

English



FUJITSU Software BS2000

AVAS / AVAS-SV-BS2 V8.5A

AVAS for the Administrator

System Administrator Guide

Edition June 2017

Comments... Suggestions... Corrections...

The User Documentation Department would like to know your opinion on this manual. Your feedback helps us to optimize our documentation to suit your individual needs.

Feel free to send us your comments by e-mail to:

manuals@ts.fujitsu.com

Certified documentation according to DIN EN ISO 9001:2008

To ensure a consistently high quality standard and user-friendliness, this documentation was created to meet the regulations of a quality management system which complies with the requirements of the standard DIN EN ISO 9001:2008.

cognitas. Gesellschaft für Technik-Dokumentation mbH

www.cognitas.de

Copyright and Trademarks

Copyright © 2017 Fujitsu Technology Solutions GmbH.

All rights reserved.

Delivery subject to availability; right of technical modifications reserved.

All hardware and software names used are trademarks of their respective manufacturers.

Contents

1	Preface	11
1.1	Objectives and target groups of this manual	12
1.2	Structure of the AVAS documentation	13
1.3	Changes since the last edition of the manual	14
1.4	Notational conventions	15
1.5	Licensing regulations	16
2	Installation and generation	21
2.1	Installing the AVAS system	21
2.1.1	AVAS-SV-BS2	21
2.1.2	AVAS-QUER	21
2.2	Generation of the AVAS system	23
2.2.1	Password encryption	24
2.2.2	Description of the system parameters	28
2.2.2.1	Definitions for the central access tasks	28
2.2.2.2	Definition of users	33
2.2.2.3	Definition of the system variables of the users	41
2.2.2.4	Defining the run control system	42
2.2.2.5	Defining the default values for processing parameters	44
2.2.2.6	Symbolic names of AVAS statements and variables	52
2.2.3	Modifying the system parameters of an installed AVAS system	53
2.2.4	Creating the generation parameters from the system parameters	56
2.2.4.1	Syntax of the generation statements and creation rules	56
2.2.4.2	Procedure for creating GENPAR descriptions	59
2.2.5	Creating and formatting the AVAS files	61
2.2.5.1	User files and their symbolic names	61
2.2.5.2	AVAS libraries	64
2.2.5.3	AVAS version changes with ABLDAT and JRNDAT	66

2.3	CC exits in the AVAS system	71
2.3.1	Linking CC routines in AVAS	77
2.3.2	Integration of MARENAV	79
2.3.3	Connecting CC routines along with MARENAV	79
2.3.4	Information exchange between the AVAS system and CC routines	81
2.3.4.1	Information area	81
2.3.4.2	Communication areas	83
2.3.5	CC exit AVEX0001 and AVEX0002	84
2.3.6	CC exit AVEX0101	85
2.3.7	CC exit AVEX0102	86
2.3.8	CC exit AVEX0401	87
2.3.9	CC exit AVEX0402	89
2.3.10	CC exit AVEX0403	90
2.3.11	CC exit AVEX2001	92
2.3.12	CC exit AVEX6601	93
2.3.13	CC exit AVEX6602	95
2.3.14	CC exit AVEX6801	96
2.3.15	CC exit AVEX6802	98
2.3.16	CC exit AVEX7101	100
2.3.17	CC exit AVEX7102	101
3	Administration of AVAS	105
<hr/>		
3.1	Starting and terminating AVAS tasks	106
3.1.1	Starting the system and system security	106
3.1.2	Start parameters for the dialog process	107
3.1.3	Starting and terminating the access tasks	108
3.1.4	Starting and terminating the run control system	115
3.1.5	Starting and terminating the server monitor process	123
3.1.6	Starting and terminating the server interface process	123
3.1.7	Displaying the statuses and versions of the configured AVAS servers	123
3.1.8	Displaying the active REP corrections	123
3.1.9	Processing the optional REP corrections	123
3.2	Working with and monitoring AVAS tasks	124
3.2.1	Working with the access tasks	124
3.2.2	Operating the run control system	127
3.2.2.1	Inputs for the run control system	128
3.2.2.2	Controlling released nets via /INFORM-PROGRAM	128
3.2.2.3	Updating the AVAS-internal processor table	128
3.2.2.4	Copying and reassigning the system files SYSLST and SYSOUT	129
3.2.3	Working with the CENTRAL task	130
3.2.4	Working with the SOUT task	131

3.2.5	Modifying the server environment in the server monitor process	133
3.2.6	Monitoring tasks via job variables	135
3.3	Runtime management tasks	137
3.3.1	Managing the system libraries	137
3.3.2	Administration of users	137
3.3.3	Managing the system parameter file	137
3.3.4	Management of runtime logs	138
3.3.4.1	Storing the logs	139
3.3.4.2	Collecting the logs	143
3.3.4.3	Editing the logs	154
3.3.4.4	Overview of log statuses in statements	155
3.3.5	Displaying the SYSOUT files	156
3.3.5.1	The SOUT task	156
3.3.5.2	Runtime environment of the SOUT task	157
3.3.5.3	Starting the SOUT task	158
3.3.5.4	Terminating the SOUT task	160
3.3.6	Optimizing the system	161
3.3.6.1	System optimization in batch mode	164
4	Backup and reorganization	169
4.1	Reorganizing the AVAS user files	169
4.1.1	Reorganizing the run control file and the journal file	172
4.1.2	Reorganizing the log file	173
4.1.3	Execution of the reorganization program	174
4.1.4	Statements for the reorganization program	175
4.2	Saving the journals and logs and outputting journal listings	190
	Saving journal records	190
	Outputting journal listings	191
	Saving log data	193
4.3	Data structures of the backup journal and the emergency journal file	194
	Record structure	194
	Journal records of the statements	198
	Record structure of the fixed portion of journal records	204
	Record structure of the variable-length portion of journal records	207
4.4	Structure of the ISAM journal log file	222
	Structure of the records	222
4.5	Structure of the history file	224

5	AVAS-QUER utility routine	239
5.1	Working with AVAS-QUER	241
5.1.1	Prerequisites for starting AVAS-QUER	241
5.1.2	Starting AVAS-QUER	242
5.1.3	Entering statements	247
5.1.4	Terminating AVAS-QUER	247
5.1.5	Structure of the output file(s) created by AVAS-QUER	248
5.2	AVAS-QUER statements	250
	SIGNON – Log in to AVAS system	251
	CREATE-FORMAT – Select output file format	252
	SELECT-OBJECTS – Select AVAS objects	256
5.3	Data storage for AVAS-QUER	259
5.3.1	Creating the database	259
5.3.2	Importing the data	259
5.3.3	Structure of the database	260
5.3.4	Structure of the database tables	262
5.3.4.1	Default format	262
5.3.4.2	Extended format	267
5.3.5	Sample database queries	276
5.4	Error handling	279
6	Coupling AVAS with MAREN	281
6.1	Performing volume checks	281
6.2	Entering VSNs in the JCL	285
6.2.1	Entering VSNs for input tapes	285
6.2.2	Entering VSNs for output tapes	287
6.2.3	Handling continuation lines	288
6.3	Creating volume lists	289
6.3.1	Transport list	291
6.3.2	Tape mount listing	293
6.3.3	Shift procedure	294
6.3.4	Restrictions pertaining to MARENAV	295
6.3.5	Example illustrating further processing of a VSN in a net	295
6.3.6	Notes on retry runs	296
6.3.7	Error messages from MARENAV	299
6.3.8	Warning messages from MARENAV	301

7	AVAS reports	303
7.1	REPORT generator	306
7.1.1	Entering REPORT statements for the REPORT generator	307
7.1.2	Selecting the REPORT generator functions	308
7.1.3	Allocating the production plan and the journal file	308
7.1.4	Allocating the work file for the REPORT generator	308
7.1.5	Allocating the report file for the REPORT generator	309
7.1.6	Output of the REPORT generator to the work file	309
7.1.7	Output of the REPORT generator to the report file	309
7.1.8	Allocating the input/output file for FUNKTION=SORT	309
7.2	REPORT statements	311
	AVAS-SYSTEM-ID – Select AVAS system	311
	AVAS-USER-ID – Select production plan	311
	END – Terminate statement sequence for REPORT generator	312
	FUNKTION – Select REPORT generator functions	312
	NET-NAME – Select nets using net names	313
	OUTPUT-REPORT-FILE – Assign BS2000 file name to report file	314
	OUTPUT-REPORT-LINK – Assign report file via link name	315
	OUTPUT-WORK-FILE – Assign BS2000 file name to work file	316
	OUTPUT-WORK-LINK – Assign work file via BS2000 link name	317
	PERIOD-NAME – Select nets using start date PLAN-START	318
	REPORT-NAME – Select reports	319
	SELECT-STATUS – Select nets according to status	320
	SORT-FIELDS-ORDER – Define sequence of sort fields	321
	SORT-STATE-ORDER – Define net sequence for report file	322
	SYSTEM-PASSWORD – Sign on to central access task	323
	THRESHOLD – Select nets according to delay	324
	USER-PASSWORD – Specify user password	324
7.3	PLANNED-NET-MODIFICATION report	325
7.4	OUT-OF-PLAN report	327
7.5	Procedures and libraries for the AVAS reports	332

8	Batch functions	335
8.1	Batch statements	337
	ADD-JOB-LOG – Add log data	337
	CANCEL-NET – Cancel or abort net because of error	339
	CHANGE-NET-DESCRIPTION – Make global changes to nets	340
	COPY-ELEMENT – Copy library elements	342
	CREATE-PERIOD – Create period	344
	CREATE-PLAN-NET – Plan net processing	349
	CREATE-PROD-JOB – Create static B2000 jobs and S procedures	351
	CREATE-PROD-NET – Create temporary BS2000 jobs and S procedures for net	352
	DELETE-DOCUMENT – Delete documentation elements	353
	DELETE-JOB – Delete BS2000 jobs, S procedures and JCL elements	353
	DELETE-JOB-LOG – Delete logs of specific net	354
	DELETE-NET-DESCRIPTION – Delete nets	354
	DELETE-PERIOD – Delete period	355
	DELETE-PLAN-NET – Delete planned nets from production plan	356
	DELETE-PROD-JOB – Delete static BS2000 jobs and S procedures	356
	END – Terminate batch statement stream	357
	HOLD-NET – Suspend nets currently being processed	358
	MODIFY-COND-DESCRIPTION – Modify condition description	359
	REPEAT-NET – Repeat release of planned net	362
	RESTART-NET – Restart net following error	363
	RESUME-NET – Cancel HOLD status	364
	SIGNON – Start batch statement stream	365
	SUBMIT-NET – Release planned nets	365
8.2	Procedures and libraries for the batch functions	366
8.3	Error information for batch functions	368
9	External creation of AVAS elements	369
9.1	Creating nets externally through programs	369
9.2	External creation of tasks	397
9.3	External creation of documents	397

10	AVAS program interface	399
10.1	AVAS program interface statements	399
	READ-AVAS-LIBRARY – Enable read access to journal and runtime files	402
10.2	Structure of the communication areas	406
10.3	Assembler Interface	412
10.3.1	Programming example in Assembler	413
10.3.2	Macros for defining the communication areas	415
	AVSASSAD – Define net contents directory entry for run control file	415
	AVSASSAN – Define net structure data	419
	AVSASSBC – Define communication area	426
	AVSASSBO – Define result area header	430
	AVSASSBW – Define work area	431
	AVSASSCD – Define directory entry for condition descriptions in run control file	433
	AVSASSCE – Define condition description in run control file	435
	AVSASSJD – Define journal directory	437
	AVSASSJN – Define journal record header	440
	AVSASSRT – Define result area	442
10.4	Cobol Interface	444
10.4.1	Programming example in Cobol	445
10.4.2	COPY elements for statements	447
	AVSCOBAD – Define net directory for run control file	447
	AVSCOBAN – Define net structure data	448
	AVSCOBBC – Define communication area	452
	AVSCOBBO – Define result area header	456
	AVSCOBQ – Define key for AVSCMDC and AVSERLTC	457
	AVSCOBW – Define work area	458
	AVSCOB CD – Define directory entry for condition descriptions in run control file	459
	AVSCOBCE – Define condition description in run control file	460
	AVSCOB JD – Define net directory specification (journal file)	461
	AVSCOB JN – Define journal record header	462
	AVSCOB RT – Define result area for net functions	463

- 11 AVAS-SV-BS2 - Job control for remote BS2000 systems 465**

- 11.1 Installation of AVAS-SV-BS2 on a BS2000 system 469**
- 11.2 Deinstallation of AVAS-SV-BS2 on a BS2000 system 469**
- 11.3 Run control and monitoring for server jobs 470**
 - 11.3.1 Structure of the configuration file 475
 - 11.3.2 Starting the AVAS agent AVSSINCM 478
 - 11.3.3 Structure of the task job variable and the server job file 484
 - 11.3.4 Job control and monitoring for remote BS2000 systems 487
 - 11.3.4.1 Starting the AVAS server under BS2000 487
 - 11.3.4.2 Terminating the AVAS server under BS2000 487
 - 11.3.5 Server monitor process 488
 - 11.3.5.1 Functions of the AVAS server monitor process 490
 - 11.3.5.2 Starting the server monitor 492
 - 11.3.5.3 Terminating the server monitor 493
 - 11.3.5.4 Operation for updating the server environment 494
 - 11.3.5.5 How to temporarily lock a server 494
 - 11.3.6 Server interface process 495
 - 11.3.6.1 Starting the server interface 496
 - 11.3.6.2 Shutting down the server interface 497
 - 11.3.6.3 Displays of the web-based server interface 497
- 11.4 Behavior after system failure 511**
- 11.5 Preparing for operation 514**
- 11.6 AVAS interface for external servers 519**

- Glossary 523**

- Related publications 535**

- Index 537**

1 Preface

The complexity and workload of computer centers are constantly increasing. DP operations therefore require clear structuring, high levels of transparency and flexibility, and constant productivity enhancement. The automation of batch production is an essential factor to reach this aim.

The AVAS (in German: **Auftragsverwaltungs- und Abwicklungssystem**) job administration and processing system provides computer center operators with a means of automating their job production to such an extent that operator intervention is reduced to an absolute minimum. Transferring batch processing into system layers that do not require operator intervention is greatly facilitated.

AVAS automates planning, preparation, release, control and monitoring of batch job processing in BS2000. The AVAS administration and control functions also run in BS2000.

From the BS2000 platform AVAS can start and monitor jobs on other systems:

- In the homogeneous BS2000 multiprocessor network AVAS uses the HIPLEX MSCF functions for job distribution and monitoring.
- The use of the AVAS-SV-BS2 server enables Remote BS2000 systems to be connected to AVAS via the Socket interface

The transfer of files to other vendors' systems is supported with an openFT connection.

In all cases the definition, preparation and monitoring of production is performed centrally by AVAS on a BS2000 system.

AVAS enables the computer center to automate its production jobs and to handle the necessary planning, preparation and monitoring tasks interactively. AVAS supports both decentralized work scheduling in the various non-DP departments and the central storage of information on jobs.

The *net description* defines the arrangement of jobs in the net, timing specifications, job characteristics, restart variants, and dependencies on other nets and jobs, and on condition values and resources.

Time scheduling uses calendars with symbolic dates or procedure names which, together with the net descriptions, form a *production plan*.

During *production preparation*, it is possible to supply the nets with runtime parameters from the production plan via user masks or from parameter files.

During the *release for production*, transport lists and tape mount listings can be created for the required data volumes by accessing the MAREN catalog. After the net has been released, it is started by the run control system in the *production execution* stage in accordance with the predefined time specifications and dependencies.

Production monitoring, like all of the preceding steps, is carried out online. If an error occurs, predefined restart processing is activated. Depending on the specifications in the net, restart is either executed automatically or initiated explicitly by the user, the latter permitting further manual intervention. The entire production sequence within the planned nets is logged and can be reconstructed on the basis of the journal. Under AVAS, the runtime logs of jobs can be saved and displayed.

1.1 Objectives and target groups of this manual

This manual addresses the AVAS administrator.

1.2 Structure of the AVAS documentation

The following documentation is available for the AVAS software product under the BS2000 operating system:

AVAS Functions and Tables

The “**AVAS Functions and Tables**” manual [1] is intended for AVAS users. It initially provides an overview of the AVAS functions, and then a detailed description of how to define and handle production runs. The manual also includes brief descriptions of multihost operation and of administration, plus tables and overviews.

AVAS Statements

The “**AVAS Statements**” [2] manual is intended for AVAS users and the AVAS administrator. It contains all the AVAS statements in alphabetical order. The masks are described together with the corresponding AVAS statements.

The manual also contains information on

- conducting a dialog
- preparing jobs for execution under AVAS
- the CHECK function.

AVAS for the Administrator

The system administrator guide “**AVAS for the Administrator**” is intended for those responsible for administrating the AVAS system. This manual describes all the tasks performed by the AVAS administrator, from generating the system to the administration of the AVAS system. The manual also includes information on

- the utility routine AVAS-QUER
- combining AVAS with MAREN
- AVAS reports
- BATCH functions
- external creation of AVAS elements
- program interface
- AVAS-SV-BS2

You can find these manuals online at <http://manuals.ts.fujitsu.com> or you can order them in printed form for a fee at <http://manualshop.ts.fujitsu.com>.

Readme file

The functional changes to the current product version and revisions to this manual are described in the product-specific Readme file.

Readme files are available to you online in addition to the product manuals under the various products at <http://manuals.ts.fujitsu.com>. You will also find the Readme files on the Softbook DVD.

Information under BS2000

When a Readme file exists for a product version, you will find the following file on the BS2000 system:

```
SYSRME.<product>.<version>.<lang>
```

This file contains brief information on the Readme file in English or German (<lang>=E/D). You can view this information on screen using the `SHOW-FILE` command or an editor. The `/SHOW-INSTALLATION-PATH INSTALLATION-UNIT=<product>` command shows the user ID under which the product's files are stored.

Additional product information

Current information, version and hardware dependencies, and instructions for installing and using a product version are contained in the associated Release Notice. These Release Notices are available online at <http://manuals.ts.fujitsu.com>.

1.3 Changes since the last edition of the manual

The following change has been made since the last edition of the manual:

The connection of other vendors' systems (Linux, Windows etc.) with AVAS-SV is no longer supported.

1.4 Notational conventions

References to other publications are specified in the text using abbreviated titles. The full title of each publication, which is referenced by a number, can be found under “Related publications” after the corresponding number.

Cross-references within this manual indicate the number of the relevant page in the manual and, if appropriate, the name of the section or chapter. References to information described in other manuals include only the abbreviated title of the corresponding manual. You can use the index of the manual indicated to find the appropriate place in the text.

Supplementary information appears under the heading “Notes”.

The following notational conventions are used in this manual:



This symbol denotes important information which you should always observe.



This symbol and the word **CAUTION!** precede warning information. In the interests of system and operating security you should always observe this information to prevent the loss of data.

`fixed`

Path and file names as well as phase names and procedure examples are displayed using a `fixed` font.

bold
`fixed`

Entered by the user on the screen.

1.5 Licensing regulations

The licensing regulations for the OpenSSL package and the TLS-FTP patch of Peter 'Luna' Runestig are printed below.

LICENSE ISSUES

=====

The OpenSSL toolkit stays under a dual license, i.e. both the conditions of the OpenSSL License and the original SSLeay license apply to the toolkit. See below for the actual license texts.

OpenSSL License

```

/* =====
 * Copyright (c) 1998-2016 The OpenSSL Project. All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 *
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in
 *    the documentation and/or other materials provided with the
 *    distribution.
 *
 * 3. All advertising materials mentioning features or use of this
 *    software must display the following acknowledgment:
 *    "This product includes software developed by the OpenSSL Project
 *    for use in the OpenSSL Toolkit. (http://www.openssl.org/)"
 *
 * 4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to
 *    endorse or promote products derived from this software without
 *    prior written permission. For written permission, please contact
 *    openssl-core@openssl.org.
 *
 * 5. Products derived from this software may not be called "OpenSSL"
 *    nor may "OpenSSL" appear in their names without prior written
 *    permission of the OpenSSL Project.
 *
 * 6. Redistributions of any form whatsoever must retain the following
 *    acknowledgment:
 *    "This product includes software developed by the OpenSSL Project
 *    for use in the OpenSSL Toolkit (http://www.openssl.org/)"

```



```
* THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY
* EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
* PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR
* ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
* SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
* NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
* LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
* STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
* ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
* OF THE POSSIBILITY OF SUCH DAMAGE.
```

```
* =====
*
* This product includes cryptographic software written by Eric Young
* (eay@cryptsoft.com). This product includes software written by Tim
* Hudson (tjh@cryptsoft.com).
*
*/
```

Original SSLeay License

```
/* Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com)
* All rights reserved.
*
* This package is an SSL implementation written
* by Eric Young (eay@cryptsoft.com).
* The implementation was written so as to conform with Netscapes SSL.
*
* This library is free for commercial and non-commercial use as long as
* the following conditions are aheared to. The following conditions
* apply to all code found in this distribution, be it the RC4, RSA,
* lhash, DES, etc., code; not just the SSL code. The SSL documentation
* included with this distribution is covered by the same copyright terms
* except that the holder is Tim Hudson (tjh@cryptsoft.com).
*
* Copyright remains Eric Young's, and as such any Copyright notices in
* the code are not to be removed.
* If this package is used in a product, Eric Young should be given attribution
* as the author of the parts of the library used.
* This can be in the form of a textual message at program startup or
* in documentation (online or textual) provided with the package.
```

```
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
* 1. Redistributions of source code must retain the copyright
*   notice, this list of conditions and the following disclaimer.
* 2. Redistributions in binary form must reproduce the above copyright
*   notice, this list of conditions and the following disclaimer in the
*   documentation and/or other materials provided with the distribution.
* 3. All advertising materials mentioning features or use of this software
*   must display the following acknowledgement:
*   "This product includes cryptographic software written by
*   Eric Young (eay@cryptsoft.com)"
*   The word 'cryptographic' can be left out if the routines from the library
*   being used are not cryptographic related :-).
* 4. If you include any Windows specific code (or a derivative thereof) from
*   the apps directory (application code) you must include an
*   acknowledgement:
*   "This product includes software written by Tim Hudson
*   (tjh@cryptsoft.com)"
*
* THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``AS IS'' AND
* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
* ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
* SUCH DAMAGE.
*
* The licence and distribution terms for any publically available version or
* derivative of this code cannot be changed. i.e. this code cannot simply be
* copied and put under another distribution licence
* [including the GNU Public Licence.]
*/
```

```
/*
 * Copyright (c) 1999 - 2002 Peter 'Luna' Runestig <peter@runestig.com>
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without modifi-
 * cation, are permitted provided that the following conditions are met:
 *
 * o Redistributions of source code must retain the above copyright notice,
 *   this list of conditions and the following disclaimer.
 *
 * o Redistributions in binary form must reproduce the above copyright no-
 *   tice, this list of conditions and the following disclaimer in the do-
 *   cumentation and/or other materials provided with the distribution.
 *
 * o The names of the contributors may not be used to endorse or promote
 *   products derived from this software without specific prior written
 *   permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 * ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED
 * TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
 * PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LI-
 * ABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUEN-
 * TIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEV-
 * ER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABI-
 * LITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF
 * THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
```

2 Installation and generation

This chapter describes the installation and generation of the AVAS system.

2.1 Installing the AVAS system

AVAS is installed with the IMON Installation Monitor.

The installation instructions contained in the Release Notice¹ must also be followed.

2.1.1 AVAS-SV-BS2

The AVAS server for remote BS2000 systems (AVAS-SV-BS2) is installed with the IMON Installation Monitor.

2.1.2 AVAS-QUER

The AVAS-QUER program is called either with the AVS.QUER procedure from the SYSPRC.AVAS.085 library or using the AVAS.SYS.LOAD.QUER program from the SYSPRG.AVAS.085.SYSTEM library.

The following prerequisites have to be fulfilled when the program is started:

- For creating SQL statements (INSERT-FORMAT):

The program needs an output file (SAM file) assigned with the link name \$AVSQUER. An existing output file will be overwritten.

¹ Release Notices are available online at <http://manuals.ts.fujitsu.com/>

- For creating a LOAD format:

The program needs an output file (SAM file) for each table to be created. Existing output files will be overwritten. The following link names must be used for the output files:

Table	Link name
caltab	\$AVSCAL
caldaytab	\$AVSCALD
caldaysymtab	\$AVSCDAY
nettab	\$AVSNET
netdoktab	\$AVSNETD
netformtab	\$AVSNETF
netpartab	\$AVSNETP
netsymtab	\$AVSNETS
netttexttab	\$AVSNETT
ordertab	\$AVSORD
orderdoktab	\$AVSORDD
orderpartab	\$AVSORDP
ordersymtab	\$AVSORDS
ordersturntab	\$AVSORDT
ordertexttab	\$AVSORDX

- In the system parameters of the AVAS system used, the user must be assigned the COPY-ELEMENT authorization for all objects which the user is to be able to access.
- The AVAS accessing tasks (ZDD, ZDL) must be active.
- The file SYSLNK.AVAS.085 must be allocated using the link name SYSLNK.
- The system syntax file SYSSDF.AVAS.085 (which contains the AVAS-QUER statements) and the message file SYSMES.AVAS.085 must be assigned. IMON takes the measures required to do this during the standard installation of AVAS (AVAS-QUER is installed automatically with AVAS). Further details are provided in the “IMON” manual [10].

2.2 Generation of the AVAS system

Every AVAS system consists of

- a central access task (batch task) which handles the UPAM accesses to the run control file and the journal file, referred to as UPAM-ZD in the following
- a central access task (batch task) which handles the PLAM accesses to all libraries, referred to as PLAM-ZD in the following
- optionally, the CENTRAL task, for collecting together the job logs; this task consists of one primary task and one or more secondary tasks
- a separate interactive task for each user of the system
- one or more run control systems (batch tasks) to control operations in the nets
- an optional run control system on an attached server system to control the processing of server jobs.
- an optional SOUT process on any BS2000 system to which the AVAS system sends jobs. The SOUT process permits access to the SYSOUT files of the AVAS jobs. Like the CENTRAL process, the SOUT process consists of one primary task and one or more secondary tasks.

All generation parameters required for describing this system are stored in the GENPAR generation file. The following items are defined in this file by means of the generation parameters:

- the AVAS system ID
- the system parameters of the central tasks
- the assignment of file names and libraries
- the description of the user groups
- the definition of the users
- the function authorizations
- the definition of the production plans
- the assignment of the mask libraries
- the assignment of the net libraries to the production plans
- the run control systems
- default values for the AVAS functions and the AVAS statements and variables.

When the AVAS system is delivered, the user is given a copy of the sample application AVAS-EXAMPLE. The values (e.g. names of libraries) assigned to the keywords can be adopted as they stand or modified or extended to meet the user's individual needs. The names of libraries, files and job variables are governed by the standard BS2000 naming conventions.

The generation parameters in the file with the symbolic name GENPAR may be modified or extended in accordance with the accompanying sample. This is done in BS2000 with EDT. In this case, the keywords of the different generation parameters must always begin in column 1 of the records (maximum length: 80 characters). A second line of a parameter description will be processed if the preceding line ended with a comma. Comment lines must start with an asterisk (*) in column 1.

When the generation parameters in the generation file are updated, the generation run must be started with the aid of the AVS.GENSYSPAR procedure (an element in the library SYSPRC.AVAS.085). This generates the AVAS system parameters (also referred to as system parameters below for short) from the generation parameters and stores them in the ISAM file with the default name AVAS.USER.SYSPAR.

The run control file and the journal file must be formatted using the procedure AVS.FORM (an element in the library SYSPRC.AVAS.085) before they can be used in the AVAS system.

Some system parameters can also be modified during the current AVAS session. Those values which can be modified are indicated under the parameter involved. When system parameters are modified using the MODIFY-SYSTEM-PARAMS statement, the associated generation parameters in the generation file remain the same.

2.2.1 Password encryption

The passwords of the AVAS users and the passwords in the AVAS procedures are encrypted as standard. The passwords of the AVAS users are encrypted at generation time, when entered in the signon mask and when entered via the MODIFY-SYSTEM-PARAMS statement.

The passwords in the AVAS procedures for signing on to the central access procedures are encrypted in the assigned programs.

Encryption of the BS2000 passwords in the nets can be set using AVAS generation parameter PASSWORD-ENCRYPTION.

The following passwords are stored in the nets:
NET-PASSWORD, JOB-PASSWORD, JVA-PASSWORD and FILE-PASSWORD

These passwords are encrypted in NPRLIB and ABLDAT.

The passwords in the nets in NETLIB are not encrypted because this library acts as an import/export interface to other AVAS systems (COPY-ELEMENT). The user can distribute the nets over more than one NETLIB so that even those with * authorization do not have access to all the nets.

Passwords in NPRLIB and ABLDAT are encrypted in the central access task ZDL, and decrypted in the run control routines.

The secret ENCRYPTION-CODE must be passed to the ZDL task and as a start parameter when run control routine AVAK is started (see [page 110](#) and [page 117](#)).

System structure

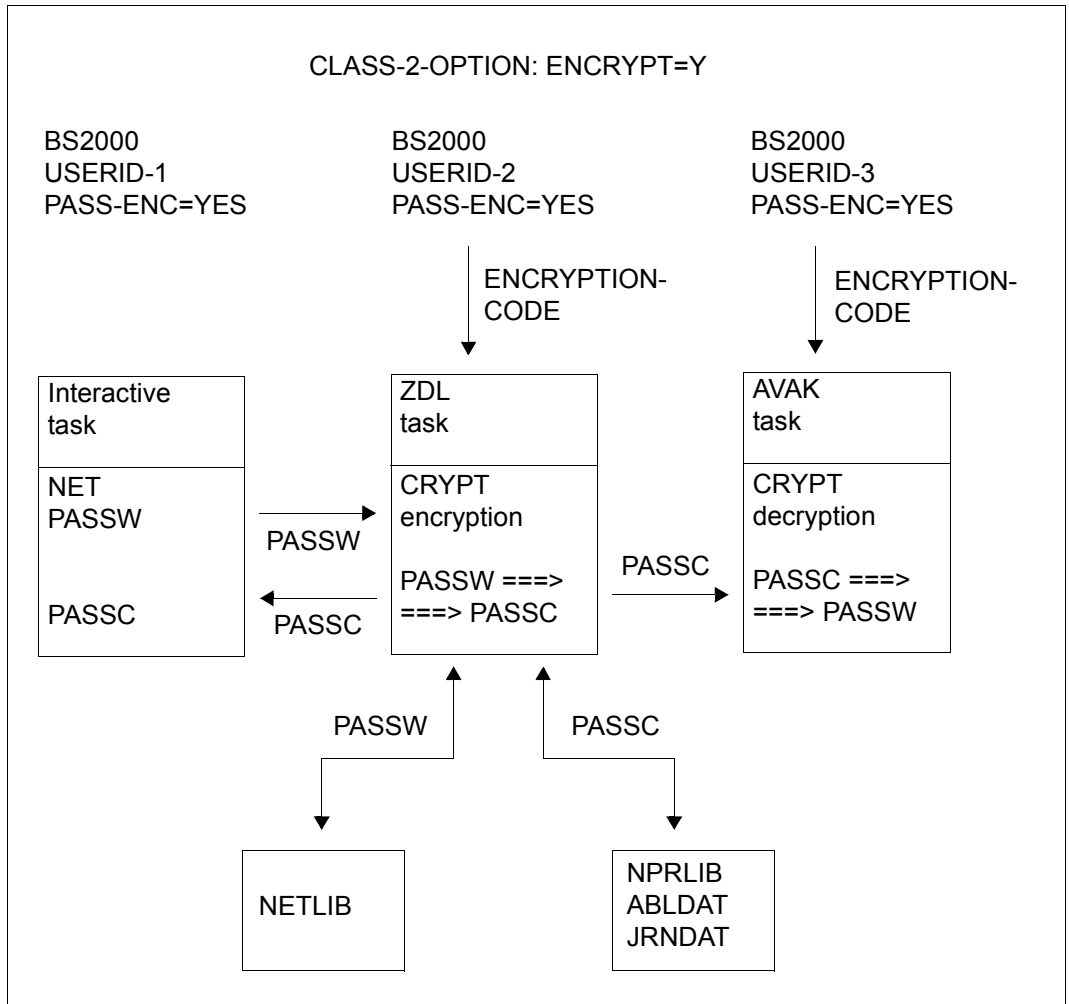


Figure 1: System structure

- NETLIB The passwords in NETLIB are not encrypted.
- NPRLIB The passwords can be stored in encrypted form.
Encryption takes place in CREATE-PLAN-NET.
Encryption takes place via ZDL.
- ABLDAT The encrypted passwords from NPRLIB are transferred to ABLDAT.
If entered via MODIFY-SUBMIT-NET the passwords are encrypted.
The old password is not checked.

