbol functions.
2. Results of symbol functions are not kept in active symbol tables to enforce recalculation of symbol function values, for example, after the change of a multiple-value symbol.

Function !D or ?D - Date Calculation and Date Formatting

(This symbol function replaces the obsolete symbol function !W or ?W.)

This symbol function calculates date values and formats them.

This section covers the following topics:

- Syntax
- Examples
- Return Format
- Parameters
- Edit Masks for Date Functions
- Reference Objects
- Output Date Formats

Syntax

```
§!D|?D<type|period|offset[offset-unit][,edit-mask][,calendar]>
```

(for all **types** except `T`)

or

```
§!D|?D<T[,edit-mask]>
```

or

```
§!D|?D<yyyymmdd,edit-mask>
```

Examples

```
§!D<AC-1>
```

Get the date before the current date (yesterday). The date is returned in the default format YYYYMMDD.

```
§?D<C20220722+1>
```

Get the next working day, for the date 2022-07-22. The date is returned in the default format YYYYMMDD.

The result will be 20220725, assuming that Saturday and Sunday are no working days in the related calendar.

```
§?D<CW+1,DDMMYY>
```

Get the first day (Monday) of the current week in the format DDMMYY.

```
§!D<T,YYMMDD>
```

Get the current date in the format YYMMDD.

```
§!D<20070629,05>
```

Get the date 2007-06-29 in the format `05` (DD.MM.YY). See ***Output Date Formats***.

```
§!D<A1+1,MM>
```

Get the previous month in the format `MM` (month only).

```
§!D<A1+1,YYYYMMDD>
```

Get the first day of the previous month.

```
§!D<A1-1,YYYYMMDD>
```

Get the last day of the previous month.

Return Format

See ***Output Date Formats***.

Parameters

The parameters are written one after the other without spaces.

Parameter/Syntax Element	Value	Description
type	T	Today's date. For this type, the parameters *period*, *offset* and *offset-unit* are not required or evaluated.
	A	All days (365 resp. 366 days a year).
	C	Calendar days (days defined as working days in the calendar).
	S	Schedule days.
period		Not evaluated for the type T.
	C	From current day.
	YYYYMMDD	From any given day, in format YYYYMMDD. Example: 20220722
	W	Week.
	M	Month.
	1 to 9	One month back...nine months back.
	Q	Quarter.
	Y	Year.
offset		Positive or negative number with prefix.
offset-unit		Unit for the offset (optional); only evaluated for the type A.
	D	Days (default).
	W	Weeks.
	M	Months.
	Q	Quarters.
	Y	Years.
edit-mask		See ***Edit Masks for Date Functions***.
calendar		See **Calendar** in *Reference Objects*.
yyyymmdd		Date in the format YYYYMMDD.

For further explanations of date and time formats, see ***Date and Time Formats***.

Edit Masks for Date Functions

The edit mask can be entered either directly or with a format sign. The format sign can be used to shorten the function call.

Format Sign	Format	Example
F1, 1	DD	29
F2, 2	DDMM	2910
F3, 3	DDMMYY	291008
F4, 4	DDMMYYYY	29102008
F5, 5	DD.MM.YY	29.10.08
F6, 6	DD.MM.YYYY	29.10.2008
F7, 7	DD/MM/YY	29/10/08
F8, 8	DD/MM/YYYY	29/10/2008
F9, 9	DD-MM-YY	29-10-08
F10, 10	DD-MM-YYYY	29-10-2008
F11, 11	MM	10
F12, 12	MMDD	1029
F13, 13	MMYY	1008
F14, 14	MM.YY	10.08
F15, 15	MMYYYY	102008
F16, 16	MM.YYYY	10.2008
F17, 17	NN	We (Weekday; language-dependent.)
F18, 18	O	4 (Number of the weekday.)
F19, 19	WW	44 (Number of the week.)
F20, 20	YY	08
F21, 21	YYJJJ	08245
F22, 22	YYMM	0810
F23, 23	YYMMDD	081029
F24, 24	YYYY	2008
F25, 25	YYYYJJJ	2008245
F26, 26	YYYYMM	200810
F27, 27	YYYYMMDD	20081029
F28, 28	YYYY.MM.DD	2008.10.29

Format Sign	Format	Example
F29, 29	YYYY/MM/DD	2008/10/29
F30, 30	YYYY-MM-DD	2008-10-29
F31, 31	YY.MM.DD	08.10.29
F32, 32	YY/MM/DD	08/10/29
F33, 33	YY-MM-DD	08-10-29
F34, 34	QQ	WE (Weekday; language-dependent; in upper case.)
F35, 35	MMDDYY	041922
F36, 36	MM/DD/YY	04/19/22
F37, 37	MM-DD-YY	04-19-22
F38, 38	JJJ	109
F39, 39	JJJYY	10922
F40, 40	JJJYYYY	1092022
F41, 41	WWYY	1622
F42, 42	WWYYYY	162022
F43, 43	YYWW	2216
F44, 44	YYYYWW	202216
F45, 45	YY.MM or YY#MM	22.04
F46, 46	YY/JJJ	22/109
F47, 47	YYYY/JJJ	2022/109

For further explanations of date and time formats, see *Date and Time Formats*.

Reference Objects

Parameter/Syntax Element	Description
Owner	Owner of the network from which the function was called.
Calendar	If a calendar was specified explicitly, this is used for calendar calculations. This calendar must exist under the current owner. If it is not found there, it is searched for under the owner SYSDBA. Otherwise, the calendar of the current schedule is used. If no calendar has been defined, all days are treated as workdays. If an explicitly specified calendar is not existing for the current year, the JCL generation or the job submission will end with an error and a log message is written. **Subnetwork considerations**

Parameter/Syntax Element	Description
	■ If the symbol function is invoked from within a subnetwork, the schedule or calendar of the top level (main calling network) is used for schedule or calendar calculation. ■ Schedules or calendars defined in the subnetwork are ignored in such calculations. ■ It is not necessary to define schedules or calendars in a network used only as a subnetwork because they are not relevant.

Output Date Formats

The output date formats described here can be used in the symbol functions `!D` or `?D` and `!W` or `?W`.

The default date output format is YYYYMMDD (see also *Date and Time Formats*). It will be used if not output date format is specified.

Format	Numeric Equivalent	Example
DD	01	29
DDMM	02	2910
DDMMYY	03	291008
DDMMYYYY	04	29102008
DD.MM.YY DD#MM#YY	05	29.10.08
DD.MM.YYYY DD#MM#YYYY	06	29.10.2008
DD/MM/YY	07	29/10/08
DD/MM/YYYY	08	29/10/2008
DD-MM-YY	09	29-10-08
DD-MM-YYYY	10	29-10-2008
JJJ	38	245
JJJYY	39	24508
JJJYYYY	40	2452008
MM	11	10
MMDD	12	1029
MMDDYY	35	102908
MMYY	13	1008
MM.YY MM#YY	14	10.08

Format	Numeric Equivalent	Example
MMYYYY	15	102008
MM.YYYY MM#YYYY	16	10.2008
MM/DD/YY	36	10/29/08
MM-DD-YY	37	10-29-08
NN	17	Sa (First two characters of the weekday name. This value depends on the current language setting of the Natural session.)
O	18	6 (Number of the weekday. This value depends on the Natural profile parameter `DTFORM` of the Natural session. Refer to the Natural *Parameter Reference* documentation.)
QQ	34	SA (Like format `NN`, but in upper case.)
WW	19	44 (Number of the week.)
WWYY	41	4408 (Week and year.)
WWYYYY	42	442008 (Week and year.)
YY	20	08
YYJJJ	21	08245
YYMM	22	0810
YYMMDD	23	081029
YYWW	43	0844 (Year and week.)
YYYY	24	2008
YYYYJJJ	25	2008245
YYYYMM	26	200810 (Year and month.)

Format	Numeric Equivalent	Example
YYYYMMDD	27	20081029
YYYYWW	44	200844 (Year and week.)
YYYY.MM.DD YYYY#MM#DD	28	2008.10.29
YYYY/MM/DD	29	2008/10/29
YYYY-MM-DD	30	2008-10-29
YY.MM YY#MM	45	08.10
YY.MM.DD YY#MM#DD	31	08.10.29
YY/MM/DD	32	08/10/29
YY-MM-DD	33	08-10-29

For further explanations of date and time formats, see *Date and Time Formats*.

Function !E or ?E - Date from Date Calculation

This symbol function is used to add or subtract a number of days from a given date in the format YYYYMMDD (see also *Date and Time Formats*).

This section covers the following topics:

- Syntax
- Examples
- Return Format

- Parameters

Syntax

```
§!E|?E<date,offset[,edit-mask]>
```

Examples

```
§!E<20081029,+28>
```

```
§!E<20081029,-7,F10>
```

Return Format

Date in the format YYYYMMDD (see also *Date and Time Formats*).

Parameters

The parameters are separated by a comma (,).

Parameter/Syntax Element	Description
date	Date in the format YYYYMMDD. See also *Date and Time Formats*.
offset	Positive or negative number with a plus (+) or minus (-) sign. Unit: days.
edit-mask	See *Edit Masks for Date Functions*.

Functions !MV or ?MV and !MM or ?MM - Access to Multiple-Value Symbols

These symbol functions allow you to use multiple symbol values.

You can get the maximum index of a multiple-value symbol.

This section covers the following topics:

- Symbol Function !MV or ?MV

- Symbol Function !MM or ?MM

Symbol Function !MV or ?MV

This symbol function allows you to use one single value of a multiple-value symbol.

Syntax

```
§!MV|?MV<symbol,index>
```

Examples

```
§!MV<ARRAY,1>
```

```
§?MV<ARRAY,§INDEX>
```

Return Format

Integer value.

Parameters

The parameters are separated by a comma.

Parameter/Syntax Element	Description
symbol	Name of the multiple-value symbol.
index	Index of the symbol value.

Errors

In the following cases, a `Symbol not found` message is returned:

- Multiple-value symbol not found,
- Index missing,
- Index not numeric, and
- Index too high.

Reference Objects

Parameter/Syntax Element	Description
Symbol table	The symbol table of the active job is always used.

Symbol Function !MM or ?MM

This symbol function allows you to get the maximum index of a multiple-value symbol.

This function accesses single occurrences of a multiple-value symbol.

Syntax

```
§!MM|?MM<symbol>
```

Example

```
§!MM<ARRAY>
```

Return Format

Symbol value.

Parameters

This function has only one parameter.

Parameter/Syntax Element	Description
symbol	Name of the multiple-value symbol.

Errors

In the following case, a `Symbol not found` message is returned:

- Multiple-value symbol not found.

Reference Objects

Parameter/Syntax Element	Description
Symbol table	The symbol table of the active job is always used.

Function !TIMN or ?TIMN - Constant Time Values

This symbol function allows you to use constant time values in multiple timestamp replacements.

This section covers the following topics:

- Syntax
- Examples
- Return Format
- Parameters

- Reference Objects

Syntax

```
§!TIMN|?TIMN<parm>
```

Examples

```
§!TIMN<A>
```

```
§?TIMN<B>
```

Return Format

A timestamp in the format of the Natural system variable `*TIMN`:

Format: A7.

Content: HHIISST where T = tenths of seconds.

Parameters

Syntax Element/Parameter	Description

parm	The parameter (for example, `A` or `B`) must be specified for syntactical reasons only and has not meaning otherwise.
	
	Note: All invocations with the same parameter create the same time value (if the same active symbol table is used). The result of each combination of symbol function and parameter(s) is kept in the active symbol table for subsequent replacements.
	
	Example:
	
	`//* line 1 - §TIMN<A>`
	`//* line 2 - §TIMN<B>`
	`//* line 3 - §TIMN<A>`
	
	The example above can be replaced as follows:

Syntax Element/Parameter	Description

	`//* line 1 - 1601331`
	`//* line 2 - 1601332`
	`//* line 3 - 1601331`

Reference Objects

None.

XII Log Information

55 Log Information

- Displaying Logged Information - Browse Log Function 668
- Displaying Extended Log Information 671
- Monitoring Entire Operations Activities 672

Entire Operations logs all important system events. You can view logs specific to an owner, network or job and specify additional information for the log report to be produced.

You can also use the monitoring function to observe all activities currently running in your Entire Operations environment.

The standard system log contains information about activities in the system as a whole such as user actions, date and time of events and messages about events. If **additional (extended) information** is available for any item on the system log, it is preceded by an asterisk (*). For details, see *Displaying Extended Log Information*.

Displaying Logged Information - Browse Log Function

To display log information

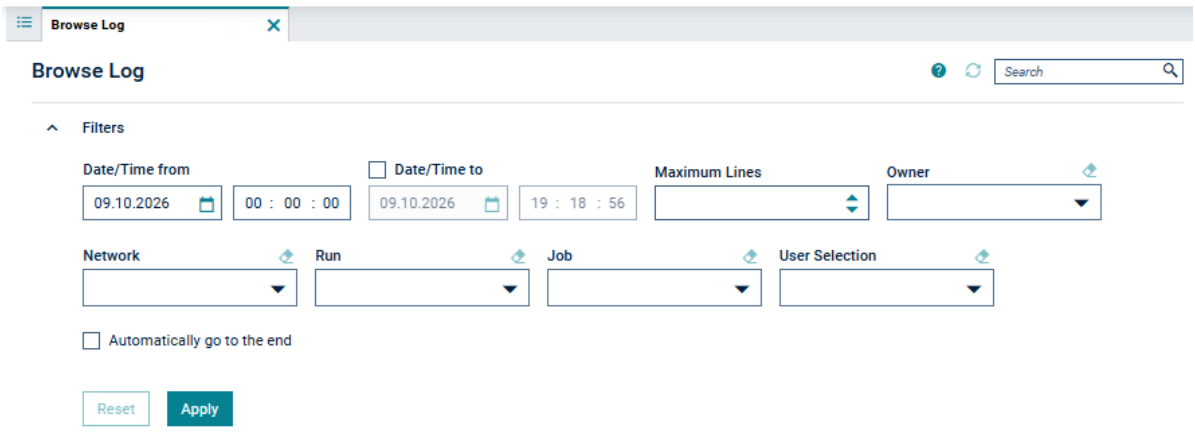
1 For a single node item:

Choose the object whose log you want to see. This opens a dialog. Choose **Browse Log**.

Or:

In the main menu, choose **General** > **Browse Log**.

A **Log Display Selection** window like the example below opens:



2 Specify the required selection criteria. The input fields are explained in *Fields: Log Display Selection*.

3 Choose **OK**.

The logging information is shown in a **Browse Log** window like the example below:

For explanations of the columns, see ***Columns: Browse Log***. The functions of the context menu are described in ***Available Functions: Browse Log***.

- Fields: Log Display Selection
- Columns: Browse Log
- Available Functions: Browse Log

Fields: Log Display Selection

The **Log Display Selection window** provides the following input fields:

Field/Option	Description
Date/Time from	Start date and time for the log display. Default: Current date and 00:00:00 for time in the format HH:II:SS.
Date/Time to	End date and time for the log display. Default: Current date and time in the format HH:II:SS.
Max. Lines	Maximum number of lines to display in the log. `0` means: no limit (default). This setting can also be specified as a default value in the user profile: see the section *User Maintenance* in the *Administration* documentation.
Owner	Name of the owner whose log is to be displayed. Select an owner from the drop-down list box or leave the field blank to display the log for all owners.
Network	Name of the network whose log is to be displayed. Select a network from the drop-down list box or leave the field blank to display the log for all networks.
Run	Run number or a range of numbers to be displayed in the log. Can be used only if an owner and a network are specified. Select a run number from the drop-down list box or leave the field blank to display the log for all run numbers. Default: 1-9999 (all).
Job	Job whose log is to be displayed. Select a job from the drop-down list box or leave the field blank to display the log for all jobs.
User	User ID or Monitor task name (or ranges) for which the log is to be displayed. Enter (or select) a user ID or Monitor task name (for example, `TASK 1`) or use asterisk (*) to specify a range (for example, `TASK*`).

Field/Option	Description
	To display the log for all users and tasks, either leave the field blank (default) or enter an asterisk only (*).
Automatically go to end	If this option is selected, scrolls down to the end of the log report to see the most recent log entries. This setting can also be specified as a default value in the user profile: see the section *User Maintenance* in the *Administration* documentation.

Columns: Browse Log

The **Browse Log window** contains the following columns:

Column	Description
Num	Log line number.
User ID	User ID or Monitor task name. An ID or name preceded by an asterisk (*) indicates that extended log information is available for the job listed in this log line. For more information, see ***Displaying Extended Log Information***.
Owner	Owner name in Entire Operations.
Job	Entire Operations job name.
Network	Name of the job network.
Run	Job run number.
Date	Date of log entry.
Time	Time of log entry.
Message	Message text. Depending on the default display option set (see *List Display Options* in the *Administration* documentation), the message text is prefixed with a message code (if one exists), for example: `EOR2260 - Network activation performed.`

Available Functions: Browse Log

The context menu of the **Browse Log window** provides the following functions:

Function	Shortcut	Description
Copy	CTRL+C	Copies the currently selected text. In case some text from the selection is not received from a server, a warning is shown, and the **Copy** function is not performed.
Select All	CTRL+A	Selects all text.
Reverse Order	CTRL+R	Shows text in the reverse order.
Find	CTRL+F	Finds the text fragment.
Find next	F3	Finds the next text fragment.
Go To	CTRL+G	Moves the cursor to the specified line number.
Print	CTRL+P	Prints log data.
Refresh	F5	Refreshes the log with "time to"=actual time
Refresh with new start time	F4	Refreshes the log with "time from"=time from the selected row "time to"=actual time
Extended Log	CTRL+E	Displays job-specific **extended log** information if available for the log item. Extended log information is marked with an asterisk (*) in the **User ID** column.

Displaying Extended Log Information

Additional (extended) log information is provided as job-specific data items appended to the standard system log.

An asterisk (*) in the **User ID** column of a log item indicates that an extended log is available for the job listed in this log line.

To display extended log information

- From the object workspace, select an active job and choose **Extended Log**. All **Extended Log** from the functions are described in ***Available Functions: Job Active*** in the section *Maintaining Active Jobs and Networks*.

  An **Extended Log** opens.

This section covers the following topics:

- Extended Log Information Available for a Job

Extended Log Information Available for a Job

Extended (additional) information that can be logged for a job are described in the following section.

JCL
Displays the JCL of a specific job run. The JCL log source is dependent upon the operating system in which the job has run.

Prerequisite for logging: Activate the **Log JCL** option in the job master definition described in *Defining Extended Log Information for a Job* in the section *Job Maintenance*.

SYSOUT
Displays the SYSOUT of a specific job run.

Prerequisite for logging: Activate the **Log SYSOUT** option in the job master definition described in *Defining Extended Log Information for a Job* in the section *Job Maintenance*.

Messages
Displays selected messages of a specific job run.

Prerequisite for logging: Define the messages to be logged in the **Log System Messages** fields of the job master definition described in *Defining Extended Log Information for a Job* in the section *Job Maintenance*.

Active/Pregenerated JCL Modifications
Displays changes made to active or pregenerated JCL of a specific job run.

Prerequisite for logging: Activate the **Log the changes made to an active/pregen. JCL** option in the Entire Operations default settings described in *Defaults: System/Log Files* in the *Administration* documentation.

Import/Export Processing
You can choose **Display** from an **Import/Export** window to view an **extended log** file containing information on an import or export operation.

Monitoring Entire Operations Activities

Activity monitoring is used to permanently display Entire Operations events that currently occur in your Entire Operations environment.

The following applies to the monitoring feature:

- The activity monitoring facility receives its data from the Entire Operations log file.
- The activity monitoring facility only displays the most important messages. Error messages and other very important messages appear highlighted.

- The default setting displays events starting from the current time.

- Viewing Entire Operations Activities
- Columns: Activity Log

Viewing Entire Operations Activities

To start monitoring and display activities

1 In the main menu, choose **General**.

2 Choose **Activity Log**.

An **Activity Log** dialog opens.

All current activities within Entire Operations are logged and displayed in this window. Error messages are highlighted in red.

The columns contained on the screen are described in *Columns: Activity Log*.

3 You can use the **Auto Refresh** and **Refresh** options to update the display and modify the update interval. See also *Refreshing Object Lists*.

You can use the **Auto scrolling** option to automatically scroll to the latest activity.

Columns: Activity Log

The columns in the **Activity Log window** are described in the following table:

Column	Meaning
Owner	Owner of the active network.
Network	Name of the active network.
Job	Name of the active job.
Run	Run number as assigned to the active network by Entire Operations.
Job ID	Job identifier as assigned by the operating system or by the job entry subsystem.
Time	Last action or check time for the job.
Message	Last message issued for the job by Entire Operations. For a list of possible messages, see *Messages in Active Jobs Lists*.

XIII Reporting

56 Reporting

- Report Types 678
- Viewing the Report Status List 681
- Generating or Regenerating Online Reports 682
- Viewing Report Properties and Deleting a Report 683
- Fields and Columns: Reporting 684
- Determination Date for Report Data 688
- Report Output Options 688
- Using Reports with Bar Charts 689
- Using Reports with Monitor Task Time Tables 692
- User Restrictions for Reports 694
- Retention Period for Reports from Entire Operations GUI Client 695
- Examples of Reports 695
- Generating Batch Reports 709

This section describes the **Reporting** function which is used to generate reports that help overview your network environment to define objects, monitor the system and plan workloads.

Related Topic:

- See also the cross-reference reports described in the section *Cross-References*.

Report Types

The report types available you can select from the **Reporting dialog** are described in the following table. Reports that require special user privileges are indicated in the table.

Report Type	Description
Log - reports (see below)	The **LOG -** reports provide job or network processing information for a given date/time range, extracted from the Entire Operations log. The following applies to all job logs: Only those networks are displayed for which the user has read access. The user has read access if: ■ the user has profile type **A** (Administrator); ■ the owner SYSDBA is currently assigned to the user; ■ the network owner can be found on the user's owner list; ■ access to the network has been explicitly granted to the user (see *Granting Definition: Authorizing Other Users or Owners to Access a Network* in the section *Network Maintenance*).
Log - Terminated Jobs	Lists all jobs that ended normally (OK). See also *Example of Log - Terminated Jobs*.
Log - Abended Jobs	Lists all jobs that ended abnormally (not OK). See also *Example of Log - Abended Jobs*.
Log - Jobs not started	Lists all jobs that were not started. A job cannot start, for example, if its latest start time exceeded, or if it is waiting for an input condition or a resource. See also *Example of Log - Jobs not started*.
Log - Jobs with permanent errors	Lists all jobs that fail to run because of permanent errors. See also *Example of Log - Jobs with permanent errors*.

Report Type	Description
Log - Networks not activated	Lists all networks that could not be activated, because an extraction or activation error occurred. See also *Example of Log - Networks not activated*.
Accounting Information	Information on job accounting data (for example, job elapsed times and CPU times) of previous network and job executions. See also *Example of Accounting Information*.
Network Description (short)	Displays information on networks and jobs as defined on the master database, including scheduling information, prerequisites and End-of-Job checking and actions. See also *Example of Network Description (short)*.
Network Description (detailed)	Displays the same information as the **Network Description (short)**, but includes all prose descriptions defined at the network, job or event level using the Editor facility. See also *Example of Network Description (detailed)*.
Schedule of Jobs	Displays a jobs schedule for a specific date range. See also *Example of Schedule of Jobs*.
Network Start Summary	Status report of all network starts for a given day, regardless of whether they: ■ are waiting for prerequisite resources; ■ have already been executed; ■ are currently being executed; ■ have already been completed. See also *Example of Network Start Summary*.
Network Schedule Overview	Overview of scheduled and/or not yet executed, system-wide network activations. See also *Example of Network Start Summary*.
Activation Overview	Overview of network activations. See also *Example of Activation Overview*.
Compare Symbol Tables	(Special user authorization required.) Displays the results of the comparison of one or more symbol tables. See also *Example of Compare Symbol Tables*.
Compare Networks	(Special user authorization required.) Displays the results of the comparison of one or more networks.

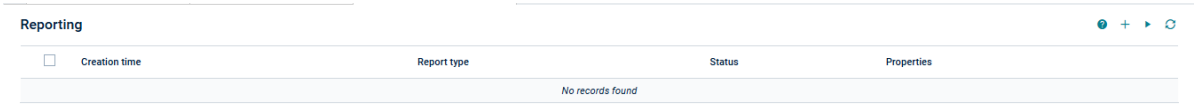
Report Type	Description
	The report shows whether jobs defined in one network exist also in another network. Additionally the attributes of the networks and of all jobs having the same name are compared. See also *Example of Compare Networks*.
Node Overview	(Special user authorization required.) Overview of available nodes. See also *Example of Node Overview*.
Network/Job Usage	(Special user authorization required.) Lists networks and related subnetworks (jobs of the type NET) as well as jobs activated by an End-of-Job action or a recovery job. See also *Example of Network/Job Usage*.
Network Start Overview (Bar Chart)	Displays the network start time and the end time of a previous network execution as a **bar chart**. See also *Network Start Overview (Bar Chart)*.
Network and Job Start Overview (Bar Chart)	Displays the network and job start as a **bar chart**. See also *Example of Network and Job Start Overview (Bar Chart)*.
Network Schedule Overview (Bar Chart)	Displays the schedule selection of a network as a **bar chart**. See also *Example of Network Schedule Overview (Bar Chart)*.
Monitor Tasks and Functions Overview	Displays data collected from performed Monitor tasks and called Monitor functions in a task time table. See also *Example of Monitor Tasks and Functions Overview*. **Note:** The **Monitor Accounting** option (see the *Administration documentation*) must be enabled to collect the data.
Monitor Tasks and Exits Overview	Displays data collected from performed Monitor tasks and called Monitor exits in a task time table. See also *Example of Monitor Tasks and Exits Overview*. **Note:** The **Monitor Accounting** option (see the *Administration documentation*) must be enabled to collect the data.

Viewing the Report Status List

To view all reports submitted for processing

1 In the main menu, choose **General** > **Reporting**.

A **Reporting** window like the example below opens:



A list of scheduled and ready reports appears.

(If no reports exist, the table in the **Reporting** window is empty.)

The reports listed in the table are used to manage and view all reports requested by the current user. Report generation is invoked within Entire Operations GUI Client whereas the reports themselves are generated asynchronously in the background via the Entire Operations Monitor.

The report list contains the following columns:

Creation time	Date and time when a report generation was initiated arranged in descending order (latest date first).
Report type	**Report type** requested.
Status	Indicates the progress in generating the report:

Queued	Report is queued for processing. A report is queued, for example, if it must wait until the Entire Operations Monitor has finished another task before it can handle the report generation (asynchronous processing). This allows you to continue working while the report is processed in the background.
Running	Report is being generated.
Completed	Report has been generated successfully.

	Aborted	Report has been aborted. A probable reason for abortion is: ■ No report data found for processing ■ An inactive Entire Operations Monitor ■ A system error
Properties	Properties settings used to generate the report.	

Press F5 or choose **Refresh** or **Auto Refresh** (specifies an automatic refresh interval) from the context menu to refresh the processing status in the **Status** column.

2 From the table in the **Reporting** window, you can select the report you want to generate by choosing **Save as File** from the context menu to determine the file to be created from the report: see *Report Output Options*.

Generating or Regenerating Online Reports

This section provides instructions for generating or regenerating a report in online mode with the **Reporting** function.

Report data is evaluated for the current day by default. If you want to change the evaluation date, use the **Determination Date** function.

Note: The selection options provided to generate a report allow extremely flexible reporting. On the other hand, it allows you to formulate requests that will cause very many Adabas calls. For this reason, it is possible to disallow asterisk/wildcard report selection for non-administrator users: see *Range Specification and Usage Restrictions*.

To generate a report

1 From the table in the **Reporting dialog**, select **Add** (+).

A **Reporting** dialog opens.

2 From the drop-down list box next to **Report type**, select the required **report type**.

Depending on the report type selected, additional input fields appear in the **Preselection** section of the **Reporting** dialog. They are explained in *Fields and Columns: Reports*.

3 In the **Preselection** section, enter the required selection criteria and output options.

In the **Completed message to E-mail** field, enter an e-mail address if you want to send a notification when the report generation has finished.

4 Choose **OK** when you are finished.

The report is added to the table in the **Reporting window** and queued for processing.

If you choose **Cancel**, the selected report type and selection criteria are retained in the **Reporting** dialog until you open it again and choose **OK**.

5 When the report has been processed successfully, the **Status** in the **Reporting** window changes to **Completed**.

You can then save the report as described in *Report Output Options*.

To regenerate a report

1 From the table in the **Reporting window**, select a generated report and choose **Regenerate** ().

A **Reporting dialog** opens with the properties settings of the previous report generation.

If required, change the current settings by replacing the values in the **Preselection** section and the **Completed message to Email** field of the dialog. The fields are explained in *Fields and Columns: Reports*.

2 Choose **OK**.

The report is added to the table and queued for processing.

The same applies as stated for generating a report.

Viewing Report Properties and Deleting a Report

To display the properties of a report

- Select the required report from the table in the **Reporting window** and choose **Properties** from the context menu.

  A **Reporting** dialog appears where the properties (report type, owner, network and e-mail) are displayed.

To delete single or multiple reports

- From the table in the **Reporting** , select one or more generated reports and choose **Delete** () .

The selected reports are removed from the table.

Fields and Columns: Reporting

The input fields for specifying selection criteria and output options for the report to be produced are explained in the following table. The input fields correspond to the columns contained in the report output file.

Depending on the report type (to be) produced, the following fields and columns are available:

Field/Column	Description
Owner	Name of an owner or a **range of names**.
Network	Name of a network or a **range of names**.
Network version	Name of a network version or a **range of names.**
Job	(Applies to **Log -** reports and **Job Schedule** only.) Name of a job or a **range of names.** If no job name is defined, the corresponding output column in the report file contains a dash (-).
Date/Time from/thru	Start date/time and end date/time of the reporting period. Valid number range: 1 to 31. Default: current date and time range 00:00 until 23:59. For **Monitor Tasks and Functions Overview** and **Example of Monitor Tasks and Exits Overview** the default time range is 00:00 until the current time. For **Network Schedule Overview** reports, the start and end dates can be in the past. In this case, only those network activations are logged, which have not yet been executed. These can be caused by symbol prompting, which has not been completed, or by an Entire Operations Monitor, that has not been active over an extended period of time.
Run from/thru	Start and end run numbers for a range of active job networks. Can be used only if owner and network have been selected. Default range: 1 - 9999 (all)
Completed message to Email	E-mail address of the user who receives an e-mail notification when the report generation completed successfully (Subject: `EOR: OGC Report ended ok`) or when the report generation was aborted (Subject: `EOR: OGC Report ended not ok`).
Save As File	See *Report Output Options*.
Type	(Applies to **Activation Overview** only.)

<table>
<tr><th>Field/Column</th><th>Description</th></tr>
<tr><td></td><td>Type of network activation.

Valid selection options:<table>
<tr><td>All types</td><td>All types of activations.</td></tr>
<tr><td></td><td></td></tr>
<tr><td>Manual</td><td>Activated manually.</td></tr>
<tr><td></td><td></td></tr>
<tr><td>Recovery</td><td>Activated by End-of-Job recovery processing.</td></tr>
<tr><td></td><td></td></tr>
<tr><td>API</td><td>Activated by the activation API. NOPUAC5N (see the section API Routines).</td></tr>
<tr><td></td><td></td></tr>
<tr><td>EOJ</td><td>Activated by an End-of-Job action.</td></tr>
<tr><td></td><td></td></tr>
<tr><td>Scheduled</td><td>Activated by a schedule.</td></tr>
</table></td></tr>
<tr><td colspan="2">Compare Networks/Symbol Tables reports only:</td></tr>
<tr><td>Show</td><td>Determines the amount of information to be shown for the compared networks.

Valid selection options:<table>
<tr><td>ALL</td><td>Shows all matching and differing objects and attributes.</td></tr>
<tr><td></td><td></td></tr>
<tr><td>Differences</td><td>Shows only the differing objects with the differing attributes (default).</td></tr>
</table></td></tr>
<tr><td colspan="2">Accounting Information/Schedule of Jobs reports only:</td></tr>
<tr><td>Step</td><td>(Applies to z/OS only.)

Job step.

Step data is only collected if the Collect z/OS step accounting data option is enabled in the Entire Operations default settings: see Defaults for Operating System Specials in the Administration documentation.</td></tr>
<tr><td>JobID</td><td>Job identifier as assigned by the operating system.</td></tr>
<tr><td>Start</td><td>Start date/time of the job.</td></tr>
<tr><td>Stop</td><td>End date/time of job.</td></tr>
<tr><td>Elapsed min

or

Elapsed Time</td><td>Estimated run time (in minutes) of the job.