Notes

- The password SERVER-PASSWORD stored in the nets is not encrypted.
- If the secret “code” is changed, it will no longer be possible to perform any further processing in production plan NPRLIB or in run control file ABLDAT on the nets planned before the change was made. This also applies if a switch is made from unencrypted storage of the passwords to encryption of the passwords.

If there is a change of procedure or a change in the secret “code”, it is advisable to delete the NPRLIB production libraries, the run control file and the journal file. The run control file and the journal file must be reformatted.

- The BS2000 macro CRYPT is used for encrypting the passwords. Encryption only takes place if the assigned class 2 option is set. Note that the AVAS tasks (DIALOG, BATCH, ZDL, ZDD, AVAK, REPORT and REORG) can run under different BS2000 user IDs and that the same value for PASSWORD-ENCRYPTION must be specified for all users.
- Encryption extends only to those passwords contained in the net description. ENTER-PARAMS=LOGON is not supported.

2.2.2 Description of the system parameters

2.2.2.1 Definitions for the central access tasks

The system parameters in the central access tasks define the amount of data in the AVAS system and how it is to be accessed. Some of these system parameters cannot be modified during the current AVAS session but only by starting a new generation run. System parameters must be defined for the following areas of the central access tasks:

- general parameters of the central access tasks
- file names of the libraries
- file names of the run control file and the journal file.

General parameters of the central access tasks

The general parameters must not be modified during the current session, i.e. they are read unchanged from the system parameter file SYSPAR.

AVAS-SYSTEM-ID=string

Character string (7 characters long) identifying the AVAS system.

JVCENTRAL=jvname

Name of the job variable for monitoring the CENTRAL task.

JVPLAMZD=jvname

Name of the job variable for monitoring the PLAM-ZD.

JVUPAMZD=jvname

Name of the job variable for monitoring the UPAM-ZD.

Job variables are used to monitor the central access tasks (ZDs), preventing them from being loaded a second time.

The job variables are set up by the tasks when these tasks are started.

If the names of the job variables are specified with a BS2000 user ID, the central access tasks must be brought to execution under the specified user ID.

PASSWORD-ENCRYPTION={YES / NO}

YES

The BS2000 passwords in the nets are encrypted via CRYPT for CREATE-PLAN-NET and stored in encrypted form in the production plan NPRLIB and in the run control file ABLDAT.

Encryption takes place in central access task ZDL, decryption in the run control routines.

NO

The BS2000 passwords in the nets are not encrypted.

MAX-BATCH-PROCESS=n

Maximum number n of batch tasks which can sign on to ZDD and ZDL at the same time. The tasks on the program interface must also be taken into account. Attempts by other tasks to sign on are rejected with a return code. The tasks are placed in the wait state and signon is repeated after a defined delay.

If the parameter is omitted, MAX-BATCH-PROCESS=2 is assumed.

Notes

The statements CREATE-PLAN-NET, CREATE-PROD-NET and SUBMIT-NET/REPEAT-NET are serialized (see [section "Starting the system and system security" on page 106](#)).

This allows a practical value to be calculated for MAX-BATCH-PROCESS:

$$\text{MAX-BATCH-PROCESS} = \text{pp} + \text{bt} + (\text{ba} + \text{ba} * \text{bt}) / 10 < 10$$

pp Number of programs permanently signed on to the program interface (e.g. CC monitor)

bt Number of batch tasks for production planning which are to run in parallel. bt must be given a value between 0 and 3.

0 No production planning via batch tasks

3 Permanent cyclic production planning via batch tasks

ba Number of other batch jobs used to analyze and modify the AVAS data which can be executed in conjunction with the production planning.

A value greater than 3 for bt is pointless because the statements CREATE-PLAN-NET, CREATE-PROD-NET and SUBMIT-NET are serialized and the fourth task must always wait for the serialization. If the production preparation is not handled via CREATE-PROD-NET, the highest practical value for bt is 2.

Example

pp = 1 (CC monitor via AVAS program interface)

bt = 2 Permanent production planning

ba = 4 COPY-ELEMENT, CREATE-PERIOD, HOLD-NET,
RESUME-NET, RESTART-NET

$$\text{M-B-P} = 1 + 2 + (2 + 2 * 4) / 10 = 4$$

MSG7-DEST-DIALOG={SYSOUT / SYSLST / BOTH}

The MSG7-DEST-DIALOG parameter determines the destination of messages issued via the MSG7 interface.

SYSOUT Messages are output to SYSOUT only (default).

SYSLST Messages are output to SYSLST only.

BOTH Messages are output to SYSOUT and SYSLST.

If the parameter is omitted, messages are output to SYSOUT only.

Restriction

The setting applies only to the AVAS dialog, starting from a successful signon. The default setting remains in force until a successful signon is made.

MAX-CENTRAL-PROCESS=n

Maximum number of CENTRAL tasks which can run simultaneously.

The maximum value for n is 30. The default value is 4.

The number applies to secondary tasks only.

CALLIB=filename File name of the calendar library.

PERDAT=filename File name of the period file (ISAM file).

SOUT-APPLICATION-NAME=name

DCAM application name of the SOUT process (up to 8 character). Alternatively the application name can be stored in a job variable (see SOUT-APPLICATION-JV parameter).

SOUT-APPLICATION-JV=job variable

Name of the job variable which contains the DCAM application name of the SOUT processes. The AVAS dialog processes must be able to access this job variable. The job variable should therefore be cataloged as shareable (USER-ACCESS=ALL-USERS) and, if required, protected by a read password (see SOUT-APPLICATION-PASSWORD parameter).

SOUT-APPLICATION-PASSWORD=password

C'....' or X'.....'

Four characters (alphanumeric or hexadecimal) must be used to specify the read password of the SOUT-APPLICATION-JV.

File names of the libraries to be used

The names (“filename”) of the following AVAS libraries cannot be changed using the MODIFY-SYSTEM-PARAMS statement. However, during the current session it is possible to assign a different file to a user group (and therefore to a user) via a keyword defined in the generation file.

The numbers (nnn = 000–999) in the keywords (e.g. NETnnn) do not have to be consecutive. They are generally denoted by LIB (e.g. NETLIB): “filename” is used to define the name of a PLAM library.

NETSYS=filename	File name of the system library where the net descriptions available for all users throughout the system are stored.
NETnnn=filename	File name of the user libraries in which the net descriptions of the users are stored.
JCLSYS=filename	File name of the system library in which the jobs and JCL elements available for all users throughout the system are stored.
JCLnnn=filename	File name of the user libraries in which the users’ jobs and JCL elements are stored.
DOCSYS=filename	File name of the system library in which the documentation elements available throughout the system to all users are stored.
DOCLIB=filename	File name of the user libraries in which the documentation elements of the users are stored.
NPRnnn=filename	File name of the user libraries in which the planned nets of the users are stored.
JMDSYS=filename	File name of the system library in which the executable jobs available for all users throughout the system are stored.
JMDnnn=filename	File name of the user libraries in which the executable jobs of the users are stored.
LOGSYS=filename	File name of the system library in which all the runtime logs of jobs are stored centrally (AVAS pool).

Notes

- At least one JCLLIB and JMDLIB must be preset in the system parameters, even if none of these files is actually needed in the application.
If the files are not accessed, they do not have to be present in the system. In the case of the JCLLIB and the JMDLIB, this applies if only JOB-TYPE=EXT is being used.
- The libraries DOCLIB and DOCSYS must be assigned, even if the assigned functions are not used.

- Libraries which do not yet exist at the time when PLAM-ZD is started up will be created.

File names of the run control file, the journal file and the history file

The names ("filename") of the run control and journal files cannot be modified during the current AVAS session.

ABLDAT=filename File name of the run control file.
This PAM file must be formatted using the procedure AVS.FORM (an element in the library SYSPRC.AVAS.085) following generation for the current session.

It can be accessed only via those run control systems which are present in the system parameter file created at system generation time using the procedure AVS.GENSYSPAR (an element in the library SYSPRC.AVAS.085).

ABLDUP=filename File name for the copy of the run control file.
This file need only be specified if the central access task is started with the statement ABLDUP=YES.
The file is not formatted for operation. It is generated by means of the statement ABLCOPY=YES when the central access task is started.

JRNDAT=filename File name of the journal file.
This PAM file must be formatted using the procedure AVS.FORM (an element in the library SYSPRC.AVAS.085) following generation for the current session.
It can be accessed only via those user groups and run control systems which are present in the system parameter file created at system generation time using the procedure AVS.GENSYSPAR (an element in the library SYSPRC.AVAS.085).

JRLDAT=filename Prefix of the file name of an alternative SAM file (emergency journal file) to which the journal records are output following an error in the PAM file JRNDAT. When the emergency journal file is created by AVAS, the file name is completed by the addition of a date and a time of day.

The prefix of the file name of the emergency journal file must be unique if more than one AVAS system is started on the same host (ZDD). Otherwise, if there is an emergency journal file of a second system, the UPAM-ZD attempts to merge it with the journal file of the first system (or vice versa) if one of the AVAS systems has not been started.

Furthermore, when the second system is started, access to the emergency journal file of the first system leads to an error because this file was already open.

HSTSYS=filename File name of the history file (ISAM file)
In a reorganization with FUNKTION=SJOUR, the net and job history records are stored in this file.

2.2.2.2 Definition of users

The system parameters of the users define all specifications required for interactive operations and user statements.

The parameters describe the users permitted to use the AVAS system, together with their function authorizations and their affiliation to a user group.

The following specifications must be defined:

- parameters of the user groups
- parameters of the users
- general parameters of the users
- parameters of the file assignments
- parameters of the function authorizations.
- system variables (S#nnn) of the users

Definition of the user groups

A user group definition defines a group of AVAS libraries as a single unit. In connection with user group definitions, a distinction must be made between system user groups and user groups to which individual users are assigned, e.g. grouped according to their subject area.

System user group

The system user group is defined by the keyword BKSYS (“BK” being the German abbreviation for “user group”). This user group must be assigned those system libraries which can be accessed by all users throughout the system. A system user group must always be defined at generation time. Its definition cannot be modified during the current AVAS session.

BKSYS=(ug,NETSYS,JCLSYS,JMDSYS,DOCSYS,LOGSYS)

- ug Name of the system user group
(up to 4 characters preceded by a dollar sign).
- NETSYS Keyword of the system library for net descriptions.
- JCLSYS Keyword of the system library for jobs and JCL elements.
- JMDSYS Keyword of the system library for executable jobs.
- DOCSYS Keyword of the system library for documentation of the nets.
- LOGSYS Keyword of the system library for the log data.

Elements with the system user group are sought only in the library JCLSYS, NETSYS or JMDSYS. Elements with another user group cannot be processed in these libraries by means of AVAS functions.

The following statements are used to process elements in the system libraries:

Statement	INPUT-LIB	OUTPUT-LIB
COPY-SYSTEM-ELEMENT	NETLIB	NETSYS
	or JCLLIB	JCLSYS
	or JMDLIB	JMDSYS
	or DOCLIB	DOCSYS
DELETE-SYSTEM-ELEMENT	NETSYS	-
	or JCLSYS	-
	or JMDSYS	-
	or DOCSYS	-
ADD-JOB-LOG	-	LOGSYS
DELETE-JOB-LOG	LOGSYS	
CREATE-PLAN-NET	NETSYS	(NPRLIB)
CREATE-PROD-NET	JCLSYS	(JMDLIB)
CREATE-PROD-JOB	JCLSYS	(JMDLIB)
SUBMIT-NET	JMDSYS	(ABLDAT)
REPEAT-NET	JMDSYS	(ABLDAT)

The elements from the system libraries can be displayed using the following statements, in which case a *-authorization is necessary in order to access the libraries.

Statement	INPUT-LIB
SHOW-JOB	JCLSYS
SHOW-NET-DESCRIPTION	NETSYS
SHOW-PROD-JOB	JMDSYS
SHOW-DOCUMENT	DOCSYS

The elements from system library LOGSYS can be displayed with the SHOW-JOB-LOG statement. For this statement a normal-user function authorization or a *-authorization can be granted.

User groups

A user group is defined by the keyword BK (this being the German abbreviation for “user group”). This definition must be entered as many times as the number of user groups required in the AVAS system. The number of user groups is unlimited. Values to be specified in parentheses must be entered in the sequence described below.

BK=(ug,NETnnn,JCLnnn,calendar,avak,avak-use,server-name)

ug	Name of a user group (up to 4 characters preceded by a dollar sign).
NETnnn	Keyword of the user library for net descriptions (net library).
JCLnnn	Keyword of the user library for jobs and JCL elements (JCL library).
calendar	Name of the calendar (up to 20 characters; element of the calendar library CALLIB) which is assigned to the user group by default.
avak	Name of the default run control system for this user group (up to 8 characters).

avak-use Specifies how names of run control systems are used.

*STD The users in this user group may only use the name defined here for the default run control system or *STD. If nets of foreign user groups are processed, the name of the run control system can only be changed to *STD or to the name of the user's own run control system. If a user copies nets of a foreign user group with COPY-NET-DESCRIPTION, or modifies the user group of the net while copying with COPY-ELEMENT (MODE=LIBIN), the value of the RUN-CONT/ROL-SYSTEM parameter is set to *STD in the net description.

*ALL The users in this user group may use the name defined here for the default run control system or the name of another run control system defined in the system.

The specification avak-use is not shown in the dialog functions SHOW-SYSTEM-PARAMS and MODIFY-SYSTEM-PARAMS and cannot be modified.

server-name Identifying name for the server (up to 8 characters, see server configuration file)

name 1..8 Server name as defined in the configuration file

*STD The server name is specified in the net description itself or in the AVSSINCM start procedure.

*NONE Members of this user group are not permitted to use server systems via AVAS.

The specification server-name is not shown in the dialog functions SHOW-SYSTEM-PARAMS and MODIFY-SYSTEM-PARAMS and cannot be modified.

BK=(ug,) Definition of additional user groups

The user group assignment of the libraries, calendar and run control system can be modified during the current AVAS session by means of the MODIFY-SYSTEM-PARAMS statement.

Definition of the users

USER serves to define the users for AVAS. It sets down the identification, password and statement authorization.

USER=(avuser,password,ug,FUTABnnn,break,password-visibility,manage-servers)

avuser User identification (up to 8 characters).

password User password (up to 8 characters).

ug Name of the user group to which the user is assigned
(no user can be assigned to the system user group).

FUTABnnn Keyword of a function authorization table.

break Authorization to interrupt using the K2 key

YES Interruption of the dialog by the K2 key permitted.

NO Break not permitted (default).

Note

Interruption of interactive mode with the aid of the K2 key is only ever possible after successful signon. Interruption is no longer possible once the SIGNON and END statements have been entered.

password-visibility

Authorization to make passwords visible

YES The BS2000 password can be made visible in the relevant masks.

NO The BS2000 password is always blanked out
(default setting).

manage-servers

Authorization to manage the server configuration using the server interface process (AVSSURF program)

YES The user is authorized to manage the server configuration.

NO No authorization to manage the server configuration.

USER=(avuser,password, . . .)

Definition of additional users.

The USER definition must be entered as many times as the number of users permitted in the user group. Values enclosed in parentheses must be entered in the sequence described above (positional parameters). The assignment of password, user group and FUTABnnn to the users can be modified during the current AVAS session by means of the MODIFY-SYSTEM-PARAMS statement.

Users may modify their passwords in the SIGNON mask even if they have no authorization to use MODIFY-SYSTEM-PARAMS.

The BS2000 passwords are always blanked out after SIGNON. In some masks the VISIBLE operation or #71 enables the AVAS user to display the passwords provided password-visibility=YES has been set for him/her (see also the descriptions of the password parameters in the manual “AVAS Statements” [2]).

General parameters of the users

The general parameters are used to assign files to the user in which he can store his user masks. They also assign to the user the file with the information functions.

MAPnnn=filename File name of a user library in which the modules of the user masks can be stored. At least one MAPLIB for net and job masks must be specified in the system parameters, even if these files are not required in the application.

HLPLIB=filename The SYSMH.AVAS.085 file contains the text elements for the AVAS information function in German and English. The HLPLIB file is assigned to all AVAS users. The file is accessed via the central access task PLAM-ZD.

The texts are output in the language which is set for outputting BS2000 system messages (see BS2000 command SHOW-MSG-FILE-ASSIGNMENT, SYSTEM LANGUAGES output field).

These parameters may be modified during the current AVAS session using the MODIFY-SYSTEM-PARAMS statement. After a reload of the central access tasks for library accesses, the “new” files are assigned.

Definition of the production plans

The production plan definitions assign a production job library (JMDnnn) to a production net library (NPRmmm). The production plans must be defined with the keyword PRODTABnnn. These definitions create a table in which the associated keywords are entered for each production plan. System-wide, the keywords NPRnnn and JMDnnn may be used only once each in the production plans.

```
PRODTAB001=(NPRmmm / JMDnnn)
PRODTAB002=(. . .)
...
```

The definitions of the production plans cannot be changed during the current AVAS session.

Definition of the assignment of net libraries to production plans

The definition of the net production plans assigns a production plan (PRODTABmmm) to a net library (NETnnn). The net production plans must be defined with the keyword NET-PRO.

```
NET-PRO=(NET001 / PRODTABmmm,  
          NET002 / PRODTABmmm,  
          ...  
          NETnnn / PRODTABmmm)
```

Definition of the assignment of mask libraries to job libraries

This assignment links the keywords of the libraries which are required when jobs are modified via user masks. To do this, each job library (JCLnnn) must be assigned one and only one mask library (MAPmmm).

The assignment must be defined with the keyword JOB-MAP.

```
JOB-MAP=(JCLSYS / MAPmmm,  
          JCL001 / MAPmmm,  
          JCL002 / MAPmmm,  
          ...  
          JCLnnn / MAPmmm)
```

The assignment of MAP to JCL can be modified.

Definition of the assignment of mask libraries to net libraries

This assignment links the keywords of the libraries which are required when nets are modified via user masks. To do this, each net library (NETnnn) must be assigned one and only one mask library (MAPmmm).

The assignment must be defined with the keyword NET-MAP.

```
NET-MAP=(NETSYS / MAPmmm,  
          NET001 / MAPmmm,  
          NET002 / MAPmmm,  
          ...  
          NETnnn / MAPmmm)
```

The assignment of MAP to NET can be modified.

Definition of function authorizations

The function authorization definition is used to set down which statements the user is authorized to use and the type of authorization involved. To change a user's authorizations a different function authorization table (FUTABnnn) must be assigned in the definition of the user (see above).

FUTABnnn=(function1[/{0/1/*}],function2[/{0/1/*}],function3[/{0/1/*}],. . .)

Following the keyword FUTABnnn the permissible statements (function1...) must be specified in parentheses and separated by commas. Statements not specified here are given the default value 0. For the SIGNON, HELP and END statements, the default value is 1 (see also manual "AVAS Statements" [2]).

function Name of a valid statement.

{0/1/*} Type of authorization

0: No authorization for the statement.

1 Normal user authorization; limited to elements in the associated user group.

*: Privileged user authorization; this user can access elements of a foreign user group within his own library.

Function authorization definitions can be modified during the current AVAS session. The number of function authorizations is unlimited.

When the function CREATE-GENPAR is used, an FUTABnnn= instruction must not be defined over several lines.

2.2.2.3 Definition of the system variables of the users

Users can define AVAS variables of type S#nnn using default values. The user-defined system variables and the values assigned to them are taken into account when generating executable jobs in the JMDLIB/JMDSYS using the statements CREATE-PROD-JOB and CREATE-PROD-NET.

The system variables must be defined using the keyword USVAR.

USVAR=(nnn,value)

nnn A 3-digit number between 201 and 999 (see *Note*)
The specified number along with the symbolic name defined during the generation (the default character string is S#) defines the name of the system variable.

Note

The value range 000 through 200 is reserved for AVAS.

value Assigned value of the system variable.
Any character string containing between 1 and 48 characters, including quotes.
If 'value' is specified in simple quotes, the character string can contain blanks. Two quotes must be entered for every quote within 'value'.

Note

The name of a system variable must be unique.

S#nnn values with syntax errors are rejected and an error message is displayed.

The definition (name and value) of the users' system variables can be modified during the current AVAS session. The number of system variables to be defined by the user in the value range 201 through 999 is limited to 400 variables.

2.2.2.4 Defining the run control system

The system parameters for the run control system cannot be modified during the current AVAS session.

AVAK defines the run control system. Check interval, routing code (authorization code) and job variable name are defined. A run control system must be defined with the keyword AVAK. The number of run control systems (max. 24) and their various characteristics must be defined in the sequence given below.

AVAK=(avak,control-time,routing-code,jvavak,mscf-control-time)

avak	Name of the run control system (up to 8 characters). “avak” is the German abbreviation.
control-time	Interval between two action cycles of the run control system, during which a check is made to see whether the planned start time of released nets has been reached (time specified in seconds). <i>Note</i> This parameter has a considerable influence on the CPU time used by the run control system and the central access tasks. At the specified regular intervals the directories of the run control file are searched for nets that are to be started. It is practical to select an interval of not less than 30 seconds, so as not to interfere with normal operation.
routing-code	Routing code for messages from the run control system. Authorization code for the assigned consoles; the default value is * (main operating console) (see the “Introductory Guide to Systems Support” [6]).
jvavak	Name of a job variable for monitoring the run control system. The job variable is set up by the task when this task is started. If the control statement NETC is to be entered by CC routines, the name must be defined with the complete path name. The function processes must have write access to the job variable.

mscf-control-time

Control time after HOSTWAIT (in seconds)

After the connection to a remote system that has failed has been reestablished, the run control system waits for this period of time before starting new jobs on the system. The pubsets required by AVAS (home pubset, shared public volume set) have to be available again after this term.

The message AVS8301 indicates when the term begins and the message AVS8500 when it ends.

Example

```
% AVS8301 INPUT '$R in MSCF-CONTROL-JVA/VPASS/00000300'
```

```
% AVS8500 INPUT '$R in MSCF-CONTROL-JVA/VPASS/00000300'  
PROCESSED
```

The characteristics control-time, routing-code and mscf-control-time can be modified during the current AVAS session. They come into effect the next time the run control system is started.

AVAK=(avak,...)

Definition of additional run control systems.

2.2.2.5 Defining the default values for processing parameters

For standard values applicable throughout the system, default values can be predefined via the generation parameters or configuration-specific settings can be made.

Some default values are assumed as parameters in the various operations if no entry is made in the relevant mask fields.

Other default values control the way in which operations are executed during processing.

Function keys on the data display terminals can be used for entering operations.

Defining the function keys for operation entry

The K and F keys of a data display terminal can be assigned for the entry of operations. Unassigned keys do not initiate any processing.

The K/F key assignments must not be changed during an AVAS session.

The values K1 through K15 or F1 through F24 are permitted for the following operations (e.g. DEFAULT-OPERATION-PLUS=K1).

DEFAULT-OPERATION-PLUS=
DEFAULT-OPERATION-MINUS=
DEFAULT-OPERATION-EXECUTE=
DEFAULT-OPERATION-RETURN=
DEFAULT-OPERATION-SAVE=
DEFAULT-OPERATION-CONTINUE=
DEFAULT-OPERATION-PRINT=
DEFAULT-OPERATION-FIRST=
DEFAULT-OPERATION-LAST=
DEFAULT-OPERATION-IGNORE=
DEFAULT-OPERATION-DOCUMENT=
DEFAULT-OPERATION-HELP=
DEFAULT-OPERATION-JOBLOG=
DEFAULT-OPERATION-CHECK=
DEFAULT-OPERATION-VISIBLE=

Note

Note that with the K keys only short messages can be transmitted to the processor; data transfer is not possible.

It is not therefore possible to change the data with the SAVE operation by means of a K key.

Defining overview processing

DEFAULT-MARK-YES={YES / NO}

- YES** An element must be marked in every selection mask. If no element is marked and the EXECUTE operation is performed, the message AVS5005 (NO OBJECT SELECTED FOR PROCESSING) is displayed.
- NO** It is not necessary to mark an element. If no element is marked and the EXECUTE operation is performed, all the elements are marked internally with Y and processed.

Notes

- For COPY-ELEMENT, COPY-SYSTEM-ELEMENT, CREATE-PLAN-NET and CREATE-PROD-NET the default value is ignored.
- This option does not apply to DELETE statements. In this case, the objects to be deleted must be marked.

Defining default values for the net description

If no default is defined, the underscored value is assumed.

DEFAULT-NET-TYPE={1 / 2 / 3}

Default value for the NET-TYPE parameter in the AVN001 mask.

DEFAULT-NET-DELAY={WAIT / START / IGNORE / CANCEL}

Default value for the NET-DELAY-SOLUTION parameter in the AVN020 mask.

DEFAULT-NET-TURNUS={1 / 2 / 3 / 4 / 5 / 6 / 7 / 8 / 9}

Default value for the SELECT-TURNUS parameter in the AVN020 mask.

DEFAULT-PLAN-TYPE={WORK / NWRK}

Default value for the SELECT-PLAN-TYPE parameter in the AVN020 mask.

DEFAULT-DOC-NAME={*NONE / *STD}

Default value for the NET-DOC / JOB-DOC / COND-DOC parameters in the AVN001, AVN002, AVN003, AVN008, AVN030, AVN031 and AVN032 masks.

DEFAULT-USER-PARFILE={*NONE / *STD}

Default value for the USER-PAR-FILE parameter in the AVN001 mask.

DEFAULT-JOB-TURNUS={0 / 1 / 2 / 3 / 4 / 5 / 6 / 7 / 8 / 9}

Default value for the SELECT-TURNUS parameter in the masks of the planning data for structure elements.

DEFAULT-JOB-TYPE={MOD / STD / EXT}

Default value for the JOB-TYPE parameter in the AVN004 mask.

DEFAULT-ENTER-PARAMS={NET / LOGON}

Default value for the ENTER-PARAMS parameter in the AVN002 mask.

DEFAULT-RESTART-TYPE={RESTART / NORMAL / RESTART-AUTO / NORMAL-AUTO}

Default value for the RESTART-TYPE parameter.

DEFAULT-RESTART-JOB={*ALL / *NAME / *ERROR}

Default value for the RESTART-JOB-NAME parameter.

DEFAULT-CONDITION-TYPE={NET / JVA}

Default value for the COND-TYPE parameter.

DEFAULT-CONDITION-DELAY={START / CANCEL / IGNORE}

Default value for the DELAY-SOLUTION parameter in the net structure descriptions in conjunction with FUNCTION=C.

DEFAULT-JOB-DELAY={START / CANCEL / IGNORE}

Default value for the DELAY-SOLUTION parameter in the net structure descriptions in conjunction with FUNCTION=J / P.

DEFAULT-JVA-POSITION=001...256

Default value for the JVA-POSITION parameter.

DEFAULT-JVA-LENGTH=001...256

Default value for the JVA-LENGTH parameter.

Defining default values for net planning

DEFAULT-LIFE-TIME={ddd.hh.mm / *NONE}

Mandatory parameter

Default value for the LIFE-TIME parameter in the AVP001 and AVP003 masks.

In the production planning, the specified value replaces the value *STD.

When the run control file is reorganized, condition descriptions for nets which have finished running without errors but whose LIFE-TIME has expired are deleted.

DEFAULT-LATEST-NETSTART=ddd.hh.mm

Mandatory parameter

Default value for the LATEST-START parameter in the net description.

In the net planning, the specified value replaces the value *NONE in the description of the start parameters for the net.

DEFAULT-LATEST-JOBSTART=ddd.hh.mm

Mandatory parameter

Default value for the LATEST-START parameter in the net structure descriptions.

In the net planning, the specified value replaces the value *NONE in the descriptions of type FU=J / P.

DEFAULT-LATEST-OCCURE=ddd.hh.mm

Mandatory parameter

Default value for the LATEST-OCCURE parameter in the net structure descriptions. In the net planning, the specified value replaces the value *NONE in the descriptions of type FU=C.

DEFAULT-OCCURE-TIME=ddd.hh.mm

Mandatory parameter

Default value for the OCCURE-TIME parameter in the net structure descriptions. In the net planning, the specified value replaces the value *NONE in the descriptions of type FU=W.

Defining default values for displaying the structures of nets

Default values used in displaying structure elements which do not get processed, in the statements CREATE-PLAN-NET, SHOW-PLAN-NET, SUBMIT-NET, REPEAT-NET, SHOW-NET-STATUS, MODIFY-SUBMIT-NET, MODIFY-SUBMIT-JOB and RESTART-NET.

DEFAULT-DISPLAY-DUMMY={YES / NO}

YES structure elements with a current status of NO-PLAN, NO-SUBMIT and DELETED are included with the structure display.

NO structure elements with a current status of NO-PLAN, NO-SUBMIT and DELETED are not displayed.

In the HOLD-NET and RESUME-NET statements, all structure elements are always displayed, as nets can also be suspended for the structure elements which do not get processed.

Defining default values for job editing

If no default value is defined, the underscored value is assumed.

DEFAULT-OVERWRITE-JOB={YES / NO}

Default value for the OVERWRITE parameter in the AVE011 mask of the EDIT-JOB operation.

DEFAULT-OVERWRITE-PRODJOB={YES / NO}

Default value for the OVERWRITE parameter in the AVE011 mask of the EDIT-PROD-JOB operation.

DEFAULT-PROCPAR-STRING={/'/* AVAS-PROC-PAR' / <c-string 1..16>}

Value for the separator in S procedures.

In S procedures, the separator is used to separate the procedure area from the parameter specification. For S procedures, this character string is usually stored as the last record in the JCLLIB, JCLSYS, JMDLIB, JMDSYS and ABLDAT.

Note

This character string must also be used when using COPY-ELEMENT with MODE=LIBIN/SAMIN and AVAS-USER-LIBRARY=JCLLIB to copy S procedures. Otherwise, the elements which are copied will be entered in the JCLLIB directory as jobs.

This character string is reserved for AVAS, and must not be used in S procedures for any other purpose.

The character string must not be the same as the value of DEFAULT-SERVERPAR-STRING.

DEFAULT-SERVERPAR-STRING={/'/* AVS-SINIX-PAR' / <c-string 1..16>}

Value for the separator in server jobs.