BS2000:

Since the BS2000 LOGOFF message does not contain seconds, the elapsed time cannot be calculated exactly.</td></tr>
</table>

Field/Column	Description

CPU Time sec	Estimated CPU time (in seconds) of the job.
	
	UNIX:
	
	CPU time information is written to the SYSOUT of UNIX jobs. For this purpose, the shell `times` command is used. The output is enclosed in the messages EOR0303 and EOR0304.
	
	Example:
	
	`%% EOR0303 - times - Begin`
	`0m0.121s 0m0.025s`
	`0m0.043s 0m0.066s`
	`%% EOR0304 - times - End`
	
	Notes:
	
	■ With Entire System Server for UNIX Version 2.1.2 PL 4 and above, this output is used to determine the CPU time consumption of UNIX jobs. The CPU time is included in the Entire Operations accounting data.
	■ For Sun Solaris:
	
	The `times` command output is useful only if the script frame (*.BF) is executed under the Korn shell. Therefore, the *.BF scripts on Sun Solaris are started under the Korn shell.

date **to** *date* **are**	Line below the report table indicating the average CPU consumption. Average values of all elapsed and CPU times for all jobs in the network.
Network/Job Usage reports only:	
Used Job	Networks and jobs used by single or multiple jobs.
Using Job	Networks and jobs using a job as a subnetwork (job type `NET`), an End-of-Job (EOJ) action or a recovery job.
Usage	Indicates the job use: `Subnet`, `EOJ Activation` or `Recovery`.
Activation Mode	Activation mode defined for the subnetwork. See also ***Time of Activation of a Subnetwork*** in the *System Overview*.
Node Overview reports only:	
Server Name	For mainframe nodes: a descriptive name. For UNIX and Windows nodes: the name of the EntireX Broker service.
Node Range	Range of node numbers from 1 to 99900.
Short	Short name of a node.
AM	Access mode used: `N` — Mainframe nodes accessed through Entire Net-Work.

Field/Column	Description
	B — UNIX and Windows nodes accessed through EntireX Broker. L — Applies to UNIX and Windows only. Local node, invoked directly on the machine where Entire Operations is running.
Op. Sys.	Operating system under which the node is running as received from the last SYSTEM-INFO call to Entire System Server or UNIX/Windows system information.
Wait a. Error	Wait time after error. Time in minutes to wait until next node access after a temporary error.
SSU	Submit Security User Type. If empty, the system-wide default is in effect for this node.
Time Diff.	Time difference between local time and GMT in hours if node is in a different time zone.
Valid	Indicates whether a node is available: yes — Node can be used. no — Node has been disabled.
NPR Version	Current Entire System Server version of the node.
OS Release	Information about the operating system (where available).
Bar Chart reports only:	
Run	Run number of the job.
Activation Time	Date/time when the job is activated.
Start time	Date/time when the job started.
Stop Time	Date/time when the job stopped.
Elapsed Time	Elapsed time (between job start time and job stop).
Minimum/Maximum Elapsed Time	Minimum or maximum elapsed time per job run.
CPU Time	CPU time in milliseconds (ms).
Stop Job	Last executed job.
Stop Message	Message of the last executed job.
Run OK?	Is true if the networks finished without errors.

Range Specification and Usage Restrictions

If indicated in the table above, you can use an asterisk (*) or a smaller/greater than (< or >) wildcard character to determine a start or an end value (see also *Valid Name Specifications* in *Direct Commands*) or open a selection list of names.

The use of asterisk/wildcard report selection can be disallowed for non-administrator users (see *User Definitions and Profile Settings* in the *Administration* documentation).

By default, users of type A (system administrator) and O (operator) are allowed to use asterisk/wild-card selection; users of type G (general user) are not. It is not possible to disallow it for users of type A.

Determination Date for Report Data

Report data is evaluated for the current day by default. This evaluation date needs to be considered, in particular, when you maintain different versions of networks and symbol tables with either fix versions or versions dynamically loaded on this date.

The determination date can also affect named filters that use the selection criterion `(current)`: see *Filtering Objects* and *Changes to the Determination Date*.

You can specify the date for which a report (or a named filter with `(current)`) evaluates the data to be generated.

To set the determination date

1 In the main menu, choose **General**.

2 Choose **Determination Date**.

A **Determination Date** dialog opens.

3 Open the drop-down list box and select the required date from a calendar.

4 Choose **Apply** to save the date.

Report Output Options

The **Save as File** dialog specifies the format for the report and the location where the report is written.

Field	Explanation
File type	The format used for report generation: ■ **HTML**: a file in HTML (Hypertext Markup Language) format. ■ **HTML5**: a file in HTML5 format is required to display **bar charts** reports. ■ This only works if the Windows **Open with** function calls an HTML5-enabled browser by default. ■ **CSV**: a comma-separated value file for Excel.

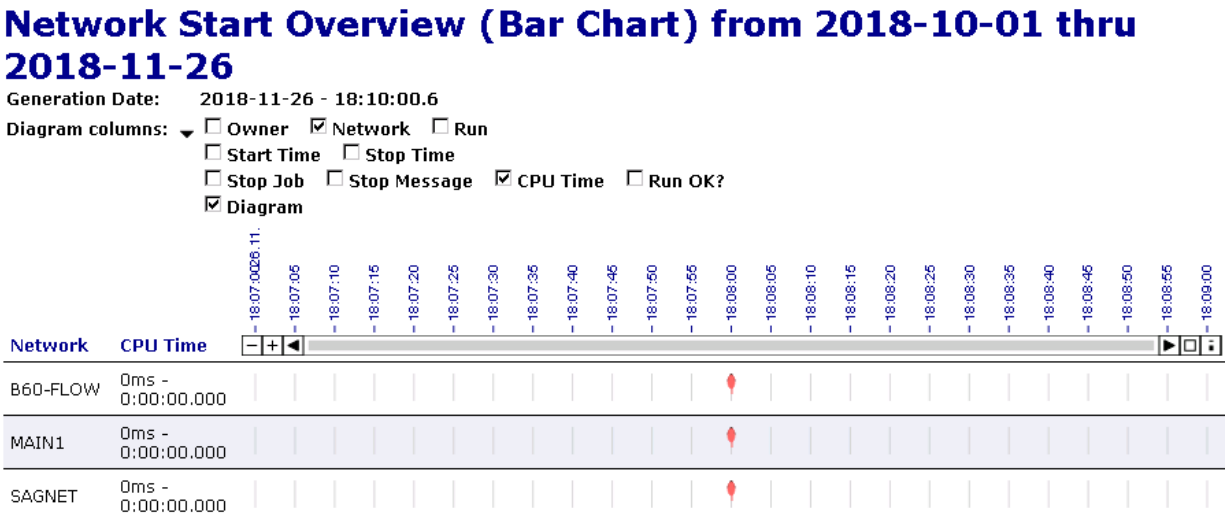
Field	Explanation
	■ **XML**: a file in XML (Extensible Markup Language) format without style sheet for HTML. ■ **XML with style sheet for HTML**: a file in XML format with a style sheet for HTML. If this option is selected, a processing instruction with a reference to an external style sheet is added to the generated XML file to transform the XML file into HTML.
File name	The name to be used for the report file. If no path is specified in the **File name** field, the file will be saved at: `%LOCALAPPDATA%\Software AG\Natural\`
Open with external application	If this option is checked (default setting), the default Windows application associated with the specific file extension opens.

The settings in the **Save as File** dialog are stored between sessions in a client-specific XML profile and are used as the default value for the next **Save as File** operation on a report of the same report type.

Using Reports with Bar Charts

The report types **Network Start Overview (Bar Chart)**, **Network and Job Start Overview (Bar Chart)** and **Network Schedule Overview (Bar Chart)** generate the usual report columns but additionally provide an interactive diagram with a bar chart for better data visualization.

Bar charts of the report types visualize report column data as an interactive diagram. Bar charts are useful for overviewing the job flow and finding out the peak and off-peak times to balance the workload (CPU load) among the different networks.

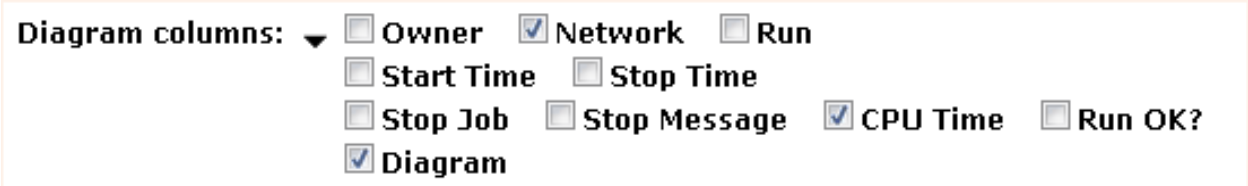


Bar charts reports are not static but have interactive elements which are described in the following section:

- Filter Area

- Diagram Part

Filter Area



You can collapse the filter area by clicking on the arrow:



Depending on the diagram columns you select, different columns of the report are shown:

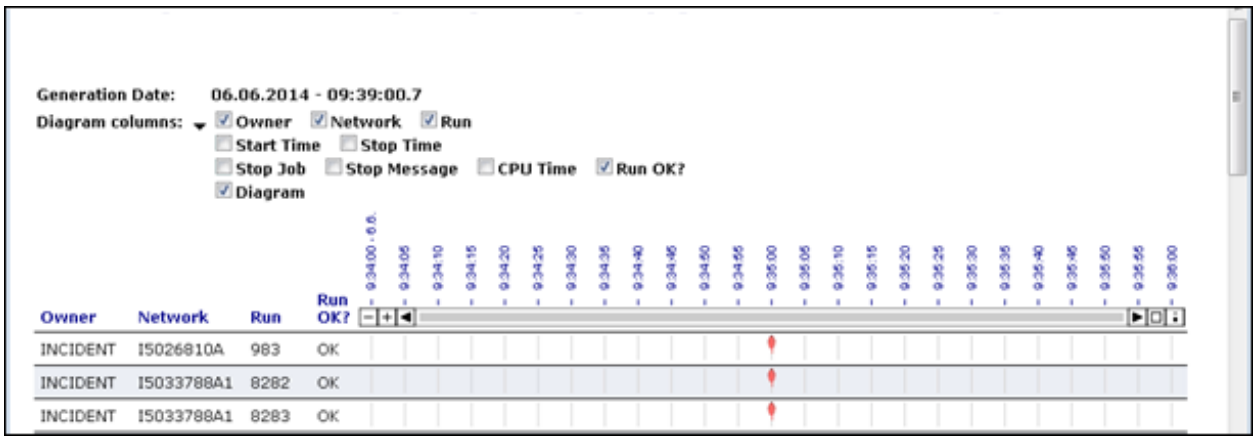
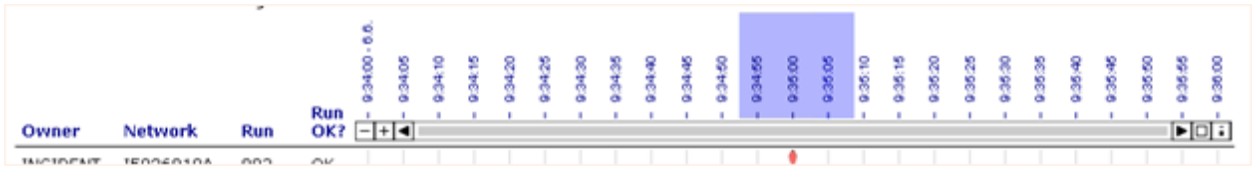
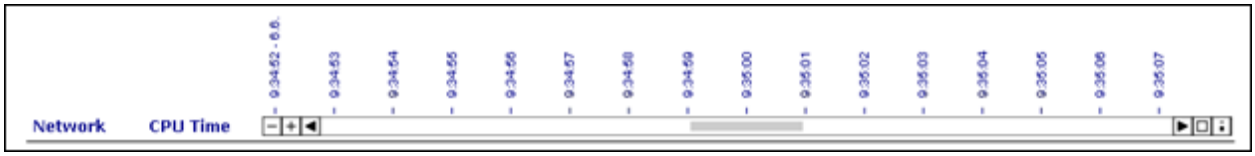


Diagram Part

If you select part of the time table as shown below,



the diagram is zoomed in so you can view more details:

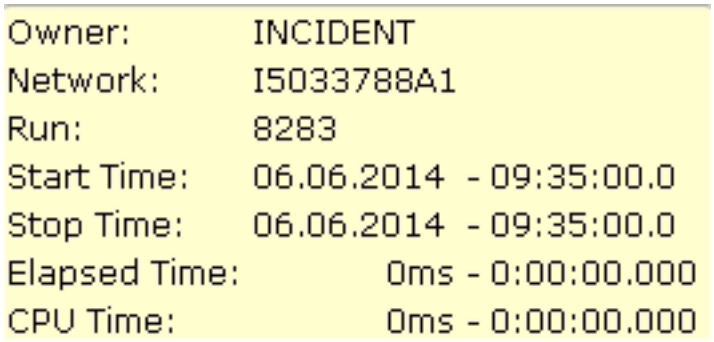


Symbol	Explanation
	Zooms the diagram out (-) or in (+).
	Scrolls left or right in the diagram.
	Shows the complete diagram.
	Shows further information/explanations.

If you move the mouse over the marker, you can view further information such as start and/or end times:



If you move the mouse over a bar, you can view additional information about the given network:



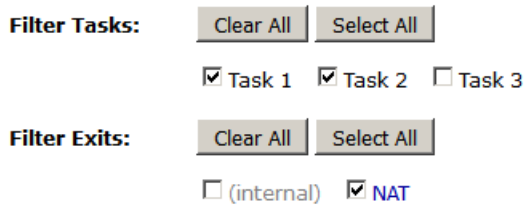
Using Reports with Monitor Task Time Tables

The report types **Monitor Tasks and Functions Overview** and **Monitor Tasks and Exits Overview** provide an interactive diagram with a bar chart and features to filter information as described in the following section.

- Filter Area

- Diagram Part

Filter Area



You can unmark or mark the items to deselect or select the items to be shown in the diagram.

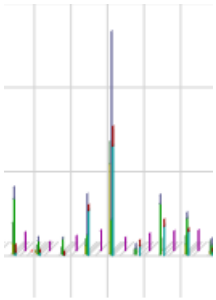
You can choose **Clear All** or **Select All** to deselect or select all items marked in the **Tasks**, **Exits** and/or **Functions** rows.

Diagram Part

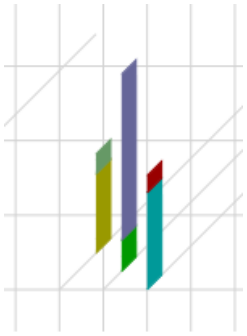
If you select part of the time table as shown below,



the diagram

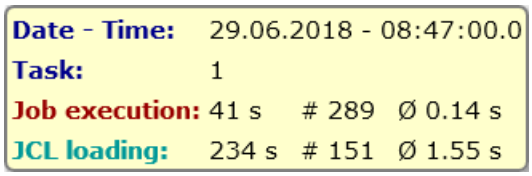


is zoomed in so you can view more details:



Symbol	Explanation
	Zooms the diagram out (-) or in (+).
	Scrolls left or right in the diagram.
	Shows the complete diagram.

If you move the mouse over the top of a bar graph in the diagram, you can view additional information such as the execution time, average execution time and the number of calls, for example:



User Restrictions for Reports

Administrator users (and reports run in batch) may report on any network in any owner. Non-administrator users, however, will only receive information on networks to which they have read access. A user is considered to have read access if:

1. The user is linked to owner SYSDBA.
2. The user is linked to the network owner.
3. Access to the network has been explicitly granted to the user.

User authorization for performing report functions are granted in the default user profile as described in *Reporting Functions* in the *Administration* documentation.

Retention Period for Reports from Entire Operations GUI Client

Data generated for a report from Entire Operations GUI Client is retained for the number of days specified for active jobs in the **Retention Periods** of the Entire Operations default settings (see *Defaults for Time Ranges* in the *Administration* documentation).

Older reports listed in the **Reporting** window of Entire Operations GUI Client are automatically removed after this period or during the next database cleanup (see also the *Administration* documentation).

Examples of Reports

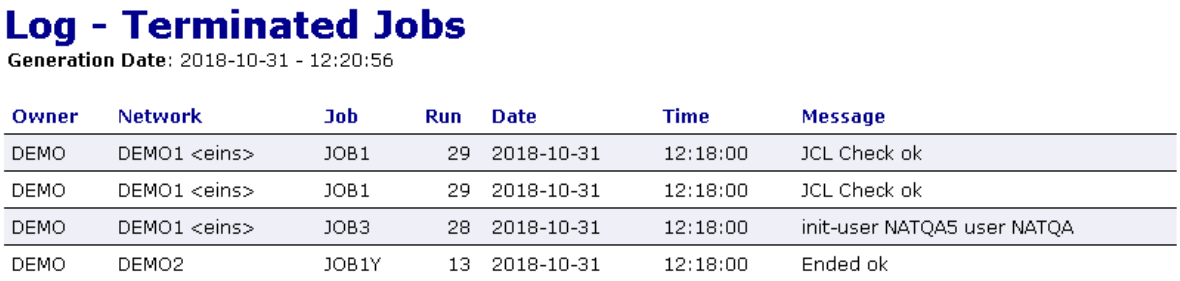
This section contains examples of all types of reports you can generate with the **Reporting** function:

- Example of Log - Terminated Jobs
- Example of Log - Abended Jobs
- Example of Log - Jobs not started
- Example of Log - Jobs with permanent errors
- Example of Log – Networks not activated
- Example of Accounting Information
- Example of Network Description (short)
- Example of Network Description (detailed)
- Example of Schedule of Jobs
- Example of Network Start Summary
- Example of Network Schedule Overview
- Example of Activation Overview
- Example of Compare Symbol Tables
- Example of Compare Networks
- Example of Node Overview
- Example of Network/Job Usage
- Example of Network Start Overview (Bar Chart)
- Example of Network and Job Start Overview (Bar Chart)
- Example of Network Schedule Overview (Bar Chart)
- Example of Monitor Tasks and Exits Overview

- Example of Monitor Tasks and Functions Overview

Example of Log - Terminated Jobs

The following is an example of the report type **Log - Terminated Jobs**:



Log - Terminated Jobs

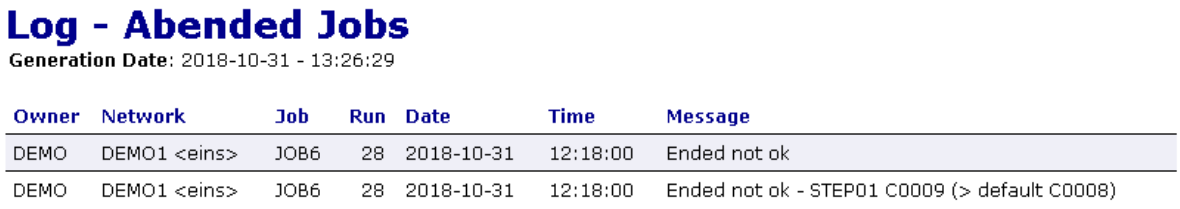
Generation Date: 2018-10-31 - 12:20:56

Owner	Network	Job	Run	Date	Time	Message
DEMO	DEMO1 <eins>	JOB1	29	2018-10-31	12:18:00	JCL Check ok
DEMO	DEMO1 <eins>	JOB1	29	2018-10-31	12:18:00	JCL Check ok
DEMO	DEMO1 <eins>	JOB3	28	2018-10-31	12:18:00	init-user NATQA5 user NATQA
DEMO	DEMO2	JOB1Y	13	2018-10-31	12:18:00	Ended ok

The columns contained in the report are described in *Fields and Columns: Reports*.

Example of Log - Abended Jobs

The following is an example of the report type **Log - Abended Jobs**:



Log - Abended Jobs

Generation Date: 2018-10-31 - 13:26:29

Owner	Network	Job	Run	Date	Time	Message
DEMO	DEMO1 <eins>	JOB6	28	2018-10-31	12:18:00	Ended not ok
DEMO	DEMO1 <eins>	JOB6	28	2018-10-31	12:18:00	Ended not ok - STEP01 C0009 (> default C0008)

The columns contained in the report are described in *Fields and Columns: Reports*.

Example of Log - Jobs not started

The following is an example of the report type **Log - Jobs not started**:

Log - Jobs not started

Generation Date: 2018-10-31 - 13:31:38

Owner	Network	Job	Run	Date	Time	Message
DEMO	DEMO1 <eins>	JOB1	28	2018-10-31	12:18:00	+BDE-KKL - 0 - ABS not found
DEMO	DEMO1 <eins>	JOB1	28	2018-10-31	12:18:00	... referenced Run: -1
DEMO	DEMO1 <eins>	JOB2	28	2018-10-31	12:18:00	JCL Load - ESY5995 File not in catalog.
DEMO	DEMO1 <eins>	JOB2	28	2018-10-31	12:18:00	... TST.NOP542.JOBS/FTPNOPCF
DEMO	DEMO1 <eins>	JOB2	28	2018-10-31	12:18:00	JCL Load Error occurred

The columns contained in the report are described in *Fields and Columns: Reports*.

Example of Log - Jobs with permanent errors

The following is an example of the report type **Log - Jobs with permanent errors**:

Log - Jobs with permanent errors

Generation Date: 2018-10-31 - 13:39:17

Owner	Network	Job	Run	Date	Time	Message
DEMO	DEMO1 <eins>	JOB2	28	2018-10-31	12:18:00	JCL Load - ESY5995 File not in catalog.
DEMO	DEMO1 <eins>	JOB2	28	2018-10-31	12:18:00	... TST.NOP542.JOBS/FTPNOPCF
DEMO	DEMO1 <eins>	JOB2	28	2018-10-31	12:18:00	JCL Load Error occurred
DEMO	DEMO1 <eins>	JOB2	28	2018-10-31	12:18:00	JCL Load - ESY5995 File not in catalog.
DEMO	DEMO1 <eins>	JOB5	28	2018-10-31	12:18:00	JCL Load - ESY5995 File not in catalog.

The columns contained in the report are described in *Fields and Columns: Reports*.

Example of Log – Networks not activated

The following is an example of a **Log – Networks not activated** report:

Log - Networks not activated

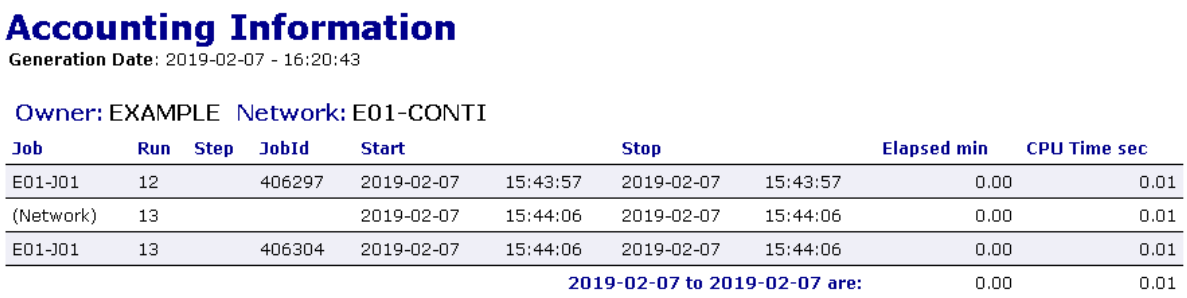
Generation Date: 2018-10-31 - 13:56:44

Owner	Network	Job	Run	Date	Time	Message
EXAMPLE	B60-FLOW	-	478	2018-10-30	14:34:00	Calendar EXAMPLE/EXAMPLECAL undefined for 2018
EXAMPLE	B60-FLOW	-	480	2018-10-30	14:51:00	Calendar EXAMPLE/EXAMPLECAL undefined for 2018
EXAMPLE	B60-FLOW	-	481	2018-10-30	14:52:00	Calendar EXAMPLE/EXAMPLECAL undefined for 2018

The columns contained in the report are described in *Fields and Columns: Reports*.

Example of Accounting Information

The following is an example of the report type **Accounting Information**:



Accounting Information

Generation Date: 2019-02-07 - 16:20:43

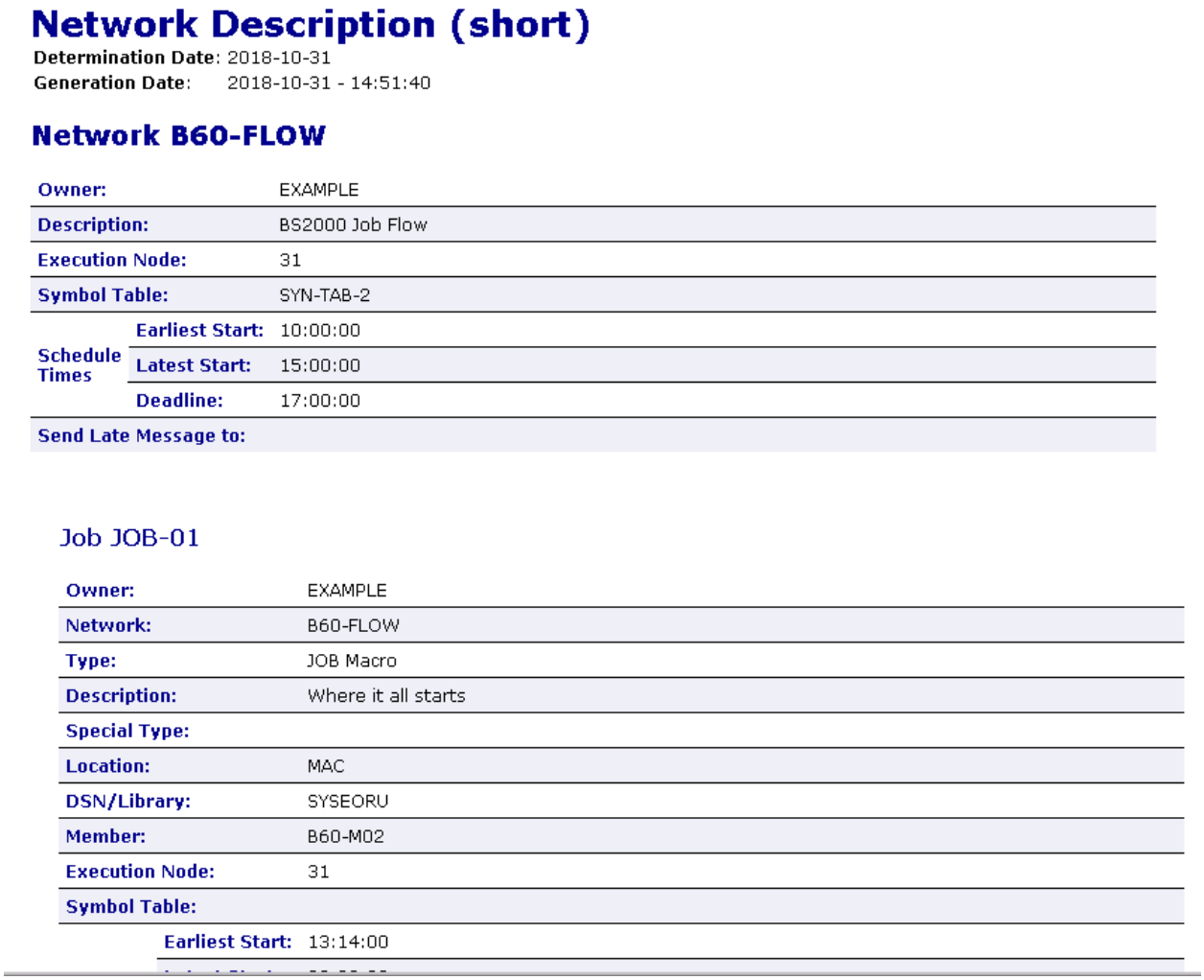
Owner: EXAMPLE Network: E01-CONTI

Job	Run	Step	JobId	Start		Stop		Elapsed min	CPU Time sec
E01-J01	12		406297	2019-02-07	15:43:57	2019-02-07	15:43:57	0.00	0.01
(Network)	13			2019-02-07	15:44:06	2019-02-07	15:44:06	0.00	0.01
E01-J01	13		406304	2019-02-07	15:44:06	2019-02-07	15:44:06	0.00	0.01
						2019-02-07 to 2019-02-07 are:		0.00	0.01

The columns contained in the report are described under *Accounting Information/Schedule of Jobs reports only* in *Fields and Columns: Reports*.

Example of Network Description (short)

The following is an example of the report type **Network Description (short)**:



Network Description (short)

Determination Date: 2018-10-31
Generation Date: 2018-10-31 - 14:51:40

Network B60-FLOW

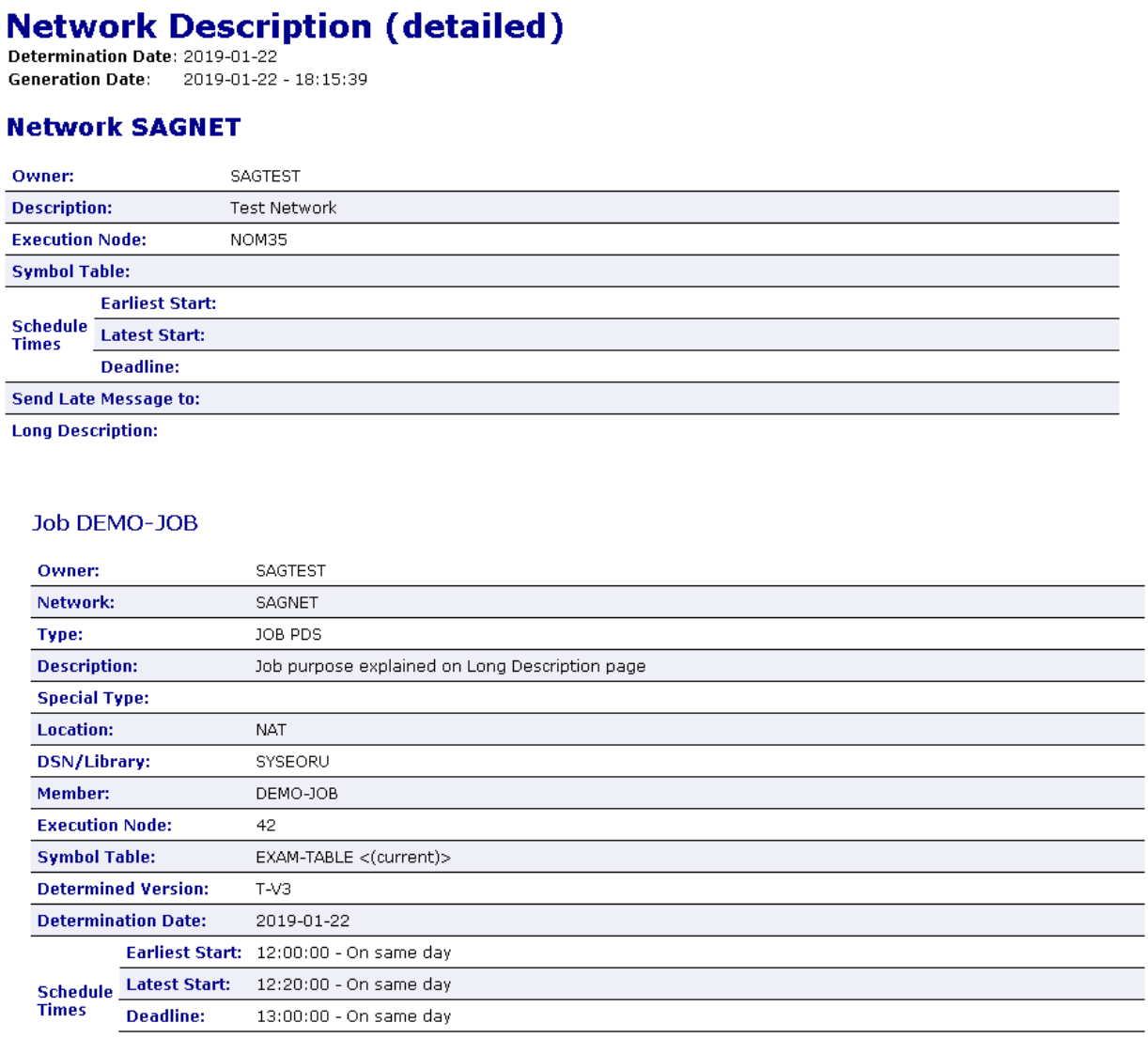
Owner:		EXAMPLE
Description:		BS2000 Job Flow
Execution Node:		31
Symbol Table:		SYN-TAB-2
Schedule Times	**Earliest Start:**	10:00:00
	Latest Start:	15:00:00
	Deadline:	17:00:00
Send Late Message to:		

Job JOB-01

Owner:	EXAMPLE
Network:	B60-FLOW
Type:	JOB Macro
Description:	Where it all starts
Special Type:	
Location:	MAC
DSN/Library:	SYSEORU
Member:	B60-M02
Execution Node:	31
Symbol Table:	
Earliest Start:	13:14:00

The fields contained in the report are described in *Fields and Columns: Reports*.