In server jobs, the separator is used to separate the job area from the parameter specification. With server jobs in the JCLLIB, JCLSYS, JMDLIB, JMDSYS and ABLDAT, the character string is generally stored as the last record, even if no parameters are being used.

Note

This character string must also be used when using COPY-ELEMENT with MODE=LIBIN/SAMIN and AVAS-USER-LIBRARY=JCLLIB to copy server jobs. Otherwise, the elements which are copied will be entered in the JCLLIB directory as jobs.

This character string is reserved for AVAS, and must not be used in server jobs for any other purpose.

The character string must not be the same as the value of DEFAULT-PROCPAR-STRING.

PARAM-JOURNAL-OUTPUT={STD / LIST / ALL / NO}

The generation parameter controls the logging of the replacement of AVAS variables in the journal in the CREATE-PROD-NET statement.

STD The journal records 25-01 and 25-02 are output for every modified record. The name of the USER-PARAM file is output with the record key 21-01.

- LIST** At the start of the modification of a net via the CREATE-PROD-NET statement, a list of all parameters from the USER-PARAM file is output with the record key 21-01.
(Structure of SSL 21-01: file_name parameter_record).
The values of the system variables S#001, S#002 and S#003 are also output.
- ALL** At the start of the modification of a net via the CREATE-PROD-NET statement, a list of all parameters from the USER-PARAM file is output with the record key 21-01.
The values of the system variables S#001, S#002 and S#003 are also output.
The journal records 25-01 and 25-02 are output for every modified record.
- NO** The parameters and the modification are not logged.

Defining default values for net release

DEFAULT-OPERATOR-START={YES / NO}

Default value for the OPERATOR-START parameter in the AVF002 and AVF012 masks of the SUBMIT-NET and REPEAT-NET statements.

Defining values for displaying hypernets

HYPERNET-COLOUR={*NONE /*STD/BLUE/CYAN/GREEN/MAGENTA/RED/WHITE/YELLOW}

Hypernets can be displayed for the sake of clarity on the AVD011, AVD012, AVD015, AVI012 and AVI022 masks with the net overviews of the CANCEL-NET, HOLD-NET, MODIFY-SUBMIT-NET, NET-CONTROL, RESTART-NET, RESUME-NET, SHOW-NET-STATUS and START-NET commands.

*NONE Hypernets are displayed in exactly the same way as other nets.

*STD The value CYAN is set.

BLUE/CYAN/GREEN/MAGENTA/RED/WHITE/YELLOW

Hypernets are displayed in the corresponding color if the emulation permits this and the color assignments of the emulation do not map the corresponding colors to others.

Defining values for restart processing

If no value is defined, the underscored value is assumed.

The values for RESTART-SKIP-ERROR and RESTART-SKIP-CONDITION define the behavior of the RESTART-NET operation if the selected restart variant prevents part of the net from running.

RESTART-SKIP-ERROR={YES / NO}

- YES** Structure elements with the ERROR status may be set to the SKIPPED status.
- NO** The restart is rejected if a structure element with the ERROR status has to be set to the SKIPPED status.
The exception to this is the structure element in POINT-OF-ERROR.

RESTART-SKIP-CONDITION={YES / NO}

- YES** Conditions with the NO-OCCURE or WAITING status may be set to the SKIPPED status.
- NO** The restart is rejected if a condition with the NO-OCCURE or WAITING status has to be set to the SKIPPED status.

The values for RESTART-WAIT-ERROR and RESTART-WAIT-CONDITION define the behavior of the RESTART-NET operation if the selected restart variant causes part of the net to run again.

RESTART-WAIT-ERROR={YES / NO}

- YES** Structure elements with the ERROR status may be set to the WAITING status.
- NO** The restart is rejected if a structure element in the ERROR status has to be set to the WAITING status.
The exception to this is the structure element with the POINT-OF-ERROR.

RESTART-WAIT-CONDITION={YES / NO}

- YES** Conditions with the OCCURRED status are to be set to the WAITING status. They are checked again for fulfillment after the restart.
- NO** Conditions with the OCCURRED status are not processed during a restart. They retain the OCCURRED status and are not checked after the restart.

Conditions which are changed from the NO-OCCURE status to the SKIPPED status by the RESTART-NET operation are considered to be fulfilled. They are treated by RESTART-NET in the same way as conditions with the OCCURRED status.

Structure elements with the status EXECUTED (FU=A/M/D) are not processed in a restart. They retain the EXECUTED status and are no longer executed after the restart. The structure element FU=M with TYPE=RES is an exception. Here, if RESTART-WAIT-CONDITION=YES is specified, the EXECUTED status is changed to WAITING and the structure element is executed again after the restart.

Definitions for function control

DEFAULT-CANCEL-TYPE={SOFT / HARD}

SOFT For the CANCEL-NET function, the CANCEL-TYPE operand is preset to SOFT if it is not specified.

HARD For the CANCEL-NET function, the CANCEL-TYPE operand is preset to HARD if it is not specified.

Note

The value does not apply to CANCEL-NET via the run control system. In this case, CANCEL-TYPE=HARD must still be specified.

2.2.2.6 Symbolic names of AVAS statements and variables

When executable jobs are created in the JMDLIB/JMDSYS, it is still possible to modify the control statements. To do this, AVAS provides special statements and variables. The preset default values can be redefined accordingly.

The symbolic names of the AVAS statements and variables (for modifying jobs and JCL elements and for processing restart statements via the run control system) are assigned the following character strings by default as AVAS statements and variables (left: symbolic name; right: assigned character strings):

#AVM#=#AVM#	Job or net modification via user masks.
#AVS#=#AVS#	Call a JCL element.
#AVD#	Call an external element.
#AVJ#=#AVJ#	Convert an #AVJ# statement to a /MODIFY-JV command for the monitoring job variable.
#AVA#	Enter information in the journal.
P#=P#	Symbolic name for a variable from job masks.
N#=N#	Symbolic name for a variable from net masks.
F#=F#	Symbolic name for a variable from the USER-PARAM-FILE.
S#=S#	Symbolic name for system variables.
#RA=#RA	Exchange next statement in the restart.
#RI=#RI	Insert statement in restart.
#RU=#RU	Suppress next statement in restart.

When a symbolic name is assigned a different character string, the preset number of characters must be maintained.

Example

```
#AVM#=$MAS$
P#=J#
N#=$J
```

It is advisable to retain the standard character strings since processing for modification is optimized on the basis of these strings.

2.2.3 Modifying the system parameters of an installed AVAS system

Modifications in the GENPAR generation file do not take effect in the AVAS system until the AVS.GENSYSPAR procedure (an element in the library SYSPRC.AVAS.085) has been used to generate a new AVAS system parameter file and the central access tasks are started using this new system parameter file.

If the AVAS system ID is changed in the generation parameters, the generation run creates system parameters for another AVAS system. For this reason, the following should be noted when changing the individual system parameters:

Modifying the system parameters of the central access tasks

AVAS-SYSTEM-ID

If the AVAS system ID is modified, system parameters are generated for another AVAS system.

This system can no longer access the run control file or the journal file, since the files were formatted under a different AVAS system ID.

CALLIB

PERDAT

If the names of the calendar library and the period file are changed, existing dates can no longer be accessed, unless they are copied to the new files.

Net and job libraries

COPY-ELEMENT can be used to copy nets and jobs to the new libraries.

Run control and journal files

If the names of the run control file and the journal file are changed, nets already released for processing can no longer be processed and edited.

Both files must be reformatted before the central access tasks can be started.

Modifying the users system parameters

BKSYS

If the name of the system user group is changed, the nets and jobs of the system user group can no longer be accessed. The jobs used in the nets, for which the previously valid system user group was specified, can no longer be retrieved.

Modification within a user group (BK)

If other NETnnn and/or JCLnnn libraries are assigned to a user group, the group will forfeit access to the elements in libraries used up until then.

By changing the name of calendar and/or avak, a different calendar and/or a different run control system can be assigned to a user.

Definition of a new user group (BK)

New user groups can be used without restrictions once the access tasks with the modified system parameters have been started.

USER

The definitions of users can be changed without restrictions, and new users can also be defined.

If the user group to which the user is assigned is changed, the user forfeits access to the elements of his previous user group, provided that he is not privileged and no other libraries have been assigned to the new user group.

Production plans and mask libraries

New production plans and mask libraries can be defined without restrictions.

If the assignment of libraries is modified, access to these old libraries and the elements therein is forfeited.

Function authorizations

The tables with function authorizations can be modified without restrictions.

New tables can be created and assigned to users.

Modifying the run control systems

The parameters of the run control systems can be changed without restrictions.

Modifying the name of a run control system is equivalent to deleting a run control system and defining a new one. It should be noted that the nets under a “deleted” run control system can no longer be processed.

In the run control and journal files, however, the run control system is not deleted, so that the nets stored under the run control system can still be accessed via the dialog functions (e.g. using the MODIFY-SUBMIT-NET command in order to shift nets to another run control system).

When starting the access task for the run control file and the journal file, newly-defined run control systems are entered in the directories of the file and can be used without restrictions.

Modifying the symbolic names of the AVAS statements and variables

If the symbolic names of the AVAS statements and/or AVAS variables are modified, all jobs in which these statements and variables are used must be adapted accordingly.

2.2.4 Creating the generation parameters from the system parameters

In an active AVAS system, the generated system parameters can be modified in dialog mode. This changes the system parameter information (SYSPAR) compared to the originally used generation parameters (GENPAR) in a productive system run. A new GENPAR file can be created from an old GENPAR file and a SYSPAR file using the AVS.CREATE-GENPAR procedure (element in the SYSPRC.AVAS.085 library).

When the new GENPAR file is created, the generation statements (GEN statements) USER, USVAR and FUTABnnn from the old GENPAR file and from the SYSPAR file are taken into consideration. By default, USER, USVAR and FUTABnnn are taken from the SYSPAR file. All other GEN statements are transferred from the old GENPAR file into the new one without any modifications.

2.2.4.1 Syntax of the generation statements and creation rules

The statement names of all descriptions must start on position 1. The parameters in continuation descriptions should be defined as of position 2. Descriptions with the same statement name must be defined consecutively in the proper order.

The GEN statements from the GENPAR file (USER, USVAR and FUTABnnn) may not be described using continuation lines.

Description of a user

USER=(--name--, -passw-, -ugr-, -futab#, -k2)

Description	Position		Meaning
	from	to	
USER	01	08	Generation statement
=	09	11	Separator between statement and parameters
(12	12	Left bracket – beginning of the parameters
name	13	20	Name of user to be generated
,	21	21	Separator between two parameters
passw	22	29	Password for each generated user
,	30	30	Separator between two parameters
ugr	31	35	Default user group of the generated user
,	36	36	Separator between two parameters
futab#	37	44	Name of the function table for the generated user
,	45	45	Separator between two parameters

Description	Position		Meaning
	from	to	
k2	46	48	K2 key authorization for the generated user
,	49	49	Separator between two parameters
visibility	50	52	Authorization for making passwords visible to the generated user
,	53	53	Separator between two parameters
manage-srv	54	56	Authorization for server management for the generated user
)	57	57	Right bracket – end of the parameters

Description of a user's system variable

USVAR=(nnn,'-----variable-----')

Description	Position		Meaning
	from	to	
USVAR	01	08	Generation statement
=	09	11	Separator between statement and parameters
(12	12	Left bracket – beginning of the parameters
nnn	13	15	Number of the users' system variables
,	16	16	Separator between two parameters
variable	17	64	Variable value, possibly in inverted commas
)	18	65	Right bracket within from/to range – at end of parameters

Description of a function table

FUTABnnn=(-----statement-----/b)

Description	Position		Meaning
	from	to	
FUTABnnn	01	08	Generation statement and number
=	09	11	Separator between statement and parameters
(12	12	Left bracket – beginning of the parameters
statement	13	42	Statement to be assigned an authorization
/	43	43	Separator between parameter descriptions
b	44	44	Authorization code for a statement
)	45	45	Right bracket – end of the parameters

When the new GENPAR file is generated, all statements per key argument (statement and first parameter) are taken from an old GENPAR file and from the SYSPAR file. This also applies to the comments (* in position 1). The order of the statements is defined by the old GENPAR statement.

Descriptions cannot be removed with the generation program. Descriptions that are contained per key argument in both the old GENPAR file and in the SYSPAR file are transferred according to fixed processing rules.

Processing the user description (USER)

The description from the SYSPAR file is always used. If the generation program recognizes that password encryption takes place in the system, the password for the relevant AVAS user is not taken from the SYSPAR file, but from the old GENPAR file. If the user is not defined in the old GENPAR file, "AVAS" is used for the password.

Processing the user's system variables (USVAR)

The description from the SYSPAR file is always used. The description does not end (right bracket) on a defined position. It is directly after the variable described.

Description of a function table

The description from the SYSPAR file is always used.

Note

The old GENPAR file does not necessarily contain the entire description of an AVAS system. It may only contain the statements used in this specific program. If the old GENPAR consists of a comment line only, for instance, the generation program will take the GEN statements to be processed (USER, USVAR etc.) only from the SYSPAR and the GEN statements will be saved in a new GENPAR file. In order to make proper use of the file, however, the file will have to be included manually in a GENPAR file for AVAS system generation.

2.2.4.2 Procedure for creating GENPAR descriptions

The program with which a new GENPAR file is generated from an old GENPAR file and from a SYSPAR file is activated using the AVS.CREATE-GENPAR procedure (element in the SYSPRC.AVAS.085 library). The following information is supplied or prompted in procedure variables in order to control the program:

&SYSPAR	Name of an ISAM file (SYSPAR) Input file with generated system parameters.
&GENOLD	Name of an SAM file (GENPAR) Input file containing descriptions for generating a SYSPAR file.
&GENNEW	Name of an SAM file (GENPAR) Output file containing descriptions for generating a new SYSPAR file.
&UPDUSER	Program start parameter UPDATE-USER={NO / <u>YES</u> } NO The user description is to be taken from the &GENOLD file. <u>YES</u> The user description parameters are to be taken from the &SYSPAR file.
&UPDFUTAB	Program start parameter UPDATE-FUTAB={NO / <u>YES</u> } NO The function tables are only to be taken from the file &GENOLD. <u>YES</u> The function table parameters are to be taken from the file &SYSPAR.
&UPDUSVAR	Program start parameter UPDATE-USVAR={NO / <u>YES</u> } NO The user system variables are to be taken only from the file &GENOLD. <u>YES</u> The user system variables are to be taken from the file &SYSPAR.

Example: generating a new GENPAR file

```
/CALL-PROC $AVAS.SYSPRC.AVAS.085(AVS.CREATE-GENPAR)
/ REMARK USERID OF THE AVAS INSTALLATION           : &USERID
&USERID=AVAS
/REMARK FILENAME, ORIGINAL / OLD GENPAR           : &GENOLD
&GENOLD=AVAS.USER.GENPAR-OLD
/ REMARK FILENAME, CURRENT SYSTEM PARAMETER       : &SYSPAR
&SYSPAR=AVAS.USER.SYSPAR
/ REMARK FILENAME, FUTURE / NEW GENPAR           : &GENNEW
&GENNEW=AVAS.USER.GENPAR-NEW
OLD GENPAR = AVAS.USER.GENPAR-OLD
  SYSPAR = AVAS.USER.SYSPAR
NEW GENPAR = AVAS.USER.GENPAR-NEW
% BLS0500 PROGRAMM 'AVASCGP', VERSION 'V8.5A' OF '2010-12-01' LOADED
% BLS0552 COPYRIGHT (C) FUJITSU TECHNOLOGY SOLUTIONS GMBH 2010. ALL RIGHTS
RESERVED
% AVS2201 START 'AVASCGP'; 'V8.5A00'; '2010-12-01/11:56:44'
% AVS8301 INPUT 'UPDATE-USER=YES'
% AVS8301 INPUT 'UPDATE-FUTAB=YES'
% AVS8301 INPUT 'UPDATE-USVAR=YES'
% AVS8301 INPUT 'END'
% AVS2310 SYSTEM 'AVASCGP'; 'V8.0A00'; NORMALLY TERMINATED
```

2.2.5 Creating and formatting the AVAS files

There are two kinds of file in the AVAS system: system files (NETSYS, DOCSYS, JCLSYS and JMDSYS), which can only be modified and evaluated by the AVAS administrator, and files with user data, which can be modified (or at least evaluated) by the user.

Within AVAS, files with user data are not accessed directly from the interactive, processing or analysis programs, but rather via central access tasks (ZDs for short, German abbreviation). The connection between the programs and the central access tasks is set up via an interface (ZDI) which forwards the requirements of the programs to the central access task via the ITC interface and passes the data received from the ZD to the programs. Communication between the programs and the ZDI takes place via an internal macro interface.

The program requirements are distributed to two access tasks via the ZDI. Access operations for libraries are forwarded to the PLAM-ZD, while those for the run control file, journal file, emergency journal file and period file are sent to the UPAM-ZD.

A program is connected to both access tasks if it wishes to access files managed by both ZDs. For this reason, both ZDs have access to the system parameters, and forward the required data from the system parameters to the signon task. If a program only accesses one group of files (e.g. the run control systems only access the run control file and the journals), it only signs on at the associated central access task (e.g. UPAM-ZD).

All AVAS files can also be maintained as NK4 files (see the “Introductory Guide to DMS” [5]).

2.2.5.1 User files and their symbolic names

This overview presents the user files with the symbolic names as given in the sample generation file. The user files contain data created by the user, e.g. jobs, net descriptions, etc.

- Generation file: GENPAR

The generation parameters described on [page 23](#) must be defined in this SAM file. The file can be created or edited with the aid of EDT.

- Period file: PERDAT

This ISAM file is used to store the periods (time intervals within the calendar).

- Calendar library: CALLIB

This library is used to store the calendars defined by the generation parameters or the net descriptions for each user group.

- Net libraries: NETSYS and NETLIB

The net libraries are used to store the net descriptions. The net descriptions in the system net library (NETSYS) are available to all users throughout the system; those in the user net library (NETLIB) are only available to users of the associated user group.

- Libraries for jobs and JCL elements: JCLSYS and JCLLIB

These libraries are used to store the jobs and JCL elements. The remarks above pertaining to net libraries also apply to JCLSYS and JCLLIB.

- Documentation libraries: DOCSYS and DOCLIB

These libraries are used to store the documentation elements. The documentation elements in the DOCSYS are available throughout the system. Editing of the elements in the DOCLIB can be restricted to the user groups of the users.

- Log data library: LOGSYS

This library is used as a central store for the job runtime logs.

- Production net libraries: NPRLIB

These libraries are used to store the planned nets.

- Libraries for production jobs: JMDSYS and JMDLIB

These libraries are used to store the executable jobs, i.e. jobs which no longer need to be modified for execution purposes. The remarks given above for net libraries also apply to JMDSYS and JMDLIB.

- Mask libraries: MAPLIB

The mask libraries with the FHS mask modules must be created by the user in accordance with the FHS guidelines. An FHS mask module library can be assigned directly as a MAPLIB.

The distinction between libraries with net modification masks (NETMAP) and those with job modification masks (JOBMAP) is made via the generation parameters during assignment.

Both of the following AVAS files are PAM files which must be formatted before being used in the AVAS system. Formatting is handled by the procedure AVS.FORM (an element in the library SYSPRC.AVAS.085). The number of blocks to be formatted is preset within AVAS with a primary allocation (PRIMARY-ALLOCATION) of 240 blocks and a secondary allocation (SECONDARY-ALLOCATION) of 30 blocks.

If the files are to be formatted with different values, these values can be specified using the /CREATE-FILE command. You must make sure that any SPACE entries are increased to a multiple of 30.

If SECONDARY-ALLOCATION=0 is specified, the file is not extended.

Example

Preset:	Formatting:
None (or SPACE=*STD)	SPACE=*RELATIVE(240,30)
SPACE=*RELATIVE(0,0)	SPACE=*RELATIVE(240,0)
SPACE=*RELATIVE(3,3)	SPACE=*RELATIVE(120,30)
SPACE=*RELATIVE(120,0)	SPACE=*RELATIVE(120,0)
SPACE=*RELATIVE(3000,3)	SPACE=*RELATIVE(3000,30)

If no explicit value is specified for the secondary allocation, the file is extended via the secondary allocation entry in the file catalog. Formatting is always performed for the run control file and the journal file. Any data in these files is then lost.

- Run control file: ABLDAT

The run control file contains all the information related to controlling the operations of the attached run control systems.

- Journal file: JRNDAT

The journal file is used to log the actions of the users at the system and the activities of the run control system.

- Emergency journal file: JRLDAT

The emergency journal file of the AVAS system is set up by AVAS during the current session. Journal records which cannot be output to the journal file because of an error are stored in the emergency journal file. To obtain a complete list of the journal records, the journal file and the emergency journal file must be merged before the list is generated.

Note

During startup of the ZDD, all the emergency journal files are merged into the journal file.

2.2.5.2 AVAS libraries

Every AVAS user, identified by a user ID and a password, is assigned libraries in the system parameters via his user group.

For some libraries, in addition to libraries assigned via the user group, there are others with elements which are available throughout the system (JCLSYS, DOCSYS, NETSYS and JMDSYS).

The only way to change the assignment is to change the system parameters using MODIFY-SYSTEM-PARAMS.

For all users of a user group, a change of assignment does not take effect until all users in this group have signed off, thereby releasing the libraries of the user group. The modified library assignments will be effective the next time a user signs on with this user group.

A library is opened the first time that it is accessed. It remains open until the last user working with it has signed off from the communication interface.

2.2.5.3 AVAS version changes with ABLDAT and JRNDAT

When changing the version, the runtime file (ABLDAT) and the journal file (JRNDAT) can be taken over to avoid having to reenter the condition entries. The SYSTEM-ID and the R-C-S names must be modified to permit the data from the old version to be used.

Transferring the runtime file and the journal data from the previous version

When converting from AVAS Version 4.0A to 4.1A or 5.0A, the data from the runtime file (ABLDAT) and the journal file (JRNDAT) can be transferred to the new version. To do so, the system ID and the names of the run control systems must be changed in the files for the new version. These modifications are made using the BS2000 utility routine DPAGE.

The following steps are required:

- Terminate the old system version **normally**.
This also applies to the registered run control systems (according to the entries in the runtime file).
- Save the old file versions under a new name.
- Generate the system parameters for the new version.
- Start the AVS.UPDATE.VERSION procedure.
The first run control system must be written in the same order as in the SYSPAR file and the ABLDAT file. The following parameters are required:

&ABLDAT=	Name of the runtime file (ABLDAT)
&JRNDAT=	Name of the journal file (JRNDAT)
&SYSP=	Name of the new system parameter file (SYSPAR)
&OLDSYSID=	Name of the old system ID(SYSTEM-ID)
&NEWSYSID=	Name of the new system ID (SYSTEM-ID)
&OLDRCS1=	Old name of the first run control system (R-C-S)
&NEWRCS1=	New name of the first run control system (R-C-S)
&LISTE=	Name of the file for logging the modifications
&JVNAM=	Prefix for job variables
&SAVE=	ABLDAT and JRNDAT saved?
Y:	continue with the procedure
N:	cancel; no UPDATE

&GEN=	Generate system parameter SYSPAR?
	Y: continue with the procedure
	N: cancel; no UPDATE
&START=	Old and new values tested; start?
	Y: continue with the procedure
	N: cancel; no UPDATE
&UPDATE=	Function:
	Y: continue with the procedure
	N: cancel; no UPDATE

Notes

- “OLD...” indicates that the values are from the runtime file and the journal file of “old” system version.
- “NEW...” indicates that the values are from the system parameters of the “new” system version.

How the procedure works

1. The user is asked whether a backup copy of the runtime file and the journal file has been made.
Answer: YES: The procedure is continued.
NO: The procedure is cancelled.
2. The user is asked whether the system parameters for the new AVAS version have been generated.
Answer: YES: The procedure is continued.
NO: The procedure is cancelled.
3. The user is prompted to enter the name of the new system parameter file SYSPAR and the name of the new runtime file ABLDAT and journal file JRNDAT.
4. The user is prompted to enter the name of the old AVAS-SYSTEM-ID (OLDSYSID) and of the new AVAS-SYSTEM-ID (NEWSYSID).
5. The user is prompted to enter the name of a LIST file, a prefix for job variables and the USER ID under which the DPAGE utility routine is located.
6. The new system parameter file SYSPAR is read with EDT. The names of the new AVAS-SYSTEM-ID (NEWSYSID), runtime file ABLDAT and journal file JRNDAT are checked as follows:

'Entry' EQ 'value from system parameter file'

If the two values do not tally, job switch 11 is set. EDT is terminated and then job switch 11 is queried.

Result: OFF: The procedure is continued.
ON: The procedure is cancelled.
7. The user is prompted to enter the name of the old first run control system OLDRCS1 and the name of the new first run control system NEWRCS1.
8. The new runtime file ABLDAT is opened with DPAGE. The file control block is output to the file allocated via &LISTE. The same is also done with the new journal file JRNDAT.

9. The file allocated with &LISTE is read with EDT (DPAGE output). The name of the old AVAS-SYSTEM-ID OLDSYSID is tested as follows:
'Entry' EQ 'value from runtime file'
If the two values do not tally, job switch 11 is set.
The name of the old run control system OLDRCS1 is tested as follows:
'Entry' EQ 'value from runtime file'
If the two values do not tally, job switch 11 is set.
It is checked whether further run control systems (OLDRCS2 to OLDRCS6) are contained in the runtime file and whether they were terminated normally. If a run control system is found that was not terminated normally, job switch 11 is set.
It is checked whether more than 5 run control systems are contained in the runtime file. If the run control file contains more than 5 run control systems, job switch 11 is set.
If ON is set, job switch 11 is queried and EDT is terminated. The new names of the run control systems are not determined.
The specified new system parameter file SYSPAR is read with EDT. The names of the new run control systems (NEWRCS1 to NEWRCS6) are determined and output to job variables called &JVNAME..NRCS1 to ..NRCS6 with /MODIFY-JV.
10. The values entered and those determined with the runtime file and the system parameter file are output to the file allocated with &LISTE.
EDT is terminated.
11. &LISTE is displayed with SHOW-FILE.
12. Job switch 11 is queried in the procedure.
Result: OFF: The procedure is continued
ON: The procedure is cancelled.
13. The user is asked whether the new values for the AVAS-SYSTEM-ID and the run control systems (NRCS1 to NRCS6) are to be entered in the runtime file and the journal file.
Answer: YES: The procedure is continued
NO: The procedure is cancelled.
14. The specified new values are included in the runtime file ABLDAT and in the journal file JRNDAT with DPAGE. The control blocks of the modified runtime file ABLDAT and the modified journal file JRNDAT are output to the file allocated with &LISTE using MODE=EXTENT.

15. The file allocated with &LISTE is read with EDT. The new values for the AVAS-SYSTEM-ID and the run control systems NEWRCS1 to NEWRCS6 are determined. The output from DPAGE is deleted and the new values are entered in the &LISTE file. &LISTE is displayed with SHOW-FILE.
16. The &JVNAME job variables are deleted.

Notes

- The job switches 10 and 11 control whether processing is cancelled. Job switch 10 is set when the user decides not to proceed with the UPDATE. Job switch 11 is set when an error is detected while checking the entries or the data in the system parameter file, runtime file and journal file.
- The name of the first old and new run control system must be specified for control purposes.
- Files (ABLDAT) with more than 5 run control systems are not updated.
- The update is not performed if the existing run control systems were not terminated normally.
- The update is not performed if more than 5 run control systems are defined in the new system parameters.
- No copies of the runtime file and the journal file are made.
- The values in the runtime file are not compared against the values in the journal file. The system only checks whether the old AVAS-SYSTEM-ID and the old name of the first run control system tally with the values in the journal file.
- The new names in the run control system are buffered in job variables. The system does not check whether their contents is modified in the course of the procedure

2.3 CC exits in the AVAS system

The CC (computer center) exits enable users to incorporate their own CC routines (written in assembly language) in processing of the AVAS system. This gives them the ability to influence subsequent processing.

CC exit AVEX0001 and AVEX0002: Journal output

Journal records are output to the journal file by individual statements (see “Table of journal records” in the “AVAS Functions and Tables” manual [1]) and by the run control system. Before AVAS outputs the records, they are passed to a CC routine via the CC exits AVEX0001 and AVEX0002.

The journal records are still transferred in the format “output date without century” in the CC exit AVEX0001.

The journal records are transferred in the format “output date with century” in the CC exit AVEX0002.

Output of the following journal records can be suppressed with the aid of the CC routine:

Statement	SSL	Function
CREATE-PLAN-NET	12-	Output structure element
	12-	Delete structure element
CREATE-PROD-NET	21-	USER-PAR-FILE parameter
	22-	Assign mask
	23-	Call element
	25-	Parameter substitution
EDIT-PROD-JOB	65-	JCL modification
SUBMIT-NET	25-	JCL statement EX7102
	32-	Output BS2000 job / S procedure
	54-	Output structure element (not BS2000 job / S procedure)
REPEAT-NET	25-	JCL statement EX7102
	32-	Output BS2000 job / S procedure
	54-	Output structure element (not BS2000 job / S procedure)
MODIFY-SUBMIT-JOB	65-	JCL modification

Notes

- Execution of the AVAS function cannot be influenced by the CC exit.
- The contents of the journal records are made available.
- The start, end and abnormal termination messages for the functions cannot be suppressed.
- Opting not to output to the journal can under certain circumstances result in considerable storage savings and a reduction in paging operations. However, record types should only be suppressed provided that auditing integrity can be guaranteed by other measures and that no subsequent procedures assume the presence of these record types.
- For the analysis of errors requiring the journal records, a journal containing all the records must be generated.

The exits AVEX0001 and AVEX0002 are activated before a journal record is output to the journal file as a result of an AVAS function.

CC exit AVEX0101: Saving a net description

After a net description has been written back to the NETLIB, AVAS transfers control to the CC routine. The user can generate a history of the creation of and modifications to the net description.

CC exit AVEX0102: Saving a job description

After a job description has been written back to the JCLIB or JMDLIB, AVAS transfers control to the CC routine. The user can generate a history of the creation of and modifications to the job description.

CC exit AVEX0401: Access the JCL of a BS2000 job or an S procedure in a temporary file

- BS2000 job, S procedure
After output of a BS2000 job to the temporary ENTER file and prior to the ENTER call, AVAS transfers control to the CC routine. This routine obtains all the information required to identify the BS2000 job and to access the ENTER file. Data (e.g. passwords) can thus be modified immediately before starting a BS2000 job. The passwords are transferred by AVAS in unencrypted form. In addition, it is also possible to prevent processing of a BS2000 job.

The BS2000 job start can be checked, permitted or rejected by the run control system.

The JCL of the BS2000 job in the temporary ENTER file can be checked and updated. The "job start" information can be passed to other program systems or work areas.

The AVEX0401 exit is activated if the run control system transfers the JCL of a BS2000 job from the run control file to the temporary ENTER file in full and without errors.

- FT request:
The AVEX0401 exit is not activated when FT requests are started.

CC exit AVEX0402: Net status changes from RUNNING

A net which is executing with the status RUNNING is given some other status value by the run control system (e.g. ENDED, ERROR). The CC routine is given control at this point.

The interruption of net execution by the run control system can be checked and logged. Information on the name of the net and the new status can be forwarded to other program systems or functional areas.

Exit AVEX0402 is activated when a net with the status RUNNING is placed in a different status and cannot be brought to execution again by means of an automatic restart.

CC exit AVEX0403: BS2000 job or S procedure termination recognized

A task has been given the status ENDED, ABENDED or ERROR. The CC routine is given control at this point and receives all the information it needs in order to detect the position in the net and the execution conditions in the system.

The contents of the task job variables are passed to the CC routine for evaluation.

Normal or abnormal job termination can be checked and logged. The "job termination" information can be passed to other program systems or work areas.

Exit AVEX0403 is activated when a BS2000 job, an FT request or an S procedure has been given the status ENDED, ABENDED or ERROR in the run control file by the run control system.

CC exit AVEX2001: Activation of a CC routine by the user via START-EXIT

Here the user can incorporate his own user routine into the processing of the AVAS system.

Exit AVEX2001 is activated when the user enters the START-EXIT statement. No data is transferred to the CC routine in the communication area. The CC routine can return a message, which is output in the MSG field of mask AVS030. Return codes in the RETURNCODE field are not analyzed by AVAS.

CC exit AVEX6601: Entries for net masks

If JCL statements for a job are to be modified with the aid of a user mask predefined in the net (operand OBJECT=MAP in the CREATE-NET-DESC statement), the COLLECT-NET-PARAMS statement must be used for collecting the entries for the user mask. After the statement has been called, the current values of the run control parameters can be assigned to the AVAS variables N#nnn by means of a CC routine. The values passed can be stored directly in the net, or storage can be aborted.

When no further user intervention is required, interactive output of the user mask can be dispensed with. Otherwise, the user mask containing the values passed will be output.

Exit AVEX6601 is activated before an assigned user mask is output.

The CC routine can pass the entries for the user mask for the purposes of editing or output.

CC exit AVEX6602: Transfer of values from mask fields of a net mask

If JCL statements for a job are to be modified with the aid of a user mask predefined in the net (operand OBJECT=MAP in the CREATE-NET-DESC statement), the COLLECT-NET-PARAMS statement must be used for collecting the entries for the user mask. At CC exit AVEX6602 the current values from the user mask which are assigned to the AVAS variables N#nnn are passed to the CC routine.

Modification using the values passed can be directly permitted, aborted or rejected with a message. If the CC routine reports an error in the RETURNCODE field, an error message is output in the MSG field of the user mask.

Exit AVEX6602 is activated before the values of the N#nnn variables are stored (CMD: CONTINUE in the user mask).

The entries from the user mask are passed to the CC routine for checking.

CC exit AVEX6801: Entries for job masks

If JCL statements for a job are to be modified with the aid of a user mask specified in the job (AVAS statement #AVM#) or a user mask assigned by means of the FORMAT-NAME operand, the current values of the run control parameters can be assigned to the AVAS variables P#nnn by means of a CC routine. These values remain valid until the user mask is next called by the job or until the end of the job.

Modification using the values passed can be directly permitted or aborted.

When no further user intervention is required, interactive output of the user mask can be dispensed with. Otherwise, the user mask containing the values passed will be output.

Exit AVEX6801 is activated before an assigned user mask is output.

The CC routine can pass the entries for the user mask for the purposes of editing or output.

CC exit AVEX6802: Transfer of values from mask fields of a job mask

If JCL statements for a job are to be modified with the aid of a user mask specified in the job (AVAS statement #AVM#) or a user mask assigned by means of the FORMAT-NAME operand, the current values assigned to the AVAS variables P#nnn are passed from the user mask of the user routine.

Modification using the values passed can be directly permitted, aborted or rejected with a message. If the CC routine reports an error in the RETURNCODE field, an error message is output in the MSG field of the user mask.

Exit AVEX6802 is activated before the values of the P#nnn variables are modified (CMD: CONTINUE in the user mask).

The entries from the user mask are passed to the CC routine for checking.

CC exit AVEX7101: Release a planned net

The release of a net can be controlled, logged, permitted or rejected.

The "release of net" announcement can be forwarded to other program systems (e.g. MAREN in order to make data volumes available).

If, for example, production area runs are controlled in the computer center by means of information which is unknown to the AVAS system and cannot be made known to it, the user has the option of preventing these production areas from starting.

Exit AVEX7101 is activated when AVAS has checked the release of the net and has not detected any faults (the net does not yet exist in the run control file; all jobs called in the net exist).

The net element in the NPRLIB is locked; the jobs in the JMDLIB and JMDSYS are not locked.

If a net group is to be released, the CC exit must be activated for each net in the group.

CC exit AVEX7102: JCL modification with net release

- BS2000 job, S procedure
Statements or parameters such as passwords and DMS statements can be entered in or deleted from the BS2000 jobs (JCL statements).
The JCL of the BS2000 job can be checked, logged and modified. The “job release” announcement can be forwarded to other program systems or work areas (e.g. tape operations with files).

If some of the JCL announcements cannot be included in the JCL run via the AVAS-specific modification options, or if the user wishes to modify or expand the JCL statements for programming reasons, each JCL statement can be presented for editing via this CC exit.

2.3.1 Linking CC routines in AVAS

The CC routines must be linked with an AVAS system module and an AVAS interface module to form a common AVAS-EXIT module AV03EXIT, AV04EXIT or AV06EXIT.

When the AVAS system is loaded, these modules are sought in the module library assigned with the link name SYSLNK or with TASKLIB.

The following link statements must be entered in the specified sequence in order to create the EXIT module:

```
MODULE AV03EXIT,...
INCLUDE AV03EXL,(avas-system-library)
INCLUDE AV03EXT,(avas-system-library)
INCLUDE cc-routines
```

The same order also applies to the AV04EXIT and AV06EXIT modules:

```
MODULE AV04EXIT,...
INCLUDE AV04EXL,(avas-system-library)
INCLUDE AV04EXT,(avas-system-library)
INCLUDE cc-routines
```

```
MODULE AV06EXIT,...
INCLUDE AV06EXL,(avas-system-library)
INCLUDE AV06EXT,(avas-system-library)
INCLUDE cc-routines
```

Example

```
MODULE AV03EXIT,...
INCLUDE AV03EXL,(avas-system-library)
INCLUDE AV03EXT,(avas-system-library)
INCLUDE AVEX7102,(user-module-library)
INCLUDE AVEX7101,(user-module-library)
etc.
```

It is not necessary to specify a CC routine for every possible CC exit.

The SYSLNK.AVAS.085 library must be assigned as the AVAS system library.

Object modules AV03EXIT, AV04EXIT and AV06EXIT without CC exits are supplied in system library SYSLNK.AVAS.085. These modules must be deleted if the user's own AV..EXIT modules are to be used with CC routines.

Notes

- The object modules which have been supplied should be saved before any deletion occurs.

- No calls to terminate the process (e.g. TERM) should be programmed in the CC routines. Use of a separate STXIT mechanism is permitted. AVAS resets the STXIT parameters to its own values after each return.

Assignment of CC exits to the EXIT module

CC exit	AV03EXIT	AV04EXIT	AV06EXIT
	DIALOG BATCH	AVAK	REORG
AVEX0001 and AVEX0002	yes	yes	yes
AVEX0101	yes	–	–
AVEX0102	yes	–	–
AVEX0401	–	yes	–
AVEX0402	–	yes	–
AVEX0403	–	yes	–
AVEX2001	yes	–	–
AVEX6601	yes	–	–
AVEX6602	yes	–	–
AVEX6801	yes	–	–
AVEX6802	yes	–	–
AVEX7101	yes	–	–
AVEX7102	yes	–	–

Register conventions

AVAS passes addresses to the CC routines in registers 1, 13, 14 and 15 with the contents given below:

- Register 1: Address of a parameter address list (two words):
 Word 1: address of the information area
 Word 2: address of the communication area
- Register 13: Address of the register backup area (18 words).
- Register 14: Address for the return to AVAS.
- Register 15: Address of the CC exit.

2.3.2 Integration of MARENAV

The following points should be noted when linking the MARENAV module to the CC exit AVEX7102:

- The entry COLBIN is indicated in MARENAV for later link runs (LINK-SYMBOLS KEEP=COLBIN).
- The MARENAV module is a COLUMBUS procedure with runtime system written in Assembler. Please refer to the program link rules relating to this (see the “ASSEMBH” manual [3]).

2.3.3 Connecting CC routines along with MARENAV

If one or more CC routines are to be connected to the CC exits AVEX7101 and AVEX7102, the AV03EXTV module can be used. AV03EXTV enables branching to individual CC routines.

To do this, the AV03EXTV source must be modified and recompiled. The AVEXSVV macro is used to define the CC routines on the relevant CC exit to which branching is to be performed.

Macro	Operands
AVEXSVV	exitno=nnnn ,addr / (list)

exitno Number (4 digits) of the CC exit to be processed (e.g. 7102 for the CC exit AVEX7102).

addr Symbolic address (ENTRY in the CC routine), to which control is passed from AV03EXTV using the interface of the CC exit.

(list) A number of ENTRY addresses can be specified in a list (e.g. (addr1,addr2,addr3)).

Note

AV03EXTV handles the return codes that are reported back from the CC routines to AVAS in the following way:

- An error situation is immediately reported back to AVAS and no further CC routines are called.
- In the normal situation, the next CC routine is called using the interface of the CC exit.

- If other CC routines are to be connected to AV03EXTV in addition to MARENAV, the call to MARENAV must be made last so that its return code (e.g. “Accept modified JCL statement”) is reported back to AVAS.
- If the user wants to read the JCL statements that were updated by MARENAV in a separate CC routine, AV03EXTV must be modified in such a way that the following CC routine uses the MARENAV interface rather than the interface of the CC exit.
- During compilation, the library SYSLIB.AVAS.085 must be assigned as OLDLIB.

In order to create the EXIT module using MARENAV, the following link statements must be specified in the given sequence:

```
MODULE AV03EXIT,...  
INCLUDE AV03EXL,(avas-system-library)  
INCLUDE AV03EXT,(avas-system-library)  
INCLUDE AV03EXTV,(avas-system-library)  
INCLUDE MARENAV,(avas-system-library)  
etc.
```

In order to create the EXIT module using the CC routines USERPROG and MARENAV, for example, the following link statements must be specified in the given sequence:

```
MODULE AV03EXIT,...  
INCLUDE AV03EXL,(avas-system-library)  
INCLUDE AV03EXT,(avas-system-library)  
INCLUDE AV03EXTV,(avas-system-library)  
INCLUDE AVEX7102,(user-module-library)  
INCLUDE MARENAV,(avas-system-library)  
etc.
```


2.3.4 Information exchange between the AVAS system and CC routines

In both directions, the exchange of information between the AVAS system and the CC routines takes place by way of data areas which are made available by AVAS and which can be emulated by the CC routines.

A distinction is made between the information area and the communication area. The information area is shared by all CC routines, while each CC routine has its own communication area.

2.3.4.1 Information area

The information area consists of fields in which AVAS passes general information and announcements to the CC routines:

These fields include:

- the BS2000 task sequence number
- the name of the AVAS system ID
- the AVAS user name
- the AVAS user group
- the name of the AVAS run control system
- the calendar name
- as well as the length of the information area.

The CC routine passes the function code to AVAS in the RETURNCODE field. These codes are described under the individual CC exits.

Field descriptions can be created by means of the AVASEXKO macro (SYSLIB.AVAS.085 macro library). It must be preceded by a DSECT statement issued by the user.

The call has the following format:

```
(name) AVASEXKO EXITNAM=USINFO[,PREFIX=(pfix)]
```

where "pfix" may be a prefix of up to 3 characters.

Example

```

(name)  DSECT
USINFO  AVASEXKO EXITNAM=USINFO,PFIX=VUS
USINFO  DS      OF
*-----*
*              AVAS - EXIT - INFORMATION AREA              *
*-----*
*
VUSLEN  DS      F              LENGTH OF INFORMATION AREA
VUSRTC  DS      OF            RETURN CODE OF EXIT ROUTINE
        DS      CL3
VUSRSC  DS      C              FUNCTION CODE
VUSOK   EQU     X'00'          EXECUTE FUNCTION
VUSUPD  EQU     X'02'          PERFORM MODIFICATION
VUSNO   EQU     X'04'          IGNORE FUNCTION
VUSIGN  EQU     X'06'          IGNORE FUNCTION STEP
VUSABE  EQU     X'08'          ABORT FUNCTION
VSNUIN  EQU     X'12'          PERFORM UPDATE / INSERT
VUSUDE  EQU     X'14'          PERFORM UPDATE / DELETE
VUSTSN  DS      F              BS2000 TSN
VUSAPP  DS      CL7            AVAS SYSTEM ID
VUSUSR  DS      CL8            AVAS USER NAME
VUSBKR  DS      CL5            AVAS/NET USER GROUP
VUSASK  DS      CL8            RUN CONTROL SYSTEM
VUSCAL  DS      CL20           CALENDAR NAME

```

2.3.4.2 Communication areas

The communication area contains data which AVAS passes to the CC routine from the function. The corresponding communication area for each CC exit must be used.

The field descriptions of the communication areas may likewise be created by means of the AVASEXKO macro. It must be preceded by a DSECT statement issued by the user.

The call has the following format:

```
(name) AVASEXKO EXITNAM=exit[,PFIX=(pfix)]
```

where "exit" specifies the name of a CC exit and "pfix" may be a prefix of up to 3 characters which can be prefixed to the field name (which in turn may be up to 4 characters long).

The following fields in the communication area are given identical values for all functions. They may be modified by the CC routine.

prefixLEN Output parameter.
 Total length of the communication area.

prefixLFU Input/output parameter.
 Length of the data which AVAS passes to the CC routine linked with this field.
 Depending on the function involved, the CC routine may modify this length if it is not the same as the return length. The modified length must not exceed the predefined boundary of the communication area.

Example

```
(name) DSECT
EX7101 AVASEXKO EXITNAM=EX7101,PFIX=A71
EX7101 DS      0F
*-----*
*                AVAS - AVEX7101 - COMMUNICATION AREA                *
*-----*
A71LEN  DS      F                LENGTH OF COMMUNICATION AREA
A71LFU  DS      F                LENGTH OF FUNCTION DATA
A71NNAM DS      CL32             NET NAME FROM NPRLIB
A71EARL DS      CL12             EARLIEST-START
A71LATS DS      CL7              LATEST-START
A71LIFT DS      CL7              LIFE-TIME
A71INTYP DS     CL1              NET-TYPE
A71NDSO DS     CL8              NET-DELAY-SOLUTION
A71RCSY DS     CL8              RUN-CONTROL-SYSTEM
```

2.3.5 CC exit AVEX0001 and AVEX0002

Before a journal record is to be output to the journal file as a result of an AVAS function, the CC routine is called by AVAS.

The journal records are still transferred in the format “output date without century” in the CC exit AVEX0001.

The journal records are transferred in the format “output date with century” in the CC exit AVEX0002.

AVAS supplies the fields in the communication area with the following values when the CC routine is called:

prefixNNAM Output parameter.
 Net name.

prefixFUNK Output parameter.
 AVAS function.

prefixAKTN Output parameter.
 AVAS action.

prefixJSSL Output parameter.
 Journal record key.

prefixJFNR Output parameter.
 Journal record sequence number.

prefixADRJ Output parameter
AVEX0001:
The journal record is transferred in the format “output date without century”.
Transfer area: F3S AVASJRN &prefix,VERSION=020

AVEX0002:
The journal record is transferred in the format of the AVAS “output date with century”.
Transfer area: F3S AVASJRN &prefix,VERSION=080

Address of the journal record (see section [“Record structure of the fixed portion of journal records”](#) on page 204)

prefixDFIX Output parameter.
 EQUATE for length of the base data section

Returning the CC routine to AVAS

The CC routine must return a return code to AVAS in the pfixRCS field of the information area. This return code has the following meanings:

pfixOK	EQU X'00'	The journal record is to be output to the journal file.
pfixNO	EQU X'04'	The journal record is not to be output to the journal file. The return code is ignored if suppression of the record output is not permissible.

Note

If other return codes are returned by the CC routine, AVAS issues a message and the journal record is output to the journal file.

2.3.6 CC exit AVEX0101

After a net description has been written back to the NETLIB, AVAS calls the CC routine.

The version number parameter always has the value C'001', and the LMS type is always C'S'.

pfixLIB	DS	CL6	Output parameter Keyword library: NETxxx
	DS	CL2	Not used
pfixLNAM	DS	CL54	Output parameter Library name
	DS	CL2	Not used
pfixNNAM	DS	CL32	Output parameter Net name
	DS	CL28	Not used
pfixVERS	DS	CL3	Output parameter Version number: 001
pfixLTYP	DS	CL1	Output parameter LMS type: S
	DS	CL4	Not used
pfixDATE	DS	CL8	Current date: yymmdd
pfixTIME	DS	CL6	Current time: hhmmss
pfixCMDN	DS	CL22	Command name

Return of the CC routine to AVAS

If return of the CC routine to AVAS is specified in the pfixRCS field of the information area, this is not evaluated.

2.3.7 CC exit AVEX0102

Before a job description is written back to the JCLLIB or JMDLIB, AVAS calls the CC routine.

The version number parameter always has the value C'001', and the LMS type is always C'J'.

pfixLIB	DS	CL6	Output parameter Keyword library: JCLxxx/JMDxxx
	DS	CL2	Not used
pfixLNAM	DS	CL54	Output parameter Library name
	DS	CL2	Not used
pfixNNAM	DS	CL57	Output parameter Job name
	DS	CL3	Not used
pfixVERS	DS	CL3	Output parameter Version number: 001
pfixLTYP	DS	CL1	Output parameter LMS type: J
	DS	CL4	Not used
pfixDATE	DS	CL8	Current date: yymmdd
pfixTIME	DS	CL6	Current time: hhmmss
pfixCMDN	DS	CL22	Command name

Return of the CC routine to AVAS

If return of the CC routine to AVAS is specified in the pfixRCS field of the information area, this is not evaluated.

2.3.8 CC exit AVEX0401

If the AVAS run control system has output an S procedure or a BS2000 job to the temporary file, AVAS calls the CC routine. This call, which is given once only, passes the following information, which differs from the fields defined in the information area:

prefixUSR=X'40'	no AVAS user name
prefixCAL=X'40'	no calendar name
prefixBKR=X'40'	no AVAS user group of the net

When the CC routine is called, AVAS supplies the fields in the communication area with the following values:

prefixNNAM	Output parameter. Net name.
prefixJIND	Output parameter. Job index.
prefixJNAM	Output parameter. Job name.
prefixRSTJ	Output parameter. Job after RESTART: Y/N
prefixRSTA	Output parameter. RESTART statement: Y/N
prefixUSID	Input/output parameter. Name of the BS2000 user ID of the ENTER file.
prefixACCN	Input/output parameter. Account number of the BS2000 user ID of the ENTER file.
prefixPASW	Input/output parameter. Password for the BS2000 user ID of the ENTER file.
prefixEFIL	Output parameter. Name of the ENTER file.
prefixECAT	Output parameter. Reference to a slave processor (value only present if a catalog ID (CATID) has been determined).
prefixSIID	Input/output parameter. Symbolic name of the server partner to whom the task is to be sent (= SERVER-NAME).

prefixSIPW	Input/output parameter. Password authorizing addressing of this partner (= SERVER-PASSWORD).
prefixJTYP	Input/output parameter Type of job (e.g. MOD/STD).
prefixFUNC	Input/output parameter Function of the job (e.g. J/P).
prefixJPAR	Input/output parameter Parameter value for JOB-PARAMETER for the BS2000 commands ENTER-JOB and ENTER-PROCEDURE.

Note

For server jobs, the prefixUSID, prefixACCN and prefixPASSW fields refer to the AVAS agent AVSSINCM. They specify the user ID, account number and password under which AVSSINCM is to be started.

If prefixSIID and prefixSIPW are deleted, the task cannot be transferred to a server partner because no entry in the configuration file can be addressed.

The values for the user ID, account number and password of the BS2000 user ID from the ENTER call are passed at CC exit AVEX0401. This information may already have been corrected by the run control system if this was initiated using the CHANGE-NET-DESCRIPTION statement.

If any one of the values is changed by the CC routine, all three values are used as parameters for the ENTER call.

The modification is logged in the journal.

The NETUSER statement (see [section “Starting and terminating the run control system” on page 115](#)) is checked before the CC routine is called. The following points must be borne in mind:

- If the run control system is started with NETUSER=*ERROR, a user ID must be specified in the net or in the job for ENTER-PARAMS=NET, even if the final user ID is not set until the CC exit (if necessary a dummy user ID, e.g. *EXIT, can be specified). If ENTER-PARAMS=LOGON is specified, the LOGON statement must contain a user ID.
- If the user ID in the CC exit is deleted, the job is brought to execution under the user ID of the run control system.

Returning the CC routine to AVAS

The CC routine must return a return code to AVAS in field pfixRSC of the information area. The return code has the following meanings:

pfixOK EQU X'00'	The function is to be performed.
pfixNO EQU X'04'	The function is not to be performed. Message AVS8509 is logged in the journal.

If other return codes are returned by the CC routine, AVAS issues a message and the message is logged in the journal. The job is not called via ENTER. The CC routine is ignored by AVAS.

2.3.9 CC exit AVEX0402

If a net with the RUNNING status is given a different status by the run control system and is not brought to execution again by the automatic restart, AVAS calls this CC routine. This call, which is given once only, passes the following information, which differs from the fields defined in the information area:

pfixUSR=X'40'	no AVAS user name
pfixBKR=X'40'	no AVAS user group
pfixCA=X'40'	no calendar name

When the CC routine is called, AVAS supplies the fields in the communication area with the following values:

pfixNNAM	Output parameter. Net name.
pfixNSTA	Output parameter. Net status.

Returning the CC routine to AVAS

The CC routine must return a return code to AVAS in the pfixRCS field of the information area. This return code has the following meaning:

pfixOK EQU X'00'	The net status was noted.
------------------	---------------------------

If other return codes are returned by the CC routine, AVAS issues a message and the CC routine is ignored.

2.3.10 CC exit AVEX0403

If a BS2000 job, an S procedure or a server job with the RUNNING status has been given the status ENDED, ABENDED or ERROR in the run control file by the run control system, AVAS calls this CC routine.

This call, which is given once only, passes the following information, which differs from the fields defined in the information area:

prefixUSR=X'40'	no AVAS user name
prefixBKR=X'40'	no AVAS user group
prefixCAL=X'40'	no calendar name

When the CC routine is called, AVAS supplies the fields in the communication area with the following values:

prefixNNAM	Output parameter. Net name.
prefixJIND	Output parameter. Job index.
prefixJNAM	Output parameter. Job name.
prefixJSTA	Output parameter. Job status.
prefixRV1I	Output parameter. Restart variant 1: Restart index.
prefixRV1T	Output parameter. Restart variant 1: Restart type.
prefixRV1J	Output parameter. Restart variant 1: Restart name.
prefixRV2I	Output parameter. Restart variant 2: Restart index.
prefixRV2T	Output parameter. Restart variant 2: Restart type.
prefixRV2J	Output parameter. Restart variant 2: Restart name.
prefixRV3I	Output parameter. Restart variant 3: Restart index.

pfixRV3T	Output parameter. Restart variant 3: Restart type.
pfixRV3J	Output parameter. Restart variant 3: Restart name.
pfixMJRV	Output parameter. Restart variant which was set by the entry RV=n in the task job variable, or blanks if RV=n was not specified.
pfixMJTL	Output parameter Length of the contents of the task job variable
pfixMJTX	Output parameter Contents of the task job variable The contents are set in the area of maximum length to a length of pfixMJTL.

Returning the CC routine to AVAS

The CC routine must return a return code to AVAS in the pfixRCS field of the information area. This return code has the following meanings:

pfixOK EQU X'00' The job status was noted.

If other return codes are returned by the CC routine, AVAS issues a message and the CC routine is not called again.

Note

The reaction to job status ERROR must be programmed.

2.3.11 CC exit AVEX2001

This CC routine is called by AVAS when the user enters the START-EXIT statement.

Passing AVAS to the CC routine and returning the CC routine to AVAS

AVAS supplies the fields in the communication area with blanks when the CC routine is called.

prefixMSG	Input parameter. Message for the AVS030 mask.
prefixOPR	Input / output parameter Information from the OPR field which was entered in the AVAS mask when START-EXIT was called and is to be output in the OPR field of mask AVS030 when the CC exit terminates.
prefixOPR1	Input / output parameter Corresponds to the OPR field in line 22 of the AVAS masks.
prefixOPR2	Input / output parameter Corresponds to the field in line 23 of the AVAS masks.
prefixNNAM	Output parameter Name of the marked net
prefixJIND	Output parameter Structure index
prefixFUNC	Output parameter FUNCTION type
prefixFNAM	Output parameter FUNCTION name
prefixCNAM	Output parameter CONDITION-NET name
prefixSNAM	Output parameter SUBNET name
prefixJNAM	Output parameter Job name
prefixCMDA	Output parameter CMD address

Notes

- The parameters from pfixNNAM on are supplied with values only if the CC exit with operation code #55 from NET-CONTROL is called and a net/nets or structure element/elements are marked.
- If it is called from mask AVI022 (LIST-OF-NETSff) only the pfixNNAM and pfixCMDA fields are supplied with values. No meaningful values are defined for the other fields.

The CC routine does not need to return a return code to AVAS in the pfixRSC field of the communication area. The pfixRSC field is not evaluated.

A message can be sent to AVAS in the pfixMSG field. If a message is sent, it is output in the AVS030 mask with message code AVS5225.

2.3.12 CC exit AVEX6601

This CC routine is called by AVAS before an assigned user mask is output for editing.

AVAS supplies the fields in the communication area with the following values when the CC routine is called:

pfixNNAM	Output parameter. Net name.
pfixFNAM	Output parameter. Name of the net mask.
pfixFMSG	not used
pfixDFIX	Output parameter. EQUATE for length of the base data section.

Passing AVAS to the CC routine and returning the CC routine to AVAS

Following the base data section, the values (data) from the mask fields are passed in the following form in the communication area (maximum length 1930 bytes):

Record length field	2 bytes
Spare	2 bytes
REM param. name	8 bytes
Field length, mask	2 bytes (maximum length 192 bytes)
Data length	2 bytes
Data	n bytes (maximum length 192 bytes)

The end of data is identified by the value zero (X'0000') in the record length field. The same also applies to the return of data by the user routine.

The CC routine must return to AVAS a return code with the following meanings in the pfixRSC field of the information area:

pfixOK	EQU X'00'	The field data is taken over and the net mask is output.
pfixUPD	EQU X'02'	The field data is taken over but the net mask is no longer output. (Modification is performed using the data passed.)
pfixIGN	EQU X'06'	Modification for this net mask is aborted (same effect as with CMD:IGNORE in interactive mode).
pfixABE	EQU X'08'	Modification is aborted (same effect as with CMD:RETURN in interactive mode).

Notes

- All the AVAS variables and their assigned values (except for S#nnn) are taken over by the CC routine.
- If the CC routine returns other return codes, AVAS issues a message in the MSG field and storage is not performed.
- Only values for the parameter fields N#nnn may be passed.
- If a mask is not output as a result of return code X'02', the parameters passed by the CC routine are not matched with the fields defined in the mask. All fields passed by the CC routine with a valid parameter code (N#nnn) are used for the modification.

2.3.13 CC exit AVEX6602

This CC routine is called by AVAS before the values of the variables N#nnn are stored.

AVAS supplies the fields in the communication area with the following values when the CC routine is called:

pfixNNAM	Output parameter. Net name.
pfixFNAM	Output parameter. Name of the net mask.
pfixFMSG	Input parameter. Error message (only when return code = pfixNO).
pfixDFIX	Output parameter. EQUATE for length of the base data section.

Passing AVAS to the CC routine

Following the base data section, the values (data) from the mask fields are passed in the following form in the communication area:

Record length field	2 bytes
Spare	2 bytes
REM param. name	8 bytes
Field length, mask	2 bytes (maximum length 192 bytes)
Data length	2 bytes
Data	n bytes (maximum length 192 bytes)

The CC routine must return to AVAS a return code with the following meanings in the pfixRSC field of the information area:

pfixOK	EQU X'00'	The field data is taken over for modification of the job. The modification is performed.
pfixNO	EQU X'04'	The field data is not taken over due to an input error. The net mask is output again with the passed message. CMD:CONTINUE is cleared.
pfixIGN	EQU X'06'	Modification for this net mask is aborted (same effect as with CMD:IGNORE in interactive mode).
pfixABE	EQU X'08'	Modification is aborted (same effect as with CMD:RETURN in interactive mode).

Notes

- All the AVAS variables and their assigned values (except for S#nnn) are taken over by the CC routine.
- With pfixRSC=X'04', a message should be passed in field (pfix)FMSG.
- If the CC routine returns other return codes, AVAS issues a message in the MSG field and modification is not performed.
- Modifications made by the CC routine to the data passed are not taken over by AVAS.

2.3.14 CC exit AVEX6801

This CC routine is called by AVAS before an assigned user mask is output.

AVAS supplies the fields in the communication area with the following values when the CC routine is called:

pfixNNAM	Output parameter. Net name.
pfixJIND	Output parameter. Job index.
pfixJNAM	Output parameter. Job name.
pfixFNAM	Output parameter. Name of the job mask.
pfixFMSG	not used
pfixDFIX	Output parameter. Equate for length of the base data section.

Returning the CC routine to AVAS

Following the base data section, the values (data) from the mask fields are passed in the following form in the communication area (maximum length 1930 bytes):

Record length field	2 bytes
Spare	2 bytes
REM param. name	8 bytes
Field length, mask	2 bytes (maximum length 192 bytes)
Data length	2 bytes
Data	n bytes (maximum length 192 bytes)

The end of data is identified by the value zero (X'0000') in the record length field. The same also applies to the return of data by the user routine.

The CC routine must return a return code to AVAS in the pfixRCS field of the information area. This return code has the following meanings:

pfixOK	EQU X'00'	The field data is taken over and the job mask is output.
pfixUPD	EQU X'02'	The field data is taken over but the job mask is no longer output. (Modification is performed using the data passed.)
pfixIGN	EQU X'06'	Modification for this job mask is aborted (same effect as with CMD:IGNORE in interactive mode).
pfixABE	EQU X'08'	Modification is aborted (same effect as with CMD:RETURN in interactive mode).

Notes

- All the AVAS variables and their assigned values (except for S#nnn) are taken over by the CC routine.
- If the CC routine returns other return codes, AVAS issues a message in the MSG field and modification is not performed.
- Only values for the parameter fields P#nnn may be passed.
- If a mask is not output due to return code X'02', the parameters passed by the CC routine are not matched with the fields defined in the mask. All fields passed by the CC routine with a valid parameter key (P#nnn) are used for the modification. This applies when the CC routine is called in interactive mode or via the AVAS-BATCH function. (In AVAS-BATCH, the CC routine can only be used if it returns return code X'02'.)

2.3.15 CC exit AVEX6802

This CC routine is called by AVAS before the values of the variables P#nnn are modified.

AVAS supplies the fields in the communication area with the following values when the CC routine is called:

prefixNNAM	Output parameter. Net name.
prefixJIND	Output parameter. Job index.
prefixJNAM	Output parameter. Job name
prefixFNAM	Output parameter. Name of the job mask.
prefixFMSG	Input parameter. Error message (only when return code = prefixNO).
prefixDFIX	Output parameter. Equate for length of the base data section.