Example of Network Description (detailed)



Network Description (detailed)

Determination Date: 2019-01-22
Generation Date: 2019-01-22 - 18:15:39

Network SAGNET

Owner:		SAGTEST
Description:		Test Network
Execution Node:		NOM35
Symbol Table:		
Schedule Times	**Earliest Start:**	
	Latest Start:	
	Deadline:	
Send Late Message to:		
Long Description:		

Job DEMO-JOB

Owner:		SAGTEST
Network:		SAGNET
Type:		JOB PDS
Description:		Job purpose explained on Long Description page
Special Type:		
Location:		NAT
DSN/Library:		SYSEORU
Member:		DEMO-JOB
Execution Node:		42
Symbol Table:		EXAM-TABLE <(current)>
Determined Version:		T-V3
Determination Date:		2019-01-22
Schedule Times	**Earliest Start:**	12:00:00 - On same day
	Latest Start:	12:20:00 - On same day
	Deadline:	13:00:00 - On same day

Example of Schedule of Jobs

The following is an example of the report type **Schedule of Jobs**:

Production Plan from 2018-10-31 thru 2018-10-31

Determination Date: 2018-10-31
Generation Date: 2018-10-31 - 15:31:37

Date: 2018-10-31

Owner	Network	Job	Description	Start	Elapsed Time
EOR3014 - Calendar EXAMPLE/EXAMPLECAL undefined for 2018					
EOR3014 - Calendar INCIDENT/I1044152 undefined for 2018					
EOR3014 - Calendar SN/A1 undefined for 2018					
EOR3014 - Calendar INCIDENT/I5060981 undefined for 2018					
EOR3014 - Calendar INCIDENT/SYSCAL undefined for 2018					
EOR3014 - Calendar INCIDENT/I5067009 undefined for 2018					
INCIDENT	I5095089A	SN000101		23:50	9.92
INCIDENT	I5095089A	SN000102		23:55	9.91

The columns contained in the report are described under *Accounting Information/Schedule of Jobs reports only* in *Fields and Columns: Reports*.

Example of Network Start Summary

The following is an example of the report type **Network Start Summary**:

Network Start Summary from 2018-10-01 thru 2018-10-31

Generation Date: 2018-10-31 - 17:42:08

Owner: EXAMPLE Network: B60-FLOW

Run	Date	Time	Job	Message
	0000-01-02			EOR3014 - Calendar EXAMPLE/EXAMPLECAL undefined for 2018

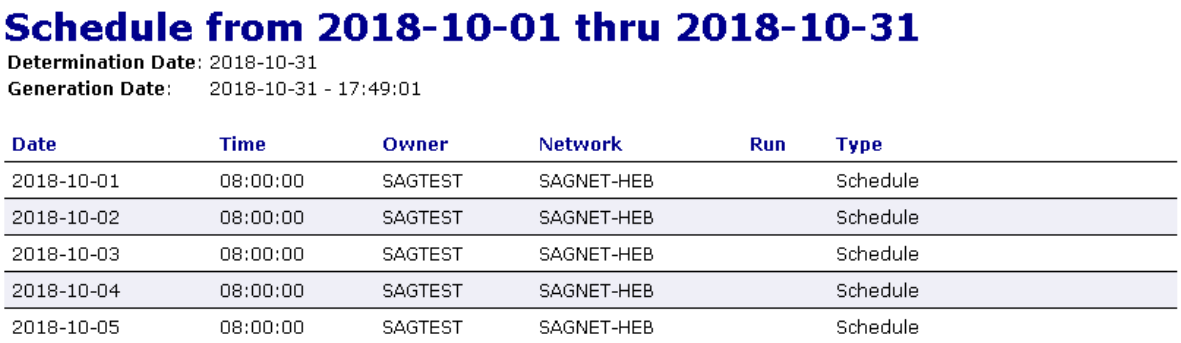
Owner: INCIDENT Network: I1042163

Run	Date	Time	Job	Message
	2018-10-01	00:00		Scheduled, but not extracted
	2018-10-02	00:00		Scheduled, but not extracted
	2018-10-07	00:00		Scheduled, but not extracted
	2018-10-08	00:00		Scheduled, but not extracted

The columns contained in the report are described in *Fields and Columns: Reports*.

Example of Network Schedule Overview

The following is an example of the report type **Network Schedule Overview**:



Schedule from 2018-10-01 thru 2018-10-31

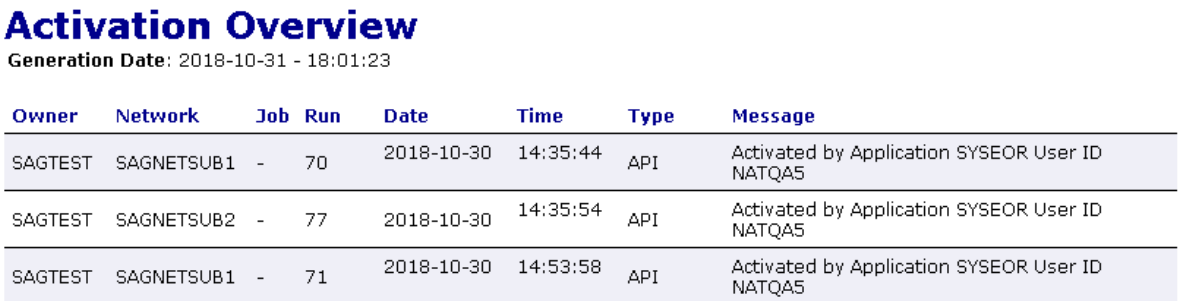
Determination Date: 2018-10-31
Generation Date: 2018-10-31 - 17:49:01

Date	Time	Owner	Network	Run	Type
2018-10-01	08:00:00	SAGTEST	SAGNET-HEB		Schedule
2018-10-02	08:00:00	SAGTEST	SAGNET-HEB		Schedule
2018-10-03	08:00:00	SAGTEST	SAGNET-HEB		Schedule
2018-10-04	08:00:00	SAGTEST	SAGNET-HEB		Schedule
2018-10-05	08:00:00	SAGTEST	SAGNET-HEB		Schedule

The columns contained in the report are described in *Fields and Columns: Reports*.

Example of Activation Overview

The following is an example of the report type **Activation Overview**:



Activation Overview

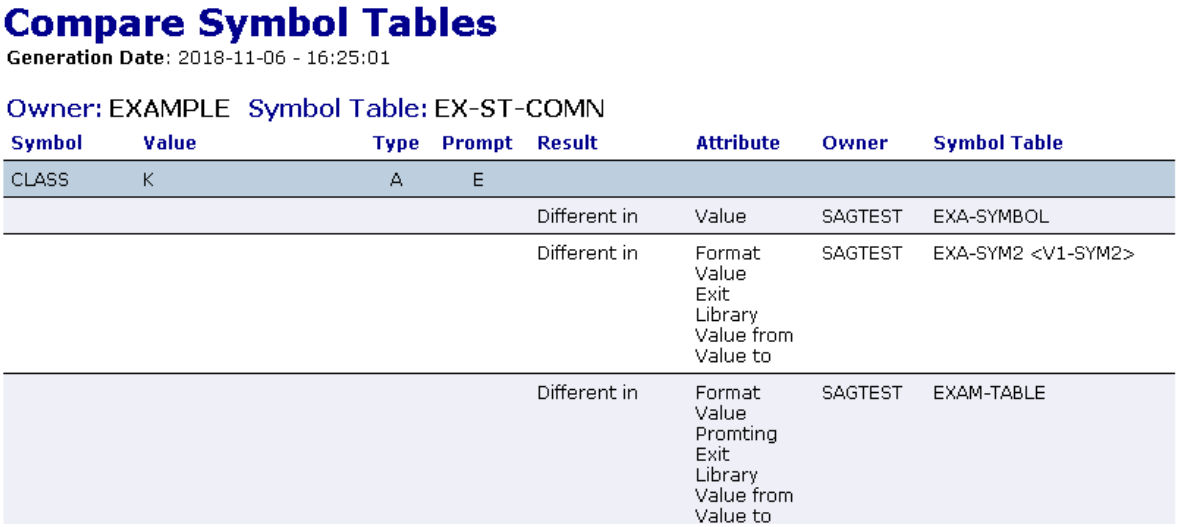
Generation Date: 2018-10-31 - 18:01:23

Owner	Network	Job	Run	Date	Time	Type	Message
SAGTEST	SAGNETSUB1	-	70	2018-10-30	14:35:44	API	Activated by Application SYSEOR User ID NATQA5
SAGTEST	SAGNETSUB2	-	77	2018-10-30	14:35:54	API	Activated by Application SYSEOR User ID NATQA5
SAGTEST	SAGNETSUB1	-	71	2018-10-30	14:53:58	API	Activated by Application SYSEOR User ID NATQA5

The columns contained in the report are described in *Fields and Columns: Reports*.

Example of Compare Symbol Tables

The following is an example of the report type **Compare Symbol Tables**:



Compare Symbol Tables

Generation Date: 2018-11-06 - 16:25:01

Owner: EXAMPLE Symbol Table: EX-ST-COMN

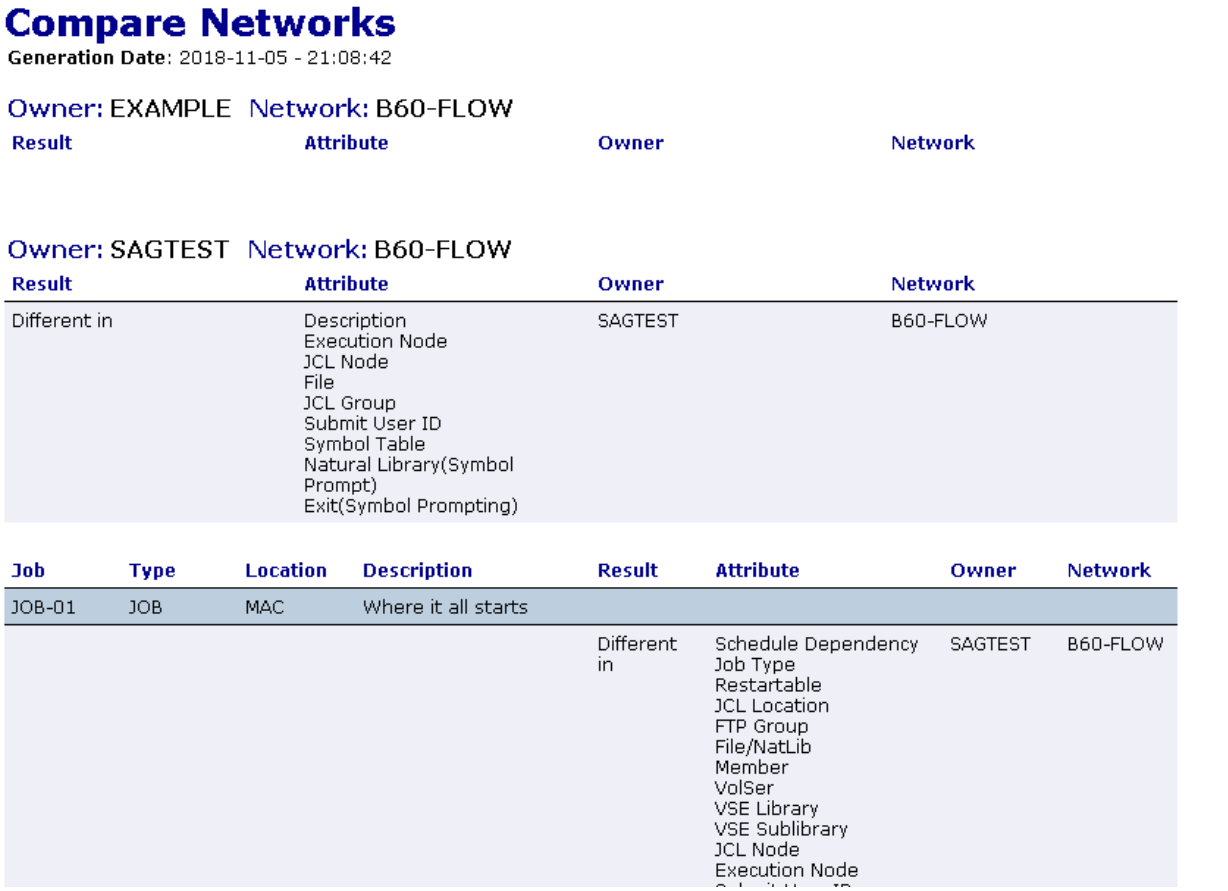
Symbol	Value	Type	Prompt	Result	Attribute	Owner	Symbol Table
CLASS	K	A	E				
				Different in	Value	SAGTEST	EXA-SYMBOL
				Different in	Format Value Exit Library Value from Value to	SAGTEST	EXA-SYM2 <V1-SYM2>
				Different in	Format Value Promting Exit Library Value from Value to	SAGTEST	EXAM-TABLE

Columns: Compare Symbol Tables

The columns contained in the report are described under *Compare Networks/Symbol Tables reports only* in *Fields and Columns: Reports*.

Example of Compare Networks

The following is an example of the report type **Compare Networks**:



Compare Networks

Generation Date: 2018-11-05 - 21:08:42

Owner: EXAMPLE Network: B60-FLOW

Result	Attribute	Owner	Network

Owner: SAGTEST Network: B60-FLOW

Result	Attribute	Owner	Network
Different in	Description Execution Node JCL Node File JCL Group Submit User ID Symbol Table Natural Library(Symbol Prompt) Exit(Symbol Prompting)	SAGTEST	B60-FLOW

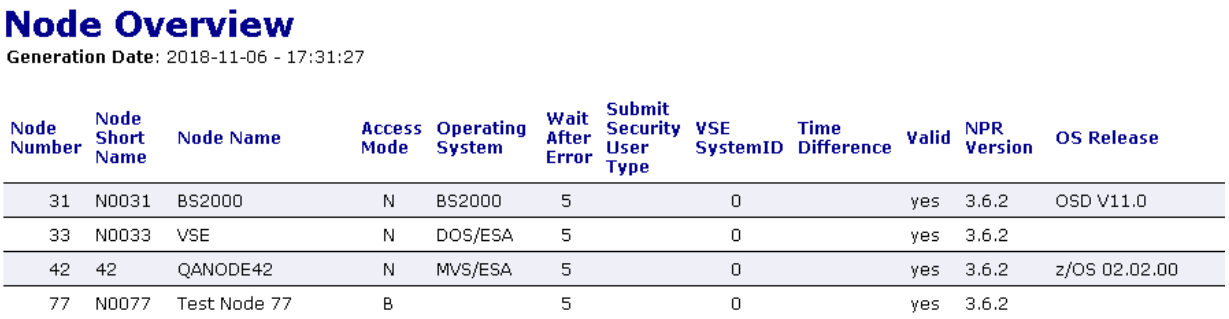
Job	Type	Location	Description	Result	Attribute	Owner	Network
JOB-01	JOB	MAC	Where it all starts				
				Different in	Schedule Dependency Job Type Restartable JCL Location FTP Group File/NatLib Member VolSer VSE Library VSE Sublibrary JCL Node Execution Node	SAGTEST	B60-FLOW

Columns: Compare Networks

The columns contained in the report are described under *Compare Networks/Symbol Tables reports only* in *Fields and Columns: Reports*.

Example of Node Overview

The following is an example of the report type **Node Overview**:



Node Overview

Generation Date: 2018-11-06 - 17:31:27

Node Number	Node Short Name	Node Name	Access Mode	Operating System	Wait After Error	Submit Security User Type	VSE SystemID	Time Difference	Valid	NPR Version	OS Release
31	N0031	BS2000	N	BS2000	5		0		yes	3.6.2	OSD V11.0
33	N0033	VSE	N	DOS/ESA	5		0		yes	3.6.2	
42	42	QANODE42	N	MVS/ESA	5		0		yes	3.6.2	z/OS 02.02.00
77	N0077	Test Node 77	B		5		0		yes	3.6.2	

The columns contained in the report are described under *Node Overview reports only* in *Fields and Columns: Reports*.

Example of Network/Job Usage

The following is an example of the report type **Network/Job Usage**:

Network/Job Usage

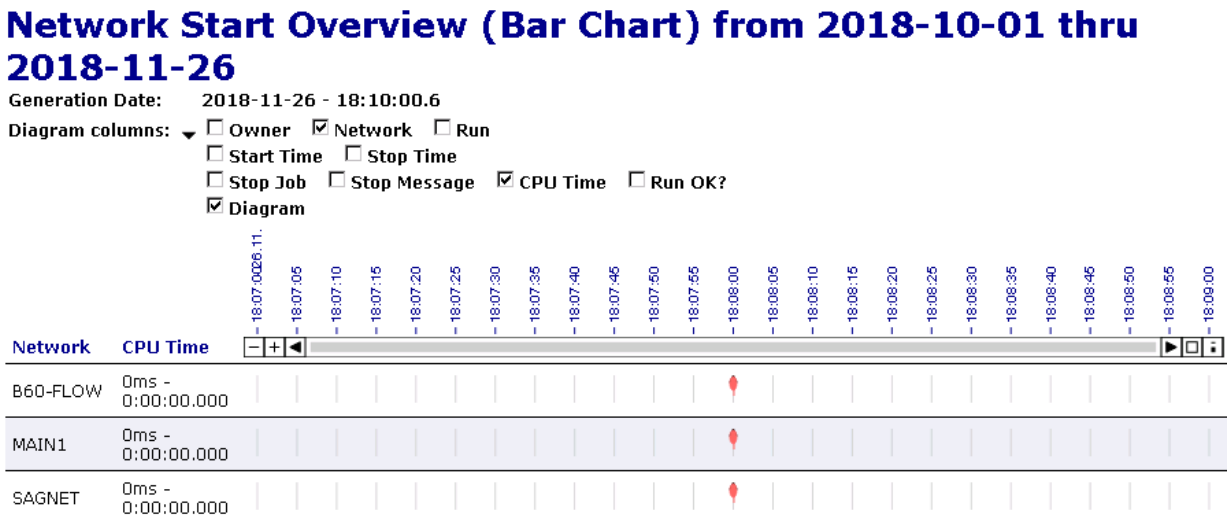
Generation Date: 2018-11-26 - 17:07:10

Used Job			Using Job				
Owner	Network	Job	Owner	Network	Job	Usage	Activation Mode
EXAMPLE	E01-CONTI	*	EXAMPLE	MAIN1	JOB-01	EOJ Activation	
EXAMPLE	E01-CONTI	*	EXAMPLE	MAIN2	JOB-01	EOJ Activation	
EXAMPLE	E40-REC-02	E40-J01-RC	EXAMPLE	E40-REC-01	E40-J02	Recovery	
EXAMPLE	E62-NET-B	*	EXAMPLE	E62-NET	E62-J02	EOJ Activation	
EXAMPLE	V40-REC-02	E40-J01-RC	EXAMPLE	V40-REC-01	E40-J02	Recovery	

The columns contained in the report are described under *Network/Job Usage reports only* in *Fields and Columns: Reports*.

Example of Network Start Overview (Bar Chart)

The following is an example of the report type **Example of Network Start Overview (Bar Chart)**:

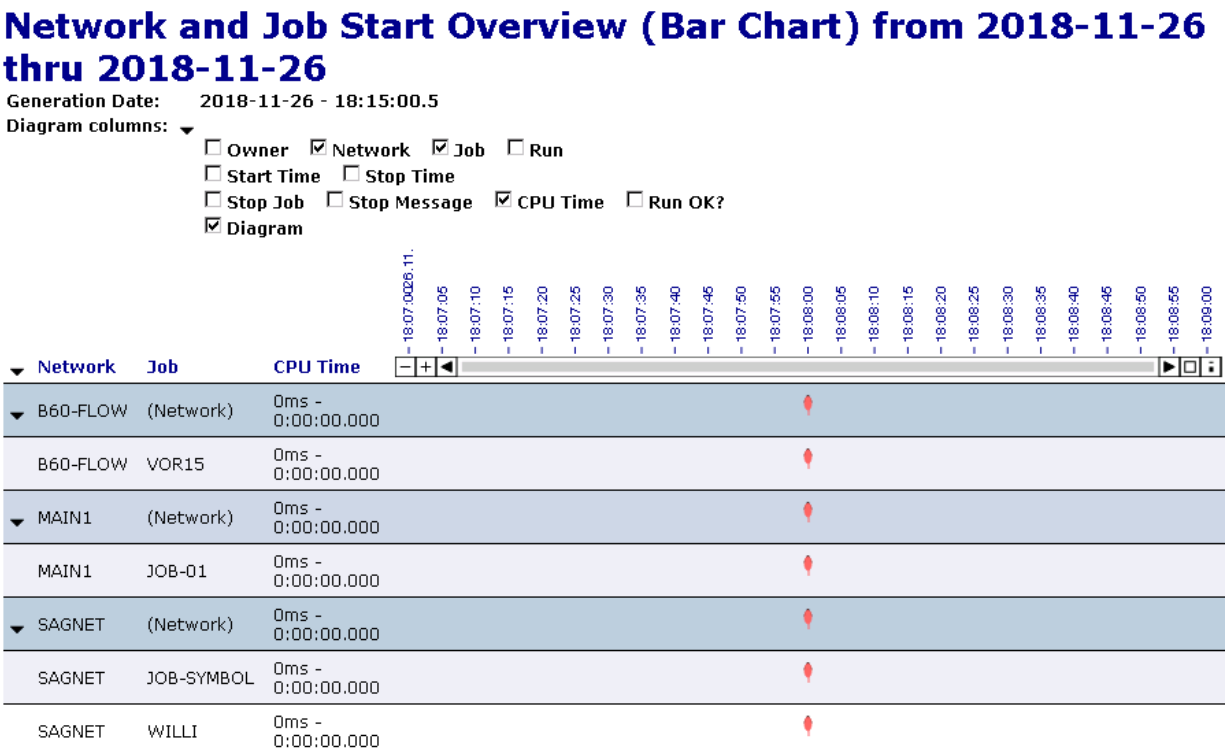


The columns contained in the report are described under *Bar Chart reports only* in *Fields and Columns: Reports*.

See also the section *Using Reports with Bar Charts* for information on how to handle bar charts.

Example of Network and Job Start Overview (Bar Chart)

The following is an example of the report type **Network and Job Start Overview (Bar Chart)**:



Network and Job Start Overview (Bar Chart) from 2018-11-26 thru 2018-11-26

Generation Date: 2018-11-26 - 18:15:00.5

Diagram columns:

☐ Owner ☑ Network ☑ Job ☐ Run
☐ Start Time ☐ Stop Time
☐ Stop Job ☐ Stop Message ☑ CPU Time ☐ Run OK?
☑ Diagram

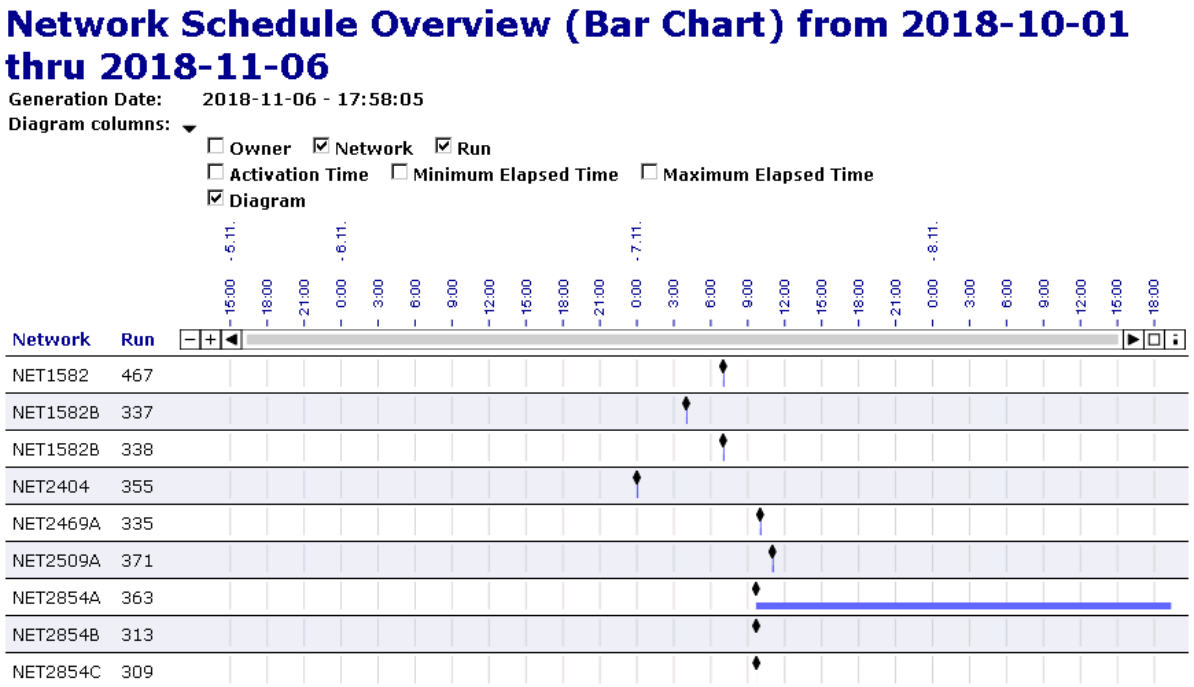
Network	Job	CPU Time
B60-FLOW	(Network)	0ms - 0:00:00.000
B60-FLOW	VOR15	0ms - 0:00:00.000
MAIN1	(Network)	0ms - 0:00:00.000
MAIN1	JOB-01	0ms - 0:00:00.000
SAGNET	(Network)	0ms - 0:00:00.000
SAGNET	JOB-SYMBOL	0ms - 0:00:00.000
SAGNET	WILLI	0ms - 0:00:00.000

The columns contained in the report are described under *Bar Chart reports only* in *Fields and Columns: Reports*.

See also the section *Using Reports with Bar Charts* for information on how to handle bar charts.

Example of Network Schedule Overview (Bar Chart)

The following is an example of the report type **Network Schedule Overview (Bar Chart)**:

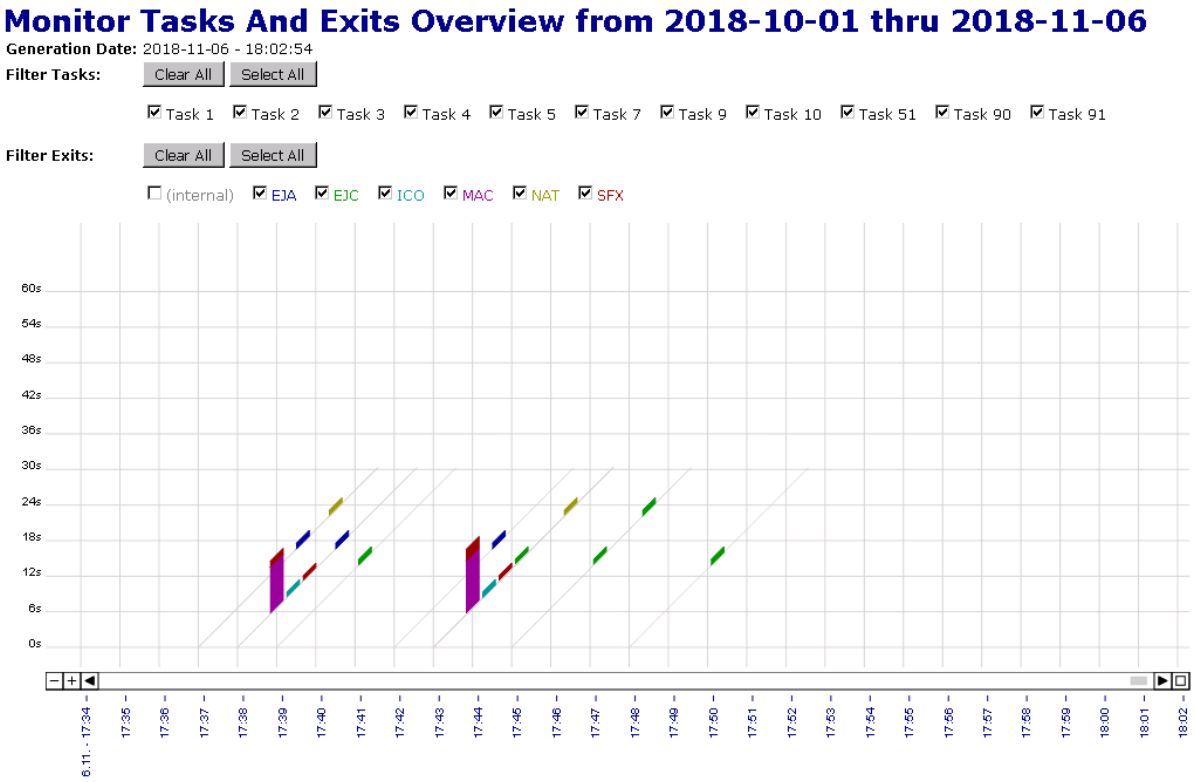


The columns contained in the report are described under *Bar Chart reports only* in *Fields and Columns: Reports*.

See also the section *Using Reports with Bar Charts* for information on how to handle bar charts.

Example of Monitor Tasks and Exits Overview

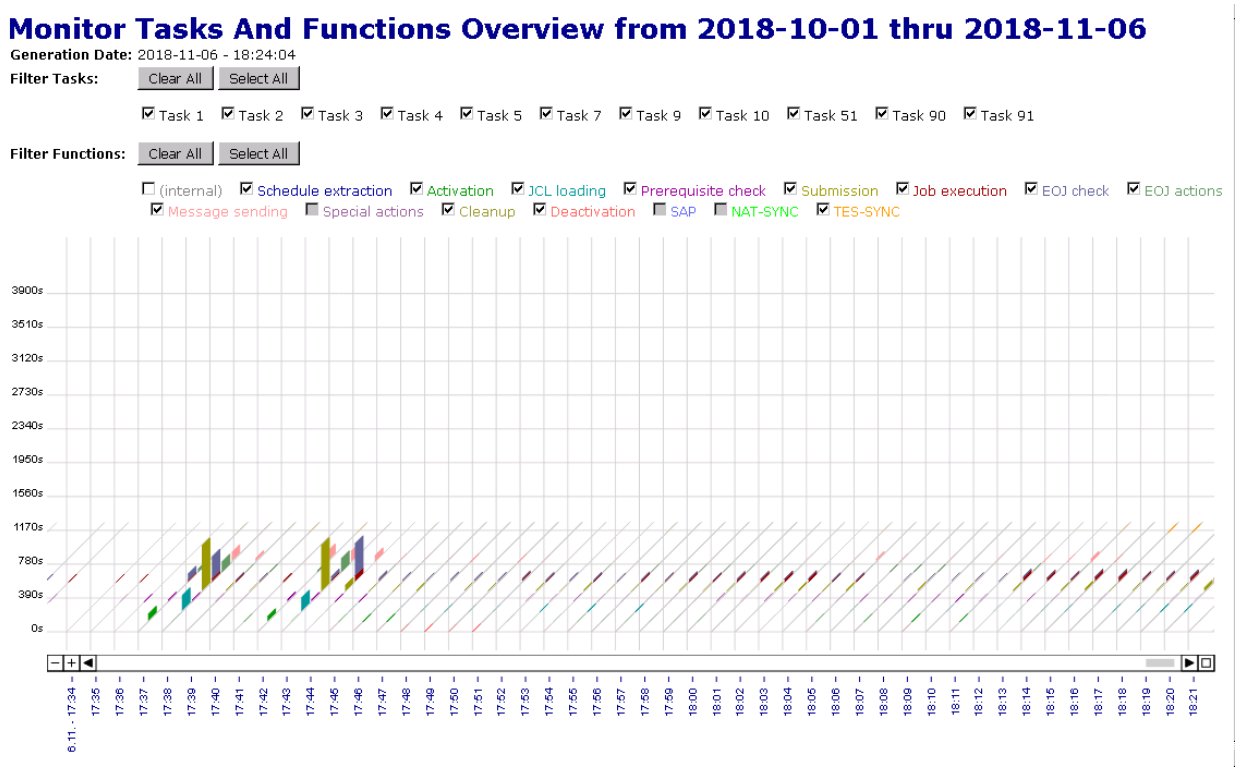
The following is an example of the report type **Monitor Tasks and Exits Overview**:



See also the section *Using Reports with Monitor Task Time Tables* for information on how to handle task time tables.

Example of Monitor Tasks and Functions Overview

The following is an example of the report type **Monitor Tasks and Functions Overview**:



See also the section ***Using Reports with Monitor Task Time Tables*** for information on how to handle task time tables.

Generating Batch Reports

You can use the batch command client of Entire Systems Management to generate a report in batch mode: see ***Commands for the Batch Command Client*** in the section *Using Entire Operations in Batch Mode*.

XIV Cross-References

57 Cross-References

- Types of Cross-Reference Reports 714
- Generating Cross-Reference Reports Online 715
- Fields and Columns: Cross-References 722
- Examples of Cross-Reference Reports 727
- Generating Cross-Reference Reports in Batch 730

The **Cross-References** function is used to cross-check the use of individual Entire Operations objects and produce a report from the data found.

This section describes how to generate cross-reference reports in online mode.

Related Topic:

- For general information on using reports, see *Generating or Regenerating Online Reports* in the section *Reporting*.

Types of Cross-Reference Reports

The types of cross-references you can select from the **Cross-References window** are described in the following table.

Report Type	Description
Exit Usage	Lists user exits defined in your environment. See also *Example of User Exit Usage*.
Symbol Table Usage	Lists symbol tables used in networks and jobs. **Note:** The cross-reference for symbol tables includes the symbol table usage for input conditions and for End-of-Job symbol settings. See also *Example of Symbol Table Usage*.
Symbol and Symbol Value Search	Lists symbols that match a specified value. See also *Example of Symbol and Symbol Value Search*.
JCL Usage	Lists JCL files used in networks and jobs. See also *Example of JCL Usage*.
Node Usage	Lists JCL nodes and execution nodes used in networks. See also *Example of Node Usage*.
Resource Usage	Lists resources used by active jobs. See also *Example of Resource Usage*.
List Undefined Objects	Lists objects that are referenced by other objects which are not defined in your environment. The missing definition of these objects might lead to errors during network activation. **Note:** JCL is not inspected for undefined symbols. See also *Example of List Undefined Objects*.

Generating Cross-Reference Reports Online

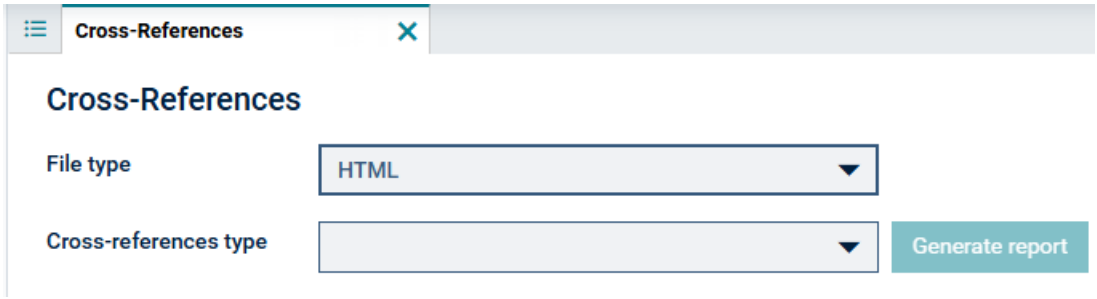
This section describes how to generate cross-reference reports online.

Report data is evaluated for the current day by default. If you want to change the evaluation date, use the **Determination Date** function described in the section *Reporting*.

To generate a cross-reference report

1 In menu **General**, choose **Cross-References**.

A **Cross-References** dialog opens.



2 In **File type** choose the desired output format for the cross-reference report to be generated.

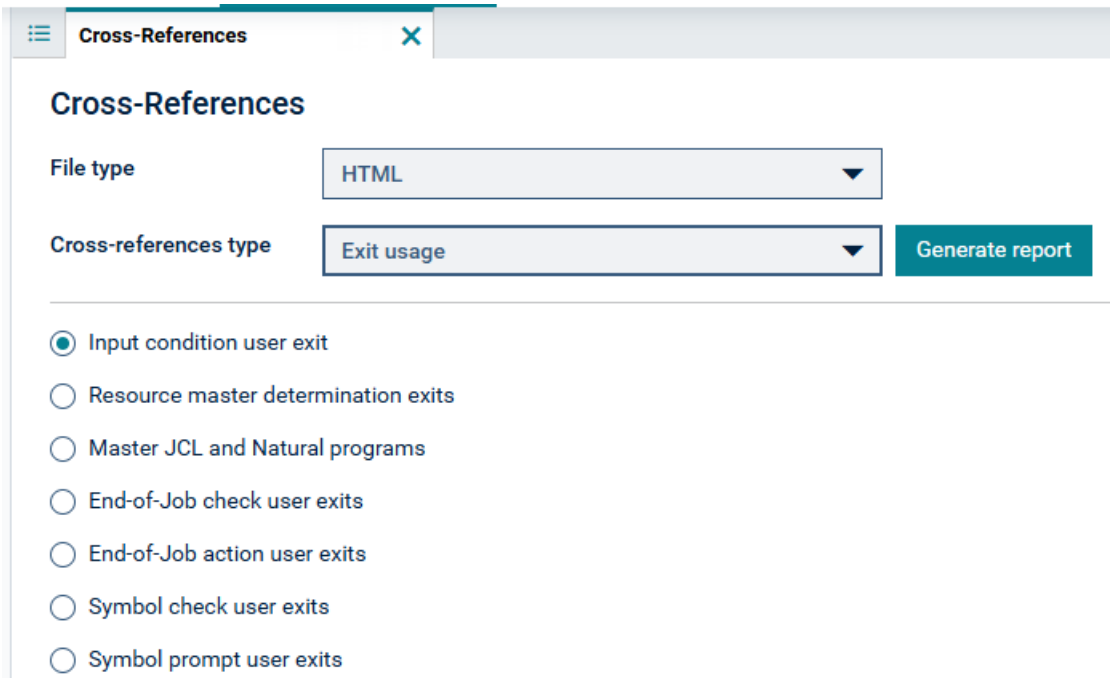
Possible formats are: `HTML` (default), `CSV` or `XML`

3 Select the **type of cross-reference** for which you want to generate a report.

Depending on the report type selected, proceed as described in the relevant step for

Exit Usage
Symbol Table Usage
Symbol and Symbol Value Search
JCL Usage
Node Usage
Resource Usage
List Undefined Objects

4 **Exit Usage**: If you select **Exit usage** from the **Cross-References window**, a selection list of user exit types appears.

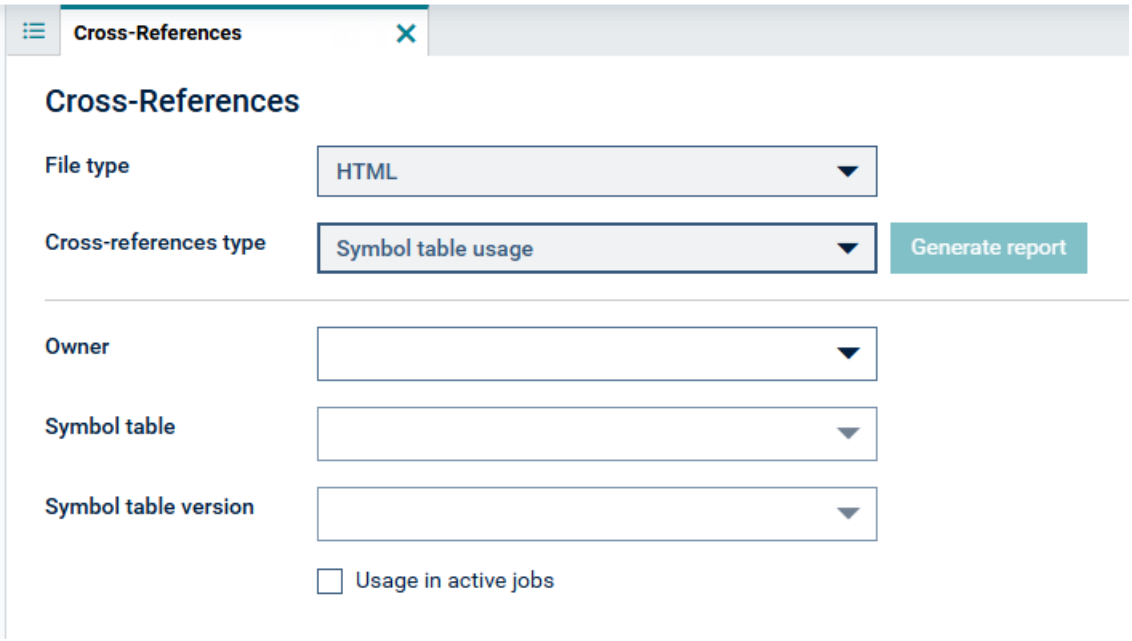


Select the required type and choose **OK**.

In the **Save as File** dialog that opens, specify the output file to be produced for the report as described in *Report Output Options* (section *Reporting*), and choose **Generate report**.

A report with a list of used user exits is generated into the output file as shown in *Example of User Exit Usage*.

5 **Symbol Table Usage**: If you select **Symbol table usage** from the **Cross-References window**, input fields appear as shown in the following example:

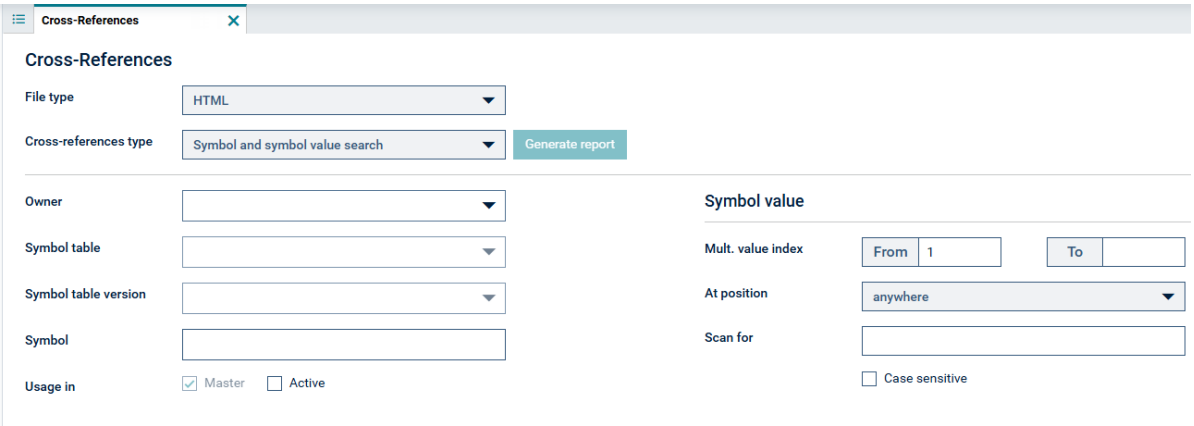


Specify the selection criteria required for the report to be produced. For valid input values, see *Fields and Columns: Cross-References*. When you are finished, choose **Generate report**.

In the **Save as File** dialog that opens, specify the output file to be produced for the report as described in *Report Output Options* (section *Reporting*), and choose **OK**.

A report with a list of used symbol tables is generated into the output file as shown in *Example of Symbol Table Usage*.

6 **Symbol and Symbol Value Search**: If you select **Symbol and symbol value search** from the **Cross-References window**, input fields appear as shown in the following example:

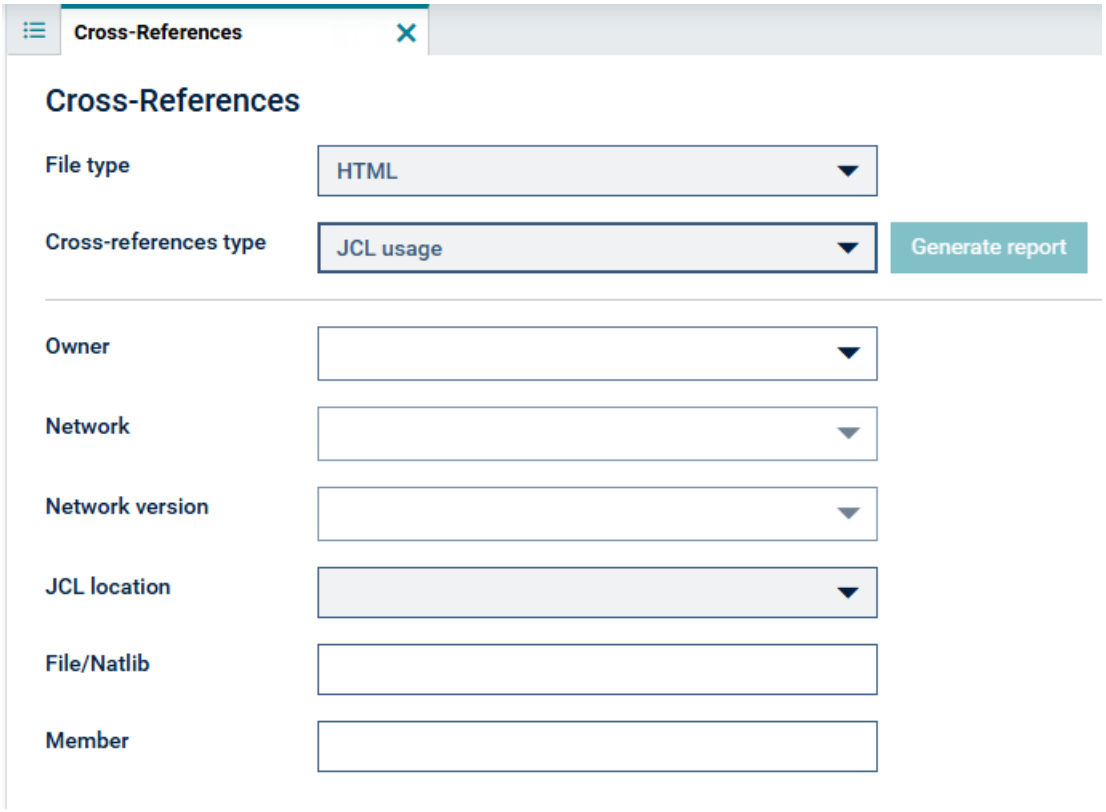


Specify the selection criteria required for the report to be produced. For valid input values, see *Fields and Columns: Cross-References*. When you are finished, choose **Generate report**.

In the **Save as File** dialog that opens, specify the output file to be produced for the report as described in *Report Output Options* (section *Reporting*), and choose **OK**.

A report with a list of used symbols is generated into the output file as shown in *Example of Symbol and Symbol Value Search*.

7 **JCL Usage**: If you select **JCL usage** from the **Cross-References window**, input fields appear as shown in the following example:

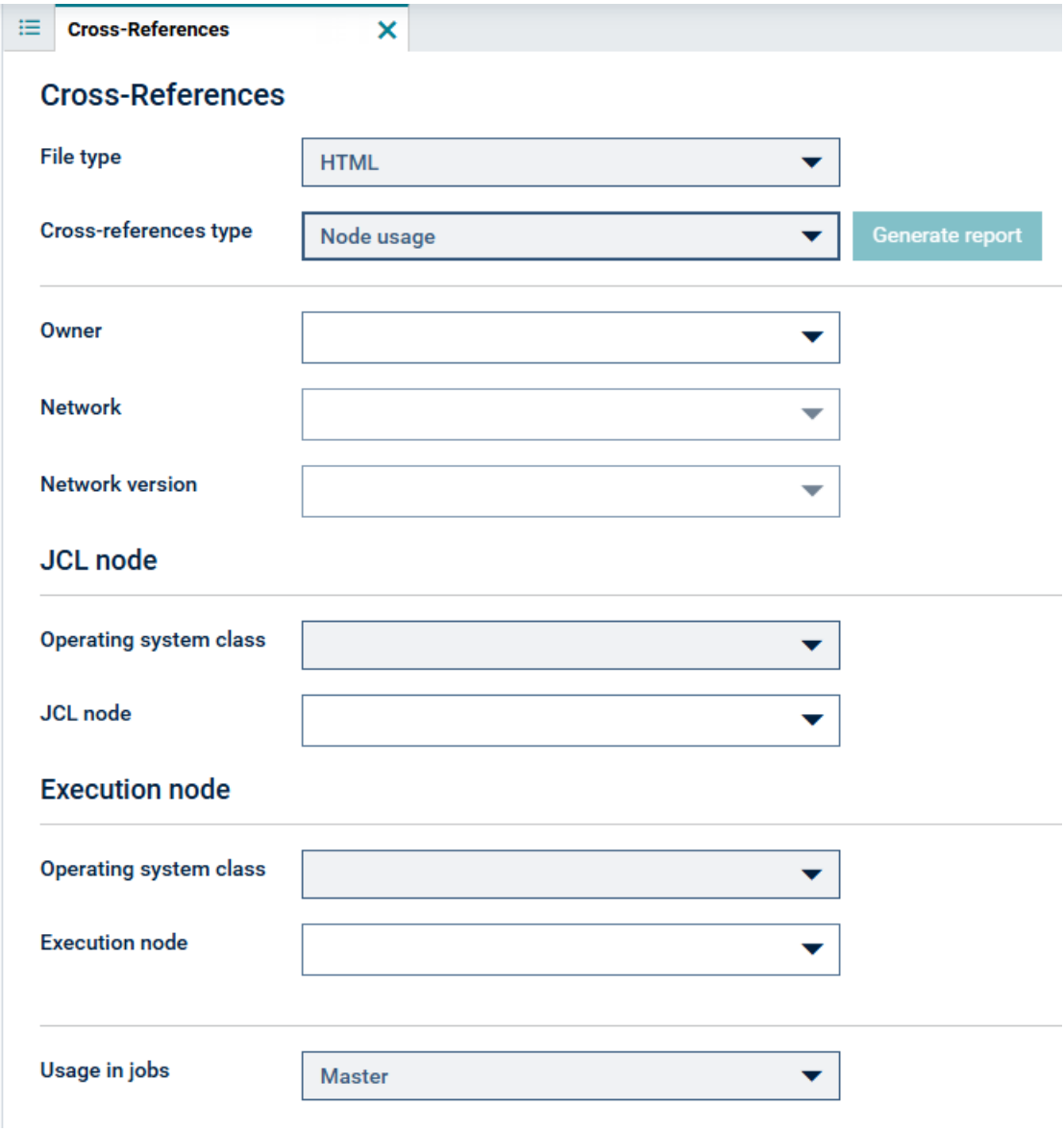


Specify the selection criteria required for the report to be produced. For valid input values, see *Fields and Columns: Cross-References*. When you are finished, choose **OK**.

In the **Save as File** dialog that opens, specify the output file to be produced for the report as described in *Report Output Options* (section *Reporting*), and choose **OK**.

A report with a list of used JCL is generated into the output file as shown in *Example of JCL Usage*.

8 **Node Usage**: If you select **Node usage** from the **Cross-References window**, input fields appear as shown in the following example:

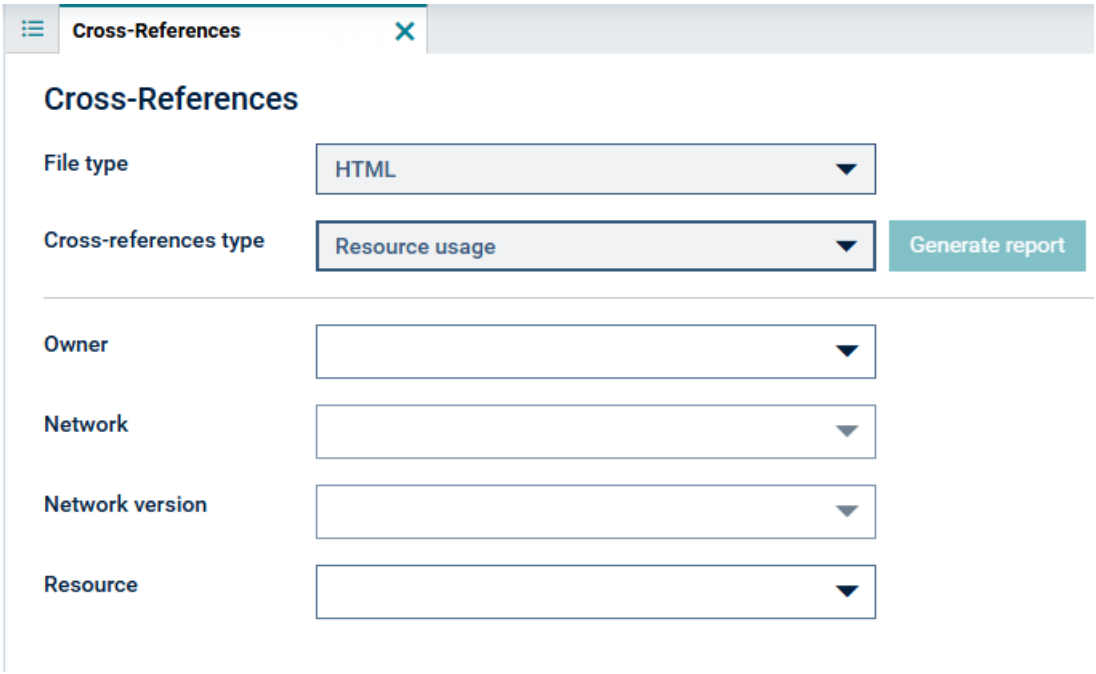


Specify the selection criteria required for the report to be produced. For valid input values, see *Fields and Columns: Cross-References*. When you are finished, choose **Generate report**.

In the **Save as File** dialog that opens, specify the output file to be produced for the report as described in *Report Output Options* (section *Reporting*), and choose **OK**.

A report with a list of used nodes is generated into the output file as shown in *Example of Node Usage*.

9 **Resource Usage**: If you select **Resource usage** from the **Cross-References window**, input fields appear as shown in the following example:

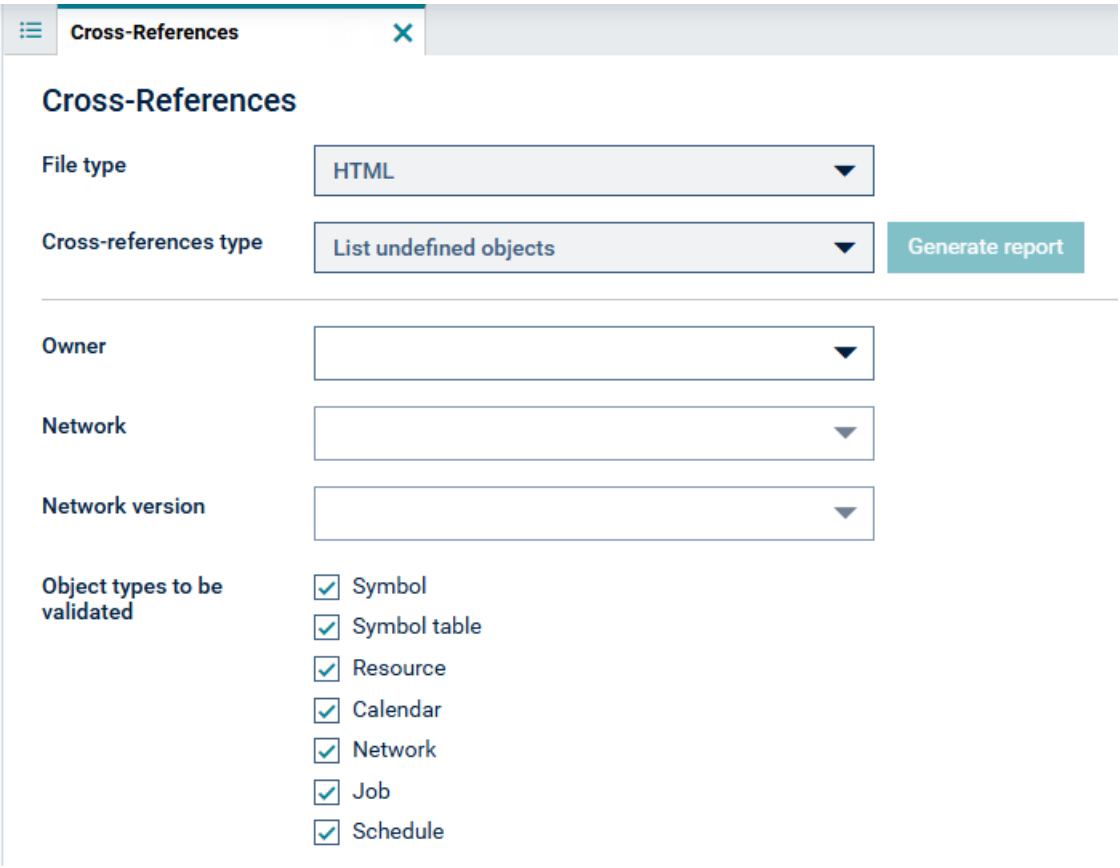


Specify the selection criteria required for the report to be produced. For valid input values, see *Fields and Columns: Cross-References*. When you are finished, choose **OK**.

In the **Save as File** dialog that opens, specify the output file to be produced for the report as described in *Report Output Options* (section *Reporting*), and choose **OK**.

A report with a list of used resources is generated into the output file as shown in *Example of Resource Usage*.

10 **List Undefined Objects**: If you select **List undefined objects** from the **Cross-References window**, input fields appear as shown in the following example:



Specify the selection criteria required for the report to be produced. For valid input values, see *Fields and Columns: Cross-References*.

Unmark the object type(s) you do not require. All object types for which you can search are selected by default.

When you are finished, choose **OK**.

In the **Save as File** dialog that opens, specify the output file to be produced for the report as described in *Report Output Options* (section *Reporting*), and choose **OK**.

A report with a list of used undefined objects is generated into the output file as shown in *Example of List Undefined Objects*.

Fields and Columns: Cross-References

The input fields for specifying selection criteria and output options for the cross-reference report to be produced are explained in the following table. The input fields correspond to the columns contained in the report output file.

Depending on the report type (to be) produced, the following fields and columns are available:

Field/Column	Explanation
Owner	Name of an owner or a range of names. Specify a valid name or range (see *Specifying Filter Criteria*) or open a selection window with a list of names. Enter an asterisk (*) to select all names.
Network	Name of a network or a range of names. Specify a valid name or range (see *Specifying Filter Criteria*) or open a selection window with a list of names. Enter an asterisk (*) to select all names.
Network version	Name of a network version or a range of names. Specify a valid name or range (see *Specifying Filter Criteria*) or open a selection window with a list of names. Leave the field blank to select a blank name. Enter an asterisk (*) to select all names.
Symbol table	Name of a symbol table or a range of names. Specify a valid name or range (see *Specifying Filter Criteria*) or open a selection window with a list of names. Enter an asterisk (*) to select all names.
Symbol table version	Version of the specified symbol table.
Resource	Name of a resource or a range of names. This can reflect real resources or can describe a fictitious resource. Specify a valid name or range (see *Specifying Filter Criteria*) or open a selection window with a list of names. Enter an asterisk (*) to select all names.
Exit Usage reports only:	
Job	Name of the job using the user exit.

Field/Column	Explanation	
Exit Library	Name of the library that contains the user exit.	
User Exit	Name of the user exit.	
Enabled	Indicates whether the user exit is enabled (`Yes`) or disabled (`No`) depending on the **type of user exit** selected.	
Background	This column is only shown for symbol prompt exits. It indicates whether symbol prompting runs in the background:	
	`Y`	Symbol prompting runs in the background.
	`N`	A window prompts the user for input.
	`B`	Performs both opens a prompt window and then runs prompting in the background.
	See also ***Specifying User Exits for Symbol Modification***.	
(User exit selection buttons)	Type of user exit(s) to be selected: Input condition exits Resource master determination exits Master JCL and Natural programs End-of-Job (EOJ) check exits End-of-Job (EOJ) action exits Symbol check exits Symbol prompt exits	
Symbol Table Usage reports only:		
Usage in active jobs	Jobs to be searched.	
	Possible check-box settings:	
	unchecked	Selects job masters only (default).
	checked	Selects both job masters and active jobs.
JCL Usage reports only:		
JCL location	A supported JCL location. All possible locations are described in ***List of JCL Locations*** in the section *Job Maintenance*. Leave the field blank to select all locations.	
File/Natlib	Name of a file or Natural library or a range of names. Specify a valid name or range (see ***Specifying Filter Criteria***). Your range specification can contain more than one leading or trailing asterisk (*).	

Field/Column	Explanation	
Member	Name of a member or a range of names. Specify a valid name or range (see *Specifying Filter Criteria*) or open a selection window with a list of names.	
Node Usage reports only:		
JCL node	Name of a JCL node. Select a name from the drop-down list box or leave the field blank to select all names.	
Execution node	Name of an execution node or a range of names. Select a name from the drop-down list box or leave the field blank to select all names.	
Operating system class	Operating system under which the JCL or execution node is used. Select an operating system from the drop-down list box or leave the field blank to select all names.	
Usage in jobs	Jobs to be selected.	
	Possible check-box settings:	
	master	Selects job masters only (default).
	active	Selects active jobs only.
Symbol Search by Value reports only:		
Symbol	Symbol for which to search. Enter an asterisk (*) to select all symbols.	
Usage in: Master/Active	Symbols in which to search. Select **Master** (default) and/or **Active** to search in symbol masters, active symbols, or in both.	
Mult. value index: From/To	Index range for symbols to which multiple values are assigned. **From**: Minimum index value in the range from `1` through `100` **To**: Maximum index value in the range from `1` through `100` or * for all multiple values available Default range: `1` to *	
At position	Position within the symbol value to be searched.	
	Possible check-box settings:	
	anywhere	Anywhere within the symbol value (default)
	nn	At the exact position: *nn* is any value in the range from `1` through `80`

Field/Column		Explanation	
		at beginning	At the beginning of the symbol value
		at end	At the end of the symbol value
Scan for		Scan value. Searches in a symbol value for a string of up to 80 characters	
Case sensitive		Mark this check box to perform a case-sensitive search.	
Undefined Objects reports only:			
This report inspects specified objects (symbol, symbol, table, resource, calendar, network, job and/or schedule) to determine which objects are referenced but not defined in your environment. **Note:** We recommend that you view this report in Entire Operations GUI Client which provides a graphical structure that helps interpret the output more easily. The columns in the GUI report are arranged in hierarchical order of object types from left (superior objects) to right. The corresponding items in the CUI report are arranged in a reverse order (subordinate object first).			
Object References	**Owner**	Owner of the network that references the undefined object.	
	Network	Network and network version (if defined) that references the undefined object. For possible **version entries**, see **Version Defined/Determined**.	
	Object: **Type** **Name** **Component**	Type and name of the object the references the undefined object. If relevant, the component of the referenced object is listed, for example, the input condition of a job.	
Undefined Object	**Owner**	Owner of the undefined object. If this column is empty or if it contains `(same)`, the owner name is identical to **Owner** in the **Object References** section.	
	Parent: **Type** **Name**	Type and name of the object that references the undefined object. If the **Name** column is empty or if it contains `(same)`, the object name is identical to **Name** in the **Object References** section.	
	Version Defined	Version defined for the referenced **Parent**. Possible version entries are: `(current)` Current object version. `(nv)` Network version. `(svn)` Symbol table version of the network. `(svj)` Symbol table version of the job.	

Field/Column		Explanation
	Version Determined	A report is generated for the current date by default. If you schedule the report to run on a different date, the object version valid on this date is listed in this column. See also **determination date** in the section *General Functions and Metanode General*.
	Object: **Type** **Name**	Type and name of the undefined object.
	Version Defined Version Determined	Version defined for the undefined object. If a **determination date** is set for the cross-reference report (see also *General Functions and Metanode General*), the version defined for the undefined object on the given specified is listed in **Version Determined**. `(void)` denotes that a version is defined for the object but was not be found on the specified determination date.
Resource Usage reports only:		
Type		Type of resource.
		Possible values:
		U — Not reusable, quantitative.
		R — Reusable, quantitative.
		N — Not quantitative (absolute).
Quantity - Initial		Initial quantity. Total amount of the resource defined to the system.
Quantity - Used		Used quantity. Amount of resource currently used by running job.
Used by Owner		Active job used by the owner.
Used by Network		Aactive job used by the network.
Run		Run job.
Job		Active job.
Begin		Date and time of the allocation.

Examples of Cross-Reference Reports

This section contains examples of all types of reports you can generate with the **Cross-References** function:

- Example of User Exit Usage
- Example of Symbol Table Usage
- Example of Symbol and Symbol Value Search
- Example of JCL Usage
- Example of Node Usage
- Example of Resource Usage
- Example of List Undefined Objects

Example of User Exit Usage

The following is an example of a cross-reference report for **Exit Usage**:

Input condition user exit
Generation Date: 2020-01-22 - 15:35:18

Owner	Network	Job	Exit Library	User Exit
EXAMPLE	E50-USRT	E50-J1-IC	SYSEORU	URD-UIC
INCIDENT	I5045008	I50450081	EOR-T531	I5045008
NATQA5	A	WILLI1	NOPNET	AAEVIT1

All user exits of the specified type (here: input condition) and the jobs that reference these exists are listed in the report.