Passing AVAS to the CC routine

Following the base data section, the values (data) from the mask fields are passed in the following form in the communication area:

Record length field	2 bytes
Spare	2 bytes
REM param. name	8 bytes
Field length, mask	2 bytes (maximum length 192 bytes)
Data length	2 bytes
Data	n bytes (maximum length 192 bytes)

The CC routine must return to AVAS a return code with the following meanings in the pfixRSC field of the information area:

pfixOK	EQU X'00'	The field data is taken over for modification of the job. The modification is performed.
pfixNO	EQU X'04'	The field data is not taken due to an input error. The job mask is output again with the passed message. CMD:CONTINUE is cleared.
pfixIGN	EQU X'06'	Modification for this job mask is aborted (same effect as with CMD:IGNORE in interactive mode).
pfixABE	EQU X'08'	Modification is aborted (same effect as with CMD:RETURN in interactive mode).

Notes

- All the AVAS variables and their assigned values (except for S#nnn) are taken over by the CC routine.
- With pfixRSC=X'04', a message should be passed in field (pfix)FMSG.
- If the CC routine returns other return codes, AVAS issues a message in the MSG field and modification is not performed.
- Modifications made by the CC routine to the data passed are not taken over by AVAS.

2.3.16 CC exit AVEX7101

This CC routine is called by AVAS before the net and structure data is entered in the run control file.

AVAS supplies these fields in the communication area with the following values when the CC routine is called:

prefixNNAM	Output parameter. Net name.
prefixEARL	Output parameter. Earliest start time.
prefixLATS	Output parameter. Latest start time.
prefixLIFT	Output parameter. Life time of the 'end-of-net' event.
prefixNTYP	Output parameter. Specifies how processing is serialized.
prefixNDSO	Output parameter. Actions to be performed in the event of an untimely start.
prefixRCSY	Output parameter. Name of the run control system.

Returning the CC routine to AVAS

The CC routine must return a return code to AVAS in the prefixRCS field of the information area. This return code has the following meanings:

prefixOK EQU X'00'	The function is to be performed.
prefixNO EQU X'04'	The function is not to be performed.

2.3.17 CC exit AVEX7102

This CC routine is called for every JCL statement of a task.

When the CC routine is called, AVAS supplies the fields in the communication area with the following values:

prefixNNAM	Output parameter. Net name.
prefixEARL	Output parameter. Earliest start time.
prefixLATS	Output parameter. Latest start time.
prefixLIFT	Output parameter. Life time of the 'end-of-net' event.
prefixNTYP	Output parameter. Specifies how processing is serialized.
prefixNDSO	Output parameter. Action to be taken in the event of an untimely start.
prefixRCSY	Output parameter. Name of the run control system.
prefixJIND	Output parameter. Index from the NPRLIB.
prefixJNAM	Output parameter. Job name from the NPRLIB.
prefixJENT	Output parameter. Key for ENTER parameters.
prefixJENTL	Equate for ENTER parameters from the LOGON statement.
prefixJENTJ	Equate for ENTER parameters from the net/job description.
prefixJCAT	Output parameter. Reference to a slave processor (value only present if NET-CAT or JOB-CAT is described in the net.)
prefixJUSE	Output parameter. BS2000 user ID for the ENTER call (value only present if prefixJENT=prefixJENTJ and the user ID is described in the net or job).

pfixOPST	Output parameter. Operator start. Y=YES N=NO
pfixJREC	Output parameter. Record area of the JCL statement.
pfixJRLG	Input/output parameter. Length of the JCL statement (up to 256 characters) plus 4 characters for the JRLG and JRES fields.
pfixJRES	Output parameter. Reserved area.
pfixJRSE	Output parameter. JCL statement.

Once all the jobs in a net have been processed, AVAS calls the CC routine again. This call passes the following information, which differs from the fields defined in the communication area:

pfixJIND=C'000'	No index level.
pfixJRLG=X'0000'	No length of JCL statement
pfixJNAM	Is not passed

Returning the CC routine to AVAS

The CC routine must return a return code to AVAS in the pfixRCS field of the information area. This return code has the following meanings:

pfixOK	EQU X'00'	The JCL statement was not modified. No entry is made in the journal.
pfixUPD	EQU X'02'	The modified JCL statement is to be adopted by AVAS. This return is possible only if a JCL statement was passed. Two records are entered in the journal: 25-01 Record with the old contents. 25-02 Record with the new contents.
pfixABE	EQU X'08'	The JCL modification could not be performed by the user routine. The statement is terminated via AVAS with ERROR. The record at which processing was aborted is entered in the journal under the number 25-01. Additionally, in the transfer area for JCL statements, it is possible to pass a text entered in the journal under the number 25-05.

pfixUIN	EQU X'12'	<p>The modified JCL statement is to be adopted by AVAS. The modified JCL statement is a block of JCL statements with variable record length, i.e. the fields pfixJRLG, pfixJRES and pfixJRSE are repeated until pfixJRLG is equal to X'0000'. This repetition is permitted only up to the length specified in the pfixLEN field.</p> <p>The actual length of the return in the pfixLFU field must extend from the pfixNNAM field to the final JCL statement.</p> <p>This return is possible only if a JCL statement was passed.</p> <p>Two records are entered in the journal: 25-01 Record with the old contents. 25-02 Record with the new contents.</p>
pfixUDE	EQU X'14'	<p>The JCL statement is not to be taken over into the run control file.</p> <p>One record is entered in the journal: 25-04 Record with the old contents.</p>

Notes

- For S procedures, the AVAS procedure parameters are passed to the CC routine, in addition to the procedure statements. These parameters are separated from the procedure statements by the separator character `'/* AVAS-PROC-PAR'` or the string that was defined at generation.
If this separator is deleted, AVAS can no longer recognize the parameters, and issues the ENTER-PROC call with no procedure parameters.
- For server jobs, not only the script statements but also the parameters are passed to the CC routines. These parameters are separated from the server script by the separator character `'/* AVAS-SINIX-PAR'` or by the string that was defined at generation.
If the separator character is deleted, AVAS can no longer recognize the parameters and starts the server job without using them.
- If the CC routine returns other return codes, AVAS issues an error message in the MSG/RESULT field and the net is not released.
- When JCL statements are returned to AVAS, the maximum length of 256 characters must not be exceeded. If the CC routine passes a greater length, processing is aborted with a corresponding message in the journal.
- In the journal, the result CHANGE/NO-CHANGE (corresponding to the return code) is stored in addition to the record with the JCL statement.

3 Administration of AVAS

In keeping with the given organization, the AVAS administrator must be defined in the function table as an AVAS user with special authorizations.

The tasks of the AVAS administrator encompass the following areas:

- starting and terminating the system tasks
- operating the system tasks
- monitoring the system tasks via job variables
- managing the system libraries
- saving and reorganizing the AVAS files
- optimizing the system and
- administering the users

The AVAS administrator should be the only user who is allowed to enter the following statements:

- MODIFY-SYSTEM-PARAMS
- SHOW-SYSTEM-PARAMS
- SHOW-USER
- CANCEL-USER and
- SEND-MESSAGE

If other AVAS elements are also to be administered centrally (calendars, periods or resources), the associated statements are also reserved exclusively for the AVAS administrator.

The AVAS administrator must authorize use of the START-EXIT statement for himself and/or other users if user functions have been implemented via the CC exit AVEX2001.

3.1 Starting and terminating AVAS tasks

3.1.1 Starting the system and system security

AVAS interactive mode assumes that the data in the run control file and the journal file can be accessed, as can the libraries.

When the user signs on to the AVAS application system, he also connects to the central access tasks.

To operate the AVAS run control system, it must be possible to access the run control file and the journal file.

The access tasks must be loaded and activated before the system is started. If a task is not activated, functions requiring the files cannot be used.

The server configuration entered in the configuration file and supported by AVAS can be monitored with the server monitor process (see the [section "Server monitor process" on page 488](#)). Server jobs are only started on active servers (servers in the RUNNING state).

The server monitor process is only needed when using server jobs. It must be started before the job controller.

Security in the AVAS system

In order to prevent unauthorized tasks from loading or signing on as a central access task, run control system or reorganization task (procedure AVS.REORG in the SYSPRC.AVAS.085 library, see [page 169f](#)), the AVAS administrator can allocate passwords in the procedures to define task associations.

The passwords are read via SYSDTA and checked in the initialization phase of the relevant tasks. Any errored input will result in termination of the task.

The prerequisites for starting the central access task and what to do in order to terminate it are described below.

Outputting diagnostic documentation SYS.ADUMP..

If unexpected errors occur, diagnostic documentation is output to a file with the prefix SYS.ADUMP...; this applies both to the AVAS dialog task and to the AVAS run control system.

If an error occurs in AVAS which leads to a problem message, a check must be made as to whether in this case diagnostic documentation was output into a file with the prefix SYS.ADUMP...; these should then be passed along with the problem message.

3.1.2 Start parameters for the dialog process

The AVAS administrator can supply the signon information for the AVAS user dialog in a job variable. This is made known to the dialog system using the LINK name *AVSUSER.

Signon information

After the dialog procedure has been started, AVAS attempts to read the signon information from the job variable with the LINK name *AVSUSER.

Signon is carried out if the following data is found in the job variable:

Positions 1..8	AVAS-USER-ID
Positions 11..18	USER-PASSWORD
Positions 21..27	AVAS-SYSTEM-ID

If the signon is successful, mask AVS020 appears, otherwise the system is terminated with an error message.

A user who has signed on in this way cannot switch to another user ID via the call of the SIGNON function. Since they are not presented with the SIGNON mask (AVS010), such users can only change their passwords if they are authorized to use MODIFY-SYSTEM-PARAMS.

The use of a job variable for signing on makes it possible to write LOGON procedures which prevent the user from doing anything in his/her BS2000 session apart from signing on under a prescribed AVAS ID and working in AVAS. It is thus possible to deny the user access to other AVAS IDs or other user programs.

It should be noted that AVAS does not convert entries in lowercase to uppercase. This may cause a signon attempt to be rejected as errored.

If the signon parameters are not supplied via a job variable, the interactive user is offered mask AVS010 so that he/she can enter them.

EDT procedure for JOBLLOG logs

If an evaluation procedure is required for monitoring the JOBLLOG information (see JOBLLOG operation and SHOW-JOB-LOG statement in the manual "AVAS Statements" [2]), the name can be entered in the positions 129 .. 256 of the job variables (name of the EDT procedure for JOBLLOG).

The name must be specified in the form FILE= or LIBRARY= and must correspond to the description of the EDT statement @INPUT (format 2).

The number of the EDT work area is defined by the EDT statement @PROC <procno> in the EDT procedure. This work area number must be made known to the interactive AVAS user for execution with DO <procno>.

3.1.3 Starting and terminating the access tasks

Starting the access tasks

An access task is created by calling an /ENTER-JOB command.

The ENTER file for the access tasks must contain the following commands:

- A file assignment (BS2000 command ADD-FILE-LINK) for the AVAS system parameter file with the link name AVASSYS.
It must be possible for the file to be accessed by two or more tasks. The file is opened with SHARED-UPDATE=YES.
- /ADD-PASSWORD commands for all files which have been given a READ-PASSWORD or WRITE-PASSWORD.

The names of the files accessed are defined in the system parameters. The file assignments for these files are issued by the access tasks.

The PLAM-ZD and the UPAM-ZD are started with the following procedures (elements (J) in the SYSENT.AVAS.085 library):

1. AVS.ZDPLAM
2. AVS.ZDUPAM

The central access tasks control access to the AVAS libraries (PLAM-ZD) as well as to the run control and journal files (UPAM-ZD).

When the access tasks are started, the following statements can be entered via SYSDTA:

Statements	UPAM-ZD	PLAM-ZD
ABLCOPY =	yes	no
ABLDUP =	yes	no
CENTRAL-PASSWORD =	yes	yes
ENCRYPTION-CODE =	no	yes
JRNDAT-ISAM-NAME	yes	no
MAX-SERIAL-LOCK =	yes	no
MAX-SERIAL-WAIT =	yes	no
RCS-PASSWORD =	yes	yes
REO-PASSWORD =	yes	yes

ABLCOPY=

- | | |
|-----|--|
| YES | Before starting the UPAM-ZD, a duplicate ABLDUP of the run control file ABLDAT is to be created. |
| NO | No duplicate of the run control file is to be created. |

If ABLCOPY is not specified, ABLCOPY=NO is assumed.

The statement ABLCOPY=YES is executed only if ABLDUP=YES is also specified.

ABLDUP=

- | | |
|-----|---|
| YES | A duplicate of the run control file is to be kept. All outputs are written both to the ABLDAT run control file and to the ABLDUP duplicate. |
| NO | No duplicate of the run control file is kept. |

If ABLDUP is not specified, ABLDUP=NO is assumed.

Notes

- If ABLDUP=YES is specified, a file assignment must be made in the system parameters via the statement ABLDUP=filename.
- When the UPAM-ZD is started, a check is made as to whether the two files have the same processing status. If not, the UPAM-ZD is not started (the duplicate must be created using the BS2000 command /COPY-FILE or with the aid of the statement ABLCOPY=YES).

CENTRAL-PASSWORD=

- | | |
|----------|---|
| *STD | The internally defined password is expected when the CENTRAL task signs on. |
| C'....' | |
| X'.....' | The CENTRAL task must sign on for the access tasks with the specified password. |

If CENTRAL-PASSWORD is not entered, *STD is assumed by default.

Notes

- Passwords must be entered with the mandatory length of 4 bytes.
- Signing on is successful only if the ZDDPW passwords defined for the run control systems and for reorganization (see [page 117](#) and [page 187](#)) match those entered here.

ENCRYPTION-CODE=

The statement is only processed if the generation parameters are used to set PASSWORD-ENCRYPTION=YES.

If the value is changed, it will no longer be possible to process the nets in the production plan or in the run control file.

'c-string'

Character string (maximum of 8 characters) for encrypting the passwords.

jvname

Name of a job variable which contains a character string of no more than 8 characters for encrypting the passwords.

If the job variable is protected by a read password, the BS2000 command /ADD-PASSWORD must already have been issued in the procedure.

Note

The AVAS administrator must ensure that the central access task ZDL and the run control routines are always started with the same ENCRYPTION-CODE, otherwise the nets cannot be processed.

JRNDAT-ISAM-NAME=

The statement controls whether the journal entries are also logged in an ISAM file. This ISAM file is opened in SHARUPD mode and can be used to transfer the journal entries to a database and to work on them there with evaluation tools.

*NONE

The journal entries are not also logged in an ISAM file - this is the default value.

*STD

The journal entries are also logged in an ISAM file. The name of this file consists of the name of the journal file and a suffix in the format .<yymmdd>.<hhmms> .

filename

The journal entries are also logged in an ISAM file. The name of this file is the same as the parameter value *filename*.

Note

The ISAM file is created with KEY-LEN=8. The key is the system time in STCK format. The records have a similar layout to the journal backup (see [section "Structure of the ISAM journal log file" on page 222](#)). The file is created as a new file when the UPAM-ZD is started up and is closed when the latter is terminated. When reorganization takes place it can be closed and created again (see ["UP-DATE-JRIDAT" on page 186](#)).

MAX-SERIAL-LOCK=

The statement describes the maximum processing time for the function execution of a serialized AVAS statement.
The value in seconds is determined from the maximum processing time for the largest net.
The processing time is accurate to plus or minus 60 seconds.

***STD** The internally defined maximum processing time is 180 seconds.
nnnnn Time in seconds. A maximum processing time of between 60 and 21,599 seconds can be defined.

See also [“Notes on the serialization run” on page 112.](#)

MAX-SERIAL-WAIT=

The statement describes the wait time of a function with serialization processing.
The processing time is accurate to plus or minus 60 seconds.

***STD** The internally defined wait time of 300 seconds is used.
nnnnn Time in seconds. A wait of time between 60 and 21,599 seconds can be defined.

See also [“Notes on the serialization run” on page 112.](#)

RCS-PASSWORD=

***STD** When the run control systems are signed on, the internally defined password is expected.

C'.....'
X'.....' The run control systems must sign on to the access tasks using the specified password.

If RCS-PASSWORD is not entered, *STD is assumed by default.

REO-PASSWORD=

- | | |
|----------|--|
| *STD | When reorganization is signed on, the internally defined password is expected. |
| C'....' | |
| X'.....' | Reorganization must sign on to the access tasks using the specified password. |

When the UPAM-ZD is started up, any emergency journal files from earlier sessions are automatically copied into the journal file. The administrator can request such copying during the current session by entering the string 'IN-JRLDAT' in the monitoring job variable for the UPAM-ZD.

Notes on the serialization run

The following must be taken into account for the serialization run which is started by the MAX-SERIAL-LOCK and MAX-SERIAL-WAIT parameters:

- The statements CREATE-PLAN-NET, CREATE-PROD-NET, SUBMIT-NET and REPEAT-NET are serialized. The SUBMIT-NET and REPEAT-NET statements work with the same serialization argument. Statements which are started at the same time with the same serialization argument are processed consecutively.
- Serialization is set for every subnet for hypernets when processing is started and is released after processing a subnet. Serialization is also released for the corresponding hypernet when the serialization is released for the subnet.
- MAX-SERIAL-LOCK=t-lock defines the maximum processing time allowed for a net.

If, when processing a net, a user occupies the serialization for longer than MAX-SERIAL-LOCK allows, the processing is terminated when the time has elapsed, as defined. The user is then forcibly signed off and must sign on again (no signon lock is set).

If a serialization argument is occupied by a statement, it will not be released when a CC exit is invoked (in CREATE-PROD-NET, SUBMIT-NET and REPEAT-NET) or when a user mask is output (CREATE-PROD-NET). This must be taken into account when defining MAX-SERIAL-LOCK.

The maximum processing times can be determined from the journals of the nets.

- MAX-SERIAL-WAIT=t-wait defines the maximum time a user must wait when processing a net if he wants to occupy a serialization argument.

If, when processing a net, a user has to wait longer to occupy a serialization argument than MAX-SERIAL-WAIT allows, the net processing is aborted.

In the overview processing with the Y/N mark, the ERROR result is displayed for the net. In the individual processing (fully qualified operand NET-NAME or S mark in the net overview), a message is output.

If a serialized statement is issued by several users simultaneously, the value for t-wait must also be a multiple of t-lock.

The following time specification is recommended as sensible:

t-wait > t-lock > maximum processing time

Statement/mask	Operation	Comment
CREATE-ORDER AVP012	EXECUTE	Overview processing with Y/N mark The serialization is set for each selected net and then reset once the net processing is completed. If the serialization cannot be occupied (MAX-SERIAL-WAIT), the ERROR result is set for the net and processing continues with the next selected net.
CREATE-PLAN-NET AVP011		
CREATE-PROD-NET AVM012		
SUBMIT-NET AVF001		
CREATE-PLAN-NET AVP001	SAVE	Overview processing with S mark or fully qualified operand NET-NAME. The serialization is not occupied until processing is initiated with SAVE or EXECUTE. If the serialization cannot be occupied (MAX-SERIAL-WAIT), processing for the net is aborted and message AVS5804 is output.
CREATE-PROD-NET AVM001	EXECUTE	
SUBMIT-NET AVF002 AVF004	SAVE	
REPEAT-NET AVF012 AVF014	SAVE	

Normal termination of access tasks

Access tasks are caused to terminate by setting the string SHUTDOWN in the corresponding job variable (see [page 125](#)). The name of the job variable is entered in the system parameters.

Access tasks for the journal and run control files and for libraries are terminated only if there are no longer any user or run control tasks active.

If active partner tasks still exist, termination of the access tasks is delayed until all partner tasks have terminated.

Abnormal termination of the access tasks

If a non-recoverable processing error arises in the access task (program error, macro error, etc.), the task will be terminated after output of a memory dump and an error message. If possible, any communicating tasks affected will be informed of the abnormal termination.

If it is not possible to effect an orderly closure of files during an abnormal termination (e.g. power failure), the BS2000 commands /REPAIR-DISK-FILES and /REMOVE-FILE-ALLOCATION-LOCKS must be issued for the files concerned. Otherwise, the job control file and journal file may be corrupted when the tasks are next started.

In the event of a non-recoverable ILAM error relating to a library, this library will be locked against further access after a memory dump and an error message have been output.

If a DMS error occurs while writing to the journal file, the output of journal records will be diverted to an emergency journal file. The AVAS administrator must have specified a prefix for the name of this file in the system parameters. AVAS extends this name by adding the date and time. The emergency journal will only be created if there is a need to output records to it. Journal records for a net which are output to the emergency journal file cannot be displayed in the dialog.

3.1.4 Starting and terminating the run control system

The BS2000 catalog ID and user ID under which a run control system is started can be selected as desired (it is not necessary to start under the TSOS user ID).

The run control system catalog ID and user ID is important for running hypernets. AVAS requires the run control system catalog ID and user ID of the hypernet for the subnets in order to pass information from the subnets to the hypernet. This means that the catalog ID and user ID of a run control system may only be changed if using more than one run control system in an AVAS system when all run control systems are reloaded.

The AVAS system ID and the name of the run control system must be placed in an AVAS system parameter file in a prior generation run (see [section “Definitions for the central access tasks” on page 28](#)).

The run control system (RUN-CONTROL-SYSTEM) is started with the following procedure:

```
AVS.AVAK
```

This procedure is an element (J) in the library SYSPRC.AVAS.085.

Starting the run control system

In order to function, the run control system requires the following entries via SYSDTA. These entries must be made upon request:

SYSID=id	AVAS system ID of the system. The AVAS system ID must be specified.
AVAK=name	Name of a run control system (must be specified). The name must have been defined in the system parameters as the run control system (RUN-CONTROL-SYSTEM). The run control file and the journal file must be formatted for this run control system.
END	End of the start parameters.

The statements SYSID, AVAK and END are mandatory.

```
MONJV-PASSWORD=
```

*NONE	The task job variables are set up without a password; *NONE is predefined in the start procedure as the default value.
*STD	The password for the task job variables is assigned by the run control system.

C'....'
X'.....'

The password must be specified with a length of 4 (alphanumeric or hexadecimal) characters. All task job variables are initialized with this password.

Note

When the run control system is restarted, the same password must be used as was assigned for the first start. The run control system can be restarted without a password only if all nets have terminated.

MONJV-RDPASS=

YES The task job variables created by the run control system when the job is started are supplied with the value specified for MONJV-PASSWORD as the read password. For SHOW-NET-STATUS the contents of the task job variables are not displayed.

NO The task job variables are set up without read passwords.

Note

This statement is ignored if MONJV-PASSWORD=*NONE is specified.

ZDD-PASSWORD=

*STD Signon for the UPAM-ZD is performed using the internally defined password.

C'....'
X'.....'

Signon for the UPAM-ZD uses the specified password.

If ZDD-PASSWORD is not specified, *STD is assumed.

Notes

- Passwords must be entered with the mandatory length of 4 bytes.
- The run control system cannot sign on successfully to the UPAM-ZD unless the password matches the one which was defined via RCS-PASSWORD when starting the UPAM-ZD.

ZDL-PASSWORD=

*STD Signon for the PLAM-ZD is performed using the internally defined password.

C'....'

X'.....' Signon for the PLAM-ZD uses the specified password.

If ZDL-PASSWORD is not specified, *STD is assumed.

Notes

- Passwords must be entered with the mandatory length of 4 bytes.
- The run control system cannot sign on successfully to the PLAM-ZD unless the password matches the one which was defined via RCS-PASSWORD when starting the PLAM-ZD.

ENCRYPTION-CODE=

The statement is only processed if the generation parameters are used to set PASSWORD-ENCRYPTION=YES.

If the value is changed, it will no longer be possible to process the nets in the production plan or in the run control file.

'c-string' Character string (maximum of 8 characters) for encrypting the passwords.

jvname Name of a job variable which contains a character string of no more than 8 characters for encrypting the passwords.

If the job variable is protected by a read password, the BS2000 command /ADD-PASSWORD must already have been issued in the procedure.

Note

The AVAS administrator must ensure that the central access task ZDL and the run control routines are always started with the same ENCRYPTION-CODE, otherwise the nets cannot be processed.

NET-STATUS=

- HOLD** If the run control system is restarted after a hard abort, it performs the HOLD-NET function for a net with the status RUNNING. This action prevents the net structure from being further processed without the necessary resources being providable following a system crash.
- RESUMED** When the run control system starts again after a hard abort, a net with the RUNNING status continues running normally if there are still structure elements available for processing. For this net the RESUME-NET statement is invoked internally. It is not invoked if structure descriptions with the HOLD status already existed before the internal HOLD-NET function.

If the NET-STATUS statement is not specified, NET-STATUS=HOLD is assumed.

NET-USER=

- The AVAS administrator can use the NET-USER parameter to define whether jobs under the BS2000 user ID of the run control system may be brought to execution if the USERID specification for these jobs is omitted in the net description and in the SET-LOGON-PARAMETER (LOGON) command.
- *ERROR** If the run control system detects an ENTER call for starting a job without a user ID, no job is started and message AVS8513 is logged in the journal. The job (and therefore also the net) are given the status ERROR.
- *NONE** An ENTER call without a user ID is executed. The job is brought to execution under the BS2000 user ID of the run control system.

If no NET-USER statement is specified, NET-USER=*ERROR is assumed.

SPVS=catid

Catalog name for the shared pubset on which AVAS is to create the ENTER files in multiprocessor operation.

The statement must be specified if jobs are to be executed via a shared pubset in a computer network and the SPD statement is not specified.

SPD=(catid,vsn,dev) Definition of the external shareable private disk volume on which AVAS is to create the ENTER files in multiprocessor operation.

catid: name of the catalog.

vsn: name of the volume.

dev: Device type in accordance with the device table (e.g. D3435)

The values for SPD must be specified if jobs are to be executed via a shareable private disk in a computer network and SPVS is not defined.

Note

The SPVS and SPD statements may only be specified if jobs are to be distributed in a computer network. Before the run control system starts, the multiprocessor system must be operational.

RESTORE-MONJV-VALUE=

YES If the job description record in the run control file is given the status ENDED or ERROR, the contents of the task job variable (positions 129 through 256) are to be saved by AVAS during termination processing for the job. If the job is rerun after a restart, the saved contents should be available again in the task job variable.

NO The contents of the task job variable are not to be saved (default value of the start procedure).

Note

If the contents of the task job variable are saved by the run control system, then any later run of the job must ensure that the correct data is contained in the task job variable (positions 129 through 256). For reasons of data security, the AVAS control data RV=n (RESTART-VARIANT=n) is changed to the value LR=n (LAST-RESTART=n), to avoid an unwanted restart variant in the case of another restart.

Start parameters for controlling log processing

The following three parameters should only be specified if the runtime logs are to be stored via AVAS.

The only logs that can be collected are those which are entered in the list of contents of the AVAS pool. This is normally done in the AVAS job by the SIGNAL program. If no such measures are taken in the AVAS job, the name of the log can be entered in the AVAS pool by the run control system.

The start parameter GENERATE-JOB-LOG must be specified for this purpose.

GENERATE-JOB-LOG=

- | | |
|------|---|
| *STD | An entry in the list of contents of the AVAS pool is initiated by the SIGNAL program. |
| *ALL | An entry in the list of contents of the AVAS pool is created by the run control system for each job started. *NONE is entered for the name (JOBLOG-NAME=*NONE). |

Note

If this parameter is specified when the run control system is started, the ZD for PLAM access must be active.

TRANSFER-PROCEDURE-NAME=

- | | |
|----------|--|
| *NONE | No follow-up job is to be started for TRANSFER by the run control system. |
| filename | <p>The name of a procedure to be started by the run control system with the /CALL-PROCEDURE command must be specified if the TRANSFER program is to run as a follow-up job (see section “Starting the SIGNAL program”, TRANSFER-OPTION parameter on page 146).</p> <p>The complete path name must be specified if the procedure is not cataloged where the AVAS job was started.</p> <p>Note that this procedure must be accessible from all IDs and mainframes addressed by the run control system.</p> <p>Make sure that there is provision in the procedure for the job run to be identified as errored by the run control system in the event of an error (for example, a transfer error is indicated via switch 30/31). For this purpose, the task job variable must be supplied starting at column 129 or the job run must be hard-terminated.</p> |

TRANSFER-PROCEDURE-PASSWORD=

*NONE	The procedure for the TRANSFER job is not password-protected.
password	The procedure for the TRANSFER job is password-protected. The password must be specified with a length of 4 characters, in alphanumeric (C'....') or hexadecimal (X'.....') notation. It is only evaluated if the parameter TRANSFER-PROCEDURE-NAME=filename was also specified.

Note

If, in connection with GENERATE-JOB-LOG=*ALL, an error occurs during communication between the run control system and the central access tasks for PLAM access, a message is output and the run control system is terminated.
The error must be eliminated and the run control system restarted.

Start parameters for working with server systems**SERVER-PROCEDURE-NAME=**

*NONE	The run control system is not to support any server jobs. If nets with server jobs (which are to be activated) are started, the structure elements with a server job are set to ERROR during net processing. IF SERVER-PROCEDURE-NAME=*NONE, the subsequent parameters can be omitted.
filename	Name of the ENTER procedure to be used by the run control system to start the AVAS agent AVSSINCM (see page 478). Note that this procedure must be accessible from all IDs and hosts addressed by the run control system.

SERVER-JV-LINK=

name	Name component (up to 8 characters) in the name of the server job variable from which the server's current status is ascertained before a server job is started. (Name of the server job variable: AVS.SERVER.STATE.<server-jv-link-name>.<server-name>)
*STD	If the parameter is not specified, the run control system enters its own name. When the server monitor is started its start parameter SERVER-JV-LINK must then be supplied with the name of the run control system.

SERVER-PROCEDURE-PASSWORD=

*NONE	The ENTER procedure is not password-protected.
password	The ENTER procedure is password-protected. The password must be specified with a length of 4 characters, in alphanumeric (C'....') or hexadecimal (X'.....') notation. The parameter is only evaluated if SERVER-PROCEDURE-NAME is not equal to *NONE.

CONFIG-FILE-NAME=

*NONE	The name of a configuration file must be entered in the start procedure for AVSSINCM. Server jobs can only be started if *STD was specified as the server partner when the user group was generated.
filename	Name of the configuration file. Note that this file must be accessible from all IDs on which the AVSSINCM program is started by the run control system.

CONFIG-FILE-PASSWORD=

*NONE	The configuration file is not password-protected.
password	The configuration file is password-protected. The password must be specified with a length of 4 characters, in alphanumeric (C'....') or hexadecimal (X'.....') notation.

Terminating the run control system

The run control system is terminated by means of the STOP and CANCEL operations. These can be executed both via the /MODIFY-JV command and via /INFORM-PROGRAM (see the manual "AVAS Statements" [2]) .

The SHUTDOWN operation, which is still permitted, can only be executed via the /MODIFY-JV command and corresponds to the STOP, LEVEL=NET operation.

Note

You are advised not to terminate the run control system with the /CANCEL-JOB command as this could lead to inconsistent status information in the AVAS files ABLDAT and JRNDAT.

3.1.5 Starting and terminating the server monitor process

How you start and terminate the server monitor process is described in [section “Server monitor process” on page 488ff.](#)

3.1.6 Starting and terminating the server interface process

How you start and terminate the server interface process is described in [section “Server interface process” on page 495ff.](#)

3.1.7 Displaying the statuses and versions of the configured AVAS servers

The AVS.SVSTATUS procedure from the SYSPRC.AVAS.085 library displays the statuses and versions of the AVAS servers configured in the CONFIG file. The information display depends on the start parameter INFORMATION:

- INFORMATION=*VERSION displays versions.
- INFORMATION=*STATUS displays status information.