The columns contained in the report are described in *Fields and Columns: Cross-References*.

Example of Symbol Table Usage

The following is an example of a cross-reference report for **Symbol Table Usage**:

Symbol table usage
Determination Date: 2020-01-23
Generation Date: 2020-01-23 - 08:52:15

Symbol Table Owner: EXAMPLE Symbol Table: EXA-TABLE

Owner	Network	Run	Job
EXAMPLE	EXA-NET1		EXA-JOB3
EXAMPLE	EXA-NET3		EXA-JOB3

Symbol Table Owner: EXAMPLE Symbol Table: EXAM-ST1

Owner	Network	Run	Job
EXAMPLE	B60-FLOW		JOB-JCL
EXAMPLE	E51-MAIL		

All networks and jobs that use the specified symbol table are listed in the report.

The columns contained in the report are described in *Fields and Columns: Cross-References*.

Example of Symbol and Symbol Value Search

The following is an example of a cross-reference report for **Symbol and Symbol Value Search**:

Symbol and symbol value search

Determination Date: 2020-01-23
Generation Date: 2020-01-23 - 09:01:20

Owner	Symbol Table	Symbol	MV	Format	Network	Run	Symbol Value
EXAMPLE	DEMO <v1>	FILE-1		A			EOR.DEMO.SRCE
EXAMPLE	DEMO <v2>	FILE-1		A			EOR.DEMO.SRCE
EXAMPLE	DEMO <v2>	X		A			99
EXAMPLE	E20-ST	CLASS		A			G
EXAMPLE	E20-ST	JOBLIB		A			NOP.EXAMPLE.LOAD
EXAMPLE	E20-ST	MSGCLASS		A			X
EXAMPLE	E20-ST	STEPLIB		A			NOP.EXAMPLE.LOAD

All symbols defined in the specified symbol table are listed in the report.

The columns contained in the report are described in *Fields and Columns: Cross-References*.

Example of JCL Usage

The following is an example of a cross-reference report for **JCL Usage**:

JCL usage

Generation Date: 2020-01-23 - 09:06:23

Owner: EXAMPLE

Network	Job	Job Type	Node	JCL Location	File/Member
B60-FLOW	JOB-JCL	JOB	148	MAC	SYSEORU E60-M02
B60-FLOW	JOB-01	NAT	148	NAT	SYSEORU ALL1338
B60-FLOW	JOB-012	JOB	31	MAC	SYSEORU B60-M01
B60-FLOW	JOB-013	JOB	31	MAC	SYSEORU B60-M01
B60-FLOW	JOB-014	JOB	31	MAC	SYSEORU B60-M01
B60-FLOW	JOB-019	JOB	31	MAC	SYSEORU B60-M01
B60-FLOW	JOB-02	JOB	31	MAC	SYSEORU B60-M02

All JCL files and referencing jobs are listed for the specified selection.

The columns contained in the report are described in *Fields and Columns: Cross-References*.

Example of Node Usage

The following is an example of a cross-reference report for **Node Usage**:

Node usage
Generation Date: 2020-01-23 - 17:34:25

Owner	Network	Run	Job	Job Type	JCL Node	Execution Node
EXAMPLE	B60-FLOW		JOB-JCL	JOB	148	148
EXAMPLE	B60-FLOW		JOB-01	NAT	148	31
EXAMPLE	B60-FLOW		JOB-012	JOB	31	31
EXAMPLE	B60-FLOW		JOB-013	JOB	31	31
EXAMPLE	B60-FLOW		JOB-014	JOB	31	31
EXAMPLE	B60-FLOW		JOB-015	DUM	31	31
EXAMPLE	B60-FLOW		JOB-019	JOB	31	31
EXAMPLE	B60-FLOW		JOB-02	JOB	31	31

All nodes and referencing jobs are listed for the specified selection.

The columns contained in the report are described in *Fields and Columns: Cross-References*.

Example of Resource Usage

The following is an example of a cross-reference report for **Resource Usage**:

Resource usage
Generation Date: 2020-01-23 - 17:43:52

		Quantity		Used by				
Resource	Type	Initial	Used	Owner	Network	Run	Job	Begin
HUGO	U	2.00	1.00	SAGTEST	B60-FLOW	11416	JOB-015	2019-12-04 - 15:58:19.9
HUGO	U	2.00	1.00	SAGTEST	B60-FLOW	11418	JOB-015	2019-12-04 - 15:58:20.2
TESTRESOURCE	R	50.00	5.00	SAGTEST	B60-FLOW	11416	JOB-015	2019-12-04 - 15:58:19.9
TESTRESOURCE	R	50.00	5.00	SAGTEST	B60-FLOW	11418	JOB-015	2019-12-04 - 15:58:20.2

All resources and referencing active jobs are listed for the specified selection.

The columns contained in the report are described in *Fields and Columns: Cross-References*.

Example of List Undefined Objects

The following is an example of a cross-reference report for **List Undefined Objects**:

List undefined objects
Determination Date: 2020-01-14
Generation Date: 2020-01-14 - 16:01:00

Object References					Undefined Object								
Owner	Network	Object Type	Name	Component	Owner	Parent Type	Name	Version Defined	Determined	Object Type	Name	Version Defined	Determined
EXAMPLE	B60-FLOW	Network Master			EXAMPLE					Symbol Table Master	SYMB-TEST		
EXAMPLE	B60-FLOW	Job Master	JOB-01	Input Condition	EXAMPLE					Network Master	SAGTEST		

All objects that are referenced but not defined in your environment are listed in the report. In the example above, the following objects of the owner `EXAMPLE` are missing:

- The symbol table `SYMB-TEST` is not defined but referenced by the network `B60-FLOW`.
- The network `SAGTEST` is not defined but referenced in the input condition `COND-SW` of the job `JOB-01` in the network `B60-FLOW`.

The columns contained in the report are described in ***Fields and Columns: Cross-References***.

Generating Cross-Reference Reports in Batch

You can use the batch command client of Entire Systems Management to generate a report in batch mode: see ***Commands for the Batch Command Client*** in the section *Using Entire Operations in Batch Mode*.

XV API Routines

This section describes the Application Programming Interface (API) routines provided by Entire Operations for batch and online processing.

Purpose and Use of Entire Operations APIs

Available Entire Operations APIs

58 Purpose and Use of Entire Operations APIs

- Features Provided by Entire Operations APIs 734
- Locating and Implementing an API 734
- API Usage Rules and Restrictions 735
- Testing Available API Routines 736
- API-specific Parameter Definitions 737

Features Provided by Entire Operations APIs

An API can be used for accessing Entire Operations data and/or performing Entire Operations functions from any Natural application.

An API provides the following features:

- Dynamic connection to the Entire Operations data file;
- Access to conditions;
- Access to symbols;
- Writing to the Entire Operations log.

An API can be used for a number of purposes within and outside Entire Operations. Among them are:

- Dynamic modification of symbol tables during the execution of a job network;
- Modification of conditions from Natural programs;
- Exchanging information between Entire Operations and any other online or batch application;
- Setting input conditions for job networks from online applications;
- Inquiring the status of job networks from applications;
- Setting Entire Operations symbols from external tables;
- Inquiring Entire Operations symbols for use in external applications.

Locating and Implementing an API

An Entire Operations API (Application Programming Interface) routine is supplied as a Natural subprogram in the Natural SYSEOR system library.

The naming convention for an API subprogram is usually as follows:

```
NOPUxxxy
```

where `NOPU` means Entire Operations User API Routine, *xxx* is the program name (sometimes containing a version number) and *y* is the program type (`N` - subprogram, `P` - program).

To make use of an API

- Call the API subprogram from a Natural program that executes a `CALLNAT` statement using the following syntax:

```
CALLNAT 'NOPUxxxN' parameters
```

where *parameters* represents the individual API parameters that can be passed to the API subprogram. See also ***API-specific Parameter Definitions***.

The CALLNAT statement is described in detail in the Natural *Statements* documentation.

API Usage Rules and Restrictions

The following general rules apply when using an API:

- We recommend that you do not rename API routines because they are also used internally by Entire Operations.
- The routines delivered with the previous Entire Operations version are still supported (except for special notice, and only if they are delivered again with the current SYSEOR library). Convert their calls to the newer version when you modify the calling programs. Change all relevant CALLNAT statements and adapt the API parameter list. The complete range of functions and functionality is only available with the latest API version.
- Entire Operations internal data structures are subject to change between versions. The usage of an outdated API may cause data inconsistencies, unexpected Monitor failures or abnormal termination of Entire Operations.
- With newer API versions, new fields might have been added. You may leave these fields blank or zero if their description says “reserved for future use”. These fields are intended to become meaningful with a future Entire Operations version.
- No END TRANSACTION statement will be executed inside of Entire Operations APIs.
- If you invoke APIs from within an Entire Operations user exit, you must not code any END TRANSACTION statement within the exit. Refer to the user exit coding rules.
- If you invoke APIs from anywhere else than an Entire Operations user exit, you must code an END TRANSACTION statement in an appropriate location after the CALLNAT statement to close Entire Operations transaction logic.
- You can change the parameter names, but not the parameter format, number and order.

API Access Restrictions

If Natural Security is installed, you can restrict Natural RPC access to APIs. For more information, see the **Service Protection** option of an RPC server profile described in the section *Allowing/Disallowing Services* in the *Natural Security* documentation.

Testing Available API Routines

You can execute the Natural NOPUT1-P program to find out how API routines react to real data. It shows how to code the calls of the supplied routines, and how to check the return codes.

The source and the cataloged object of NOPUT1-P are supplied in the SYSEOR system library.

When you execute the program NOPUT1-P, you can select and test the required API routine from a screen like the example below:

```
    Entire Operations - API Routines

Please Select:
  Network / Job Status       ST                  NOPUST3N
  Conditions                 CO                  NOPUCN3N
  Resource Master            RM                  NOPURS1N
  Resource Usage             RU                  NOPURE2N
  Symbols                    SY                  NOPUSY7N
  Calendars/Schedules        CS                  NOPUCS1N
  Activation                 AC                  NOPUAC5N
  Milestone                  MI                  NOPUMI1N
  Subnetwork                 SN                  NOPUSN2N
  Job Import                 JI                  NOPUJI4N
  Version Info               VE                  NOPUVI2N
  Symbol Prompt Texts        SP                  NOPUSP3N
  Job Schedule               SC                  NOPUJS2N
  Log Monitor Activity       LM                  NOPMLA1N
  Active Job Statistics      AS                  NOPUAS1N
  Message Text               MT                  NOPUMT3N
                        ===> __  (blank or . or PF3  ==>  End)
```

API-specific Parameter Definitions

The parameters that can be passed to an API subprogram with the CALLNAT statement are described for each API in the section *Available Entire Operations APIs*.

The data types listed in the individual parameter description of an API represent the Natural data format and length, for example, A1 for an alphanumeric variable with a one-byte length. For details, see *User-Defined Variables* in the Natural *Programming Guide*.

By default, parameters are passed to an API subprogram *by reference*, that is, the data is transferred via address parameters, the parameter values themselves are not moved. The parameter descriptions indicate parameters that are defined with BY VALUE, BY VALUE RESULT and/or OPTIONAL.

OPTIONAL

For a parameter defined with OPTIONAL, a value can, but need not be supplied.

Since the parameter sequence is fixed, usually a value must be passed from the calling program to each parameter. However, a parameter defined with OPTIONAL, can be skipped by substituting it with 1X in the CALLNAT statement as indicated in *Example of Using NOPUST3N*.

For further information about OPTIONAL parameters, refer to the description of the DEFINE DATA statement, section *Defining Parameter Data*, *Syntax Description*, OPTIONAL, in the Natural *Statements* documentation.

BY VALUE

A parameter defined with BY VALUE is passed to the API subprogram by value; that is, the actual parameter value (instead of its address) is passed. Consequently, the field in the subprogram need not have the same format/length as the parameter passed in the CALLNAT statement.

BY VALUE RESULT

A parameter defined with BY VALUE RESULT is passed by value in both directions; that is, the actual parameter value is passed from the calling program to the API subprogram and, on return to the calling program, the actual parameter value is passed from the subprogram back to the calling program.

For further information, refer to *Defining Parameter Data* in the *DEFINE DATA* section of the Natural *Statements* documentation.

59 Available Entire Operations API Routines

- EORUCB1N: Check Use of BS2000 User IDs 740
- NOPFB2-N: Generate SYSOUT File Names for BS2000 740
- NOPMLA1N: Start and Stop the Monitor Activity Log 743
- NOPU--1N: Read Network Available to a Specific User 744
- NOPU--2N: Return all Usable Symbol Tables for a Network 745
- NOPU--3N: Get Correlation ID for an Activated Network 746
- NOPU--4N: Store New Event in Entire Operations System File 746
- NOPUAC5N: Activate Job Networks or Jobs 747
- NOPUAS1N: Retrieve Numbers of Active Jobs in Defined Status Ranges 752
- NOPUCN3N: Access Entire Operations Conditions 753
- NOPUCS1N: Access Calendars and Schedules 754
- NOPUJI4N: Import Operating System Jobs into the Active Queue 758
- NOPUJS2N: Job Schedule Inquiry and Modification 760
- NOPULW9N: Write Messages to System Automation Tools Log 761
- NOPUMI1N: Set/Reset Text Milestones in Master and Active Jobs 763
- NOPUMT3N: Expand Message Texts 764
- NOPUNI1N: Invalidate Entire System Server Node Table Entries 766
- NOPUNX1N: Entire System Server Calls to Access UNIX and Windows Files 766
- NOPURE2N: Handle Resource Allocations 780
- NOPURS1N: Access Entire Operations Resource Masters 784
- NOPUSN2N: Inquire Calling Job or Called Network for Subnetworks 785
- NOPUSP3N: Display Long Texts for Symbol Prompting 787
- NOPUST3N: Inquire Network and Job Status, Symbol Table 788
- NOPUSY7N: Access Entire Operations Symbols 792
- NOPUVI2N: Obtain Entire Operations Version Information 797
- NOPUXD1N: Maintain End-of-Job User Exits for a Network 798
- NOPUXI0N: Add Input Condition to an Activated Job 800

EORUCB1N: Check Use of BS2000 User IDs

You can use the exit EORUCB1N contained in the library SYSEORU for a user-defined check if Natural users are allowed to use BS2000 user IDs.

Meaning of the input and output fields:

Parameter Description

Parameter	Format/Length	Use	
P-RC	N4	out	Return code (not evaluated, should be 0).
P-RT	A70	out	Return text (not evaluated).
P-USER	A8	in	*USER in Natural.
P-BS2000-USERID	A8	in	BS2000 user ID.
P-ACCOUNT	A8	in	BS2000 account number.
P-SUBMIT-PSWD	A8	in	BS2000 submit password.

In the following you find the example of the exit EORUCB1N delivered with Entire Operations which you can adapt according to your needs. The exit always returns P-RC=0 (authorized).

Example

```
* NOPFB2-N

END
```

NOPFB2-N: Generate SYSOUT File Names for BS2000

The exit NOPFB2-N may be copied into the library SYSEOR to enable the user-defined generation of SYSOUT file names in BS2000. Only one instance of this exit exists in the system and it is executed only if it is present. Otherwise, standard name generation is used.

This exit is called with the parameter list **NOPXPL-A**, so the first line of the exit must be:

```
DEFINE DATA PARAMETER USING NOPXPL-A
```

This section covers the following topics:

- Parameter Description
- Rules for Exit Coding
- Use with Work File Deletion
- Example

Parameter Description

Meaning of the parameters in **NOPXPL-A**:

Parameter	Format/Length	Use		
P-CALL-PLACE	A3	in	Constant FSB (file name SYSOUT BS2000).	
P-RC	N4	out	Return code:	
			0	Function OK (name generation OK).
			1	A name could not be generated. The default name routine of Entire Operations is to be called.
			2	A name could not be generated. The action (activation) is to be cancelled.
			If the activation is cancelled by means of the name routine, a corresponding log entry will be written.	
P-RT	A66	out	Return code (not evaluated).	
P-OWNER	A10	in	Network owner.	
P-NETWORK	A10	in	Job network.	
P-JOB	A10	in	Job.	
P-RUN	P13	in	Run number.	
P-ACTIVATION-TIME	T	in	Activation time of the network.	
P-EXECUTION-NODE	N3	in	Entire System Server execution node.	
P-EXECUTION-OPSYS	A8	in	Operating system of Entire System Server node.	
P-SYMBOL-TABLE	A10	in	Name of the defined symbol table.	
P-FSB-OBJECT-TYPE	A5	in	File type:	
			SO-C	Current SYSOUT file.
			SO-P	Previous SYSOUT file.
			SO-P1	Previous SYSOUT file without user ID.

Parameter	Format/Length	Use		
			SO-A	All SYSOUT files of a job.
P-FSB-USERID	A8	in	BS2000 user ID.	
P-FSB-CATID	A4	in	BS2000 Cat ID.	
P-FSB-SUFFIX	A2	in	BS2000 suffix.	
P-SYSOUT-FILE	A54	out	Generated file name.	

Rules for Exit Coding

- File names must be created which are unique system-wide. Otherwise, the proper running of Entire Operations is no longer possible.
- File names which are unique system-wide can be produced by using the fields P-OWNER, P-NETWORK, P-RUN, P-JOB and P-FSB-SUFFIX simultaneously.
- You must be able to process all object types defined above.
- Except for the object type SO-P1, a fully-qualified file name must be created which includes the BS2000 user ID.
- A wildcard file entry valid for all SYSOUT files of an active job must be created for the type SO-A. This type is used for deleting files.
- The field P-FSB-SUFFIX is transferred and must be used to differentiate SYSOUT files when jobs are repeated. The caller ensures that the suffix for the currently active job is unique. For the object type SO-C, this field is blank.

Note: If no file name is returned, then standard name generation is used.

Use with Work File Deletion

Even in case of work file deletion (in connection with deactivation or cleanup) this exit is called, if it exists at all. The following will apply in this context:

- At any rate, an attempt is made to also delete work files, which correspond to the default name conventions of Entire Operations, independent of the existence of the exit.
- It is presumed that the exit will generate the run number with exactly 5 digits, which are enclosed by dots:
  - aaa.00000.bbb (example: aaa.01234.bbb)

  For the Entire Operations file search, this pattern is replaced by a wildcard:
  - aaa.00000.bbb --> aaa.*.bbb
- If the name syntax of the exit is modified, files cannot be found any longer, the names of which have been generated before the modification. It is not recommended to exchange the exit, while active jobs are running in the operating system.

Example

```
* NOPFB2-N
* EXAMPLE EXIT FOR SYSOUT FILE NAME
*
* MODIFICATIONS:
* 29.10.08 (139240) SYSOUT FILE NAME EXIT                       SN210
* ------------------------------------------------------------------
DEFINE DATA PARAMETER USING NOPXPL-A
END-DEFINE
* ------------------------------------------------------------------
COMPRESS P-OWNER P-NETWORK P-RUN P-JOB
  INTO P-SYSOUT-FILE LEAVING NO SPACE
IF P-FSB-OBJECT-TYPE NE 'SO-P1'
* -- ALL OBJECT TYPES EXCEPT SO-P1  M U S T  HAVE A BS2000 USERID --
  COMPRESS '$' P-FSB-USERID '.' P-SYSOUT-FILE
    INTO P-SYSOUT-FILE LEAVING NO SPACE
END-IF
IF P-FSB-OBJECT-TYPE EQ 'SO-A'
* -- ALL SYSOUT FILES FOR A JOB --
  COMPRESS P-SYSOUT-FILE '*'
    INTO P-SYSOUT-FILE LEAVING NO SPACE
ELSE
  COMPRESS P-SYSOUT-FILE '.S' P-FSB-SUFFIX
    INTO P-SYSOUT-FILE LEAVING NO SPACE
END-IF
RESET P-RC        /* OK
END
```

NOPMLA1N: Start and Stop the Monitor Activity Log

You can start and stop the monitor activity log using the following calls from your Natural application.

```
CALLNAT 'NOPMLA1N'
P-FUNCTION
```

Parameter Description

Parameter	Format/Length	Use		
P-FUNCTION	A1	in	Function code:	
			Y	Starting the monitor activity log.
			N	Stopping the monitor activity log.

The call of this API has the same effect as setting the field **Log Monitor Activity** in the **Monitor Defaults** (see the *Administration* documentation).

This enables you to automatically start and stop the monitor activity log at particular times or on account of specific conditions.

NOPU--1N: Read Network Available to a Specific User

You can get a list of networks owned or granted to a user using the following call from your Natural application.

```
CALLNAT 'NOPU--1N'
  P-USER
  P-SELECTION
  P-RC
  P-RESULT-GRANTED-NETWORKS (*)
```

The parameters have the following meaning:

Parameter	Format/Length	Use		
P-USER	A8	in	User name.	
P-SELECTION	A10	in	Selection criteria.	
			Select by P-SELECTION (combination of values possible):	
			O	Networks of owner.
			G	Owner granted networks.
			A	Active networks only. **Note:** Only usable together with O, G or U.
			U	User granted networks.
P-RC	N3		1	User not defined.
			99	Natural runtime error.
P-RESULT-GRANTED-NETWORKS				

Parameter	Format/Length	Use	
Note: This is a dynamic structure consisting of:			
P-OWNER	A10	out	Owner of network.
P-NETWORK	A10	out	Network name.
P-NETWORK-VERSION	A10	out	Network version.
P-DESCRIPTION	A70	out	Short description of network.

NOPU--2N: Return all Usable Symbol Tables for a Network

You can get a list of usable symbol tables for a given Network using the following call from your Natural application.

```
CALLNAT 'NOPU--2N'
  P-OWNER
  P-NETWORK
  P-NETWORK-VERSION
  P-RC
  P-RESULT-SYMBOL-TABLES  (*)
```

The parameters have the following meaning:

Parameter	Format/Length	Use	
P-OWNER	A10	in	Owner name.
P-NETWORK	A10	in	Network name.
P-NETWORK-VERSION	A10	in	Network version.
P-RC	N3	1	Error when determining tables.
		99	Natural runtime error.
P-RESULT-SYMBOL-TABLES **Note:** This is a dynamic structure consisting of:			
P-SYMBOL-TABLE_TYPE	A2	out	Symbol table type.
P-SYMBOL-TABLE_OWNER	A10	out	Symbol table owner.
P-SYMBOL-TABLE	A10	out	Symbol table name.
P-SYMBOL-TABLE-VERSION	A10	out	Symbol table version.

NOPU--3N: Get Correlation ID for an Activated Network

You can retrieve the correlation ID for a specific Network using the following call from your Natural application.

```
CALLNAT 'NOPU--3N'
  P-OWNER
  P-NETWORK
  P-RUN-NUMBER
  P-RC
  P-CORRELATION-SUBSYS
  P-CORRELATION-ID
  P-CORRELATION-TYPE
```

The parameters have the following meaning:

Parameter	Format/Length	Use	
P-OWNER	A10	in	Owner name.
P-NETWORK	A10	in	Network name.
P-RUN-NUMBER	I4	in	Run number.
P-RC	N3	1	Activated network does not exist.
		99	Natural runtime error.
P-CORRELATION-SUBSYS	A32	out	Correlation subsystem.
P-CORRELATION-ID	A36	out	Correlation ID.
P-CORRELATION-TYPE	A32	out	Correlation type.

NOPU--4N: Store New Event in Entire Operations System File

You can store an event record for a given network run using the following call from your Natural application.

```
CALLNAT 'NOPU--4N'
          P-EVENT-TYPE
          P-EVENT-VERSION
          P-EVENT-DATA (*)
          P-OWNER
          P-NETWORK
          P-NETWORK-RUN
          P-RC
          P-ERROR-NR
```

The parameters have the following meaning:

Parameter	Format/Length	Use		
P-EVENT-TYPE	A32	in		
P-EVENT-VERSION	A5	in		
P-EVENT-DATA	A80/1:19			
P-OWNER	A10	in	Owner of network.	
P-NETWORK	A10	in	Network.	
P-NETWORK-RUN	I4	in	Run number.	
P-RC	N4	out	1	Network run does not exist.
			99	Natural runtime error.
P-ERROR-NR	N7	out	Only filled with the contents of the Natural system variable *ERROR-NR in case of a runtime error.	

NOPUAC5N: Activate Job Networks or Jobs

You can activate job networks or jobs from your Natural application. The network of job definitions must exist in Entire Operations. Use:

```
CALLNAT 'NOPUAC5N'
    FUNCTION RC OWNER NETWORK NETWORK-VERSION JOB SYMTAB  SYMTAB-VERSION
START-TIME RUN [8X] [LATEST-START] [DEADLINE]
```

This section covers the following topics:

- Parameter Description
- Activation with Symbol Modification
- Repetition of an Active Network

Parameter Description

Parameter	Format/Length	Use		
FUNCTION	A1	in	Function code:	
			A	**Activate**, no hold for symbols.
			1	Like A, but time frame as defined in schedule is used.
			C	Change start time.
			D	RUN > 0: Deactivate network. RUN = -1: Delete network master.

Parameter	Format/Length	Use		
			H	**Activate**, but hold task until released with function R. See ***Activation with Symbol Modification***.
			I	Set run number initial value. The next network run will use the next run number available after the number entered. The run number must be between 0 and the highest run number permitted.
			J	Modify the earliest start time, latest start time and deadline time of an active job. The three timestamps are shifted with the same offset.
			K	Modify the earliest start time of an active job only. The latest start time and deadline time will not be modified.
			R	Release activation that is in hold. ***Activation with Symbol Modification***.
			2	Like R, but time frame as defined in schedule is used.
			S	Repetition of an active job.
			T	**Repetition of an active network.**
			X	Network or job existence test only.
RC	N4 BY VALUE RESULT	out	**Return code:**	
			0	OK.
			1	Owner, network, job not found.
			2	Activation entry not found.
			3	Network not active.
			4	Job cannot be restarted. For the function S (repetition), this return code is issued if there is a SYSOUT copy in progress.
			5	Run number has already been used.
			6	Run number not in valid range.
			7	No free run number.
			8	Active objects existing.
			9	Current version cannot be determined.
			10	Version default usage found. This return may be issued at a deletion attempt.

Parameter	Format/Length	Use		
			11	Invalid time frame. One or both of the following has occurred: P-LATEST-START is specified and is earlier than P-START-TIME. P-DEADLINE is specified and is earlier than P-START-TIME or P-LATEST-START.
			12	Invalid activation origin. This return code may be issued for internal calls of the API only.
			101	Invalid function code.
			102	Parameters missing.
			111	Symbol table name contained reserved prefix =EOR=.
			120	Owner must not be used. Applies to the reserved owner WASTEBIN, for example.
			121	Owner does not exist.
			122	Network version is required.
			901	OK; subnetwork symbol prompting.
DBENV	A10 BY VALUE OPTIONAL	in	Database environment (reserved for future use).	
OWNER	A10 BY VALUE	in	Owner of the job network. If a job is placed in hold status by means of the function H, this will be written to the log.	
NETWORK	A10 BY VALUE	in	The job network.	
NETWORK-VERSION	A10 BY VALUE OPTIONAL	in	Version name of the job network or (current) (see ***Reserved Version Names for Networks***).	
JOB	A10	in	Job. If empty, whole network is activated, deactivated or tested.	
SYMTAB	A10 BY VALUE OPTIONAL	in	The symbol table.	

Parameter	Format/Length	Use	
SYMTAB-VERSION	A10 BY VALUE OPTIONAL	in	Symbol table version.
START-TIME	T	in	If zero (0), the network is activated immediately. Otherwise, the network is activated at the designated time. The field must contain both date and time.
		out	Start time actually set.
RUN	I4 BY VALUE RESULT	in	Run number used by Entire Operations.
			Functions A and H: ■ When activating a single job, an existing run number can be entered. This should only be used for post activations of an active network. Functions C, D, R and S: ■ Must be an existing run number of the job network. Function X: ■ If 0, the network master is tested. Otherwise, an active network with this run number is tested.
		out	Functions A and H: ■ If 0, the next free run number is assigned. ■ If not 0, this number is used; if this number is still active, the next free number in ascending order is used.
8X	OPTIONAL		Placeholder for 8 parameters that are intended for internal use. Required if you want to pass one or more of the following parameters.
LATEST-START	T OPTIONAL	in	If specified, the field must contain both date and time. For the activated run, the latest start passed here will override any other definitions on job or network level.
DEADLINE	T OPTIONAL	in	If specified, the field must contain both date and time. For the activated run, the deadline passed here will override any other definitions on job or network level.

Note: You can use this function from outside Entire Operations, as well as from user exits within Entire Operations. This function is comparable to the manual activation of networks or jobs in the Entire Operations online system.

Activation with Symbol Modification

To activate a job or network and pass specific symbol values to this activation, proceed as follows

1 Activate the network and keep it in hold, with function `H`.

2 Use the returned run number to set symbols in the active symbol table(s) for this run. The API routine must be called **NOPUSY7N**.

3 After setting the symbol(s), release this activation with the `R` function.

Notes:

1. The parameters `START-TIME`, `EARLIEST-START` (optional), `DEADLINE` (optional) will be taken from the `R` (release) call.
2. Active symbols, which are set here, will not be overwritten by a subsequent complete symbol table activation.

Repetition of an Active Network

The field `P-JOB` may remain empty.

- If `P-JOB` is empty, all executed jobs of the active network will be repeated.
- If `P-JOB` contains a valid job name, the active network will be repeated, but the given job will be repeated as "dummy due to repetition".

The repetition also works correctly for jobs of type `NET` (subnetwork): subnetworks are also repeated, with their original run number.

The conditions `NET-BEGIN`, `NET-END` and `NET-END-NOTOK` will be reset before the repetition.

This API function enables you to repeat whole active networks, including subnetworks. Active conditions which are already set or reset will be backed out.

Caution: You should exercise caution if the API NOPUAC5N function `T` is invoked for a network, in which jobs are currently executing. These will not be repeated. We recommend that you use this API function only for active networks, in which no job is currently executing.

NOPUAS1N: Retrieve Numbers of Active Jobs in Defined Status Ranges

This section covers the following topics:

- Function Call

Function Call

You can retrieve the numbers of active jobs in defined status ranges by using the following call:

```
CALLNAT 'NOPUAS1N'
    NOPUAS-VERSION NOPUAS-PARMS
```

In the calling program, define the parameter list for this API as follows:

```
DEFINE DATA
...
LOCAL USING NOPUAS1L
```

Meaning of the parameters:

Parameter	Format/Length	Use		
NOPUAS-VERSION	N2	in	Interface version (currently this is 01 constantly).	
NOPUAS-PARMS	A250	mod	Parameter data area.	
Redefinition of NOPUAS-PARMS:				
P-FUNCTION	A1	in	Function code:	
			S	Get statistics.
P-RC	N3	out	Return code:	
			0	OK.
			101	Invalid function code.
			102	Parameter(s) missing.
P-NUMBER-WAITING	I4	out	Number of jobs waiting.	
P-NUMBER-HOLD	I4	out	Number of jobs in hold.	
P-NUMBER-INPUT-QUEUE	I4	out	Number of jobs in the input queue.	
P-NUMBER-EXECUTING	I4	out	Number of jobs executing.	
P-NUMBER-PERM-ERROR	I4	out	Number of jobs in permanent error status.	

NOPUCN3N: Access Entire Operations Conditions

You can handle conditions using the following statement in your Natural application:

```
CALLNAT 'NOPUCN3N'
    P-FUNCTION P-RC P-OWNER P-NETWORK P-CONDITION P-RUN-FROM P-RUN-TO
    P-TIME-FROM P-TIME-TO P-CONFIRM P-JCL-CHECK
```

This section covers the following topics:

- Parameter Description
- Description of Function Codes

Parameter Description

Parameter	Format/Length	Use		
P-FUNCTION	A1	in	Function code:	
			R	Reset a condition.
			S	Set a condition.
			T	Test a condition.
			See also *Description of Function Codes*.	
P-RC	N3	out	Return code:	
			0	Function OK.
			1	Condition not found.
			2	Time or run required for reset.
			3	Invalid run number range.
			101	Invalid function code.
			102	Condition name missing.
			104	Network name missing.
			121	Owner does not exist.
P-OWNER	A10	in	Owner of the job network/condition.	
P-NETWORK	A10	in	Job network.	
P-CONDITION	A20	in	Condition name.	
P-RUN-FROM	I4		Corresponding run number.	
		in	Absolute condition: -1.	
		out	Function T and field empty: Last run number found is returned.	
P-RUN-TO	I4	in	Run number to (end of interval). If zero, only P-RUN-FROM will be handled.	
P-TIME-FROM	T	in	Starting time of condition.	