3.1.8 Displaying the active REP corrections

The SYSPRC.AVAS.085.SCANREPS procedure from the SIPLIB.AVAS.085 library displays the active REP corrections of the AVAS libraries SYSLNK.AVAS.085, SYSPRG.AVAS.085.SYSTEM and SYSPRG.AVAS.085.DIALOG.

3.1.9 Processing the optional REP corrections

The SYSPRC.AVAS.085.SCANOPTREPS procedure from the SYSPRC.AVAS.085 enables the optional REP corrections of the AVAS libraries SYSLNK.AVAS.085, SYSPRG.AVAS.085.SYSTEM and SYSPRG.AVAS.085.DIALOG to be displayed, activated and deactivated.

3.2 Working with and monitoring AVAS tasks

3.2.1 Working with the access tasks

The access tasks are administered via input in the job variables which monitor them. The names of the job variables are defined in the system parameters.

Three different types of job variable input can be distinguished:

- input for administration of the users who are signed on
- input for termination of the access tasks,
- input for the merging in of the emergency journal file and
- input for copying and reassigning the system files SYSLST and SYSOUT.

All inputs are made via the monitoring job variable. The operation to be executed must be the first entry and consist of job variable values:

```
/MODIFY-JV JV=jvname,SET-VALUE=C'operation'
```

Operations for administering signed-on users

CANCEL-SOFT	Signs off all interactive, batch and PI (program interface) users currently signed on as soon as they have finished the present statement. The relevant ZD (central access task) is locked against new signons.
CANCEL-HARD	Signs off all interactive, batch and PI users currently signed on as soon as they have finished the present dialog step. The ZD is locked against new signons.
HOLD	HOLD locks the relevant ZD against new signons.
RESUME	RESUME cancels the signon lock.

Notes

- The signon lock for the JV operation applies until the end of the AVAS session at the latest, if it is not canceled beforehand by the JV entry RESUME. The next time AVAS is started, all locks will have been canceled.
- There are no JV operations for excluding only specific users from signing on.
- In the CANCEL and HOLD operations, no signon lock is set in the system parameters for the affected users.

- The RESUME operation must be applied to the same ZD to which the HOLD command was previously applied, otherwise it has no effect.

Operations for terminating the access tasks

SHUTDOWN The ZD tasks can only be terminated via BS2000 command /MODIFY-JV for the job variable monitoring the task.

```
/MODIFY-JV JV=jvname,SET-VALUE=C'SHUTDOWN'
```

The tasks are terminated as soon as

- no more users are active (ZDPLAM),
- no more nets are running (UPAM-ZD) and
- no more run control systems are active (UPAM-ZD).

New users can no longer sign on and no new nets are started.

KILL

The task is terminated irrespective of any active users or any objects (elements) currently being processed.

New users can no longer sign on or process the objects managed by the task.

```
/MODIFY-JV JV=jvname,SET-VALUE=C'KILL'
```

Merging in of the emergency journals by the UPAM central access task (ZD)

If the administrator has set the UPAM-ZD's monitoring job variable to IN-JRLDAT, the emergency journals will be copied into the journal file immediately (during the current AVAS session).

Input:

```
/MODIFY-JV JV=jvname,SET-VALUE=C'IN-JRLDAT'
```

- The previous emergency journal file is closed.
- A new emergency journal file is opened.
- The entries from the old emergency journal file are copied into the journal file. If errors occur during this activity, the entries will be copied into the new emergency journal file.

This function is performed automatically each time the UPAM-ZD is started up.

IN-JRLDAT is displayed as the status in the monitoring job variable (see also [page 135](#)).

When emergency journal records have been transferred, their number will be logged by the UPAM-ZD.

Operations for copying/reassigning the system files SYSLST and SYSOUT

The system files SYSLST and SYSOUT of the access tasks can be evaluated during ongoing operation after they have been copied or reassigned.

Input:

```
/MODIFY-JV JV=jvname,SET-VALUE=C'operation'
```

The following entries are possible for operation:

COPYLST	Creates a copy of the current SYSLST file.
COPYOUT	Creates a copy of the current SYSOUT file.
NEWLST	Assigns a new SYSLST file. The old SYSLST file can then be accessed.
NEWOUT	Assigns a new SYSOUT file. The old SYSOUT file can then be accessed.

AVAS creates the copied or new file in the tasks' execution ID under the following name:

AVAS.LST.<tsn>.<yymmdd>.<hhmmss> for the SYSLST file and

AVAS.OUT.<tsn>.<yymmdd>.<hhmmss> for the SYSOUT file.

Notes

- SYSLST and SYOUT can be copied only if a file is currently assigned. Copying is not possible in the event of primary allocation (PRIMARY) or if a PLAM library member is assigned.
- The current system file is closed to execute the copy and reassignment operations. When copying takes place, it is opened again when the copy operation has been completed. Otherwise no operations on the files would be possible. To perform the operations it may be necessary to change the call procedure level. Complex call hierarchies within customer-specific AVAS start procedures can therefore be disturbed. If the sample procedures for starting the AVAS processes which are supplied in the SYSPRC.AVAS.085 library are used, this is no problem.

3.2.2 Operating the run control system

Two user interfaces have been provided for administration of the run control system:

- run control system calls via /INFORM-PROGRAM commands
- run control system calls via entries in the job variable of the run control system; the name of the job variable is defined by means of the system parameters (see [section “Working with the access tasks” on page 124](#)).

This section deals with entering statements using /INFORM-PROGRAM command calls. With these, a distinction is made between:

- inputs for the run control system itself,
- inputs for controlling the nets which run under the control of the run control system,
- input for updating the AVAS processor table,
- input for copying and reassigning the system files SYSLST and SYSOUT.

All the commands calls are described in full in the manual “AVAS Statements” [2].

In their impact, the command calls correspond to those statements which may be issued via the interactive components. However, certain calls have fewer parameters, and hence fewer opportunities of influencing or viewing system behavior. This applies in particular to all inputs valid for more than one run control system, since all /INFORM-PROGRAM commands relate only to the control of nets which are running under the run control system involved.

The statement is entered via the BS2000 command /INFORM-PROGRAM¹. Its syntax is as follows:

```
/INFORM-PROGRAM MSG='statement', JOB-ID=*TSN(<tsn>)
```

- `statement` is one of the AVAS statements listed below
- `tsn` is the task sequence number of the run control system under the operating system
- `tsn` and `statement` are passed as character strings to an STXIT routine of the AVAS system.

The statements may be abbreviated in accordance with the same rules that apply to the AVAS system. Since not more than 64 characters may be entered, the name of the statement and the names of the operands must, in part, be abbreviated.

¹ The call via /INTR <tsn>,<message> is still supported.

3.2.2.1 Inputs for the run control system

The following statements can be called using the /INFORM-PROGRAM command in order to administer the run control system:

CANCEL	Abort run control and monitoring system
HOLD	Suspend run control and monitoring system
NETC	Perform net start check
RUNC	Perform abortion control for running nets
RESUME	Reactivate run control and monitoring system
STOP	Terminate run control and monitoring system

3.2.2.2 Controlling released nets via /INFORM-PROGRAM

This section deals with the functions which support the execution and monitoring of nets and jobs.

The /INFORM-PROGRAM commands for controlling net execution should only be used when there is no interactive terminal available for administering the run, or when the interactive system cannot be loaded.

The following statements can be called using the /INFORM-PROGRAM command in order to control released nets:

CANCEL-NET	Terminate running net abnormally.
HOLD-NET	Suspend running net.
RESTART-NET	Restart net which terminated abnormally.
RESUME-NET	Restart suspended net.
SHOW-NET-STATUS	Display status of running nets.
START-NET	Start nets with status OPWAIT.

All net-related actions triggered by the /INFORM-PROGRAM commands are logged in the journal. Should an error occur, a corresponding message appears at the console, where messages with no net name are always concluded by a message with a net name.

3.2.2.3 Updating the AVAS-internal processor table

When the run control system is started it uses the MSCF job variable to ascertain from the MRSCAT the host/pubset configuration to be supported. The run control system keeps this configuration data in a separate host table.

The same procedure is followed for AVAS servers by means of the SERVER job variable. Modifications to the server configuration during ongoing operation can be reported to the run control system using the /INFORM-PROGRAM call. In the process the server table is updated.

Active jobs on hosts and servers which are no longer in the network are aborted and assigned the ERROR status.

The ONEVT events for monitoring the MSCF-SERVER job variable are updated.

The following statements can be called using the /INFORM-PROGRAM command in order to update the AVAS host and server tables:

UHOST	Updates the table of MSCF hosts
USERVER	Updates the server table

For information on modifying a server configuration, please also refer to the [section “Server monitor process” on page 488](#).

3.2.2.4 Copying and reassigning the system files SYSLST and SYSOUT

The system files SYSLST and SYSOUT of the access tasks can be evaluated during ongoing operation after they have been copied or reassigned.

AVAS creates the files in the tasks' execution ID under the following names:

AVAS.LST.<tsn>.<yymmdd>.<hhmmss> for the SYSLST file and

AVAS.OUT.<tsn>.<yymmdd>.<hhmmss> for the SYSOUT file.

The following statements enable the system files SYSLST and SYSOUT to be copied or reassigned using the /INFORM-PROGRAM command:

COPYLST	Copies the SYSLST file
COPYOUT	Copies the SYSOUT file
NEWLST	Reassigns the SYSLST file
NEWOUT	Reassigns the SYSOUT file

Note

SYSLST and SYSOUT can be copied only when a file is currently assigned. Copying is not possible in the event of primary allocation (PRIMARY) or if a PLAM library member is assigned.

3.2.3 Working with the CENTRAL task

The CENTRAL task is managed via inputs in the job variable which is used to control it. The name of the job variable is defined in the system parameters.

With the inputs in job variables a distinction must be made between

- inputs for terminating the CENTRAL task and
- inputs for copying and reassigning the system files SYSLST and SYSOUT

All inputs are made via the monitoring job variable. The operation to be executed must be the first entry and consist of job variable values:

```
/MODIFY-JV JV=jvname,SET-VALUE=C'operation'
```

Operations for terminating the CENTRAL task

SHUTDOWN	The CENTRAL task is terminated. <ul style="list-style-type: none">– The secondary tasks are terminated if no further job is being processed.– The primary task is terminated if all secondary tasks have been terminated.
CANCEL	The CENTRAL task is terminated immediately. <ul style="list-style-type: none">– The primary task is terminated immediately without taking the secondary tasks into account.

Operations for copying and reassigning the system files SYSLST and SYSOUT

The system files SYSLST and SYSOUT of the CENTRAL tasks can be evaluated during ongoing operation only after they have been copied or reassigned.

AVAS creates the files in the tasks' execution ID under the following names:

AVAS.LST.<tsn>.<yymmdd>.<hhmmss> for the SYSLST file and

AVAS.OUT.<tsn>.<yymmdd>.<hhmmss> for the SYSOUT file.

COPYLST	Creates a copy of the current SYSLST file.
COPYOUT	Creates a copy of the current SYSOUT file.
NEWLST	Assigns a new SYSLST file. The old SYSLST file can then be accessed.
NEWOUT	Assigns a new SYSOUT file. The old SYSOUT file can then be accessed.

Note

- SYSLST and SYSOUT can be copied only when a file is currently assigned. Copying is not possible in the event of primary allocation (PRIMARY) or if a PLAM library member is assigned.
- The operations always affect all CENTRAL tasks.

3.2.4 Working with the SOUT task

The SOUT task is managed via inputs in the job variable which is used to control it. The name of the job variable is defined in the call parameters in the start procedure.

With the inputs in the job variable a distinction must be made between

- inputs for terminating the SOUT task and
- inputs for copying and reassigning the system files SYSLST and SYSOUT

All inputs are made via the monitoring job variable. The operation to be executed must be the first entry and consist of job variable values:

```
/MODIFY-JV JV=jvname,SET-VALUE=C'operation'
```

Operations for terminating the SOUT task

SHUTDOWN	The SOUT task is terminated. <ul style="list-style-type: none">– The secondary tasks are terminated if no further job is being processed.– The primary task is terminated if all secondary tasks have been terminated.
CANCEL	The SOUT task is terminated immediately. <ul style="list-style-type: none">– The primary task is terminated immediately without taking the secondary tasks into account.

Operations for copying and reassigning the system files SYSLST and SYSOUT

The system files SYSLST and SYSOUT of the SOUT tasks can only be evaluated during ongoing operation after they have been copied or reassigned.

AVAS creates the copied or new files in the tasks' execution ID under the following names:

AVAS.LST.<tsn>.<yymmdd>.<hhmmss> for the SYSLST file and

AVAS.OUT.<tsn>.<yymmdd>.<hhmmss> for the SYSOUT file and.

COPYLST	Creates a copy of the current SYSLST file.
COPYOUT	Creates a copy of the current SYSOUT file.
NEWLST	Assigns a new SYSLST file. The old SYSLST file can then be accessed.
NEWOUT	Assigns a new SYSOUT file. The old SYSOUT file can then be accessed.

Note

- SYSLST and SYSOUT can be copied only when a file is currently assigned. Copying is not possible in the event of primary allocation (PRIMARY) or if a PLAM library member is assigned.
- The operations always affect all SOUT tasks.

3.2.5 Modifying the server environment in the server monitor process

A controlling job variable is provided as a user interface to administer the server monitor process.

The name of a job variable can be selected as desired and is not stored in the generation parameters. It needs to be passed as a parameter when calling the AVAS.SYS.LOAD.SVDOG server monitoring program.

Operation for updating the server environment

The controlling job variable can be changed via

```
/MODIFY-JV JV=jvname,SET-VALUE=C'CONFIG '
```

starting at the first position of the job variables to be monitored. The operation used is:

CONFIG Server entries in the configuration file have been changed. The server monitor process reads the configuration file and updates the server environment to be monitored.

In order to be able to synchronize with other processes, the server monitor places status information in the controlling job variable (just like other AVAS processes).

The controlling job variable can contain the following values (starting in column 2 of length 10, just like for other AVAS processes):

STARTING1	Program parameters are processed, the configuration file is read
STARTING2	SV-JVs are created or deleted
STARTING3	Server statuses are determined
READY	SV-JVs are updated
RUNNING	The configuration file is reread and/or the server statuses are determined (again)
ENDING	SV-JVs are reset, i.e. they are set to '\$U'
ENDED	AVAS server monitor process is terminated

Changes to the controlling job variable are not processed during the start phase (STARTING1-3), update phase (RUNNING) and after a SHUTDOWN.

How to temporarily lock a server

Job distribution by the run control system can be disabled for a short time for a server in order to configure the server, for example. The corresponding server job variable is set to \$I (Ignore) to accomplish this.

If the lock is to be removed, set the value back to '\$R' (RUNNING).

3.2.6 Monitoring tasks via job variables

Some batch tasks in the AVAS system are monitored by means of the job variables assigned to these tasks (see [section "Definitions for the central access tasks" on page 28](#)). These job variables can be used to query the status of the task concerned.

Querying the status of the tasks

Input: /SHOW-JV JV=jvname

Output: jvvalue (= status of the task)

The job variable can contain up to five status values. It begins with the current status and ends with the oldest status.

The job variable can contain up to five status values. The status entry in the job variable begins with the current status and ends with the oldest status.

The entry TSN=<tsn> is always contained at the end of the job variable (bytes 57 through 64). Here <tsn> is the TSN of the batch task, and in the case of CENTRAL and SOUT tasks the TSN of the primary task. In the case of CENTRAL and SOUT the TSNs of the secondary tasks follow, separated by blanks.

Possible values for the status of a task:

STARTED	The task has been started and is currently in the initialization phase.
READY	The task is ready for operation; no user is active or no net is currently running.
RUNNING	At least one user is active or one net is currently running.
ENDED	The task has terminated normally.
ABENDED	The task has terminated abnormally.
STOP	Task termination was requested (run control system only). The task terminates when there is no net (job) currently running.
HOLD	ZDs: A signon lock has been set for all users. Users cannot sign on again until this lock has been canceled. Run control system: Suspension of processing has been requested. No further nets (jobs) will be started.
RESUME	The signon lock has been canceled. Users can now sign on again (ZDs only).
RESUMED	The processing of further nets (jobs) continues after a HOLD (run control system only).

CANCEL-SOF	Forced signoff of the users has been requested. The users are signed off when they have finished processing the current element.
CANCEL-HAR	Forced signoff of the users has been requested. The users are signed off at the next dialog step. Any changes made to the current element are lost.
SHUTDOWN	Task termination has been requested. The task terminates when there are no more signed-on users.
KILL	Immediate task termination has been requested (ZDs only). The task is terminated immediately.
IN-JRLDAT	The UPAM-ZD has performed the “merge in the emergency journals” function.

Example

The command `/INFORM-PROGRAM MSG='STOP,LEVEL=NET',JOB-ID=*TSN(<tsn>)` is issued for the run control system. There are still nets currently being processed.

Entry of `/SHOW-JV JV=jvavak` produces the following display:

```
RUNNING    STOP        RUNNING    ...        TSN=<tsn>
```

This means that the input has been accepted but that there are still some nets running. Once these nets have terminated, the run control system will also terminate.

Reentering `/SHOW-JV JV=jvavak` produces the following display:

```
ENDED      READY       RUNNING    STOP        RUNNING    TSN=<tsn>
```

If there were no nets running at the time the command `/INFORM-PROGRAM MSG='STOP,LEVEL=NET',JOB-ID=*TSN(<tsn>)` was issued, the following display is output directly:

```
ENDED      READY       STOP       READY       ...        TSN=<tsn>
```


3.3 Runtime management tasks

3.3.1 Managing the system libraries

The system libraries are managed with the following statements:

COPY-SYSTEM-ELEMENT	Copy library elements to a central library
DELETE-SYSTEM-ELEMENT	Delete elements from a central library

3.3.2 Administration of users

For administering the users of an AVAS system, the AVAS administrator can use the following statements:

CANCEL-USER	Forcibly sign off users
MODIFY-SYSTEM-PARAMS	Modify the system parameters
SEND-MESSAGE	Send a message to users
SHOW-SYSTEM-PARAMS	Display the system parameters
SHOW-USER	Display the users who are currently signed on
START-EXIT	Activate the CC exit AVEX2001

For details of inputs for user administration via the job variables that monitor the access tasks, see [section “Working with the access tasks” on page 124](#).

3.3.3 Managing the system parameter file

The parameters in the system parameter file (see [section “Generation of the AVAS system” on page 23](#)) can be displayed by means of the statements described below and modified by the AVAS administrator. It is not always possible to modify the displayed definitions (see [page 23](#) onwards). Information on the connections and relations both among the individual parameters within a parameter record and between the individual parameter records themselves can be found in the aforementioned pages of this manual whenever they cannot be recognized from the displays of these statements.

3.3.4 Management of runtime logs

The runtime logs of BS2000 jobs started via the AVAS run control system can be collected via AVAS and stored in a central library.

Two programs in BS2000, which have to be integrated in the job, are used for transferring the job runtime logs:

- the SIGNAL program notifies AVAS that a log is to be transferred.
- the TRANSFER program transfers the log from the user area to the AVAS area. The TRANSFER program must be called in the AVSSINCM start procedure to capture LOG files from server jobs

The DCAM application CENTRAL, which accepts the log and stores it in the log library (AVAS pool), has been added to the AVAS system.

The interactive statement ADD-JOB-LOG is available for subsequent storage of signaled logs.

A maximum of 99 logs can be signaled and stored for each job run.

The logs can be displayed by means of the SHOW-JOB-LOG, SHOW-NET-STATUS and SHOW-JOURNAL statements.

As part of reorganization the logs can be saved and deleted.

Individual logs can also be deleted using the interactive statement DELETE-JOB-LOG.

For log files to be saved and deleted they must have the status ADDED or TRANSFERRED.

Collecting the logs

A DCAM application is available for storing logs centrally at the time of the job run. The application consists of a number of DCAM programs which communicate with one another. The SIGNAL and TRANSFER programs notify the central task CENTRAL that the logs are available for transfer and then transfer them for central storage.

CENTRAL task

CENTRAL is a shareable DCAM application.

CENTRAL consists of a primary task and a certain number of secondary tasks. The primary task provides the resources for the central DCAM task and controls the application. The secondary tasks take on the requirements of the DCAM programs and store the logs at a central location.

In addition to the central access tasks (ZDs) the CENTRAL task is started by means of an ENTER call on the same mainframe and under the same user ID. CENTRAL logs on to the ZDs as a special task (in the same way as AVAK).

SIGNAL program

SIGNAL is a non-shareable DCAM program.

Immediately after the LOGON statement and the ASSIGN-SYSOUT statement, SIGNAL is loaded as the first program in an AVAS job. It ascertains the AVAS job ID and the name of the file with the SYSOUT information and sends this information together with other control parameters to CENTRAL. SIGNAL indicates any problems in a program job variable. If further logs (e.g. SYSLST) are to be stored centrally, the SIGNAL program must be called for each log.

TRANSFER program

TRANSFER is a non-shareable DCAM program.

Either it is loaded in the AVAS job immediately before the /EXIT-JOB or /LOGOFF statement as the last program or it runs after the end of the job (activated by AVAS) as an independent job.

TRANSFER ascertains the AVAS job ID and sends this information together with any specified parameters to CENTRAL. If the information is correct, CENTRAL requests the TRANSFER program to transfer the log data. Any problems which occur in the TRANSFER program are reported in a program job variable.

3.3.4.1 Storing the logs

The logs of the jobs which have run under AVAS control are stored in a PLAM library. This means that these log elements are accessed via the central access task (PLAM-ZD) for PLAM libraries. The BS2000 file name of the library is defined by generation parameter LOGSYS.

A log is entered in the library

- as a JOBLIST record in a type S element and
- as a data collection in a type D element.

The name of a log element comprises the following components:

Type S

\$ug_netname_plandate_plantime_catid_tsn_date

Type D

\$ug_netname_plandate_plantime_catid_tsn_date_nn

Meanings and lengths (in brackets) of the name components

ug	user group (1 .. 4)
netname	net name (1 .. 12)
plandate	planned start date (6)
plantime	planned start time (6)
catid	catalog ID (1 .. 4)
tsn	task sequence number (4)
date	date of log entry (8)
nn	serial log number (2)

Element structure for type S

- Directory record

The directory record contains AVAS information needed for editing the logs of a job.

Structure: *SYSLOG*<s1><s2><s3><s4>

Meaning and length (in brackets) of the name components:

SYSLOG	Record ID (8)
s1	Name of the run control system (8)
s2	Index level of the structure element in the net (3)
s3	Name of the structure element in the net (24) (without user group)
s4	Internal AVAS area

- Log message record (maximum of 99 per job)
A record with the ID *JOBLOG* is entered for each log signaled. In its other elements the record with this unique feature defines the necessary information on a log.

Structure: *JOBLOG*<j1><j5><j6><j7><j8><j9><j2><2t><j3><3t><j4><4t>

Meaning and length (in brackets) of the name components:

JOBLOG	Record ID (8)
j1	Log number (2)
j5	AVAS return code (4)
j6	SUBSYSTEM return code (7)
j7	JOBLOG name (54)
j8	Date yyymmdd (8)
j9	Time hhmmss (6)
j2	Log status code (1)
2t	Log status text (11)
j3	TRANSFER parameter code (1)
3t	TRANSFER parameter text (5)
j4	Log delete code (1)
4t	Log delete text (3)

Element structure for type D

- Function record
A function record exists only if a log has been incorporated by means of a function (e.g. ADD-JOB-LOG). This record defines the information relating to the execution of the function.

Structure: *function*<f1><f2><f3><f4>

Meaning and length (in brackets) of the name components:

function	Record ID (24) (e.g. ADD-JOB-LOG)
f1	Name of the AVAS user (8)
f2	Date yyymmdd (8)
f3	Time hhmmss (6)
f4	Name of the file assigned (54)

- Log records

The log records are the records read from the transferred file.

Status of the elements

An internal hexadecimal AVAS status is maintained in the DIRECTORY area of the type S element and in the log message records.

This status of the log entries provides the following information:

ADDED

The log data of the job has been collected using the ADD-JOB-LOG function.

ASSIGNED

No log data for the job. In the job a log name was signaled and no request made for data transfer.

CREATED

No log data for the job. The log entry was created by the run control system (GENERATE-JOB-LOG=YES).

ERROR

An error occurred in the job during transfer of the log data. No log data is stored.

IGNORE

A log name was signaled in the job run. The log data was later not required.

SAVED

The log(s) of a jobs were stored in the JOBLOG save operation.

TRANSFERRED

All the log data was collected without errors during the job.

Status priority

The status is determined in the DIRECTORY entry of the type S elements and the display of the group line according to the following priority:

```

ERROR,
      CREATED,
            ASSIGNED,
                  ADDED,
                        TRANSFERRED,
                              SAVED,
                                    IGNORE

```

Restrictions

The lengths of the log records must not exceed the maximum length supported by the software being used. This means that AVAS can collect log records with a maximum length of 2028 bytes in the log library. The SHOW functions with the aid of the EDT, however, can only display records with a length of 256 bytes. If a longer record is encountered, the current function is terminated with an error.

3.3.4.2 Collecting the logs**Starting the CENTRAL task**

In addition to the central access tasks (ZDs) the CENTRAL task is started on the same mainframe and under the same ID. As far as the ZDs are concerned, the task is a single and unique user.

The CENTRAL task is started by calling the AVS.CENTRAL procedure (an element in the library SYSPRC.AVAS.085), once as a primary task and more than once as a secondary task (the number of times can be set via the GENPAR parameter MAX-CENTRAL-PROCESS).

The following information for signing on to the ZD and for task execution is read via SYSDTA as parameter statements.

APPLICATION={{(name) / jobvariable}

(name)	Name of the CENTRAL task; the name of this DCAM application must be unique in BS2000.
jobvariable	The name of DCAM application CENTRAL starts at position 1 in the specified job variable.

AVAS-SYSTEM-ID=string

7-character string (ID) which identifies the AVAS system. The AVAS system ID must be specified.

DEFAULT-TRANSFER-OPTION={ENDED / ERROR / NO}

The default value of the TRANSFER-OPTION control parameter for the SIGNAL function is ENDED. It can be changed here to one of the above values. See the function for the meanings.

DEFAULT-DELETE={YES / NO}

The default value of the DELETE control parameter for the SIGNAL and TRANSFER functions is NO. It can be changed here to YES. See the function for the meanings.

DEFAULT-TRANSFER-FILENAME={*ALL / *LAST}

The default value of the TRANSFER-FILENAME control parameter for the TRANSFER function is *ALL. It can be changed here to *LAST. See the function for the meanings.

START={PRIMARY / SECONDARY}

PRIMARY The primary task of CENTRAL is to be loaded.

SECONDARY A secondary task of CENTRAL is to be loaded.
If START is omitted, SECONDARY is assumed.

ZDD-PASSWORD={*STD / password}

*STD The internally defined password is used for signing on to UPAM-ZD.

password The password must be entered with a length of 4 characters, in alphanumeric (C'....') or hexadecimal (X'.....') form.
If ZDD-PASSWORD is omitted, *STD is assumed.

Note

CENTRAL can only sign on successfully to UPAM-ZD if the password matches the password defined with CENTRAL-PASSWORD when UPAM-ZD was started.

ZDL-PASSWORD={*STD / password}

*STD	The internally defined password is used for signing on to PLAM-ZD.
password	The password must be entered with a length of 4 characters, in alphanumeric (C'....') or hexadecimal (X'.....') form. If ZDL-PASSWORD is omitted, *STD is assumed.

Note

CENTRAL can only sign on successfully to PLAM-ZD if the password matches the password defined with CENTRAL-PASSWORD when PLAM-ZD was started.

For CENTRAL to be able to receive the logs directly, before the CENTRAL task is called, the passwords for the log files of the application tasks should be defined (BS2000 command /ADD-PASSWORD ...) for the secondary tasks in the AVS.CENTRAL procedure.

Terminating the CENTRAL task

The CENTRAL task can be terminated normally or abnormally.

Normal termination

The CENTRAL task is made to execute a termination by setting the SHUTDOWN string in the assigned job variable.

The name of the job variable is entered in the system parameters.

A single secondary task terminates only if no more jobs are processed. When all the secondary tasks have terminated the primary task is also closed.

Abnormal termination

An unrecoverable task error (program or macro error etc.) causes a memory dump to be output and an error message to be issued.

If an error occurs in a secondary task, only the task affected is terminated. In this case it is possible to start a new secondary task by means of the start procedure.

An error in the primary task also leads to termination of all secondary tasks. When the cause of the error has been eliminated, the CENTRAL task must be started in its entirety.

The same procedure applies if a CENTRAL task is subjected to hard termination by a BS2000 system command.

Starting the SIGNAL program

The SIGNAL program must be loaded in an AVAS job and requires a task job variable. It ascertains all the information for log transfer and sends this information to the CENTRAL task. SIGNAL may be called more than once within a job run, but an error occurs if the same log name is reported more than once. The log records of the first signaled file of a job run can be called up for display in the SHOW-NET-STATUS or SHOW-JOURNAL functions.

The SIGNAL program is started by calling the AVS.SIGNAL procedure (an element in the library SYSPRC.AVAS.085) or by means of a user-defined procedure (or command sequence) within an AVAS job.

The following parameters can be defined via SYSDTA for controlling the run.

APPLICATION={{(ptnname,proname) / jobvariable}}

(ptnname,proname)

ptnname is the application name of the CENTRAL task.
proname is the processor name of the CENTRAL task.

jobvariable

The application name of the CENTRAL task starts at position 1, the processor name at position 9 in the specified job variable.

The APPLICATION parameter must be specified.

FILE-NAME={*STD / *SV / filename}

*STD

The file name for the SYSOUT log is ascertained by SIGNAL. If SYSOUT was not assigned to a file, the SIGNAL program is terminated with an error.

The SIGNAL is called within a job immediately after a SYSOUT assignment. Note that all the information prior to the SYSOUT assignment does not flow into the assigned file.

filename

Indicates the name of the log file to be sent to CENTRAL. Any SAM or ISAM file (without an index key) can therefore be transferred.

Note

This parameter is only required if a log file other than that assigned via ASSIGN-SYSOUT is to be collected.

TRANSFER-OPTION={ENDED / ERROR / NO}

ENDED

The run control system starts a follow-up job for transferring this log if the log is not transferred in the job run as a result of a calling TRANSFER. Error-free transfer of this log is not a requirement for the normal termination of the job in the AVAS system.
The job status is not affected by the result of log transfer.

ERROR The run control system starts a follow-up job for transferring this log if the log is not transferred in the job run as a result of a calling TRANSFER. Error-free transfer of this log is a requirement for the normal termination of the job in the AVAS system. Failure to transfer the log leads to job status ERROR. In this case a restart must be initiated for the net/job.

NO The run control system does not start a follow-up job for transferring this log. If log transfer is performed in the job run, users must themselves set normal or abnormal termination of the job by evaluating the return codes in the job and therefore also set the status of the job in the AVAS system.

Notes

- If the TRANSFER-OPTION parameter is omitted, the value specified when CENTRAL was started is assumed.
- The run control system does not start follow-up jobs for logs for which transfer has been attempted in the job but was not performed owing to an error.
- If transfer of a log has been aborted owing to an error, the data already transferred is not stored. The status ERROR is assigned to the signaled log in the directory of logs.
- If logs with different TRANSFER-OPTIONs are signaled, specifying ENDED or ERROR always triggers the start of the TRANSFER follow-up job. In this case all logs with the status ASSIGNED are handled by the TRANSFER program.