Parameter	Format/Length	Use	
		out	Function T and field empty: Time belonging to run number is returned.
P-TIME-TO	T	in	End time of condition.
P-CONFIRM	A1	in	Confirm deletions: Y (yes), N (no).
P-JCL-CHECK	A1	in	If this field contains C, the condition is treated like one which is used during a JCL check.

Description of Function Codes

Meaning of the function codes:

Code	Description
R	Reset a condition. The date or the run number must be specified. If a run number is specified, the date is not considered. If a date is specified, the deletion of several conditions with the same date, but different run numbers might occur.
S	Set a condition. The date and/or run number are optional.
T	Test a condition. If a run number is specified, the time frame is not considered. If no time frame and no run number are specified, each condition with the specified name matches. If no run number is specified, the last run number for the time frame up to the current time is returned. If no time frame was given, the time belonging to this run number is returned in P-TIME-FROM.

NOPUCS1N: Access Calendars and Schedules

This section covers the following topics:

- Accessing Dates within Calendars or Schedules

- Example for Using NOPUCS1N

Accessing Dates within Calendars or Schedules

You can access dates within calendars or schedules using the following statement in your Natural application:

```
CALLNAT 'NOPUCS1N'
    P-FUNCTION P-RC P-OBJECT-TYPE P-OWNER P-OBJECT P-DATE-A8
```

Meaning of the parameters:

Parameter	Format/Length	Use		
P-FUNCTION	A1	in	Function code:	
			A	Delete a calendar or schedule.
			B	Add calendar.
			C	Determine the calendar used.
			D	Determine the schedule used.
			N	Determine the next set date, from a start date.
			P	Determine the previous set date, from a start date.
			R	Reset a date. Date will be removed from calendar or schedule.
			S	Set a date. Date will be added to calendar or schedule.
			T	Test a date. If date exists within calendar or schedule: RC contains 0, otherwise 1.
			1	Set all dates in the calendar. Not supported for schedule.
			2	For calendar: Reset all dates in the calendar. For schedule: Delete all schedule definitions. The schedule will remain as empty body.
P-RC	N3	out	Return code:	
			0	Function OK, or date is set. **Note:** For calendars, this means workday.
			1	Date is not set. **Note:** For calendars, this means holiday.

Parameter	Format/Length	Use		
			2	Object not found.
			3	Object used in definitions.
			4	Object already existing.
			101	Invalid function code.
			102	Parameter(s) missing.
			103	Invalid parameter combination.
			104	Invalid object type.
			105	Invalid date.
			106	Table of explicit dates full.
			121	Owner does not exist.
P-OBJECT-TYPE	A1	in	Object type:	
			N	Network.
			C	Calendar.
			H	Schedule history.
			S	Schedule.
P-OWNER	A10	in	Owner of object.	
		out	Functions C and D: owner of determined object.	
P-OBJECT	A10	in	Name of object: owner name resp. schedule name. If P-FUNCTION is N, P, or T, and if a calendar is expected as object: ■ You may use the reserved calendar name =EOR=WORK7. This pseudo-calendar is valid for any year and has all days defined as workdays. ■ The value of P-OWNER is meaningless in this case.	
		out	Function codes C and D: name of determined object.	
P-DATE-A8	A8	in	Date in format: YYYYMMDD. ALL in this field means all years. This value applies for deletions.	
		out	Function codes N and P: the date found.	

Notes:

1. Only function T (test a date) is allowed for object type H (schedule history).
2. Schedule history is usually available for the current year and previous two years.
3. Schedule dates can be set for the current year and the next year.
4. In calendars, a set date means workday, and a non-set date means holiday. The functions N and P work for a range of years. Limitations:
   For schedules, only the current and the previous year are available.

5. Calendars must be defined for all years in which a search is to be performed.
6. Function A deletes calendar/schedule master when P-OBJECT-TYPE='C' or P-OBJECT-TYPE='S', respectively. In case calendar/schedule is for the current year, and if is referenced in schedule/network the P-RC value 3 (Object used in definitions) is returned. Calendars for past years will be deleted without reference check.
7. When a calendar is deleted (P-FUNCTION='A' and P-OBJECT-TYPE='C'), the value of P-DATE-A8 shall be either in form YYYY or YYYYMMDD or constant ALL. Value ALL means: the calendar will be deleted for all defined years.

Example for Using NOPUCS1N

The program calculates the first Monday of each month in 2008 and subsequently enters the network REQUEST/NET001 for the calculated days in the schedule.

```
* DETECT FIRST MONDAY IN MONTHS
*
DEFINE DATA
LOCAL
1 #YYYYMMDD (A8)
1 REDEFINE #YYYYMMDD
 2 #YYYY    (N4)
 2 #MM      (N2)
 2 #DD      (N2)
1 #D1      (D)
1 #WD      (A1)
1 #OWNER   (A10) INIT <'REQUEST'> /* Owner
1 #OBJECT  (A10) INIT <'NET001'>  /* Network
1 #RC3     (N3)
END-DEFINE
*
#YYYY := 2010                     /* Year
F1. FOR #MM = 01 TO 12
 RESET #DD
 R1. REPEAT
  ADD 1 TO #DD
  MOVE EDITED #YYYYMMDD TO #D1 (EM=YYYYMMDD)
  MOVE EDITED #D1 (EM=O) TO #WD
  WRITE #D1 (EM=YYYYMMDD) #WD
  UNTIL #WD = '1'      /* R1.
 END-REPEAT        /* R1.
 WRITE #D1 (EM=YYYYMMDD' 'N(10))
*
* CALL NOP SCHEDULE API TO SET THESE DAYS
*
 CALLNAT 'NOPUCS1N'
  'S' #RC3 'S' #OWNER #OBJECT #YYYYMMDD
 WRITE #YYYYMMDD #RC3
END-FOR      /* F1.
```

```
END TRANSACTION
END
```

NOPUJI4N: Import Operating System Jobs into the Active Queue

You can activate a job in Entire Operations together with an existing operating system job number. This is possible if the job was submitted by any application and set to HOLD status.

Use the following statement:

```
CALLNAT 'NOPUJI4N'
   P-FUNCTION P-RC P-OWNER P-NETWORK P-JOB P-RUN P-EXECUTION-NODE P-SYSOUT-NODE
              P-SYSOUT-FILE P-BS2000-MONJV P-JOB-ID P-OS-JOB-NAME
```

Parameter Description

Parameter	Format/Length	Use		
P-FUNCTION	A1	in	Function code:	
			I	Import job, without prerequisite check.
			J	Import job, with prerequisite check. The new active job is created with input and output condition definitions.
P-RC	I4 BY VALUE RESULT	out	Return code:	
			0	Function OK.
			101	Invalid function code.
			102	Parameter(s) missing.
			103	P-JOB-ID value invalid:
			121	Owner does not exist.
P-OWNER	A10	in	Optional[1].	
P-NETWORK	A10	in	Optional[1].	
P-JOB	A10	in	Optional[1].	
P-RUN	I4 BY VALUE RESULT	in	Optional[1]. If P-RUN is set to zero (0), Entire Operations assigns a new run number; otherwise, the job is activated under the run number given. Job masters for imports with run number must be defined with the **special type** R to prevent automatic activation.	
P-EXECUTION-NODE	I4 BY VALUE	in	Required. Execution node.	

Parameter	Format/Length	Use	
P-SYSOUT-NODE	I4 BY VALUE	in	Optional. Necessary only if P-SYSOUT-FILE is specified, and if P-SYSOUT-NODE is different from P-EXECUTION-NODE. Default: same value as P-EXECUTION-NODE.

P-SYSOUT-FILE	A250
	
	BY VALUE
		in	For BS2000, UNIX and Windows.
			
			File which contains the SYSOUT.
			
			UNIX (for example, Bourne Shell):
			
			End your script with
			
			`set -x`
			`exit $EXITCODE`
			
			assuming the EXITCODE file contains the exit code you want to set so that Entire Operations End-of-Job checking can parse the exit code from the SYSOUT.
			
			Linux:
			
			If a SYSOUT file is not given here and if a process with PID = P-JOB-ID is currently executing, Entire Operations tries to determine the SYSOUT file from the stdout assignment of the PID.

P-BS2000-MONJV	A250 BY VALUE	in	Optional. BS2000 monitor job variable. Specify this parameter only for BS2000, and only if the job was started in conjunction with a monitor job variable.
P-JOB-ID	A10	in	Required. The job ID can be passed left-justified in this field, regardless of whether it is numeric or alphanumeric.
P-OS-JOB-NAME	A8	in	Optional; from operating system.

[1] The parameters P-OWNER, P-NETWORK, P-JOB and P-RUN are optional. If any of these are empty, IMPORT is used instead as owner, network and/or job name. Entire Operations will try to find a job definition for these values. If a definition is found, the new active job entry is supplied from it.

NOPUJS2N: Job Schedule Inquiry and Modification

You can inquire and modify a job schedule by using the following statement in your Natural application:

```
CALLNAT 'NOPUJS2N'
    P-FUNCTION P-RC P-DBENV P-OWNER P-NETWORK P-NETWORK-VERSION
    P-RUN P-JOB P-AVERAGE-TIME P-EARLIEST-START
    P-LATEST-START P-DEADLINE
```

Parameter Description

Parameter	Format/Length	Use		
P-FUNCTION	A1	in	Function code:	
			N	Inquire job schedule for the next job (in alphabetical order). At the end of the list, RC = 1 is returned.
			R	Inquire job schedule.
			S	Change job schedule.
P-RC	N3	out	Return code:	
			0	Function OK.
			1	Owner, network, run number or job not found.
			101	Invalid function code.
			102	Parameter missing.
			103	Earliest start time invalid.
			104	Latest start time invalid.
			105	End time invalid.
			106	Latest start must be greater than earliest start.
			107	End time must be greater than latest start.
			108	Job has already been started.
			109	Earliest start must be greater than current time.
			110	Job is in hold status.
			121	Owner does not exist.
P-DBENV	A10 OPTIONAL	in	Database environment (reserved for future use).	

Parameter	Format/Length	Use		
P-OWNER	A10	in	Owner of the network.	
P-NETWORK	A10	in	Job network.	
P-NETWORK-VERSION	A10	in	Network version.	
P-RUN	I4	in	Run number.	
			0	Data of the network master is inquired or modified.
			Other value	Data of the specified active network run is inquired or modified.
P-JOB	A10	in	Job.	
P-AVERAGE-TIME	I4	in/out	Expected runtime tenths of a second.	
P-EARLIEST-START	A14	in/out	Earliest possible start time. Format: YYYYMMDDHHIISS	
P-LATEST-START	A14	in/out	Latest possible start time. Format: YYYYMMDDHHIISS	
P-DEADLINE	A14	in/out	Deadline. Format: YYYYMMDDHHIISS	

NOPULW9N: Write Messages to System Automation Tools Log

You can write messages to the System Automation Tools log file. The messages will be written with the product identifier of Entire Operations so that they will appear in the Entire Operations log display.

Use the following statement:

```
CALLNAT 'NOPULW9N'
    P-TASK-ID P-MESSAGE-CODE P-MESSAGE P-DBENV P-OWNER P-NETWORK P-NETWORK-VERSION
     P-RUN P-JOB P-JOB-ID P-LOG-TARGET P-USE-NAT-MSG-CACHE
```

Parameter Description

Parameter	Format/Length	Use	
P-TASK-ID	A8 BY VALUE	in	User ID of person initiating the Entire Operations log messages. This can remain blank, in which case it is supplied with the content of the Natural *USER system variable. Entire Operations-internal use: name of the current Monitor task.
P-MESSAGE-CODE	N4 BY VALUE	in	Internal message code (not displayed); used for later message selection. Use message codes in the range 8000 - 8999 to avoid conflicts with internal messages.
P-MESSAGE	A70/1:V BY VALUE	in	The message text line(s). **Note:** This parameter is a dynamic array. One or several text lines may be passed.
P-DBENV	A10 BY VALUE OPTIONAL	in	Database environment (reserved for future use).
P-OWNER	A10 BY VALUE OPTIONAL	in	Owner; used for later message selection; can remain blank.
P-NETWORK	A10 BY VALUE OPTIONAL	in	Network; used for later message selection; can remain blank.
P-NETWORK-VERSION	A10 BY VALUE OPTIONAL	in	Network version; used for later message selection; can remain blank.
P-RUN	I4 BY VALUE OPTIONAL	in	Run number assigned to the message; can remain blank.
P-JOB	A10 BY VALUE OPTIONAL	in	Job; used for later message selection; can remain blank.
P-JOB-ID	A10 BY VALUE OPTIONAL	in	Operating system job identifier; can remain blank. Numeric values must be right-justified with leading zeros, for example: 0000004711.
P-LOG-TARGET	A3	in	Defines where to write the log message:

Parameter	Format/Length	Use		
			NOP	Writes to Entire Operations log (default).
			SO	Writes to the SYSOUT of the current monitor task or of the batch job.
			ALL	Writes to both the Entire Operations log and the SYSOUT.
P-USE-NAT-MSG-CACHE	L OPTIONAL	in	Use the Natural internal message cache. If this routine is called from outside the Entire Operations Monitor, this parameter must be skipped or set to FALSE, because the Natural message cache is not initialized in this case. If this routine is called from within the Entire Operations Monitor, you may set this parameter to TRUE. In case of any problems with this setting, skip the parameter or use FALSE.	

The timestamp and the originating Natural user ID are inserted automatically.

This routine is also invoked internally by the other user API routines.

NOPUMI1N: Set/Reset Text Milestones in Master and Active Jobs

You can inquire on or modify milestone definitions in master and active jobs (see also **Milestone types** in the section *Job Maintenance*) by using the following statement in your Natural application:

```
CALLNAT 'NOPUMI1N'
    P-FUNCTION P-MILESTONE-TYPE P-DBENV P-OWNER P-NETWORK P-NETWORK-VERSION
    P-RUN P-JOB P-RC
```

The parameters have the following meaning:

Parameter	Format/Length	Use		
P-FUNCTION	A1	in	Function code:	
			S	Set milestone.
			R	Reset (delete) milestone.
			T	Test milestone.
P-MILESTONE-TYPE	A1	in	Milestone type:	
			B	Milestone set at network begin.
			E	Milestone set at network end.
			I	Milestone set for any job (any network position except begin and end).

Parameter	Format/Length	Use		
			A	All milestone types (not for function code S).
P-DBENV	A10 OPTIONAL	in	Database environment (reserved for future use).	
P-OWNER	A10	in	Owner of the network.	
P-NETWORK	A10	in	Name of the job network.	
P-NETWORK-VERSION	A10 OPTIONAL	in	Version of the job network.	
P-RUN	I4 BY VALUE OPTIONAL	in	Run number of the job network.	
P-JOB	A10	in	Name of the job.	
P-RC	I4 BY VALUE RESULT	out	Return code:	
			0	Function OK. Or: Milestone was set.
			1	Owner, network, version, run and/or job not found.
			2	Milestone was not set.
			9	Current version cannot be determined.
			101	Invalid function code.
			102	Parameter missing.
			103	Invalid milestone type.
			121	Owner does not exist.
			122	Network version is required.

NOPUMT3N: Expand Message Texts

Entire Operations stores its message texts in the Log file in a language-independent, compressed format.

To read the proper message text, you must use the following routine:

```
CALLNAT 'NOPUMT3N'
    P-APPLIC-ID P-MESSAGE P-USE_NAT_MSG_CACHE P-LANGUAGE P-PREFIX-HANDLING
```



Note: Expansion is only necessary if the message text begins with a semicolon (;).

Parameter Description

Parameter	Format/Length	Use		
P-APPLIC-ID	A8	in	Name of the application from which the message is to be taken.	
P-MESSAGE	A70	in	The message text. Input: compressed. Output: readable text.	
P-USE-NAT-MSG-CACHE	L	in	Use the Natural internal message cache. If this routine is called from outside the Entire Operations Monitor, this parameter must be set to FALSE, because the Natural message cache is not initialized. If this routine is called from within the Entire Operations Monitor, you may set this parameter to TRUE. In case of any problems with this setting, use FALSE.	
P-LANGUAGE	I1	in	Language code for the message. Values:	
			0	Use current setting of the Natural *LANGUAGE system variable.
			1	English.
			2	German.
			other	English.
P-PREFIX-HANDLING	A1	in	A (or blank)	Do not remove message prefix.
			,	
			B	If prefix AAA9999 - , return from position 9. Example: AAAA999 - text --> - text
			C	If prefix AAA9999 - , return from position 11. Example: AAAA999 - text --> text
			D	If prefix is missing in SYSERR, insert a prefix. The resulting message will be EOR1234 - Just an example. Example: The message in internal format is ;EOR;1234, and the SYSERR message text is Just an example.

Parameter	Format/Length	Use		
				The resulting message will be `EOR1234 - Just an example.`

NOPUNI1N: Invalidate Entire System Server Node Table Entries

You can use this API to invalidate session node table entries.

This API must be invoked before exiting a user exit if logons to Entire System Server nodes were made. The invalidation forces new Entire System Server logons to the nodes after termination of the calling user exit.

Use the following statement:

```
CALLNAT 'NOPUNI1N'
    P-FUNCTION P-NODE(*) P-RC
```

Parameter Description

Parameter	Format/Length	Use		
`P-FUNCTION`	A1	in	Function code:	
			`I`	Invalidate nodes.
`P-NODE`	I4/1:V	in	Node number(s) of nodes to be invalidated. The array may have an arbitrary size.	
`P-RC`	I4	out	Return code:	
			`0`	Function OK.
			`101`	Invalid function code.

NOPUNX1N: Entire System Server Calls to Access UNIX and Windows Files

You can use this API to access UNIX and Windows files out of Natural programs.

For `FILE / READ` or `FILE / WRITE`, and if `BUFFER-TYPE` `B`, `C` or `D` is used:

```
CALLNAT 'NOPUNX1N' NOPUNX1A data buffer[caller-module]
```

For all other calls:

```
CALLNAT 'NOPUNX1N' NOPUNX1A [1X caller-module ]
```

For calls to NOPUNX1N, use the parameter data area NOPUNX1A described in the following sections.

- Parameter Description
- Runtime Environment
- Programming Requirements
- Opening and Closing a File
- Reading a File
- Searching for specific Data Sets in a File
- Writing a File
- Reading and Writing a File Using a Large Buffer
- Deleting a File
- Copying a File
- Moving or Renaming a File
- Verifying the Existence of a File
- Creating a List of Files of a Directory

Parameter Description

This section covers the following sections:

- Data area NOPUNX1A
- data buffer
- caller module (input only, optional)

Data area NOPUNX1A

The parameters in the parameter data area NOPUNX1A have the following meaning:

Parameter	Format/Length	Use		
OBJECT	A16	in	Possible values: USER FILE JOB SERVER	
FUNCTION	A8	in	Possible values for USER:	
			LOGON	Logon to a node.
			LOGOFF	Logoff from a node.
			Possible values for FILE:	
			APPEND	Append to an existing file.
			COPY	Copy file.
			CLOSE	Close file.

Parameter	Format/Length	Use		
			DELETE	Delete file.
			DELDIR	Delete file directory.
			DIR	List files.
			DTA	List files sorted by modification time in descending order.
			DTD	List files sorted by modification time in descending order.
			GETATTR	Get file attributes.
			LINECNT	Count lines in a file.
			MOVE	Move file or rename.
			READ	Read file.
			SCAN	Search for strings in a file.
			WRITE	Write file.
			Possible values for JOB:	
			CANCEL	Cancel job.
			COMMAND	Execute command.
			GETEXIT	Get exit code of a terminated job.
			Possible values for SERVER:	
			CL-TL-G	Get command log level and trace level.
RETURN-CODE	N8	out	0	Function was OK.
			4	No more data
			101	Object is not implemented
			102	Not a Unix or Windows node
			404	User undefined
			5509	Logon failed. Check USERID and / or PASSWORD.
			5510	Logon required
			5995	File not found
			other value	See content of RETURN-TEXT.
RETURN-TEXT	A80	out	Explanatory text for RETURN-CODE.	
NODE	N5	in	Node, as defined in Entire Operations.	
NODE-NAME	A16	in	Node name. ■ Can be passed as an alternative to NODE.	

Parameter	Format/Length	Use		
			■ Will be used only if NODE=0.	
NODE-ACCESS-MODE	A1	Can only be changed by the API	*empty*	Has not yet been checked.
			N	Node is not local.
			Y	Node is local.
			This field is set by API at USER / LOGON. It should not be changed by the application.	
USERID	A20	in	User ID for logon.	
GROUP	A20	in	Optional:	
			UNIX	Group for logon.
			Windows	Domain for logon.
PASSWORD	A16	in	Password for logon. Obsolete. Please use UL-PASSWORD-A256. We recommended that you clear this field after a successful USER / LOGON.	
SID or SESSION-ID	N10	Can only be changed by the API	Will be provided by the API after a successful USER / LOGON call. For other calls it is an input parameter.	
SERVER-COMMAND-LOG-LEVEL	N5	out	Output of SERVER / CL-TL-G.	
SERVER-TRACE-LEVEL	N5	out	Output of SERVER / CL-TL-G.	
HTML-CODE-HANDLING	A1	in/out	For FILE functions.	
CODEPAGE-ID	N5	in	For the FILE functions WRITE and APPEND.	
OPSYS-CLASS	A1	Can only be changed by the API	Operating system class of NODE. Possible values:	
			B	BS2000
			M	z/OS
			X	UNIX
			W	Windows
USE-PAM	A1	in	For USER / LOGON on UNIX nodes.	
			N	Do not use PAM for LOGON.
			Y	Use PAM for LOGON.
INHERIT-ACCESS-TIME	A1	in	For FILE functions.	
BUFFER-FORMAT	A1	in	For the FILE functions READ, WRITE and APPEND. Possible values:	
			T	Text, single record.

Parameter	Format/Length	Use		
			3	Text, multiple records, 3-byte fields.
			L	Text, multiple records, variable-length fields.
			A	Base64-encoded format.
UNUSED-1	A10	-	Unused.	
ESC-TRIGRAPH	A1	in	Trigraph usage:	
			empty or N	No trigraph converted.
			Y	Use question mark (?) as trigraph escape character.
			other	Use this character as trigraph escape character.
			For information on **trigraph encoding**, see the relevant section in *Submission of Jobs by Entire Operations*.	
TRACE-LEVEL	N1	in	0	No trace.
			> 0	Trace activated.
TRACE-TARGET	A1	in	Destination where trace will be stored.	
			S	SYSOUT.
			L	Entire Operations log.
			B	Both: SYSOUT and Entire Operations Log (only relevant if TRACE-LEVEL > 0).
BUFFER-TYPE	A1	in	Type of buffer. Mainly used for the FILE functions READ, WRITE and APPEND. Possible values:	
			A (or blank)	XS-FIRWA-BUFFER-TYPE, size 600.
			B	XS-FIRWB-BUFFER-TYPE, size 9700.
			C	XS-FIRWC-BUFFER-TYPE, size 99000.
			D	XS-FIRWD-BUFFER-TYPE, size 59000.
			For the buffer types B, C and D: The buffer is to be passed as a separate parameter in the CALLNAT 'NOPUNX1N' statement (mentioned earlier).	

Parameter	Format/Length	Use		
OBJECT-AREA	A1200	-	Object-specific fields.	
			Object: FILE (redefinition of OBJECT-AREA).	
F-FILE	A250	in	File name. The file name must be fully qualified. It must contain a complete path starting at the root of the file system. The file name may contain environment variables.	
F-HANDLE	N10	Can only be changed by the API	File handle. Is allocated by Entire System Server at the first access of a file. Is active until FILE / CLOSE.	
F-LINE	N10	in/out	In	Line number where reading begins (0 or 1: start reading at the beginning of the file).
			Out	(Last) read line number.
F-NUMBER-RECORDS	N10	out	Number of read data sets.	
F-SEARCH-TEXT	A100	in	Only FILE / SCAN: Text to be searched for in the file.	
F-RECORD-BUFFER	A600	in/out	Whole record buffer (redefinition of F-RECORD BUFFER). Used for BUFFER-TYPE='A' only.	
F-RECORD-250	A250	in/out	FILE / READ, FILE / WRITE: Effective record. Used for BUFFER-TYPE='A' only.	
F-RECORD-SEPARATOR	A5	in	For future use.	
F-LINE-LIMIT	N10	in	FILE/COPY, FILE/MOVE: For text files: The maximum number of records to be copied or moved.	
			Object: JOB Function: COMMAND (redefinition of OBJECT-AREA).	
JCM-COMMAND	A500	in	The command to be executed.	
JCM-OUTPUT-FILE	A250	in	Output file for the command.	
JCM-REMOVE-OUTPUT-FILE	A1	in	N	Keep the output file.
			Y	Delete the output file after successful command execution.

Parameter	Format/Length	Use		
JCM-USER	A20	in	User ID for the command.	
JCM-GROUP	A20	in	Optional:	
			UNIX	Group for the command.
			Windows	Domain for the command.
JCM-WAIT-FOR-TERMINATION	A1	in	N	The termination of the command is not to be waited for.
			Y	The termination of the command is to be waited for.
JCM-OUTPUT-MODE	A1	in	C	Overwrite existing output file. (default value)
			A	If the output file already exists, append the output to it.
JCM-EXIT-CODE-MSG-APPEND	A1	ein	N	Do not append an exit code message to the output file.
			Y	Append the exit code message NPR3355 to the output file. (default value)
JCM-EXIT-CODE	N10	out	Exit code of the command.	
JCM-PID	N10	out	Process ID of the command.	
			Object: JOB Function: SUBMIT (redefinition of OBJECT-AREA)	
JSU-SUBMIT-FILE	A250	in	The file to be submitted (e.g. shell script, Windows BAT file).	
JSU-SYSOUT-FILE	A250	in	Output file for the file to be submitted.	
JSU-EXIT-CODE	N10	out	Only for the functions	
			GETEXIT	Read exit code from an output file.
			CANCEL	Cancel the process.
JSU-EXIT-CODE-SIGN			Sign + or - of exit code. If the exit code itself has 10 digits, exitCode contains the absolute part, and the sign must be read from exitCodeSign. exitCodeSign is always being set, regardless of a sign in exitCode.	
JSU-USER	A20	in	User ID of the file to be submitted.	
JSU-GROUP	A20	in	Optional:	

Parameter	Format/Length	Use		
			UNIX:	Group for the file to be submitted.
			Windows:	Domain for the file to be submitted.
JCA-KILL-MODE	A1	in	Only for UNIX, only for function CANCEL:	
			K	Use SIGKILL. (default value)
			T	Use SIGTERM.
JSU-PID	N10	out	Process ID of the submitted file.	
			Object: USER Function: LOGON (redefinition of OBJECT-AREA)	
UL-PASSWORD-MODE	A1	in	C	UL-PASSWORD-A256 contains a password in plain text. (default value)
			E	UL-PASSWORD-A256 contains an encrypted password.
UL-PASSWORD-A256	A256	in	The password. The encryption is determined by UL-PASSWORD-MODE. When this field is empty, the password is taken from PASSWORD. **Note:** We recommended that you clear this field after a successful USER / LOGON.	

data buffer

Mandatory for FILE / READ or FILE / WRITE, and if **BUFFER-TYPE** B, C or D is used. Optional for all other cases.

caller module (input only, optional)

You can pass the name of a calling module here. It will be passed to NPR / UNIX for an easier identification of the caller. If you want to pass your own module name here, you can use the following code:

```
...
1 #PROGRAM (A8)
...
#PROGRAM := *PROGRAM
...
CALLNAT 'NOPUNX1N' NOPUNX1A 1X #PROGRAM (AD=O)
...
↩
```

Runtime Environment

- The API can be run on mainframes, UNIX or Windows.
- With this API you can access only UNIX and Windows nodes.
- In the selected Natural environment you must define the steplib SYSEOR and the steplibs used by SYSEOR.
- The system files (System File 1, System Automation Tools Log File) used by Entire Operations must be defined by `LFILE` assignments.

Programming Requirements

API Invocation

In general, API `NOPUNX1N` is invoked with the requested object and function.

Windows File Names

Windows file names can also be specified with a slash instead of a backslash.

Example:

```
c:/work/file1.txt
```

is equivalent to

```
c:\work\datei1.txt
```

Return Code Checking

Checking **NOPUNX1A.RETURN-CODE** is mandatory after each API call.

`NOPUNX1A.RETURN-CODE = 0` (zero) indicates that the API call was ok.

NOPUNX1A.RETURN-TEXT can contain more information related to the return code.

If an error occurs, an appropriate reaction is required.

If processing is to be terminated as a reaction to an error, please take care of the following:

- If a file is currently open, it must be closed using `FILE / CLOSE`.

- If a session is currently open, it must be closed using USER / LOGOFF.

Session

All file operations are functional only during a valid session.

A session must be started with USER / LOGON, and must be terminated with USER / LOGOFF.

During a session (including USER / LOGOFF), the NOPUNX1A.SESSION-ID must not to be changed.

A USER / LOGOFF at the end of a session is mandatory. After the USER / LOGOFF, the NOPUNX1A.SESSION-ID is invalid and must not be used anymore for any call.

Course of Session

```
USER / LOGON
FILE / ...
FILE / ...
...
USER / LOGOFF
```

Opening and Closing a File

- Files must not be explicitly opened. This will be done automatically with the first call of FILE / READ or FILE / WRITE.
- At the first successful FILE / READ or FILE / WRITE, the API returns a file handle in NOPUNX1A.F-HANDLE. This must be passed by all subsequent calls to the same file (read or write session), including the final FILE / CLOSE call.
- After a read or write session, the file must be closed with the function FILE / CLOSE.

  NOPUNX1A.F-HANDLE becomes invalid then.

Reading a File

With the function FILE / READ, a file is read in sequential mode.

In NOPUNX1A.F-LINE you can provide a line number to start.

If NOPUNX1A.F-LINE contains the value 0 (zero) or 1, the read sequence starts at the beginning of the file.

Scheme

```
USER / LOGON

R1. repeat
   FILE / READ
   if (NOPUNX1A.RETURN-CODE ne 0)
       escape bottom (R1.)
   end-if
   ... handle record
end-repeat       /* R1.

FILE / CLOSE

USER / LOGOFF
```

Searching for specific Data Sets in a File

With the function `FILE / SCAN`, it is possible to search for a specific string in a file. The search string must be present in `NOPUNX1A.F-SEARCH-TEXT`.

Scheme

```
USER / LOGON

R1. repeat
   FILE / SCAN
   if (NOPUNX1A.RETURN-CODE ne 0)
       escape bottom (R1.)
   end-if
   ... handle record
   add 1 to NOPUNX1A.F-LINE
*          -- line start for follow-up scan
end-repeat       /* R1.

FILE / CLOSE

USER / LOGOFF
```

Writing a File

With the function `FILE / WRITE`, a file is written in sequential mode.

Scheme

```
USER / LOGON

R1. repeat
   ... allocate record  in NOPUNX1A.F-RECORD-250
FILE / WRITE
   if (...)
       escape bottom (R1.)
   end-if
end-repeat       /* R1.

FILE / CLOSE

USER / LOGOFF
```

Reading and Writing a File Using a Large Buffer

The following is an example of using a large data buffer to read or write files.

1. Define a large data buffer:

   ```
   1 #BUFFER-99000 (A99000)
   ```

   Choose one of the following buffer types:

   ```
   NOPUNX1A.BUFFER-TYPE := 'B'              /* 9900 bytes
   NOPUNX1A.BUFFER-TYPE := 'C'              /* 99000 bytes
   NOPUNX1A.BUFFER-TYPE := 'D'              /* 59000 bytes
   ```

2. Call the API:

   ```
   CALLNAT 'NOPUNX1N' NOPUNX1A #BUFFER-99000
   ```

   The parameter definition of NOPUNX1N is:

   ```
   DEFINE DATA
   PARAMETER USING NOPUNX1A
   PARAMETER
   1 P-BUFFER-99000 (A99000)
     BY VALUE RESULT OPTIONAL
   ...
   END-DEFINE
   ```

Note: #BUFFER-99000 can contain a maximum of 9900, 99000 or 59000 bytes depending on the size specified with NOPUNX1A.BUFFER-TYPE. The data written to the buffer can have less bytes but must not exceed the specified buffer size.

Deleting a File

With the function FILE / DELETE, you can delete one or several files. If F-FILE contains a wildcard (*), all files will be deleted that match the wildcard.

Copying a File

With the function FILE / COPY, the target file is passed on to F-RECORD-250.

Moving or Renaming a File

With the function FILE / MOVE, the target file is passed on to F-RECORD-250.

Verifying the Existence of a File

To verify the existence of a file, you can use the function FILE / DIR with a defined file name. If a 0 (zero) is returned to NOPUNX1A.RETURN-CODE, the files exists.

Creating a List of Files of a Directory

With the function FILE / DIR, you can create a list of files of a directory. With the wildcard expression contained in NOPUNX1A.F-FILE you can keep the list small.