DELETE={YES / NO}

The DELETE parameter should only be specified for SIGNAL if

- several logs are signaled and
- the files are to be handled differently after error-free transfer and
- TRANSFER is called in a follow-up job by the run control system.

The files are deleted by TRANSFER.

YES The file is to be deleted after error-free storage.

NO The file is retained under the user ID created.

Note

If the DELETE parameter is omitted, the value specified when CENTRAL was started is assumed. The value thus set can be changed by an entry in TRANSFER.

Terminating the SIGNAL program

The SIGNAL program terminates when all the information has been sent to the CENTRAL task.

If errors occur during the SIGNAL operation (e.g. incorrect parameters entered, CENTRAL task not available, etc.), a message is issued. If a job variable was specified when SIGNAL started, the message number (without AVS) is entered in the return code display field of the job variable.

Notes

- If there is more than one log for a job, only the first log is displayed via the JOBLOG operation for the SHOW-NET-STATUS and SHOW-JOURNAL functions. It is therefore advisable to signal the file with the ASSIGN-SYSOUT assignment as the first file.
- Tasks for which TYPE=EXX is set can only work with the SIGNAL program if the external monitoring job variable can be addressed as a task job variable via the JV link name *SMONJVJ.

Starting the TRANSFER program

The TRANSFER program requests the CENTRAL task to collect the specified log file. If CENTRAL itself cannot read the file, TRANSFER transfers it using DCAM, for which it requires access to the task job variable. The transfer takes place in the same job or in a follow-up job.

- For a transfer in the same job, the AVS.TRANSFER procedure (an element of the library SYSPRC.AVAS.085) or a user-defined procedure (or command sequence) is incorporated in the AVAS job.
- Transfer in a follow-up job is controlled via the TRANSFER-OPTION parameter in the SIGNAL program.

Before the TRANSFER program is called, the log files should (as far as possible) be made accessible to the CENTRAL task (e.g. by the BS2000 command MODIFY-FILE-ATTRIBUTES...).

The following parameters can be specified for controlling the execution of TRANSFER.

APPLICATION={ (ptnname,praname) / jobvariable}

(ptnname,praname)

ptnname is the application name of the CENTRAL task.

praname is the processor name of the CENTRAL task.

jobvariable

The application name of the CENTRAL task starts at position 1, the processor name at position 9 in the specified job variable.

The APPLICATION parameter must be specified.

FILE-NAME={ *ALL / *LAST / filename}

*ALL

All the logs signaled by the SIGNAL program are to be transferred.

*LAST

The last log signaled is to be transferred. The log not yet processed by TRANSFER is transferred.

filename

Name of a particular log file.

The name of the log must be signaled to the CENTRAL task and therefore known to the CENTRAL task. Otherwise the specification is ignored and no message issued.

Note

The default value of the parameter is *ALL. The default value may be changed to *LAST in the CENTRAL task.

DELETE={ YES / NO}

YES

The file is to be deleted after error-free storage.

NO

The file is retained under the user ID created.

Note

If the DELETE parameter is omitted, the value specified when SIGNAL was started or the value set via CENTRAL is assumed.

The DELETE value for TRANSFER overrides the DELETE value set for SIGNAL.

Terminating the TRANSFER program

The TRANSFER program terminates once all the information and data have been sent to the CENTRAL task.

If errors occur during the TRANSFER operation (e.g. incorrect parameters entered, CENTRAL task not available, file protected by a READ password, etc.), a message is issued. If a job variable was specified when TRANSFER started, the message number (without AVS) is entered in the return code display field of the job variable.

Notes

- The JCL statements must ensure that errors in the TRANSFER routine lead to an abnormal termination of the job run, or to an entry in the task job variable. This is the only way in which an errored transfer can result in the ERROR status being set for the structure element in the net.
- With ADD-JOB-LOG and in the TRANSFER program, the data of the logs is read via SYSDTA. Possible file formats are described under the BS2000 command ASSIGN-SYSDTA (see manual “Commands” [7]).
- Tasks for which TYPE=EXX is set can only work with the TRANSFER program if the external monitoring job variable can be addressed as the task job variable via the JV link name *SMONJVJ.

Comments on the control parameters

When the SIGNAL and TRANSFER programs are started, parameters are defined via SYSDTA for controlling the run. The default values for these control parameters have been selected so that data for a log can be easily collected. In addition, these parameters enable further SAM or ISAM files to be transferred to the AVAS pool so that job production in the computer center is as complete and revision-proof as possible.

Notes on the control parameters

- FILE-NAME= in the TRANSFER program

This parameter is only needed if only a particular log is to be collected even though more than one log has been signaled. By specifying a name a 1-to-1 relationship can be established between SIGNAL and TRANSFER.

The value *LAST enables a run to be controlled on the “last in – first out” principle, where even an error in the TRANSFER program fulfils the principle.

If the TRANSFER program is started by the run control system as a follow-up job FILE-NAME=*ALL is assumed.

- TRANSFER-OPTION in the SIGNAL program

Transfer of the logs signaled in an AVAS job via SIGNAL can take place when TRANSFER is called in the AVAS job or when the run control system starts a follow-up job with TRANSFER. The run control system normally checks whether TRANSFER was started in the AVAS job. It starts a follow-up job if TRANSFER was not called. The user does not have to take this follow-up job into account in the AVAS net structure

- If TRANSFER-OPTION=NO is specified, the start of a follow-up job for transferring logs can be suppressed, irrespective of whether TRANSFER was called in the AVAS job or not.

If TRANSFER was called in the AVAS job and an error occurred during transfer, the user can determine normal or abnormal termination of the job by evaluating the return code.

The signaled logs can be subsequently collected with the ADD-LOG-JOB statement.

If TRANSFER was not called in the AVAS job and an error occurred during transfer of a log, the TRANSFER-OPTION parameter can be used to determine whether this error is to be transferred to the job status or not.

- If TRANSFER-OPTION=ENDED is specified, the job status is not affected by an error in log transfer. As far as the job status is concerned, the only status which matters is the one set at the end of the AVAS job.

- If TRANSFER-OPTION=ERROR is specified, the ERROR job status is set if an error occurs during log transfer. Job status ENDED is only achieved if the AVAS job has terminated normally and the signaled logs have been transferred error-free.
- DELETE={YES / NO} in the SIGNAL and TRANSFER programs
 This parameter controls the deletion of log files after error-free transfer by TRANSFER. If the parameter is omitted from SIGNAL and from TRANSFER, the default value specified when the CENTRAL task was started is used.
 If the parameter is omitted from SIGNAL, the default value of the CENTRAL task is used.
 If the parameter is specified for TRANSFER, the value set for SIGNAL is overwritten.

Measures for optimizing the collection of logs

If the procedures previously used for collecting together logs run satisfactorily, they need not be changed. However, if large log files need to be transferred, the run times for previous procedures may become inconveniently long. In this situation, the collection of logs can be accelerated by applying the measures in the upper part of the following matrix:

	Ensure that CENTRAL can read the log file	Y	Y	N	N
	Ensure that CENTRAL can write to the LOGSYS library	Y	N	Y	N
(A)	CENTRAL reads the JOBLOG files directly	x	x	-	-
	The log is read by TRANSFER and sent to CENTRAL by DCAM	-	-	x	x
(B)	CENTRAL writes directly into the LOGSYS library	x	-	x	-
	CENTRAL sends the log for writing by ITC to the PLAM-ZD	-	x	-	x
	Fastest procedure	x	-	-	-
	Most resource-intensive procedure	-	-	-	x

Effects of (A):

Since CENTRAL can access the log file (OPEN INPUT, SAM/ISAM), the file transfer to CENTRAL by TRANSFER using DCAM is not required.

Effects of (B):

Since CENTRAL can write to the LOGSYS library via ILAM (PMOPEN OUT), it is not necessary to transfer the file between CENTRAL and PLAM-ZD using ITC.

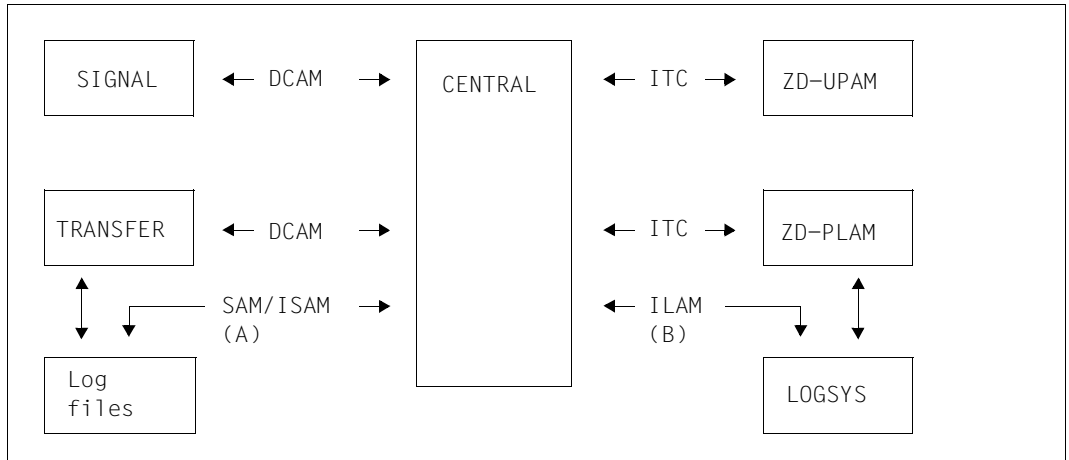
Overview

Figure 2: Sequence of log processing activities

Route (A) or (B) will be selected when the log file or the LOGSYS library, respectively, can be processed from CENTRAL.

Route (A) is only possible for operations on a single computer, not for multiprocessor operations.

The measures which should be considered in order to make the log files accessible must include the following, for example:

Before the TRANSFER program is called, CENTRAL must be given authorization to access all the log files, e.g. by:

```
/MODIFY-FILE-ATTRIBUTES FILENAME=.....,-
/ PARAMETERS(USER-ACCESS=ALL-USERS)
```

For details of how to define such access authorization for AVAS alone, please refer to the SECOS access control system.

Before the CENTRAL program is called, the passwords for the log files for all applications must be specified: ADD-PASSWORD '.....'

3.3.4.3 Editing the logs

The runtime logs of jobs are stored in an AVAS pool. They can be added, deleted and displayed by means of the following statements.

ADD-JOB-LOG	Add log data
DELETE-JOB-LOG	Delete logs
SHOW-JOB-LOG	Display logs

The log data is displayed via EDT.

If the NET-NAME operand is specified in the statements, the qualified net name determines whether a net or a directory of all nets is displayed from the AVAS pool. Individual nets can be selected for editing by marking them with S or Y.

If there are no more logs in the AVAS pool, they cannot be edited using the interactive AVAS functions. The logs can then only be accessed via the log save operation. The logs must have been saved during reorganization (see reorganization parameters).

The display of the net overview in mask AVI016 is sorted according to the NET-NAME parameter.

The display of the job runs of a net in mask AVI017 is sorted according to the IND, DATE, TSN and JOB-NAME parameters.

The logs of a jobs in mask AVI018 are displayed in the order signaled.

3.3.4.4 Overview of log statuses in statements

Statement/ function	LOGSYS Old status	LOGSYS New status	Explanation
AVAK	---	CREATED	A log was signaled in the job run by SIGNAL.
SIGNAL	---	ASSIGNED	The log was collected in the job run.
TRANSFER	ASSIGNED	TRANSFERRED	The log could not be collected in the job run. An error occurred.
	ASSIGNED	ERROR	The log could not be collected in the job run. An error occurred.
ADD-JOB-LOG	CREATED	ADDED IGNORE	A first log has been added to this job run. No log data is to be added to this log entry.
	ASSIGNED	ADDED IGNORE	A signaled log has been added. A signaled log is no longer relevant.
	ERROR	ADDED IGNORE	A signaled log which could not be collected in the job run has been added. A signaled log is no longer relevant.
	ADDED	ADDED	A new or additional log has been added.
DELETE-JOB-LOG	ADDED ASSIGNED CREATED ERROR IGNORE SAVED TRANSFERRED	---	The log(s) of a job run has or have been deleted. Log library editing is no longer possible for this job run.

3.3.5 Displaying the SYSOUT files

The SYSOUT files of BS2000 jobs which are started using the AVAS run control system can be made available to the AVAS dialog tasks and, for example, be displayed there using EDT. This applies both for jobs in the local BS2000 system, i.e. for jobs which were started via an MSCF connection, and for BS2000 server jobs.

In the AVAS system the SOUT task is provided for this purpose. The SOUT task opens the SYSOUT file, temporarily copies it, and transfers the copy to the dialog task.

Transfer of the SYSOUT file is initiated in the dialog task by means of the OUTSYS (#79) operation under the NET-CONTROL command.

3.3.5.1 The SOUT task

The SOUT task is a shareable DCAM application which can access AVAS jobs via privileged system interfaces. Like the CENTRAL task it consists of a primary task and a certain number of secondary tasks. The primary task provides the resources of the central DCAM task and controls the application. The secondary tasks receive the requests of the DCAM programs and store the logs at a central location.

The SOUT task must be started on every BS2000 system on which AVAS-SERVER jobs are started.

The AVAS dialog tasks address the SOUT task via its DCAM application name, which they obtain from GENPAR when signing on to the AVAS access tasks. The name can be specified there in two different ways:

- The DCAM application name must be specified explicitly (SOUT-APPLICATION-NAME parameter).
- The name of a job variable, together with its read password, which contains the DCAM application name must be specified (SOUT-APPLICATION-JV and SOUT-APPLICATION-PASSWORD parameters).
The job variable must be accessible for dialog tasks. This must be ensured above all when the SOUT task's application job variable is used.

The AVAS dialog task, AVAS job and SOUT task can run on different BS2000 systems. In this case the SOUT task is addressed using the DCAM application name and using the BCAM name of the host which the dialog task obtains from the job's AVAS runtime data.

The AVAS dialog tasks therefore address SOUT tasks on various hosts via the host name and **one** identical DCAM application name.

The DCAM application name must consequently be the same for all SOUT tasks which work with an AVAS system.

As the SOUT task (in contrast to CENTRAL) does not sign on to the AVAS access tasks, it can also work with multiple AVAS installations on different hosts. However, multiple independent SOUT tasks can be started on a host on an AVAS-system-specific basis.

The figure below provides a general view of OUTSYS processing activities:

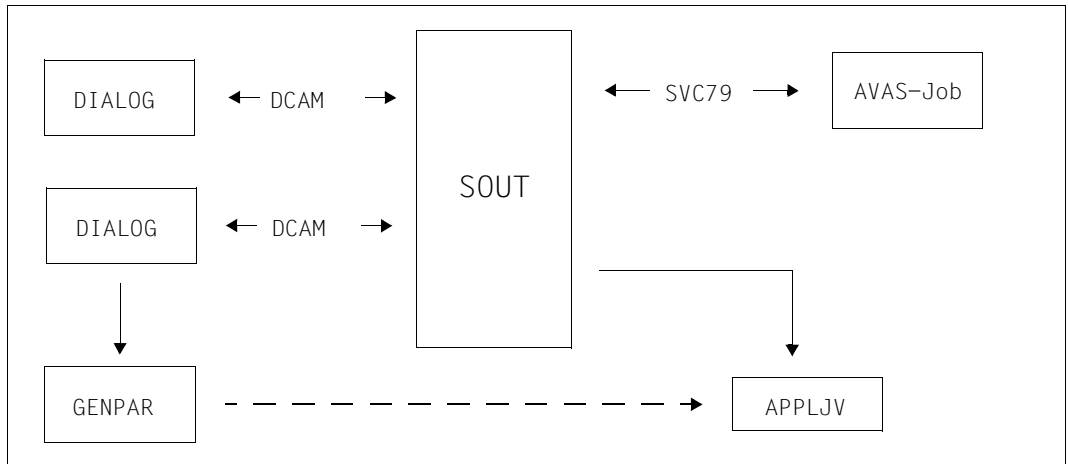


Figure 3: Sequence of OUTSYS processing activities

3.3.5.2 Runtime environment of the SOUT task

Privileged system interfaces of the BS2000 operating system are required to open and provide the SYSOUT files of the AVAS server jobs. For this purpose an SOUT subroutine is started in privileged mode using the BS2000 subsystem CAPRI (i.e. ultimately via SVC79).

The following conditions must be satisfied to permit privileged code to be called using CAPRI:

- In BS2000 the CLASS-2 option SVC79 must be set so that CAPRI or SVC79 calls can be executed without a query at the console (setting SVC79=0).
- The SOUT task must be started in such a way that a CAPRI or SVC79 call is permissible. This is generally the case under the BS2000 ID TSOS.

If the GUARDS subsystem (component of the software product SECOS) is available, the CAPRI functionality can also be provided under other IDs. An ACCESS-CONDITION and SVC79 co-ownership must be configured for this purpose.

Configuring the ACCESS-CONDITION

```
/ADD-ACCESS-CONDITION GUARD-NAME=<protection guard>
, SUBJECTS=*USER(USER-ID=(<userid1>,<userid2>,...))
, ADMISSION=*YES
```

<protection guard> Freely selectable name for the connection between the ACCESS-GUARD and the COOWNER-GUARD. This name must also be specified after SVC co-ownership has been configured.

<userid1> BS2000 ID which is to be privileged for executing the SOUT task. Up to 20 IDs can be specified in a list.

Configuring SVC79 co-ownership

```
/ADD-COOWNER-PROTECTION-RULE RULE-CONTAINER-GUARD=$TSOS.SYS.UCC
, PROTECTION-RULE=<management name>,
, PROTECTION-OBJECT=*PARAMETER(NAME=<avas-syslnk>
, CONDITION-GUARD=<protection guard>)
```

<protection guard> Name for the connection between the ACCESS-GUARD and the COOWNER-GUARD (specified as in ACCESS-GUARD).

<management name> Sort key within the RULE-CONTAINER.

<avas-syslnk> CAPRI load library CAPLNK from the start procedure of the SOUT task (generally \$<avas-userid>.SYSLNK.AVAS.085)

3.3.5.3 Starting the SOUT task

The SOUT task must be started on every host on which AVAS server jobs run whose SYSOUT files are to be made accessible to the AVAS dialog tasks.

As the SOUT task does not sign on to the AVAS access tasks, all control data which is relevant to execution is defined as parameters of the start procedure AVS.SOUT (member of the SYSPRC.AVAS.085 library).

The SOUT task (like the CENTRAL task) is started once as a primary task and multiple times as a secondary task. In contrast to the CENTRAL task, the number of secondary tasks cannot be defined for the SOUT task in GENPAR.

Procedure parameters of the start procedure AVS.SOUT

APPLICATION={ (name) / job variable }

- | | |
|--------------|---|
| (name) | Name of the SOUT task; the name of this DCAM application must be unique in BS2000. |
| job variable | Name of the job variable which contains the DCAM application name of the SOUT task starting at position 1.
If the job variable is protected by a read password, this must be specified in the call parameter JVAPP-PASSWORD. |

CONTROL-JVA=job variable

- | | |
|--------------|---|
| job variable | Name of a job variable for monitoring the SOUT task.
The job variable is configured by the task when it starts. The SOUT task must be able to access this job variable in write mode.
The primary task and all the associated secondary tasks of a SOUT task (specified via the call parameter APPLICATION) must be monitored with the same job variable. |
|--------------|---|

JVAPP-PASSWORD={ *NONE / password }

- | | |
|----------|---|
| *NONE | The job variable which contains the DCAM application name of the SOUT task is not protected with a read password. |
| password | The password must be specified in 4 characters, in alphanumeric (C'....') or hexadecimal (X'.....') format. |

If JVAPP-PASSWORD is not specified, *NONE is assumed.

START={ PRIMARY / SECONDARY }

- | | |
|-----------|---|
| PRIMARY | The primary task of SOUT is to be loaded. |
| SECONDARY | A secondary task of SOUT is to be loaded. |

If START is not specified, SECONDARY is assumed.

3.3.5.4 Terminating the SOUT task

The SOUT task can be terminated normally or abnormally.

Normal termination

The SOUT task is instructed to terminate by setting the SHUTDOWN string in the assigned job variable.

The name of the job variable is entered in the system parameters.

The single secondary task terminates only if no further job is being processed. When all secondary tasks have terminated, the primary task is also terminated.

Abnormal termination

A non-recoverable process error (program or macro error) leads to a memory dump being output and an error message being issued.

If the error occurs in a secondary task, only the task affected is terminated. In this case it is possible to start a new secondary task using the start procedure.

An error in the primary task also results in all secondary tasks being terminated. In this case a complete startup of the SOUT task is required after the error has been corrected.

The same behavior is encountered when the SOUT task is terminated using the BS2000 command CANCEL-JOB.

3.3.6 Optimizing the system

If the use of AVAS is accompanied by problems regarding runtime throughput or unsatisfactory response times in the interactive components, they can be remedied either by instituting organizational measures or by making appropriate technical adjustments to the application.

To assist you in making the best use of the AVAS system, attention is drawn to the following measures:

Organizational measures

When AVAS is used there is a wide variety of options for streamlining the operation of production planning, production preparation, release for production and production execution. To a considerable extent, execution of these operations can be conveniently staggered.

The following basic guidelines for the timing of these operations can be put forward:

- Time scheduling via the calendar should be performed in advance for the longest period of time possible (e.g. one to three months or more). The statements do not constitute a major workload for the system, nor do they create any data redundancies.
- Production planning should be performed in advance for a medium period of time (up to 14 days or two months, depending on the scope of the production). Production planning is not a critical function, and generates only minimal data redundancies (modified net structure). The collection of net parameters can also be taken care of in advance for the same period of time via COLLECT-NET-PARAMS. In this case, only limited amounts of data accrue and there are no redundancies. The data can be updated at any time with no great overhead.
- Parameter modification should be performed in advance for a shorter period of time (2 to 14 days, depending on the scope of the production and on whether the parameters are known). When parameters are modified via CREATE-PROD-NET, all jobs subject to modification are mapped in the JMDLIB.
- Release for production execution (SUBMIT-NET) should take place as closely as possible to the scheduled time of processing (one or two work days before). During release for production, complete duplicates of the nets are created in the run control file ABLDAT. Nets which are released too soon create an unnecessary load on the directories of the run control file and hinder the search operations of the run control system.

To help the user decide what to do, the table below lists those statements which involve movement of the greatest amounts of data and overlapping use of files.

Statement	Input	Output
COLLECT-NET-PARAMS	NPRLIB	NPRLIB JRNDAT
CREATE-PROD-NET	NPRLIB +++ JCLLIB JCLSYS	NPRLIB +++ JMDLIB ### JRNDAT
SUBMIT-NET	NPRLIB +++ JMDLIB ### JMDSYS	NPRLIB ABLDAT ### JRNDAT
Run control system	ABLDAT +++ JRNDAT	ABLDAT +++ JRNDAT

The critical files are indicated as follows:

+++ Large number of access operations on the file.

Large amounts of data moved.

This summary shows that the critical point can be found at the SUBMIT-NET statement. Improved response time and throughput can be expected for all statements when this statement is performed largely separated from the others.

We recommend that the user adapt his organizational concept accordingly.

It should also be pointed out that these considerations only apply to the release for production of large numbers of nets, not to the individual nets themselves.

Application-specific steps

To obtain optimum response time and throughput, the AVAS user should adhere as closely as possible to the following guidelines:

- On average, a job should have no more than 50 to 100 commands.
- On average, a net should not have more than 20 to 40 jobs.
- /REMARK or /WRITE-TEXT commands in the jobs can be kept to a minimum in the AVAS jobs.
- Statements which are used solely for documentation purposes should be stored externally in documentation elements for the jobs.
- All jobs which are brought to execution via the nets, but are not subject to modification and do not have any AVAS restart statements, can be managed outside the AVAS system (where they can be edited under AVAS by means of EDT), provided JOB-TYPE=EXT is specified in the net. This offloads the run control file, the SUBMIT-NET statement and the run control system.
- It is better to collect run parameters for the jobs in a net via USER-PARAM-FILE or via COLLECT-NET-PARAMS rather than to modify jobs by calling user masks in the job with CREATE-PROD-NET.
- For all jobs which are called, the user group \$ug_ or \$ugsys_ must be specified in the net. If no user group is specified, the job is sought first in the JMDLIB of the user's own user group and then, if it cannot be found there, in the system library JMDSYS.
- Release for production should not take place until one or two days prior to the scheduled date of production, so as to prevent the run control file from getting bloated. It is possible to optimize the scanning of the run control file by working with different run control systems when using different "control-times" (see [section "Defining the run control system" on page 42](#)).
- When it is necessary to collect logs together, the CENTRAL tasks (PRIMARY and SECONDARY) should be run under the same BS2000 ID as the PLAM-ZD. If the LOGSYS library is password-protected, ADD-PASSWORD must be used to assign its password in the secondary CENTRAL task before the program is started. The CENTRAL task then records the logs directly in the LOGSYS library, rather than via the PLAM-ZD.

3.3.6.1 System optimization in batch mode

In situations where batch functions are being used intensively, timing problems may occur when processing AVAS statements via BATCH, particularly when AVAS-BATCH is called up from within AVAS nets.

Indications that the system is reaching a critical state are:

- 'RESULT: LOCKED' is displayed for a net or a number of nets when preparing nets under AVAS-BATCH.
- The run control file and journal file temporarily show a different status for a net.
- The runtime of nets containing jobs in which AVAS-BATCH statements are being issued increases considerably.
- 'RESULT: ERROR' is displayed for one or more nets when the CREATE-PROD-NET and/or SUBMIT-NET statements are issued because the associated jobs could not be read.

These situations are caused by lengthening wait queues at the interfaces to the run control file ABLDAT, the journal file JRNDAT and the libraries, and consequently around the central access tasks ZDD and ZDL as well.

To avoid critical situations, it is essential that repeated attempts to access locked elements or blocks be prevented or at least minimized. This is achieved with the generation parameter MAX-BATCH-PROCESS (see [page 29](#)) and the MAX-SERIAL-LOCK and MAX-SERIAL-WAIT statements at the start of the UPAM-ZD access task (see [page 111](#)).

The critical statements CREATE-PLAN-NET, CREATE-PROD-NET, SUBMIT-NET and REPEAT-NET are serialized via the following wait queues.

The following queues are created:

- Queue of the batch tasks
If more than n batch tasks are started in MAX-BATCH-PROCESS=n, only n tasks will be processed. All others are put in a queue.
- Queue of the CREATE-PLAN-NET statement
- Queue of the CREATE-PROD-NET statement
- Queue of the SUBMIT-NET and REPEAT-NET statements

Example

Before execution of the critical processing sequence, the SUBMIT-NET function passes a serialization argument to the UPAM-ZD. This causes the processing to be exclusively controlled by this argument right through to the end message.

A second function task with the SUBMIT-NET function and the same serialization argument must wait in the UPAM-ZD until the exclusively working function has finished. During the wait time, there is no communication between the second function task and the UPAM-ZD. The maximum wait time is specified by the MAX-SERIAL-WAIT statement.

This procedure serializes processing sequences of a function task. Processing only comes to a standstill if a function task working with a serialization does not reach the end message and one or more function tasks with the same serialization argument are waiting. The end of the serialization can only be omitted by a function task in exceptional circumstances. A possible resulting obstruction of the SUBMIT-NET function is prevented by means of the MAX-SERIAL-LOCK statement.

The following sequence of serialized statements occurs:

1. Every serialization task is allocated a processing time defined via the MAX-SERIAL-LOCK statement. These times are determined according to the longest possible processing time of the net. If the MAX-SERIAL-LOCK time is exceeded, the process is aborted. The user is forcibly signed off and must sign on again.

Note

When MAX-SERIAL-LOCK has been processed, a CANCEL-USER (HARD) is performed internally for the user in order to release the lock state. No signon lock is set. The overlong interruption of the processing could have been caused, for example, by a USER-EXIT.

2. At task submission, **every** waiting serialization task receives the wait time defined by the MAX-SERIAL-WAIT statement. This time is determined from the average size of a net and the number of nets that have to be processed simultaneously via a serialized statement at the peak time of the system.
If the MAX-SERIAL-WAIT time is exceeded, the process is aborted with a message.

To avoid wait queues, the access tasks must be given a considerably higher priority than the processes using AVAS-BATCH statements. Similarly, the ZDD process must be given a higher priority than the ZDL process. In the case of the critical statements, more than twice as many accesses are performed through the ZDD process as through the ZDL process.

Example

AVAS-BATCH	PRIO=255
Ablaufsteuerung	PRIO=250
AVAS-Dialog	PRIO=240
CENTRAL	PRIO=220
ZDL (PLAM)	PRIO=220
ZDD (UPAM)	PRIO=200

When formatting the run control file ABLDAT and the journal file JRNDAT, you should select and format the maximum file size necessary for medium-term requirements. This will avoid having to extend the file during online operation.

When a file is extended, the new blocks have to be read, formatted, written and then added to the free block chain. The file is locked until this has been done.

Organizational measures

The AVAS nets and jobs are stored in PLAM libraries until they are released for production by SUBMIT-NET. When processing nets and jobs, each element in the library is locked while it is being processed.

Journal records for a net are first output by the CREATE-PLAN-NET statement when the planned net is entered in the production plan NPRLIB. All subsequent functions that are going to be used when processing a net in the production plan also output journal records and modify the status of the net in the net directory of the journal file.

The statements SUBMIT-NET and REPEAT-NET output nets to the run control file ABLDAT. All subsequent statements that control net processing change the status in the net directory of the run control file **and** the journal file and output at least two journal records as well.

To avoid critical situations, the following guidelines should be observed when processing statements through AVAS-BATCH:

- Optimum run times are attained when only one statement is ever active in just one run (complete serialization of planning through AVAS-BATCH). An element can thus never be locked in any case.

Complete serialization of all runs through the use of batch statements is achieved by specifying MAX-BATCH-PROCESS=1 when generating the system.

- With regard to the libraries, comparatively good results can be achieved by using the NET-NAME operands to isolate the nets during parallel processing (run a: NET-NAME=A1*, run b: NET-NAME=A2*).

Optimum isolation is achieved when two different user groups are allocated different libraries NETLIB (and NPRLIB). Contending access when outputting journal records cannot, however, be prevented in this case.

- High system loads and run times follow when several batch runs want to access the same elements.

To avoid this, observe the following:

- When planning with the aid of the calendar (CREATE-PLAN-NET with PERIOD-NAME), every net of a user group must always be read and scanned for symbolic start dates if no partially-qualified net name is specified.
 - If a net cannot be read using CREATE-PLAN-NET because it is locked, it will not be possible to evaluate the start dates for the net. Whether run variants for the net are to be planned or not remains open (no RESULT: NO PLAN).
 - If CREATE-PLAN-NET is handled via BATCH, an adequately qualified net name must always be specified.
- High system loads and run times also occur when a reorganization is started together with AVAS-BATCH runs. During the reorganization, the net directory of the run control file or the journal file is locked when nets are being deleted from the file. The files are locked when the released blocks are added to the chain of free blocks.

4 Backup and reorganization

This chapter describes the backup and reorganization of files.

4.1 Reorganizing the AVAS user files

Reorganizing the files is essentially a matter of cleaning up the contents of the run control file (ABLDAT) and journal file (JRNDAT). If you so wish, the contents of the production plan libraries (NPRLIB and JMDLIB) the log collection library (LOGSYS) can be reorganized.