Example

/tmp/*	All files in the directory /tmp.
/tmp/*.txt	All files in the directory /tmp that end with .txt.

The file names are stored in F-RECORD-BUFFER. The file names are separated by a semicolon (;) as indicated for SHOW-DIR-BUFFER in the following example.

If the buffer is not large enough for the file names, then the value 19 (more objects) is passed on to NOPUNX1A.RETURN-CODE.

Scheme

```
USER / LOGON

... allocate file name (with wildcard) in NOPUNX1A.F-FILE

R1. repeat
FILE / DIR
   decide on first value of NOPUNX1A.RETURN-CODE
       value 0, 19      /* 0: ok, 19: more objects
            perform SHOW-DIR-BUFFER
       none value
            ignore
   end-decide
   if  NOPUNX1A.RETURN-CODE ne 19
       escape bottom (R1.)
   end-if
   add 1 to NOPUNX1A.F-LINE
end-repeat       /* R1.

FILE / CLOSE

USER / LOGOFF

* ----------------------------------------------------------------------
DEFINE SUBROUTINE SHOW-DIR-BUFFER
*                 ---------------
* -- Function FILE / DIR:
* -- The record buffer contains a file list, with delimiter ';'
* -- This routine separates the single file names out of the buffer.
* -- #DIR-FILE and #DIR-REST should be declared as (A600)
*
SDB1. REPEAT
     EXAMINE  NOPUNX1A.F-RECORD-BUFFER FOR ';' REPLACE FIRST WITH ':'
SEPARATE NOPUNX1A.F-RECORD-BUFFER INTO #DIR-FILE #DIR-REST
              WITH DELIMITER ':'
*  -- now #DIR-FILE contains a single file name, without qualifiers

         IF #DIR-REST = ' '
             ESCAPE BOTTOM (SDB1.)                                      ↩

         END-IF
  NOPUNX1A.F-RECORD-BUFFER := #DIR-REST
END-REPEAT     /* SDB1.
END-SUBROUTINE
```

NOPURE2N: Handle Resource Allocations

You can handle resource allocations and deallocations with the following statement:

```
CALLNAT 'NOPURE2N'
    P-FUNCTION P-RC P-DBENV P-OWNER P-NETWORK P-NETWORK-VERSION P-RUN
    P-JOB P-RESOURCE P-QUANTITY P-DEALLOCATION P-JOB-EXECUTED
    [P-ALLOCATION-MODE] [P-DEALLOCATE-NOT-OK]
```

Parameter Description

Parameter	Format/Length	Use		
P-FUNCTION	A1	in	Function code:	
			A	Allocate resource.
			R	Release one or several resources. ■ If job and resource name given: release only one allocated resource of one job. ■ If job given, but no resource: release all allocated resources of one job. ■ If run number, but no job and no resource given: release all allocated resources of one network run. ■ If no run number and no job and no resource given: Release all allocated resources of all network runs.
			N	Network release. Release resources with deallocation = N as well. For other parameters, see R.
			F	Forced release. Release resources with deallocation = N or K as well. For other parameters, see R.
			G	Forced release, by resource usage. Release resources with **deallocation** = N or = K as well. For other parameters, see R.

Parameter	Format/Length	Use		
			P	Modify (active) prerequisite resource definition.
			T	Test resource usage, sorted by active jobs. Subsequent calls read in the order of active jobs, then resource names. **Sequential reading** ■ The sequential reading starts at the first found resource in the order of active jobs, then resource names. ■ Fields returned by the previous call must be kept and passed for the next call. ■ P-RC = 5 (resource usage entry not found) is returned, if there are no more entries for this resource.
			U	Test resource usage, sorted by resources, then by active jobs. **Sequential reading** ■ The sequential reading starts at the first found resource, in alphabetical order, where the name of the first found resource is equal or subsequent to the name passed in P-RESOURCE. ■ Fields returned by the previous call must be kept and passed for the next call. ■ P-RC = 5 (resource usage entry not found) is returned at the end of all resource usage entries.
			V	Test resource usage, sorted by active jobs. Subsequent calls read in the order of resource names, then active jobs. **Sequential reading** ■ The sequential reading starts at the first found resource followed by the superdescriptor of the active job (identified by P-OWNER, P-NETWORK, P-RUN, P-JOB).

Parameter	Format/Length	Use		
			■ Fields returned by the previous call must be kept and passed for the next call. ■ P-RC = 5 (resource usage entry not found) is returned at the end of all resource usage entries.	
P-RC	N3	out	Return code:	
			0	Function OK.
			1	Active job not found.
			2	Resource not defined for job.
			3	Resource (master) definition not found.
			4	Quantity is not available.
			5	OK, but no resource usage entries were found. This code may be returned: ■ if the resource has already been freed. ■ if the resource was never allocated.
			6	Prerequisite resource definition not found.
			101	Invalid function code.
			102	Parameter(s) missing.
			103	Wildcard not allowed.
			104	P-DEALLOCATION has wrong value.
			121	Owner does not exist.
P-DBENV	A10 OPTIONAL	in	Database environment (reserved for future use).	
P-OWNER	A10	in	Owner. Wildcard allowed. Function A: obligatory, no wildcard allowed.	
P-NETWORK	A10	in	Network. Wildcard allowed. Function A: obligatory, no wildcard allowed.	
P-NETWORK-VERSION	A10	in	Network version.	
P-RUN	I4	in	Run. A value of zero means all runs of a network. Function A: obligatory, no wildcard allowed.	

Parameter	Format/Length	Use	
P-JOB	A10	in	Job. If empty, the whole network is meant. Wildcard allowed. Function A: obligatory, no wildcard allowed. **Note:** If this API is used to free a resource and if the allocation was made by another job, this field must remain empty.
P-RESOURCE	A20	in	Name of the resource. If empty, all prerequisite resource definitions of a job are meant. Function A: obligatory, no wildcard allowed.
P-QUANTITY	N7.2	in	For function A and B: quantity to be allocated. The value must be greater than or equal to zero.
P-DEALLOCATION	A1	in	How to deallocate this allocation (for function A only).
			J — After job termination.
			N — After network termination.
			K — Keep until explicit release.
			For a detailed description of deallocation modes, see ***Resource Deallocation Modes*** in the section *Job Maintenance*.
P-JOB-EXECUTED	L	in	Not reusable resources are decreased only if the job was really executed.
P-ALLOCATION-MODE	A1 OPTIONAL	in	Allocation mode
			A — Always allocate the resource (default).
			T — Do not allocate the resource if the active job becomes a temporary dummy, see ***Temporary Dummy Jobs***.
P-DEALLOCATE-NOT-OK	A1 OPTIONAL	in	Deallocate if job not ok. **Note:** This definition is only relevant for deallocation mode J, see ***Resource Deallocation Modes*** in the section *Job Maintenance*.
			Y — Deallocate the resource even if the job was not ok (default).
			N — Keep the resource allocated if the job ended not ok.

NOPURS1N: Access Entire Operations Resource Masters

This section covers the following topics:

- Handling Entire Operations Resource Masters
- Description of Function Codes

Handling Entire Operations Resource Masters

You can handle resource masters with the following statement:

```
CALLNAT 'NOPURS1N'
    P-FUNCTION P-RC P-NODE P-RESOURCE P-TYPE P-INIT-QTY P-USED-QTY
```

Meaning of the parameters:

Parameter	Format/Length	Use		
P-FUNCTION	A1	in	Function code:	
			A	Add a resource.
			D	Delete a resource.
			M	Modify a resource.
			T	Test a resource.
			See also *Description of Function Codes*.	
P-RC	N3	out	Return code:	
			0	Function OK; resource found.
			1	Resource not found.
			2	Resource already exists.
			3	Inconsistent values.
			4	Initial quantity may not be modified.
			101	Invalid function code.
			102	Resource name missing.
P-NODE	N3	in	Not used; use constant with value=0.	
P-RESOURCE	A20	in	Name of the resource.	
P-TYPE	A1	in	Type of resource (only for FUNCTION = 'A').	
			N	Not quantitative.
			R	Quantitative, reusable.
			U	Quantitative, not reusable.
P-INIT-QTY	P7.2	in/out	Total quantity.	
P-USED-QTY	P7.2	in/out	Currently used quantity.	

Description of Function Codes

Meaning of the function codes:

Code	Description
A	Adds a new resource definition. You must give values for P-TYPE, P-INIT-QTY and P-QTY.
D	Deletes a resource definition.
M	Used to modify the values P-INIT-QTY.
T	Returns the current values of P-INIT-QTY and P-QTY.



Notes:

1. Before using M for modifications, you should use T to read the present values (unless you want to enter an absolute value).
2. The currently used amount of a resource can no longer be modified directly. You must perform a resource allocation or deallocation with the API **NOPURE1N** to modify the currently used resource amount.

NOPUSN2N: Inquire Calling Job or Called Network for Subnetworks

You can use this API to:

- Find the current subnetwork for a job of type NET (subnetwork);
- Find the current job calling a subnetwork.

Use the following statement:

```
CALLNAT 'NOPUSN2N'
    P-FUNCTION P-RC P-DBENV *P-OWNER P-NETWORK P-NETWORK-VERSION
    P-RUN P-JOB
```

The parameters P-OWNER, P-NETWORK, P-RUN and P-JOB are input and output parameters. They are overwritten with the values found by the API. Therefore the caller must supply these fields with new entries before each new call.

Parameter Description

Parameter	Format/Length	Use		
P-FUNCTION	A1	in	Function code:	
			C	Find calling job (available for active subnetworks only).
			S	Find subnetwork.
			T	Find the topmost calling job (available for active subnetworks only). This function recursively goes back in the subnetwork hierarchy, until the caller job is no longer part of a subnetwork.
P-RC	N3	out	Return code:	
			0	Function OK.
			1	Input object not found.
			2	Wrong job type.
			3	No calling job.
			4	No symbol(s) with caller information.
			101	Invalid function code.
			102	Parameter missing.
			121	Owner does not exist.
			141	Symbol access error.
P-DBENV	A10 OPTIONAL	mod	Database environment (reserved for future use).	
P-OWNER	A10	mod	Owner of the network.	
P-NETWORK	A10	mod	Network.	
P-NETWORK-VERSION	A10	mod	Network version.	
P-RUN	I4	mod	Run number. If the function S is used for a job master, this parameter must be zero.	
P-JOB	A10	mod	Job.	

NOPUSP3N: Display Long Texts for Symbol Prompting

You can display the symbol prompting long text for a specified symbol by using the following statement in your Natural application:

```
CALLNAT 'NOPUSP3N'
    P-FUNCTION P-DBENV P-OWNER P-NETWORK P-RUN P-SYMTAB P-SYMTAB-VERSION
               P-SYMBOL P-PROMPT-TYPE P-RC P-PROMPT-TEXT (*)
```

Parameter Description

Parameter	Format/Length	Use		
P-FUNCTION	A1	in	Function code:	
			G	Get prompting attributes (for master and active).
			S	Set prompting attributes (for master and active).
P-DBENV	A10 OPTIONAL	in	Database environment (reserved for future use).	
P-OWNER	A10	in	Owner of the symbol table.	
P-NETWORK	A10	in	Job network (for active symbols).	
P-RUN	I4	in	Job run number (for active symbols).	
P-SYMTAB	A10	in	Symbol table.	
P-SYMTAB-VERSION	A10	in	Symbol table version.	
P-SYMBOL	A40	in	Name of the prompted symbol.	
P-PROMPT-TYPE	A1	in/out	Input parameter for the function code S, and output parameter for the function code G. Possible values:	
			A	Prompt for a symbol at each activation.
			E	Prompt only if no value is specified in the symbol table.
			N	Never prompt for a symbol.
P-RC	N3	out	Return code:	
			0	Function ok.
			1	Symbol not found.
			99	Invalid parameter value.

Parameter	Format/Length	Use		
			111	Symbol table name begins with reserved prefix =EOR=.
P-PROMPT-TEXT	A70/1:V	out	Symbol prompting long text. Caller should provide an array of at least 5 text lines.	

NOPUST3N: Inquire Network and Job Status, Symbol Table

You can inquire the current status of an active job network or of a single active job by using the following statement in your Natural application:

```
CALLNAT 'NOPUST3N'
    P-FUNCTION P-RC P-DBENV P-OWNER P-NETWORK P-NETWORK-VERSION P-JOB P-RUN
    P-SYMTAB P-SYMTAB-VERSION P-JOB-ID P-STATUS-TIME
```

This section covers the following topics:

- Parameter Description
- Inquire Status of Whole Active Network
- Inquire Status of all Jobs of an Active Network Individually
- Inquire Symbol Table Used
- Example of Using NOPUST3N

Parameter Description

Parameter	Format/Length	Use		
P-FUNCTION	A1	in	Function code:	
			A	Get next active run number (in numerical order) beginning with a starting run number. If the starting run number is equal to the run number limit, the range from 1 to the starting run number minus 1 is checked.
			R	Get last activated run number (in chronological order).
			S	Inquire status.
			N	Inquire status of next job (alphabetically).
			W	Status of next job waiting for anything, or of next job in status permanent error.

Parameter	Format/Length	Use		
			Y	Get name of symbol table either from network master or from job definition.
P-RC	N3	out	Return code:	
			0	Function OK.
			1	Network/job not found, or run number does not exist.
			20	Waiting for symbol prompting.
			21	Waiting for job/network activation.
			25	Waiting for prerequisite.
			26	Job(s) are in hold.
			27	Waiting for next action or start time.
			28	Waiting for input condition.
			29	Waiting for resource.
			30	Waiting for operating system node.
			31	Waiting for JCL to be loaded.
			32	Network activation error.
			33	Job activation error.
			34	Symbol replacement error.
			35	Schedule extraction or network activation error.
			36	Job is being submitted.
			37	Job submission error.
			38	Job is executing.
			39	Job execution error.
			41	Network/job is executing.
			42	Performing End-of-Job checking.
			43	End-of-Job checking error.
			44	Performing End-of-Job actions.
			45	End-of-Job actions error.
			65	All jobs ended successfully.
			66	Job(s) ended not successfully.
			69	Permanent error(s).
			93	Waiting for network to be deactivated.
			101	Invalid function code.
			102	Parameters missing.
			121	Owner does not exist.
			999	Status not defined.

Parameter	Format/Length	Use	
P-DBENV	A10 OPTIONAL	in	Database environment (reserved for future use).
P-OWNER	A10	in	Owner of network.
P-NETWORK	A10	in	Job network.
P-NETWORK -VERSION	A10	in	Version of job network.
P-JOB	A10	in	Job. If left blank, inquiry is for whole network.
		out	The job for which the return code is returned.
P-RUN	I4	in	Function code:
			A — Starting run number (can be 0).
			S — Run number to be checked.
		out	Function code:
			A — Next active run number (in numerical order).
			R — Last run number of the network in chronological order.
P-SYMTAB	A10	out	Name of the defined symbol table. Function R: Name of the defined symbol table on the network level. Function Y: Name of the defined symbol table on the network level (empty P-JOB), or the job level (given P-JOB). Empty, if a symbol table is not defined on the requested level.
P-SYMTAB-VERSION	A10	out	Version of the symbol table. The parameter only applies if P-SYMTAB contains a name.
P-JOB-ID	A10	out	ID of the job (for single jobs only, and only if the job was already submitted).
P-STATUS-TIME	A14	out	Timestamp when the current status of the job was set. Format: YYYYMMDDHHIISS.

Inquire Status of Whole Active Network

Use function `S`. Leave the `P-JOB` parameter blank.

Inquire Status of all Jobs of an Active Network Individually

Use function `N`. Delete the `P-JOB` parameter. Then invoke this API in a `REPEAT` loop until you get RC = 1 (network end).

Each call returns the status of a job. The name is contained in `P-JOB`. Do not change the content of `P-JOB`, because it is used as starting value for the next call.

Inquire Symbol Table Used

The symbol table used is always returned for active networks and jobs.

For network masters and jobs, you can use the function `Y`.

Notes:

- The status inquiry functions independently of Monitor activity, because it uses database entries.
- For a whole network inquiry (with function `S`), the statuses of the individual jobs are linked with logical AND. In the worst case, if at least one job has failed, the status `failed` is returned for the whole network.
- An active network is considered to have `terminated OK` only if all its active jobs have `terminated OK`.
- The status inquiry is only possible as long as the network or job involved has not been deactivated.
- To obtain all active run numbers of a network, proceed as follows:
  1. Use function `A`.
  2. Begin with starting run number 0.
  3. Call the API.
  4. Terminate when RC is not equal to 0, or when the number returned for the active run is less than the number of the previous run.
  5. Keep result as next starting run number and continue with Step 3 (Call the API).

Example of Using NOPUST3N

```
...
 1 #RC (N3)
 1 #OWNER (A10)
 1 #NETWORK (A10)
 1 #NETWORK-VERSION (A10)
 1 #JOB (A10)
 1 #RUN (I4)
 1 #SYMTAB (A10)
 1 #SYMTAB-VERSION (A10)
 1 #JOB-ID (A10)
 1 #STATUS-TIME (A14)
 ...
```

```
CALLNAT 'NOPUST3N'
 'S' /* get status
  #RC 1X #OWNER #NETWORK #NETWORK-VERSION #JOB #RUN
  #SYMTAB #SYMTAB-VERSION #JOB-ID #STATUS-TIME
```

In the above example, `1X` is the placeholder for the `OPTIONAL` parameter `P-DBENV`.

NOPUSY7N: Access Entire Operations Symbols

You can handle symbols in symbol tables using the following statement:

```
CALLNAT 'NOPUSY7N'
    FUNCTION RC DBENV OWNER NETWORK NETWORK-VERSION RUN JOB SYMTAB SYMTAB-VERSION ↩
SYMBOL FORMAT
    VALUE USER TIME
   1X 1X
   VALUE-INDEX
```

Where:

`1X 1X` are placeholders for `OPTIONAL` parameters required for internal purpose only.

This section covers the following topics:

- Description of Parameters
- Description of Function Codes
- Sequential Reading in a Symbol Table
- Example of Sequential Symbol Table Reading

Related Topic:

- *Symbol Table and Symbol Maintenance*

Description of Parameters

Parameter	Format/Length	Use		
FUNCTION	A1	in	Function code:	
			D	Delete one multiple value.
			M	Add one multiple value.
			N	Test next symbol.
			P	Test next symbol to be prompted.
			R	Reset a symbol.
			S	Set (add or modify) a symbol.
			T	Test existence of a symbol and inquire its value.
			1	Test existence of a symbol and inquire its value, with recursive resolution.
			4	Applies to active symbol tables only. Test next symbol and inquire its value, with recursive resolution.
			See also *Description of Function Codes*.	
RC	N3	out	Return code:	
			0	Function OK; symbol found.
			1	Symbol not found.
			2	Invalid numeric value.
			3	Invalid format.
			4	Format modification attempted.
			5	Multiple table full.
			6	Value missing.
			7	Attempted to write a predefined (read-only) symbol. Refer to *Predefined Symbols Ranges*.
			10	Invalid value by user exit.
			11	User exit not found.
			12	Access to user exit not allowed (Natural Security).
			20	OK; is a multiple value.
			30	OK; the symbol master was also modified.
			101	Invalid function code.
			102	Parameter missing.
			111	Symbol table name starts with reserved prefix =EOR=.

Parameter	Format/Length	Use		
			121	Owner does not exist.
			131	Invalid symbol table name.
			132	Invalid symbol table version name.
			777	Internal parameter error. Refer to the Entire Operations log for more information.
DBENV	A10 OPTIONAL	in	Database environment (reserved for future use).	
OWNER	A10	in	Owner of the symbol table.	
NETWORK	A10	in	Network (for active symbol table only).	
NETWORK-VERSION	A10	in	Network version.	
RUN	P13	in	Run (for active symbol table only).	
JOB	A10	in	Job.	
SYMTAB	A10	in	Symbol table.	
SYMTAB-VERSION	A10	in	Symbol table version.	
SYMBOL	A40	in	Symbol.	
		out	For function codes N and P.	
FORMAT	A1	in	Symbol format. For the function 1 for symbol table masters, this parameter has a special meaning: FORMAT may contain the escape character to be used for the recursive resolution. The format values (below) and a question mark (?) must not be used as escape character.	
		out	For function codes N, P and T.	
			blank or A	Alphanumeric. No case conversion.
			D	Date in the format: YYYYMMDD
			H	Alphanumeric. Hidden.
			L	Alphanumeric. Conversion to lower case.
			N	Numeric.
			U	Alphanumeric. Conversion to upper case.
VALUE	A250	in	Symbol value.	
		out	For function codes N, P and T.	
VALUE-INDEX	I4	in	Index for multiple-value symbols. Set VALUE-INDEX to an appropriate value. After the call, this value is adapted to the index of the next multiple-value symbol read.	

Parameter	Format/Length	Use	
			If the value set for VALUE-INDEX is higher than the highest index of the given symbol, the next symbol is read.
		out	For function code 4 only.
USER	A8	out	User who made the last modification.
TIME	T	out	Time of the last modification.

Description of Function Codes

Meaning of the function codes:

Code	Description
D	Delete one multiple value. Resets one value in a multiple-value symbol. If this is the last symbol value, the whole symbol is removed from the table.
M	Add one multiple value. Sets one more value in a multiple-value symbol. Even if the quantity of the multiple values is 1, the single value remains a multiple value. It will not be converted into a standard value.
N	Test next symbol. Tries to find the next symbol (in alphabetic order) from the specified symbol. The given name is overwritten by the name found. The other fields are returned as in function T. To find the first symbol of a symbol table, the symbol name can be omitted. If the end of the symbol table is reached, code 1 is returned. See also *Sequential Reading in a Symbol Table*.
P	Test next symbol to be prompted. Finds the next symbol in alphabetical order to be prompted for the current network/job run. Otherwise works like function code N.
R	Reset a symbol. The symbol is removed from the symbol table.
S	Set a symbol. If the symbol does not exist in the symbol table, it is inserted; if it already exists, it is overwritten.
T	Test a symbol. If the symbol is not found, code 1 is returned. If the symbol exists, its format and value are returned. The fields USER and TIME contain the user and timestamp of the last modification.

Code	Description
1	Test a symbol. If the symbol is not found, code 1 is returned. If the symbol exists, its format and value are returned. If the symbol value contains symbols, they will be resolved recursively. The fields USER and TIME contain the user and timestamp of the last modification.
4	Applies to active symbol tables only. Test next symbol and inquire its value, with recursive resolution. This function requires you to specify the following: ■ A network and run number (in addition to owner). ■ The VALUE-INDEX **parameter**. See also ***Sequential Reading in a Symbol Table***.

Notes:

1. To access a symbol table master, the fields NETWORK and RUN must be empty. To access an active symbol table, the fields NETWORK and RUN must be supplied by the caller.
2. If a **validation check user exit** (see the section *User Exits*) is defined for the symbol, it is invoked from this API routine, too. Make sure that the user exit is accessible in the calling environment. The library containing the user exit must be defined as a steplib for the executing environment of these APIs. A symbol value is rejected if the exit returns not OK (RC=10) or if the exit is missing (RC=11).
3. All actions will be logged.
4. The API is can also read the values of **predefined symbols** (described in *Symbol Table and Symbol Maintenance*).
5. Symbol created by this API will get the symbol prompting setting N (never). See also ***Symbol Prompting during Network or Job Activation*** in the section *Symbol Table and Symbol Maintenance*.

Sequential Reading in a Symbol Table

To read symbols sequentially from a master or an active symbol table

1 Set the function to N.

2 If you want to start reading at the start of the table, reset SYMBOL first; otherwise put a start name into SYMBOL.

3 Call the user exit in a REPEAT loop; leave it if RC is not 0 and not 20.

4 Leave SYMBOL unchanged as the start value for the next call.

5 Reset VALUE before the next call if RC=0.

6 Do not reset VALUE if RC=20.

To read symbols sequentially from an active symbol table

1 Set the function to 4.

2 If you want to start reading at the start of the table, reset SYMBOL first; otherwise put a start name into SYMBOL.

3 Leave SYMBOL and VALUE-INDEX unchanged as the start values for the next call.

4 Reset VALUE before the next call if RC=0.

5 Do not reset VALUE-INDEX if RC=20.

Example of Sequential Symbol Table Reading

```
MOVE 'N' TO FUNCTION
RESET SYMBOL-NAME
R1.  REPEAT
     CALLNAT 'NOPUSY7N' RC  ...

     DECIDE ON FIRST VALUE OF RC
     VALUE 0, 20 IGNORE
     VALUE 1 ESCAPE BOTTOM (R1.)
     NONE VALUE
            /* Error handling
     END-DECIDE
            /* process symbol here
     IF RC NE 20
          RESET VALUE
     END-IF
END-REPEAT
```

NOPUVI2N: Obtain Entire Operations Version Information

You can use this API to find out the version of Entire Operations. Use the following statement:

```
CALLNAT 'NOPUVI2N'
    P-FUNCTION P-VERSION P-VERSION-DATE P-UPDATE-DATE
```

Parameter Description

Parameter	Format/Length	Use	
P-FUNCTION	A1	in	Function code:
			V Version information.
P-VERSION	A20 BY VALUE RESULT	out	Version, in the format *vv.rr.ss.pppp*: see ***Format for Product Versions***.
P-VERSION-DATE	A8	out	Date of version. Format: YYYYMMDD.
P-UPDATE-DATE	A8	out	Date of last updates. Format: YYYYMMDD.

Format for Product Versions

A product version in the format *vv.rr.ss.pppp* denotes the following:

vv	Major version.
rr	Minor version.
ss	Service pack.
pppp	Fix.

Leading zeros are omitted, for example: 5.5.1.1.

Note: For further information on product versions, see *Version* in the *Glossary* of the *Natural* documentation.

NOPUXD1N: Maintain End-of-Job User Exits for a Network

You can maintain End-of-Job (EJA) definitions of a specific activated network using the following call from your Natural application.

```
CALLNAT 'NOPUXD1N' UXD-PARAMETER-AREA
```

The parameters have the following meaning:

Parameter	Format/Length	Use		
UXD-FUNCTION	A1		A	Add.
			M	Modify.
			D	Delete.
UXD-RC	N4		Return code:	
			0	Function OK.
			1	Input object not found.
			2	Wrong job type.
			3	Not a calling job.
			4	Call place not supported.
			5	Invalid exit mode.
			6	Run number not in range.
			7	Exit is already defined. **Note:** For UXD-FUNCTION A only.
			101	Invalid function code.
			102	Parameter(s) missing.
			121	Owner does not exist.
UXD-DBENV	A10		Reserved for future use.	
UXD-OWNER	A10		Reserved for future use.	
UXD-NETWORK	A10		Reserved for future use.	
UXD-RUN	I4		Reserved for future use.	
UXD-JOB	A10		Reserved for future use.	
UXD-EVENT-TYPE	A3		JOK	Job OK.
			JNO	Job not OK.
			UEX	No influence.
UXD-EVENT-NAME	A30	in	Selected event. If empty, a new additional job-ok or job-not-ok event will be added.	
		out	Event name used.	
UXD-EXIT-CALL-PLACE	A3		EJA End-of-Job action.	
UXD-EXIT-LIBRARY	A8		Reserved for future use.	
UXD-USEREXIT	A8		Reserved for future use.	
UXD-EXIT-MODE	A1		blank	Synchronous execution.
			A	Asynchronous execution.

NOPUXI0N: Add Input Condition to an Activated Job

You can add an input condition to an activated job using the following call from your Natural application.

```
CALLNAT 'NOPUXION'
  P-OWNER
  P-NETWORK
  P-JOB
  P-RUN-NUMBER
  P-RC
  P-CONDITION
  P-NATURAL-LIBRARY
  P-NATURAL-SUBPROGRAM
```

The parameters have the following meaning:

Parameter	Format/Length	Use		
P-OWNER	A10	in	Owner name.	
P-NETWORK	A10	in	Network name.	
P-JOB	A10	in	Job name.	
P-RUN-NUMBER	I4	in	Run number.	
P-RC	N3		1	Activated job does not exist.
			2	Maximum number of existing conditions.
			3	Global condition only allowed for owner SYSDBA.
			4	Invalid parameters for condition passed.
			5	Condition already exists.
			99	Natural runtime error.
P-CONDITION	A20	in	Name of the new condition.	
P-NATURAL-LIBRARY	A8	in	Name of exit library.	
P-NATURAL-SUBPROGRAM	A8	in	Name of exit subprogram.	

XVI User Exits

60 User Exits

- What User Exits Can Do 804
- Global User Exits 805
- Front-end User Exits 805
- Common User Exit Parameter Data Area NOPXPL-A 805
- Generation of Dynamic JCL and SYSOUT File Names (BS2000 only) 812
- User Exits for Resource Master Determination 812
- User Exits for Setting Input Conditions 812
- User Exits for End-of-Job Checking and Actions 813
- User Exits for Symbol Functions 815
- User Exits for Validation Checks of Symbol Values 818
- Reserved Symbols for UNIX and Windows Environment Variables 822

User exits (user routines) must be coded as Natural subprograms and are automatically invoked by the Entire Operations Monitor according to where they are defined.

Entire Operation user exits require information specified in the Natural parameter data area NOPXPL-A.

When implementing Entire Operation user exits, you use the Entire Operations editor (see also ***Editing Master JCL and Natural Sources*** in the section *Job Maintenance*) and must obey the ***User Exit Coding Restrictions***.

What User Exits Can Do

User exits can be used to check job results and influence job flow by their return codes. Apart from running checks, however, the use of Entire System Server and Adabas technology allows user exits to perform various actions such as

- Database updates;
- File access and handling;
- Access to operating system functions;
- Validation checks.

User exits for the Monitor can perform any Adabas or Entire System Server function, but must not perform any type of screen I/O.

Entire Operations checks the existence of a user exit while it is being defined, as well as at invocation time. Entire Operations treats the non-existence of a user exit as an error. When such an error occurs at invocation time or during job processing, the job network execution is interrupted.

User exits can affect the Entire Operations job flow by sending a return code or other information back to the caller after execution, depending on the type and result of the routine.

Notes:

1. User exits are invoked by the Entire Operations Monitor, which performs them as subroutines. Do not forget that intensive use of user exits can influence other work to be performed by the Monitor.
2. For user exit coding, the **user exit coding restrictions** must be adhered to.

The following section describes the different types of user exit in more detail.

Global User Exits

Global user exits can be used in your whole Entire Operations environment. For more information, see the following sections in *Global User Exits* in the *Administration* documentation:

- *Global Exit for Version Names*
- *Global JCL Activation Exit*
- *Global Symbol Modification Exit*
- *Global Symbol Not Found Exit*
- *Global Message Sending Exit*

Front-end User Exits

Front-end user exits are coded as Natural subprograms and are invoked by the Entire Operations front end for specific processing.

They can use any user-defined map with the following restrictions:

- The map (Natural object type M) must be accessible from the SYSEOR system library. The easiest way is to copy the required maps into the SYSEOR system library.
- Note that Entire Operations can be executed with the following Natural language codes: 1 = English; 2 = German. So it could be useful to supply two maps and to refer to them with the ampersand & as a placeholder representing the current language code. For more information on this technique, see the section *Multilingual User Interfaces* in the Natural *Programming Guide*.

You can define the following front-end user exit:

- Symbol prompting for a network or job activation. For more information, see ***Symbol Prompting during Network or Job Activation*** in the section ***Symbol Table and Symbol Maintenance***.

Common User Exit Parameter Data Area NOPXPL-A

All user exits under Entire Operations must use the common parameter data area NOPXPL-A, which is available in both object and source format in the SYSEOR library.

For this reason, the initial coding of a user exit must always resemble:

```
DEFINE DATA PARAMETER USING NOPXPL-A
```

This section covers the following topics:

- User Exit Types, P-CALL-PLACE
- Parameters Used for Different Call Places
- Additional Independent (AIV) Variables
- Predefined Symbols
- Common Considerations for User Exits
- Decimal Character (DC) Setting
- User Exit Coding Restrictions
- User Exit Execution

User Exit Types, P-CALL-PLACE

The field `P-CALL-PLACE` in the parameter list shows for which purpose the user exit was called:

P-CALL-PLACE	Description
EJA	End-of-Job Action See also ***Defining Action User Exits*** in the section *Defining and Managing End-of-Job (EOJ) Checking and Actions*.
EJC	End-of-Job Checking See also ***Defining Action User Exits*** in the section *Defining and Managing End-of-Job (EOJ) Checking and Actions*.
FSB	SYSOUT File Name Generation (BS2000) See also ***NOPFB2-N: Generate SYSOUT File Names for BS2000*** in the section *API Routines*.
ICO	Input Condition Value Determination See also ***Input Condition with User Exit*** in the section *Job Maintenance*.
MAC	Dynamic JCL Generation See also ***Dynamic JCL Generation (JCL Location MAC)*** in the section *Job Maintenance*.
NAT	Natural Program Standalone under Entire Operations (NAT-type Job) See also ***Job Types*** in the section *System Overview*.
NVN	Network version name, syntax check See also *Global Exit for Version Names* in the *Administration* documentation.
RMD	Resource Master Determination Exit See also *Using a Resource Master Determination Exit* in the *Administration* documentation.
SFX	Symbol Function Exit See ***User Exits for Symbol Functions***.

P-CALL-PLACE	Description
SNF	Global Symbol not Found Exit See *Global Symbol not Found Exit* in the *Administration* documentation.
SVN	Symbol table version name, syntax check See *Global Exit for Version Names* in the *Administration* documentation.
SYC	Symbol Validation Check See ***User Exit for Validating Symbol Values***.
SYF	**Symbol Function** See ***User Exits for Symbol Functions***.

Parameters Used for Different Call Places

Field Name	Format	Call Place												
P-CALL-PLACE	A3	EJA	EJC	FSB	ICO	MAC	NAT	NVN	RMD	SFX	SNF	SVN	SYC	SYF
P-RC (Return code)	N4	out	out	out (15)	out	n/a	out	out	out	out	out	out	out	out
P-RT (Return text)	A66	out	out	out (15)	out	n/a	out	out	out	out	out	out	out	out
P-OWNER	A10	in	in	in	in	in	in	in	in	in	in	in	in	in
P-NETWORK	A10	in	in	in	in	in	in	in	in	in	in	in	(3)	in
P-NETWORK-VERSION-1	A10	n/a	n/a	n/a	n/a	n/a	n/a	in	n/a	n/a	n/a	n/a	n/a	n/a
P-JOB	A10	in	in	in	in	in	in	n/a	in	in	in	n/a	n/a	in
P-RUN	P13	in	in	in	in	in	in	n/a	in	in	in	n/a	(3)	in
P-ACTIVATION-TIME	T	in	in	in	in	in	in	n/a	in (7)	n/a	n/a	n/a	(3)	n/a
P-EXECUTION-NODE (10)	N3	in	in	in	in	in	in	n/a	in	in	in	n/a	n/a	in
P-EXECUTION-OPSYS	A8	in	in	in	in	in	in	n/a	in	n/a	n/a	n/a	n/a	n/a
P-SYMTAB	A10	in	in	in	in	in	in	n/a	in	in	in	n/a	n/a	in
P-SYMTAB-VERSION-1	A10	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	in	n/a	n/a
P-CONDITION	A20	(6)	(6)	n/a	in	(2)	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
P-RESOURCE	A20	in	n/a	n/a	n/a	n/a	n/a	n/a	in	n/a	n/a	n/a	n/a	n/a
P-SYMBOL (13)	A20	(1)	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	mod	n/a	n/a	in
P-JOB-ID	A10	out	out	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
P-JOB-NUMBER (9) (obsolete)	N5	out	out	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
P-SYSOUT-FILE (11)	A54	(1)	(1)	out	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
P-FSB-OBJECT-TYPE	A5	n/a	n/a	in	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a

Field Name	Format	Call Place												
P-FSB-USERID	A8	n/a	n/a	in	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
P-FSB-CATID	A4	n/a	n/a	in	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
P-FSB-SUFFIX	A2	n/a	n/a	in	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
P-SYMBOL-VALUE [(14)]	A54	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	out	n/a	n/a	in
P-RMD-OLD-QTY-INITIAL	N7.2	n/a	n/a	n/a	n/a	n/a	n/a	n/a	in	n/a	n/a	n/a	n/a	n/a
P-RMD-NEW-QTY-INITIAL	N7.2	n/a	n/a	n/a	n/a	n/a	n/a	n/a	out	n/a	n/a	n/a	n/a	n/a
P-RMD-PARAMETER	A36	n/a	n/a	n/a	n/a	n/a	n/a	n/a	in	n/a	n/a	n/a	n/a	n/a

where:

n/a = Parameter can contain invalid values, do not modify.

in = Parameter contains input values.

out = Parameter contains output values.

mod = Parameter modifications are saved.

Notes:

[1] in; BS2000 only

[2] in; contains `P-SUFFIX` in first 10 bytes

[3] in; only for active symbols

[6] in; contains `P-JOB-ID` (A5) in first 5 bytes

[7] in; here: timestamp of previous exit call

[9] This field is obsolete. Use `P-JOB-ID` instead. The size of `P-JOB-NUMBER` is too small for some operating systems. For compatibility reasons, the field will be filled on return, if the job number is in the range of 1 to 99999.

[10] Node numbers from 1 thru 999 are provided here. For node numbers greater than 999, P-EXECUTION-NODE is zero. Node numbers of any size are provided in the AIV variable `+P-EXEC-NODE-I4` (I4).

[11] For longer file names, you must use the AIV variable `+P-SYSOUT-FILE-250` (A250).

[13] This field is only maintained for compatibility reasons. Parameter modifications using the internal A40 format/length can truncate values thus causing data loss.

[14] For returned symbol values longer than 54 bytes (A54), the AIV variable `+P-SYMBOL-VALUE-250 (A250)` must be used.

[15] For the fields `P-RC` and `P-RT` with `P-CALL-PLACE` set to `ICO`: see ***Input Condition with User Exit*** in the section *Job Maintenance*.

Additional Independent (AIV) Variables

In addition to the parameter list NOPXPL-A, some parameters are provided as independent (AIV) variables.

Field Name	Format	Call Place												
		EJA	EJC	FSB	ICO	MAC	NAT	NVN	RMD	SFX	SNF	SVN	SYC	SYF
+P-NETWORK-VERSION	A10	in	in	in	in	in	in	in	in	in	in	in	n/a	in
+P-SYMTAB-VERSION	A10	in	in	in	in	in	in	n/a	in	in	in	n/a	n/a	in
+P-EXEC-NODE-I4	I4	in	in	in	in	in	in	n/a	in	in	in	n/a	in	in
+P-SYSOUT-FILE-250	A250	in	in	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
+P-SYMBOL-VALUE-250	A250	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	out	n/a	n/a	n/a	n/a
+P-JI-JOB-TYPE	A3	in	in	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
+P-JI-DUMMY-FLAG	A1	in	in	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a

Field Descriptions

+P-NETWORK-VERSION	Version of the network.	
+P-SYMTAB-VERSION	Version of the symbol table.	
+P-EXEC-NODE-I4	This field provides node numbers of any size (1 thru 99900). It is provided for each node number. You must use it instead of `P-EXECUTION-NODE`, if the node number is greater than 999.	
+P-SYSOUT-FILE-250	This field provides the SYSOUT file name in a field with a length of 250 bytes. You must use this field for the name of the SYSOUT file for a UNIX or Windows job (instead of `P-SYSOUT-FILE`) to avoid truncation.	
+P-SYMBOL-VALUE-250	A Symbol Function result value must be returned in this field if it exceeds the size of P-SYMBOL-VALUE (A54). The effective maximum size for single and multiple symbols is 120 (A120).	
+P-JI-JOB-TYPE	The job type (A3). For valid values, see the three-letter codes in ***Available Job Types*** in the section *Job Maintenance*.	
+P-JI-DUMMY-FLAG	Reason why a job became a temporary dummy job (dummy due to …).	
	`J`	JCL check.
	`D`	Definition.
	`S`	Schedule.
	`R`	Recovery.
	`T`	Repetition.
	`E`	Empty JCL.
	`C`	Condition.
	`M`	Multiple suffix.
	`K`	To be deactivated.

Predefined Symbols

For a list of predefined symbols, see ***Predefined Symbols*** in the section ***Symbol Table and Symbol Maintenance***.

Common Considerations for User Exits

All user exits must return the field `P-RC` (return code). 0 (zero) means OK.

The field `P-RT` (return text) can be used optionally to pass back some specific information. Entire Operations writes this text into the log and displays it on the **Active Jobs** screen, if supplied.

All other parameters pass environment information to the user exit. Some **parameters** and **AIV variables** are not always required; see the relevant field descriptions for details.

Decimal Character (DC) Setting

The parameter data area `NOPXPL-A` contains numbers with decimal characters. The default setting for decimal character in the delivered source of `NOPXPL-A` is a period (`.`). If you want to catalog your user exits and Natural programs with another decimal character (e.g. comma, `DC=,`), you must also modify `NOPXPL-A`:

- Make sure that the decimal character of your Natural session is set correctly. You can verify this with the system command `GLOBALS` (described in the Natural *System Commands* documentation).
- Use the Natural Editor to edit `NOPXPL-A`.
- Change the decimal character to your current DC parameter setting in the numeric fields, which contain a decimal character. Stow `NOPXPL-A`.

You will now be able to use `NOPXPL-A` with your own decimal character.

User Exit Coding Restrictions

1. User exits must reside in a Natural library on the FUSER system file but may not reside in the Entire Operations SYSEOR system library.

   Sub-objects invoked by user exits (subprograms, etc.) and global user exits (see the *Administration* documentation) and must reside in the SYSEORU system library.
2. `ON ERROR` routines are allowed in Entire Operations user exits, with the following restriction: The `ON ERROR` block must not be left by any `ESCAPE ROUTINE, ESCAPE MODULE, TERMINATE` or similar statement. The purpose of this restriction is that the common Entire Operations Monitor (or Entire Operations online) error transaction gets control after the execution of the user-defined `ON ERROR` block. The common Entire Operations error transaction performs some common error handling, logging, notifications etc.
3. No programs with their own GDA (global data area) may be invoked.

4. The names of user exits must not begin with the letter `V`, particularly if they are to be stored in the library SYSEOR, because the prefix `V` is reserved for the internal Entire Operations exit directory.
5. The following Natural statements must *not* be coded in any user exit invoked by Entire Operations:
   - `END TRANSACTION`

     The transaction logic is handled by the Entire Operations Monitor itself.
   - `BACKOUT TRANSACTION`
   - `STOP`
   - `TERMINATE`
   - `FETCH` without `RETURN` (`FETCH RETURN` is allowed)
6. Entire System Server node logons and logoffs should be avoided. This applies for mainframe nodes (view NATPROC-LOGON), UNIX nodes, and Windows nodes. If any logons and/or logoffs to Entire System Server nodes were made in a user exit, the exit must invoke the Entire System Server node invalidation **API NOPUNI1N** prior to termination. This forces subsequent new logons to these nodes.

   Note that no logon and no logoff is required
   - to the job's JCL node if the JCL is managed by the exit;
   - to the job's execution node if the exit is managing the job's execution, end-of-job checking or end-of-job actions.

   In these cases, implicit logon's were made already by the exit caller.
7. Do not code any screen and map input/output in user exits. An exception is the symbol prompting user exit in online usage mode. See also ***Specifying User Exits for Symbol Modification*** in the section ***Symbol Table and Symbol Maintenance***.
8. In general, the execution of all kinds of code, which can disturb the Entire Operations Monitor execution, is forbidden in user exits (exits). If in doubt, contact technical support.



Notes:

1. If you want to use the values of the special parameters in Macro JCL, you must first read them with the special statement **#GET-SYMBOL** (described in the section *System Overview*).
2. See also ***User Exits for Validation Checks of Symbol Values***. For foreground usage, this exit may use maps (with some restrictions).

User Exit Execution

If a user exit fails due to a Natural runtime error (e. g. a NAT0082 error if an invoked module does not exist), the user exit's Natural object will be deleted by the Entire Operations (NOP) Monitor. This prevents a Monitor failure due to the repeated execution attempt of an erroneous user exit. The event will be logged, and the active job's status text will be set accordingly. In this case, the user exit's source must be corrected, and the user exit must be stowed (cataloged) again.

Generation of Dynamic JCL and SYSOUT File Names (BS2000 only)

For MAC-type jobs, you can generate dynamic JCL: see ***Dynamic JCL Generation (JCL Location MAC)*** in the section *Job Maintenance*.

In a BS2000 environment, you can generate file names for SYSOUT: see ***NOPFB2-N: Generate SYSOUT File Names for BS2000*** in the section *API Routines*.

User Exits for Resource Master Determination

For each resource master, you can define a resource amount determination exit: see *Using a Resource Master Determination Exit* in the *Administration* documentation.

User Exits for Setting Input Conditions

Before Entire Operations can submit a job, all its input conditions must be fulfilled.

If a user exit is defined for a condition, Entire Operations invokes the defined exit (of the type **ICO**) each time it checks the condition status. See ***Return Code Settings for an Input Condition User Exit*** for the meaning of the return codes.

Otherwise, the condition checking will be continued.

You can set a return text in the `P-RT` parameter additionally.

User Exits for End-of-Job Checking and Actions

You can define user exits as part of End-of-Job checking. Entire Operations bases its End-of-Job checking on the occurrence of defined events during job execution. You can define a user exit as an event check. Entire Operations automatically executes the user exit when the job terminates.

Typically, such a user exit routine can check the result of a job, for example. The user exit returns a code which determines the event status. A return code zero means `Check OK`, any other return code means `Check not OK`. Condition codes returned by user exits can determine End-of-Job status.

End-of-Job action user exits can be used to perform any actions after job termination.

End-of-Job checking and action user exits are supplied with both the job number and the name of the SYSOUT file in which Entire Operations collected the job SYSOUT (BS2000 only).

For End-of-Job user exits, the common exit parameter list **NOPXPL-A** must be used. The parameter `P-CALL-PLACE` contains `EJC` for End-of-Job checking user exits or `EJA` for End-of-Job action user exits.

Related Topic:

- *Defining Action User Exits* in the section *Defining and Managing End-of-Job (EOJ) Checking and Actions*

This section covers the following topics:

- Example

Example

An End-of-Job checking user exit is shown in the example below.

This user exit routine adds up the contents of a numeric field in the SYSOUT and compares it with a given value. If the sum of the contents equals the sum of the given value, the event is treated as `ok`, otherwise as `not ok`.

```
DEFINE DATA PARAMETER USING NOPXPL-A
LOCAL
1 READ-SPOOL VIEW OF READ-SPOOL      /* AN ENTIRE SYSTEM SERVER VIEW
  2 RECORD
  2 REDEFINE RECORD
    3 RECORD-HEADER    (A05)
*
1 #NF              (I02) CONST <10>   /* MAX. NUMBER FOR SEPARATION
```

```
1 #SUM             (N08) INIT  <0>
1 #SUM-EXPECTED    (N08) CONST <2000>
1 #FIELD           (A10/1:#NF)
1 #USERID          (A08)
1 #DBENV           (A10)
1 #NETWORK-VERSION (A10)

1 #LOG-STATUS      (N04)
1 #LOG-MESSAGE     (A60)
1 #LOG-TARGET      (A03) CONST <'NOP'>
1 #USE-NAT-MSG-CACHE (L) CONST <FALSE>
END-DEFINE
```

```
* ------------------------------------------------
* This EOJ User Exit will read the Job SYSOUT (data set 2)
* and will summarize a defined field.
* If the sum is equal to a given value, this routine returns
* an 'OK' (P-RC = 0), otherwise 'NOT OK'.
*
* A log record will be written to document the result.
*
RESET #SUM
RS. FIND READ-SPOOL
    WITH NODE = P-EXECUTION-NODE AND JOB-NUMBER = P-JOB-ID
    AND TYPE = 'SO' AND DATA-SET = 2 AND RECORD-NUMBER >= 1
  ACCEPT IF RS.RECORD-HEADER = 'REC01'     /* interesting only
  SEPARATE RS.RECORD LEFT JUSTIFIED INTO #FIELD(*)
  ADD VAL(#FIELD(3)) TO #SUM
END-FIND
*
IF #SUM = #SUM-EXPECTED
  RESET P-RC
  MOVE 'Sum check is ok' TO #LOG-MESSAGE
ELSE
  MOVE 1 TO P-RC
  COMPRESS 'Sum:' #SUM 'Expected:' #SUM-EXPECTED INTO #LOG-MESSAGE
END-IF
#USERID := *USER
MOVE 9999 TO #LOG-STATUS
CALLNAT 'NOPULW93N'          /* WRITE LOG RECORD
  #LOG-STATUS #LOG-MESSAGE #DBENV
  P-OWNER P-NETWORK #NETWORK-VERSION P-RUN P-JOB
  P-JOB-ID #LOG-TARGET # USE-NAT-MSG-CACHE
END
```

User Exits for Symbol Functions

You can perform user-specific symbol functions via a user exit. For the symbol functions available, see *Functions for Symbol Replacement* in the section *Symbol Table and Symbol Maintenance*.

- Symbol Function Syntax
- Symbol Function Coding Rules
- Symbol Function Parameter List
- Symbol Function Usage and Execution
- Example of a Symbol Function

Symbol Function Syntax

Syntax for symbol functions:

```
§!|?function[parm,...]
```

Symbol Function Coding Rules

- As for other user exits (exits), the **user exit coding restrictions** must be adhered to.
- For a function to be recognized, an exclamation mark (`!`) or a question mark (`?`) must always follow the escape character. In BS2000 the exclamation mark (`!`) is represented by an `ö` character.
- The parameter list is optional and is enclosed in square brackets (`[ ]`). No spaces are allowed between function name and parameter list.
- Parameters must be separated by commas.
- The total length of the function call may not be longer than 20.
- The symbol functions must be located in the SYSEORU library.
- The Entire Operations Monitor performs logging of symbol replacements. Additional logging may also be performed.
- After successful replacement, the symbol value is written to the active symbol table of the job. The symbol function is not called for subsequent replacements. Instead, the value is read from the active table.

The names of the symbol function exits begin with `SX`. Since the length of the names of Natural objects is limited to 8 characters, the function name (the exclamation mark (`!`) not counted) can have a maximum length of 6 characters.

Example:

The function `!RANDOM` is handled by the Natural subprogram SXRANDOM.

Symbol Function Parameter List

The symbol function exits are called with the parameter list **NOPXPL-A**, so the first line of the exit should be:

```
DEFINE DATA PARAMETER USING NOPXPL-A
```

Notes:

1. Caller supplied `SFX` (symbol function exit) for `P-CALL-PLACE`.
2. Set `P-RC` to either of the following return codes:

   `0` (Function ok.), or

   `4513` (EOR4513 - Symbol :1: not found.)
3. `P-SYMBOL` only transfers the function parameters (without brackets). The function names can be identified from the Natural variable `*PROGRAM` (described in the Natural *System Variables* documentation).

Symbol Function Usage and Execution

- Symbol functions can also be used for **input conditions that depend on a symbol value**. Please keep in mind the overhead of the repeated execution until the input condition is satisfied.
- Depending on the **escape character** being used, symbol functions are executed at JCL loading time, or at job submission time. If a symbol function fails due to a runtime error, or if its Natural object is not available, the affected active job will be set to a permanent error state.

Example of a Symbol Function

User Exit Subprogram SXQS

```
* SYMBOL FUNCTION EXIT EXAMPLE
* SXQS<PARM1>
* CROSSFOOTING OF THE PARAMETER
* ------------------------------------------------------------
DEFINE DATA PARAMETER USING NOPXPL-A
LOCAL
1 #I   (I02)
1 #N   (I02)
1 #SUM (I02)
1 #A   (A01)
END-DEFINE
* ------------------------------------------------------------
RESET #SUM
F1. FOR #I = 1 TO 20
```

```
  #A := SUBSTR (P-SYMBOL,#I,1)
  IF #A IS (N1)
    #N := VAL(#A)
    ADD #N TO #SUM
  END-IF
END-FOR    /* F1.
P-SYMBOL-VALUE := #SUM
RESET P-RC P-RT
END
```

Master JCL

```
//SN000001 JOB ,SN,CLASS=K
//IEFBR14  EXEC PGM=IEFBR14
//*
//* +!D<AQ+1>
//* §!D<AQ+1>
//*
//* +!QS<4711>
//* §!QS<4711>
//*
//* +!QS<+*DATE>                NESTED SYMBOLS
//* +!QS<§*DATE>                NESTED SYMBOLS
//* §!QS<§*DATE>                NESTED SYMBOLS
//*
//* +!QS<+!D<AM-1>>             NESTED FUNCTIONS
//* +!QS<§!D<AM-1>>             NESTED FUNCTIONS
//* §!QS<§!D<AM-1>>             NESTED FUNCTIONS
//*
```

Active JCL

```
//SN000001 JOB ,SN,CLASS=K
//IEFBR14  EXEC PGM=IEFBR14
//*
//* +!D<AQ+1>
//* 20080101
//*
//* +!QS<4711>
//* 13
//*
//* +!QS<+*DATE>                NESTED SYMBOLS
//* +!QS<29/10/08>              NESTED SYMBOLS
//* 18                          NESTED SYMBOLS
//*
//* +!QS<+!D<AM-1>>             NESTED FUNCTIONS
//* +!QS<20080131>              NESTED FUNCTIONS
//* 30                          NESTED FUNCTIONS
//*
```

Active Symbol Table

```
Symbol            F A Value                          modified by
!D<AM-1>          A E 20080131                       EORMON   11.01.08 17:25
!D<AQ+1>          A E 20080101                       EORMON   11.01.08 17:25
!QS<11/01/08>     A E 20                             EORMON   11.01.08 17:25
!QS<20080131>     A E 15                             EORMON   11.01.08 17:25
!QS<4711>         A E 13                             EORMON   11.01.08 17:25
X                 A E x                              SN       10.11.08 13:13
```

User Exits for Validation Checks of Symbol Values

You can write a user exit routine (**front-end user exit**) that defines your own prompting screens and validation (plausibility) checks for symbol prompting during job network activation. For more information on symbol prompting, see ***Symbol Prompting during Network or Job Activation*** in the section ***Symbol Table and Symbol Maintenance***.

The user exit is called by the Entire Operations Monitor after the creation of the active symbol tables, but before the JCL loading. If the user exit is missing or receives a run time error, network activation is stopped and a message is sent to the mailbox, which is assigned to the network as the message recipient. If no mailbox is assigned, the message is sent to the SYSDBA mailbox. For details, see the section ***Mailboxes***.

When writing the user exit routine, use the mandatory parameter data area NOPXPL-A and the parameter list **NOPSYP3A**.

NOPXPL-A contains the owner, symbol table, symbol name, etc. It can therefore be used for different symbols.

NOPSYP3A contains all needed environment parameters. The list of symbol tables used for this network activation will be passed to the Entire Operations API routine.

The symbols must be read and updated by the Entire Operations API routine NOPUSY*x*N which allows sequential reading in the active symbol table. *x* denotes a sequential version number of 1 through 6, where 6 is the most recent version of the API.

The parameters `P-RC` (return code) and `P-RT` (return text) are examined by the caller after execution of the user exit. If `P-RC` is zero, the symbol is accepted as OK. Otherwise, the symbol is not accepted.

If the text from `P-RT` is not blank, it will be displayed to the user with message code EOR1855. If `P-RT` is not zero and blank, a standard error message is displayed.

As in other user exits, Adabas, Entire System Server and Natural system variables can provide a lot of flexibility to symbol checking.

This section covers the following topic:

- Parameter List NOPSYP3A
- Symbol Modification without Prompting

Parameter List NOPSYP3A

```
*    NOPSYP3A
*    Entire Operations
*    Exit parameter list
*    for symbol prompting
*
*    27.09.12 N1399    SN541                      /* successor of NOPSYP2A
*    11.03.13 N1574    SN541                      /* N1574
*    20.08.13 N1649    SN541                      /* P-USED-SYMTAB-34
*    --------------------------------
   1 P-CALL-PLACE                    A          3
*      'SYP'  symbol prompting online
*      'SYM'  symbol modif.    batch
   1 P-RC                            N          4 /* in/out  return code
*      0  out: ok
*      1  out: no symbols prompted
*      2  out: activation cancelled
*      4  out: end (no modification)
*
*      3  in:  rewrite modified
*              symbols to symbol
*              master too
   1 P-RT                            A         70 /* out  return text
*
   1 P-OWNER                         A         10 /* in
   1 P-NETWORK                       A         10 /* in
   1 P-NETWORK-VERSION               A         10 /* in  312641 add
   1 P-RUN                           I          4 /* in  312641 fmt, order
   1 P-JOB                           A         10 /* in  312641 order
   1 P-ACTIVATION-TIME               T            /* in
   1 P-EARLIEST-START                T            /* in
   1 P-NETWORK-SYMTAB                A         10 /* in
R  1 P-NETWORK-SYMTAB                              /* REDEF. BEGIN : P-NETWORK-SYMT
   2 P-NETWORK-SYMBOL-TABLE          A         10 /* 312641.3
   1 P-NETWORK-SYMTAB-VERSION        A         10 /* 312641 add
   1 P-USED-SYMTAB-34                A         34 (1:V) /* N1649
*      -- Use LDA NOPSYT1L for the                N1649
*      -- layout / redefinition of                N1649
*      -- the table elements.                     N1649
   1 P-DATE-FORMAT                   A          1 /* in
*      'A' American
*      'E' European
*      'G' German
*      'I' International
```

```
*
*    -- end of parameter data area --
```

This section covers the following topics:

- Supplied by the Caller on Input
- Return Codes to Be Set by the Symbol Modification User Exit

Supplied by the Caller on Input

P-RC	Meaning
3	Rewrite modified symbols to the symbol table master.

Return Codes to Be Set by the Symbol Modification User Exit

P-RC	Meaning
0	OK, modifications were done.
1	OK, no symbols prompted or modified.
2	Activation cancelled.
4	End; no modification.

Symbol Modification without Prompting

Code the user exit analogous to the symbol prompt user exit and use the provided **parameter list NOPSYP3A**, in conjunction with the local data area `NOPSYT1L`. Do not code any screen I/O if you do not use foreground prompting.

This section covers the following topics:

- Data Area NOPSYT1L

Data Area NOPSYT1L

The layout of the field `P-USED-SYMTAB-34` is specified in the data area NOPSYT1L.

```
*    NOPSYT1L
*    'Symbol tables used'
*    table entry
*
*    Maintenance
*    13.08.13  N1649     SN541                    created
*    07.10.13  N1649.A   SN541                    'origin object type' val.
*    --------------------------------             -------------------------
*    -- values for SYTU-SYMTAB-STATUS
*
```

```
   1 SYTU-SYMTAB-STATUS-EV-DUP        A          2 CONST<'ED'>
*    -- effective symbol table
*    -- version is a duplicate --
*    -- entry should be skipped
*    -- during prompting
   1 SYTU-SYMTAB-STATUS-UNDEF         A          2 CONST<'00'>
*    -- undefined
   1 SYTU-SYMTAB-STATUS-NPN           A          2 CONST<'10'>
*    -- no prompting necessary
   1 SYTU-SYMTAB-STATUS-PN            A          2 CONST<'20'>
*    -- prompting necessary
   1 SYTU-SYMTAB-STATUS-PD            A          2 CONST<'30'>
*    -- prompting done
*
*    --------------------------------
*
*    -- values for
*    -- SYTU-ORIGIN-OBJECT-TYPE
*
   1 SYTU-ORIGIN-OBJECT-NV            A          2 CONST<'NV'>
*    -- network version definition
   1 SYTU-ORIGIN-OBJECT-JM            A          2 CONST<'JM'>
*    -- job (master) definition
   1 SYTU-ORIGIN-OBJECT-IM            A          2 CONST<'IM'>
*    -- input cond. (mult. suffix)
   1 SYTU-ORIGIN-OBJECT-IS            A          2 CONST<'IS'>
*    -- input cond. (dep. on symbol)
   1 SYTU-ORIGIN-OBJECT-AS            A          2 CONST<'AS'>
*    -- EOJ action (symbol setting)
*
*    --------------------------------
*
   1 SYTU-ENTRY-34                    A         34
R  1 SYTU-ENTRY-34                                 /* REDEF. BEGIN : SYTU-ENTRY-34
   2 SYTU-SYMTAB-STATUS               A          2
*    -- values see above
   2 SYTU-ORIGIN-OBJECT-TYPE          A          2
*    -- values see above                           /* N1649.A
*    ...
   2 SYTU-SYMTAB                      A         10
   2 SYTU-SYMTAB-VERSION-DEFINED      A         10
   2 SYTU-SYMTAB-VERSION-EFFECTIVE    A         10
*      blank:
*        not determined yet
*      KVE-TEXT-UNNAMED = '(none)'
*        determined and blank
*      other:
*        determined and non-blank
*
*    -- end of area --
```

Reserved Symbols for UNIX and Windows Environment Variables

A set of reserved symbols can be used to set environment variables in the JCL frame script (UNIX) or in the JCL frame BAT file (Windows) for jobs submitted on UNIX or Windows nodes.

Prerequisite: The symbols must be defined in the Natural text object NOPVS001 supplied in the SYSEORU system library.

- Special Usage Considerations
- Example of Using Reserved Symbols for Environment Variables

Special Usage Considerations

Consider the following when using reserved symbols as environment variables:

- For valid format/length specifications of the reserved symbols, see ***Table of Predefined Symbols*** in the section *Symbol Table and Symbol Maintenance*.
- Dash signs (-) within a symbol name are converted to underscore signs (_) as indicated in the following example.
- For UNIX:

  Environment variables are exported. Therefore, they are available in all subscripts and executables called from the frame script.
- For Windows:

  Environment variables are set in the frame BAT file. They are available in the user BAT or PS1 file.

Example of Using Reserved Symbols for Environment Variables

The following is an example of using reserved symbols to set environment variables.

Variables defined in the NOPVS001 text object:

```
P-SYSF1
P-OWNER
P-NETWORK
P-NETWORK-VERSION
P-RUN
P-RUN5
P-JOB
P-REPEAT
P-ESC-ACT
P-ESC-SUB
P-SYMTAB
```

```
P-SYMTAB-VERSION
P-JCL-NODE
P-EXEC-NODE
P-SUL
P-SUG
P-NADIR
P-SYSOUT
P-DATE
P-C-OWNER
P-C-NETWORK
P-C-NETWORK-VERSION
P-C-RUN
P-C-RUN5
P-C-JOB
```

Excerpt of the generated JCL frame script for UNIX:

```
...
# setting of variables defined in SYSEORU/NOPVS001
P_SYSF1="0000900017" ; export P_SYSF1
P_OWNER="NOPALL" ; export P_OWNER
P_NETWORK="N2838S02" ; export P_NETWORK
P_NETWORK_VERSION="" ; export P_NETWORK_VERSION
P_RUN="6" ; export P_RUN
P_RUN5="00006" ; export P_RUN5
P_JOB="J101" ; export P_JOB
P_REPEAT="0" ; export P_REPEAT
P_ESC_ACT="@" ; export P-ESC-ACT
P_ESC_SUB="^" ; export P-ESC-SUB
P_SYMTAB="" ; export P_SYMTAB
P_SYMTAB_VERSION="" ; export P_SYMTAB_VERSION
P_JCL_NODE="517" ; export P_JCL_NODE
P_EXEC_NODE="517" ; export P_EXEC_NODE
P_SUL="sag" ; export P_SUL
P_SUG="" ; export P_SUG
P_NADIR="$EOR_WORK/sag/0000900017/NOPALL/N2838S02/0000/0006"
export P_NADIR
P_SYSOUT="$EOR_WORK/sag/0000900017/NOPALL/N2838S02/0000/0006/J101.00000000.sysout.txt"
export P_SYSOUT
P_DATE="20161219" ; export P_DATE
P_C_OWNER="NOPALL" ; export P_C_OWNER
P_C_NETWORK="N2838M02" ; export P_C_NETWORK
P_C_NETWORK_VERSION="" ; export P_C_NETWORK_VERSION
P_C_RUN="7" ; export P_C_RUN
P_C_RUN5="00007" ; export P_C_RUN5
P_C_JOB="S002" ; export P_C_JOB
# setting of variables - end
...
```

Excerpt of the generated JCL frame BAT file for Windows:

```
...
rem - setting of variables defined in SYSEORU/NOPVS001
set P_SYSF1=0000900018
set P_OWNER=SN
set P_NETWORK=W401-2
set P_NETWORK_VERSION=
set P_RUN=2382
set P_RUN5=02382
set P_JOB=J001
set P_REPEAT=0
set P_ESC_ACT=@
set P_ESC_SUB=$
set P_SYMTAB=W401-2
set P_SYMTAB_VERSION=
set P_JCL_NODE=401
set P_EXEC_NODE=401
set P_SUL=sn
set P_SGL=eur
set P_NADIR=c:\SAG\eor_work\sn\0000900018\SN\W401-2\0000\2382
set ↩
P_SYSOUT=c:\SAG\eor_work\sn\0000900018\SN\W401-2\0000\2382\J001.00000000.sysout.txt
set P_DATE=20180503
set P_C_OWNER=
set P_C_NETWORK=
set P_C_NETWORK_VERSION=
set P_C_RUN=0
set P_C_RUN5=00000
set P_C_JOB=
rem - setting of variables - end
...
```