Files which may optionally be reorganized in the reorganization run:

JMDLIB	Library of production jobs
LOGSYS	Library of log data
NPRLIB	Library of production nets

Files which are ignored in the reorganization run:

CALLIB	Calendar library
DOCLIB	Library of documentations
HSTSYS	File with net and job data, which is filled during a reorganization run.
JCLnnn	Library of jobs and JCL elements
MAPnnn	Library of mask modules
NETnnn	Library of net descriptions
PERDAT	ISAM file with the defined periods

The elements no longer needed in these files must be checked and deleted with the assigned statements (DELETE-....). The user must monitor the occupancy of the storage area and compress it if necessary.

This also applies to files for which reorganization is optional if they were ignored in the reorganization run.

Reorganization is performed by the reorganization procedure AVS.REORG. AVS.REORG is an element (J) in the SYSPRC.AVAS.085 library.

In the reorganization procedure, the password defined by the central access tasks with REO-PASSWORD must be specified via parameters ZDD-PASSWORD and ZDL-PASSWORD (see [page 112](#)). If ZDD-PASSWORD and REO-PASSWORD for ZDL or ZDD-PASSWORD and REO-PASSWORD for ZDD do not match, the signon procedure to the central access tasks is rejected.

During reorganization the processed nets are removed from the run control file and from the journal file. The journal records are saved, and also the log entries if this is required by the JOBLOG-SAVE parameter. Deletion of the processed nets and jobs in the production files is performed by the DELETE-PLAN-NET parameter and takes place during reorganization of the run control file. However, this only applies to nets which have been released via SUBMIT-NET or REPEAT-NET.

If you are working with DELETE-PLAN-NET=YES, at least one user must be assigned to each user group. If this is not the case for a user group, the function can only be executed with the help of the dialog/batch statement and if the person executing the function has * authorization.

The JOBLOG-DELETE-STATUS parameter deletes all the log entries with the specified status, if required by the parameter DELETE-JOB-LOG=YES.

When deleting nets from the production plan and the run control and journal files, the following points should be borne in mind:

- A net in the production plan which was released for production via SUBMIT-NET or REPEAT-NET cannot be deleted using DELETE-PLAN-NET until it has been deleted from the run control file during reorganization. Nets which were not released for production (no SUBMIT-NET or REPEAT-NET) can be deleted at any time using DELETE-PLAN-NET=YES.
- The prerequisite for deleting items from the run control file is the net status ENDED, SHIFTED or ABENDED.
- The journal records of a net are not saved until the net has been deleted using DELETE-PLAN-NET (net status in the journal file is DELETED).
- The journals of a net are deleted from the journal file if they were saved in the preceding reorganization run (net status in the journal file is DELETED / SAVED).

This results in the following deletion sequence for nets which have been released for production but which have reached the status ENDED, SHIFTED or ABENDED in the run control file:

1. The nets are deleted from the run control file in the subsequent reorganization run.
2. The nets must be deleted from the production plan with DELETE-PLAN-NET. The statement is issued automatically during reorganization of the run control file if the latter is started with the parameter DELETE-PLAN-NET=YES.
3. The journal records of the nets are written to the backup file during the subsequent journal backup operation.
4. The nets are deleted from the journal file during the subsequent reorganization run.

If the nets are to be deleted from the journal file immediately after the backup operation, the reorganization program can be started a second time following the backup, in which case only the function FUNKTION=JRNDAT need be specified.

4.1.1 Reorganizing the run control file and the journal file

Reorganization can take place during the current session. The program accesses the run control file and the journal file by means of the access task for run control and journal files. In addition, a backup file is created for journals which are removed from the journal file. This backup file is created under the user ID under which the reorganization program is running.

For reorganization to take place the central access task UPAM-ZD must be available and, if the reorganization statement DELETE-PLAN-NET=YES is specified, so too must the central access task PLAM-ZD.

Reorganization takes place in the following steps:

1. All nets whose journals have already been written to a backup file are removed from the journal file. This can be recognized on the basis of the net status DELETED / SAVED (reorganization statement FUNKTION=JRNDAT).
2. Journals of all nets whose status is DELETED are saved from the journal file. At the same time, their net status is set to DELETED / SAVED (reorganization statement FUNKTION=SJOUR).

If the reorganization statement HISTORY-SAVE-OPTION is specified, specific data from nets which have the status ENDED are also saved in the history file HSTSYS.

3. The data concerning nets specified with the reorganization statements HISTORY-DELETE-DATE and HISTORY-KEEP-RECORDS are removed from the history file (reorganization statement FUNKTION=HISTORY).
4. All nets whose net status is ENDED, SHIFTED or ABENDED are removed from the run control file. After the reorganization these nets are no longer available, and will not be displayed by NET-CONTROL and SHOW-NET-STATUS.
5. Condition descriptions are deleted from the run control file if:
 - the condition description has the status ENDED or ABENDED and the LIFE-TIME of the condition description has expired (COND-TYPE=NET / JOB),
 - the condition description was deleted by a DELETE-CONDITION-DESCRIPTION statement, or was logically deleted by the function FU=D (COND-TYPE=VAL / RES / NET / JOB).

For a reorganization run one or more of these stages can be requested by entering the FUNKTION parameter the appropriate number of times. These stages are then performed in the order shown above.

You can define a different order for the individual reorganization stages by specifying the functions in the following form: FUNKTION=(function1,function2,function3).

In this form of reorganization all three stages are performed. You must therefore specify the three functions in the order in which you want them to be performed.

Note

Reorganization does not reduce the size of the file but merely reorganizes its contents.

4.1.2 Reorganizing the log file

Reorganization can take place during the current session.

If the reorganization program can access the AVAS-LOGSYS library directly, the program itself will copy the logs into the specified file or library. If it cannot do so, the logs are read via the PLAM access task (ZD), and are written to the backup file or library.

Direct access by the reorganization program is possible if

- it is running under the same user ID as the PLAM ZD
- the AVAS-LOGSYS library is not password-protected, or if the password was assigned in the reorganization job before the program was started by means of the BS2000 command ADD-PASSWORD.

Reorganization is started by the AVS.REORG procedure (an element in the library SYSPRC.AVAS.085) and control information is received via parameters. The saving and deleting of log data can then take place in a single run or in separate runs. The following principles apply:

- All log entries with the status TRANSFERRED, ADDED or IGNORE are saved.
- All log entries which have the SAVED status are deleted, as are all the log entries which the control parameter indicates are to be deleted.
- The DELETE-JOB-LOG parameter must be set to YES.

Note

Reorganization does not reduce the size of the log library but merely reorganizes its contents.

4.1.3 Execution of the reorganization program

The following diagram illustrates the execution of a reorganization (default sequence) and files required for the individual steps.

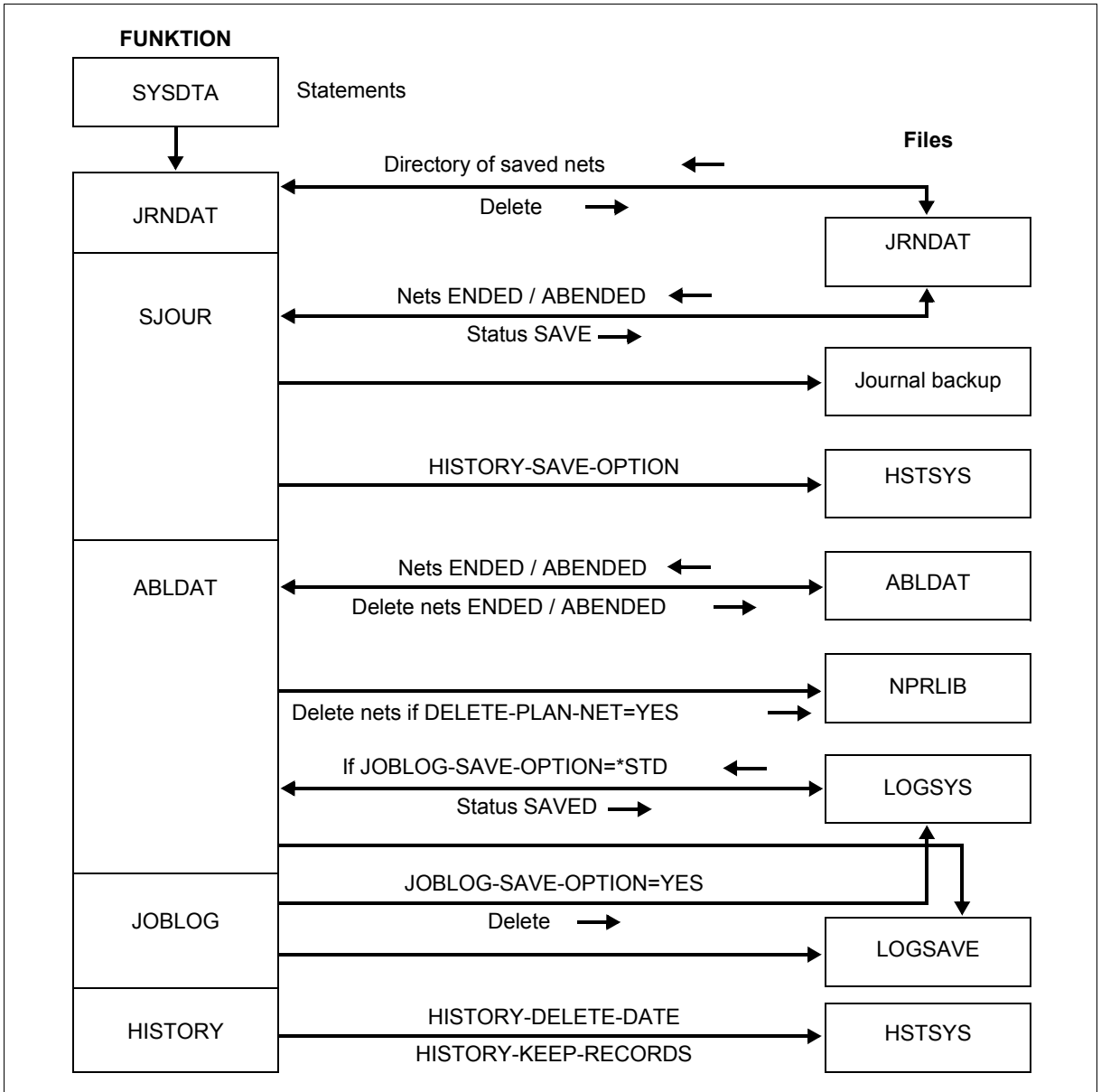


Figure 4: Execution of a reorganization

4.1.4 Statements for the reorganization program

Statements for the reorganization program are read from SYSDTA. Depending on the form of the FUNKTION= statement, additional statements are optional or mandatory.

Overview of the reorganization functions and the required reorganization statements

Reorganization statement	FUNKTION =				
	JRNDAT	SJOUR	ABLDAT	JOBLOG	HISTORY
AVAS-SYSTEM-ID	yes	yes	yes	yes	yes
AVAS-USER-ID	opt (2)	opt (2)	opt (2)	opt (2)	opt (2)
DELETE-JOB-LOG	no	no	no	opt	no
DELETE-PLAN-NET	no	no	opt	no	no
END	opt	opt	opt	opt	opt
HISTORY-DELETE-DATE	no	no	no	no	opt
HISTORY-DELETE-OPTION	no	no	no	no	opt
HISTORY-KEEP-RECORDS	no	no	no	no	opt
HISTORY-SAVE-OPTION	no	opt	no	no	no
JOBLOG-DELETE-DATE	no	no	no	opt	no
JOBLOG-DELETE-STATUS	no	no	no	opt	no
JOBLOG-SAVE-LINK	no	no	opt (1)	opt (1)	no
JOBLOG-SAVE-OPTION	no	no	opt	opt	no
NET-NAME	opt	opt	opt	opt	opt
OUTPUT-FILE	no	opt	no	no	no
OUTPUT-LINK-NAME	no	opt	no	no	no
PERIOD-NAME	no	no	yes	no	no
START	opt	opt	opt	opt	opt
UPDATE-JRIDAT	ww	ww	ww	ww	ww
USER-PASSWORD	opt (2)	opt (2)	opt (2)	opt (2)	opt (2)
ZDD-PASSWORD	opt (4)	opt (4)	opt (4)	opt (4)	opt (4)
ZDL-PASSWORD	opt (4)	opt (4)	opt (3, 4)	opt (3, 4)	opt (4)

yes The statement must be specified.

no The statement is not required for the function.

opt The statement is optional. If it is not specified, a default value is assumed.

- (1) The statement is mandatory if `JOBLOG-SAVE-OPTION=YES` is specified.
- (2) The statement is mandatory if `NET-NAME={sug_netname / sug_group*}` is specified.
- (3) The statement is mandatory if `NET-NAME={*ALL / sug_*}` is specified.
- (4) The statement only has to be specified if a special password has been declared for the reorganization by `REO-PASSWORD=` for either the `PLAM-ZD` or the `UPAM-ZD`.

The reorganization statements are, in alphabetical order, as follows:

AVAS-SYSTEM-ID

The reorganization statement `AVAS-SYSTEM-ID` is used to select the AVAS system which is to be reorganized.

The statement must be specified for each function.

`AVAS-SYSTEM-ID=string`

string Seven-character name of the AVAS system which is to be reorganized. The statement is used to select the access tasks, and hence the files.

AVAS-USER-ID

This reorganization statement is used to specify the name of a user in situations where a single net (`NETNAME=sug_netname`) or a group of nets (`NET-NAME=sug_group*`) is to be reorganized, and the production plan (`NPRLIB`) is also to be processed with `DELETE-PLAN-NET=YES`. The user must be assigned to the user group `sug` for the net or net group. It is not sufficient to specify a user with `*` authorization. The user must be authorized to use the `SHOW-SYSTEM-PARAMS` statement.

If all the nets in the AVAS system are to be reorganized, this statement is not specified. The statement is only processed with `USER-PASSWORD=C'string'`.

`AVAS-USER-ID=string`

string AVAS user ID.

DELETE-JOB-LOG

This reorganization statement determines whether or not logs in the LOGSYS library are deleted when the JOBLOG function is executed.

This statement may only be specified for the JOBLOG function.

DELETE-JOB-LOG = {YES / NO}

YES All the log records which are identified by the reorganization statement JOBLOG-DELETE-DATE or JOBLOG-DELETE-STATUS are deleted. If neither of these statements is specified, all the logs with the status SAVED are deleted.

NO Log records are not deleted.

DELETE-PLAN-NET

This reorganization statement specifies whether nets which have been planned in the production plan are to be deleted when they are removed from the run control file.

DELETE-PLAN-NET={YES / NO}

YES When the reorganization deletes planned nets from the run control file, they are also removed from the production plan if it is possible to determine their access path.

Every user group whose nets are reorganized must have at least one user assigned to it.

NO Planned nets are not removed from the production plan (NPRLIB).

The access path is determined from the user specified by AVAS-USER-ID and the entries in the system parameters. The specified user must be authorized to use the SHOW-SYSTEM-PARAMS statement. If the user does not have this authorization, the reorganization is aborted with an error message. If it is impossible to determine an access path, the nets will remain in the production plan. No message about this will be output.

END

This reorganization statement terminates the input of statements. It is optional. If it is not specified, the reorganization identifies the end of the input from end-of-file on SYSDTA.

FUNKTION

This reorganization statement is used to inform the reorganization program which files in the AVAS system are to be reorganized. This is a mandatory statement.

The statement has two formats:

FUNKTION=function1 *or*

FUNKTION=(function1,function2,function3[,function4][,function5])

function1 – function5

One of the functions JRNDAT, SJOUR, ABLDAT, JOBLOG or HISTORY.

Specification of the JOBLOG and HISTORY functions is optional. The functions must be enclosed in parentheses. When specifying the list, at least the three functions JRNDAT, SJOUR and ABLDAT must be included, in any sequence. This sequence has an effect on the lifetime of the journals and the logs in the AVAS system (see below).

Each of the reorganization functions may be specified once only.

If only a single function is specified, this may be any one of the four.

The default sequence for the functions is JRNDAT, SJOUR, ABLDAT, JOBLOG.

JRNDAT: Journals which have been saved (net status *SAVED*) are removed from the journal file.

SJOUR: Journals from nets with the status *ENDED* or *ABENDED* are saved. The net status is set to *SAVED*. If the reorganization status *HISTORY-SAVE-OPTION=*STD / NET* is specified, the data from nets whose status is *ENDED* is saved in the history file.

ABLDAT: The function searches for all the nets with the status *ENDED* or *ABENDED* in the run control file.

The reorganization statement *JOBLOG-SAVE-OPTION=*STD* saves all the logs for these nets and identifies them with the status *SAVED*.

The reorganization statement *DELETE-PLAN-NET=YES* deletes the nets from the production plan. In *JRNDAT* the nets are assigned the *DELETED* status. After this, the nets are deleted from the run control file.

Nets whose signaled log files have the *ASSIGNED* or *ERROR* status are not reorganized.

JOBLOG: All logs with the status *TRANSFERRED*, *ADDED* or *IGNORE* are saved and their status set to *SAVED*. If *JOB-LOG-DELETE-STATUS* is set, then all the logs with the prescribed status are deleted.

HISTORY The data records in the history file are deleted (TYPE NET and JOB) or updated (TYPE SUM). The reorganization statements NET-NAME, HISTORY-DELETE-OPTION, HISTORY-DELETE-DATE and HISTORY-KEEP-RECORDS are considered.

The records to be processed are selected with the HISTORY-DELETE-OPTION reorganization statement.

Data records with KEY 03 and 04 are deleted using the HISTORY-DELETE-DATE and/or HISTORY-KEEP-RECORDS reorganization statements in conjunction with NET-NAME.

Compressed data records with KEY 01 and 02 are deleted if no further run is specified (Number=0).

If the reorganization statement HISTORY-DELETE-DATE is specified, only records with PLAN-START are selected and only the records with a PLAN-START value before the specified date are deleted.

With HISTORY-KEEP-RECORDS, you specify how many records are to be retained in the history file.

HISTORY-DELETE-DATE

This reorganization statement determines that the net's history records that were scheduled up to a specific date or within a specific term are to be deleted from the history file. If this affects nets contained in the compressed data records, the net data is also deleted from the compressed records and the mean values, the range and the standard difference are recalculated.

The parameter HISTORY-DELETE-OPTION is taken into consideration during deletion if *STD, NET or JOB are specified.

HISTORY-DELETE-DATE={period / dd.mm.yy / (dd.mm.yy,dd.mm.yy)}

period Symbolic name of a period
The NET and JOB data records of nets whose PLAN-START is within the range of the period are deleted from the history file. The SUM data records are updated accordingly. The parameters NET-NAME and HISTORY-DELETE-OPTION are considered in deleting and updating.

dd.mm.yy The NET and JOB data records of nets whose PLAN-START is before the specified data are deleted from the history file. The SUM data records are updated accordingly. The parameters NET-NAME and HISTORY-DELETE-OPTION are considered in deleting and updating.

dd.mm.yy,dd.mm.yy

The NET and JOB data records of nets whose PLAN-START is within the specified term are deleted from the history file. The SUM data records are updated accordingly. The parameters NET-NAME and HISTORY-DELETE-OPTION are considered in deleting and updating.

HISTORY-DELETE-OPTION

This reorganization statement is used to specify which data records are to be deleted from the history file.

HISTORY-DELETE-OPTION={*STD / JOB}

***STD** The NET and JOB data records are deleted from the history file. The SUM data records are updated accordingly.

The parameters NET-NAME, HISTORY-DELETE-DATE and HISTORY-KEEP-RECORDS are considered in deleting and updating.

If only the NET data records were saved (HISTORY-SAVE-OPTION=NET), only the NET data records are deleted in the history file. The SUM data records are updated accordingly.

JOB The JOB data records are deleted in the history file, the SUM data records are updated accordingly. The parameters NET-NAME, HISTORY-DELETE-DATE and HISTORY-KEEP-RECORDS are considered in deleting and updating. The related NET data records are not deleted. It is thus possible to retain NET data records longer than JOB data records or to retain a larger number of NET data records.

HISTORY-KEEP-RECORDS

This reorganization statement specifies how many NET and JOB data records are saved in the history file. If there are more net or job records than the specified number, they are to be deleted. The NET records with the oldest PLAN-START dates are deleted.

HISTORY-KEEP-RECORDS=nnn (nnn=012 ... 999)

nnn The NET and JOB data records are deleted from the history file and the SUM data records are updated accordingly if there are more records for a net or a job than the number specified.
The data records of the nets with the oldest PLAN-START value are deleted. If a value larger than 64 is specified for HISTORY-KEEP-RECORDS, the entries in the compressed records are also deleted. The mean values, the range and the standard difference are recalculated on the basis of the specified number.

If the HISTORY-DELETE-DATE and HISTORY-KEEP-RECORDS parameters are both specified, the data records that comply with HISTORY-DELETE-DATE are deleted first. Next those complying with HISTORY-KEEP-RECORDS are deleted.

The compressed records of nets and jobs below the nets if they do not contain any further runs (Number=0).

Notes

- If different numbers of JOB and NET data records are to be saved, the following parameter combinations should be used:
HISTORY-SAVE-OPTION=*STD with
HISTORY-DELETE-OPTION=JOB and HISTORY-DELETE-DATE=WOCHEV
or HISTORY-KEEP-RECORDS=64 and
HISTORY-DELETE-OPTION=*STD and HISTORY-DELETE-DATE=MONATV
oder HISTORY-KEEP-RECORDS=200.
The symbolic periods and quantities are examples.
- If only the NET data records are to be saved, the parameters HISTORY-SAVE-OPTION=NET and HISTORY-DELETE-OPTION=*STD should be used.

HISTORY-SAVE-OPTION

This reorganization statement specifies which data records are to be saved in the history file with the SJOUR function. Only the data records of net that have been backed up are saved.

HISTORY-SAVE-OPTION={NO / *STD / NET}

- NO No data records are saved in the history file.
- *STD The NET and JOB data records are saved in the history file. The SUM data records are updated accordingly.
- NET The NET (KEY=03) data records are saved in the history file. The relevant SUM (KEY=01) data records are updated accordingly. The JOB (KEY=03) data records are not saved. The relevant SUM (KEY=02) data records are not updated.

It is recommended you use the same value allocations in all reorganization runs, as the history data obtained will otherwise not be particularly informative.

The SUM data records are required for operation #52 (HISTORY) in the NET-CONTROL statement.

JOBLOG-DELETE-DATE

This reorganization statement specifies that all the logs which were created by a particular day or within the specified time period are to be deleted from the LOGSYS library.

The statement may only be specified if DELETE-JOB-LOG=YES is also specified.

JOBLOG-DELETE-DATE={period / dd.mm.yy / (dd.mm.yy,dd.mm.yy)}

- period Symbolic name of a period
- dd.mm.yy Specifies the date of the log which is to be deleted.
If a second date is specified in addition, all the logs which were generated between the first and second dates inclusive are deleted.

JOBLOG-DELETE-STATUS

This reorganization statement specifies the status of the logs which are to be deleted during the reorganization of the LOGSYS library.

The statement may only be specified in conjunction with the reorganization statements FUNKTION=JOBLOG and DELETE-JOB-LOG=YES.

JOBLOG-DELETE-STATUS={*STD / *ALL / status / (status,status,...)}

*STD Log records which have the status SAVED are deleted.

*ALL All log records for a net are deleted
(only in conjunction with NETNAME=\$ug_name).

status All log records with the specified status are deleted.

(status,status,...)

All the log records with the same status as any in the list in parentheses are deleted.

JOBLOG-SAVE-OPTION

This reorganization statement specifies whether or not the items of log data are saved.

JOBLOG-SAVE-OPTION={*STD / YES / NO}

*STD Those items of log data for the net which have reached the status ENDED or ABENDED in the run control file are saved. The save operation will be carried out as part of a reorganization of the run control file.
The assignment of the value *STD is only processed if the functions ABLDAT and JOBLOG are specified.
Nets whose signaled log files have the ASSIGNED or ERROR status are not reorganized.

YES All the log data with the status TRANSFERRED, ADDED or IGNORE is saved. This is the default value assigned when the function FUNKTION=JOBLOG was specified.
This assignment is not permitted for other functions.
After they have been saved the log files are placed in the SAVED status.

NO No logs are saved. This assignment is only permissible in conjunction with FUNKTION=JOBLOG.

JOBLOG-SAVE-LINK

This reorganization statement defines where the logs are to be saved to.

JOBLOG-SAVE-LINK={linkname / LIB(linkname)}

linkname File link name of a SAM file to which the logs are to be saved.

LIB(linkname)

File link name of a PLAM library to which the logs are to be saved. The library must be directly accessible from within the reorganization program (see [page 172](#)). It can be read into an (audit) AVAS system by means of the SHOW-JOB-LOG statement if it is recorded there as the LOGSYS library.

NET-NAME

This reorganization statement identifies the nets to be reorganized.

NET-NAME={*ALL / \$ug_netname / \$ug_* / \$u*}

*ALL All the nets are considered during the reorganization.

\$ug_netname

The specified net or, if the net name is abbreviated, the specified net group is reorganized. The user group must be fully specified. The net name may be abbreviated.

\$ug_* All the nets of the specified user group are reorganized.

\$u* All the nets of the user groups which begin with the specified character are reorganized.

OUTPUT-FILE

This reorganization statement defines the prefix for the name of the file in which the journal records are to be saved. This statement is only processed in conjunction with FUNKTION=SJOUR.

OUTPUT-FILE=name

name The first 22 characters of “name” are extended by adding the date and time (format `yyyymmdd.hhmmss`), and are used as the file name for saving the journal.

If the name is longer than 22 characters it is truncated to the maximum length of 22 characters without any notification.

OUTPUT-LINK

This reorganization statement specifies the output file for the journal save by a file link name. The statement is only processed in conjunction with `FUNKTION=SJOUR`, but not if `OUTPUT-FILE=name` is specified.

`OUTPUT-LINK=name`

`name` File link name of the output file for the journal save.
The BS2000 command `/ADD-FILE-LINK` must be used to assign the save file before the reorganization program is started.

Note

The use of a link name to assign a file is preferable, especially for large save files, because the user can adjust the space allocation to the size of the file (in the case of `OUTPUT-FILE=name`, the file is always created with `SPACE=RELA(PRIM-ALLOC=30, SEC-ALLOC=15)`).

PERIOD-NAME

As well as `NET-NAME`, the reorganization statement determines the nets that are to be removed from the run control file.

`PERIOD-NAME={period / dd.mm.yy / (dd.mm.yy,dd.mm.yy)}`

`period` Symbolic name of a period.
Only nets with the `ENDED` or `ABENDED` status whose `EARLIEST-START` lies within the date limits for the period are removed from the run control file.

`dd.mm.yy` Only nets with the `ENDED` or `ABENDED` status whose `EARLIEST-START` matches the specified date are removed from the run control file.

`(dd.mm.yy,dd.mm.yy)`
Only nets with the `ENDED` or `ABENDED` status whose `EARLIEST-START` lies within the specified date range (from-date,to-date) are removed from the run control file.
The first date is assigned the time of 00:00:00 and the second date the time of 23:59:59.

Notes

A time cannot be specified.

START

The reorganization statement START terminates the input sequence of statements for a reorganization run. It informs the reorganization program that statements for another reorganization run follow.

Value assignments from the preceding reorganization run are not transferred. All the statements that are needed for another reorganization run must be specified.

UPDATE-JRIDAT

The reorganization statement causes the ISAM log of the journal file (see also “[JRNDAT-ISAM-NAME=](#)” on page 110) to be closed and created anew.

UPDATE-JRIDAT={YES / NO}

- YES The ISAM log file is closed at the end of reorganization and then created anew.
 If the file has the name suffix .<yymmdd>.<hhmmss> (JRNDAT-ISAM-NAME=*STD), this suffix is updated for the new file. Otherwise (JRNDAT-ISAM-NAME=<filename>) the file retains its name and the closed old version is assigned the name suffix .<yymmdd>.<hhmmss> (the name being abbreviated if appropriate).
- NO The ISAM log file is not reassigned when reorganization takes place.

USER-PASSWORD

This reorganization statement passes the AVAS password for the user specified by AVAS-USER-ID.

USER-PASSWORD=C'.....'

- C'.....' The password is to be entered in 8-character form. The statement is only processed in conjunction with the reorganization statement AVAS-USER-ID.

ZDD-PASSWORD

This reorganization statement passes the password to be used when the reorganization signs on to the UPAM-ZD. The password must be entered in 4-byte form.

The reorganization will only succeed in signing on to the UPAM-ZD if this password matches that specified in the REO-PASSWORD parameter when the UPAM-ZD was started.

ZDD-PASSWORD={*STD / C'....' / X'.....'}

*STD The internally defined password is to be used for signing on to the UPAM-ZD.

C'....' Password entry in alphanumeric form (4 characters).

X'.....' Password entry in hexadecimal form (8 characters).

If ZDD-PASSWORD is not specified, then *STD is assumed.

ZDL-PASSWORD

This reorganization statement passes the password to be used when the reorganization signs on to the PLAM-ZD. The password must be entered in 4-byte form.

The reorganization will only succeed in signing on to the PLAM-ZD if this password matches that specified in the REO-PASSWORD parameter when the UPAM-ZD was started.

ZDL-PASSWORD={*STD / C'....' / X'.....'}

*STD The internally defined password is to be used for signing on to the PLAM-ZD.

C'....' Password entry in alphanumeric form (4 characters).

X'.....' Password entry in hexadecimal form (8 characters).

If ZDL-PASSWORD is not specified, then *STD is assumed.

Example

In a reorganization run, it is necessary to perform the following for all the nets which have the status ENDED or ABENDED, as defined in the reorganization statement NET-NAME:

- to save and delete the logs and
- to save and delete the journals.

The logs are to be output to a library.

Required statements:

```
FUNKTION=(ABLDAT,SJOUR,JRNDAT,JOBLOG)
NET-NAME={*ALL / $ug_netname}
DELETE-PLAN-NET=YES
JOBLOG-SAVE-OPTION=*STD
JOBLOG-SAVE-LINK=LIB(linkname)
DELETE-JOB-LOG=YES
JOBLOG-DELETE-STATUS=*STD
```

If the logs and journals which are saved during a reorganization run are not to be deleted until the next reorganization run, it is only necessary to change the sequence of the operands in the reorganization statement FUNKTION:

```
FUNKTION=(JRNDAT,SJOUR,JOBLOG,ABLDAT)
```

Sample procedure for reorganizing a net which has had data removed from all reorganizable files

Example

```

/BEGIN-PROC LOGG=CMD,PAR=YES(PROC-PAR=(&SYSID=,-
/&FUNKTION='(ABLDAT,SJOUR,JRNDAT,JOBLOG)',-
/&IGNLT=YES,&DELPN=YES,&ZDDPW=*STD,&ZDLPW=*STD,-
/&AVSID=,&USRPW=,&JLDELS=*STD,&JLSVO=YES,&DELJL=YES,-
/&NETN=,&JVZDD=,-
/&SFILE=AVAS.USER.JRNDAT.SAVE),ESC-CHAR=C'&')
/REMARK *****
/REMARK &SYSID::= SYSTEM-ID
/REMARK &ZDDPW::= ZDD-PASSWORD (SEE REOPW FOR ZDD)
/REMARK &ZDLPW::= ZDL-PASSWORD (SEE REOPW FOR ZDL)
/REMARK &FUNKTION::= FUNKTION ABLDAT;SJOUR;JRNDAT;JOBLOG
/REMARK &NETN::= NETNAME ($UG_NETNAME / ALL)
/REMARK &DELPN::= EXECUTE DELETE-PLAN-NET ? YES / NO
/REMARK &AVSID::= AVAS-USERID FOR DELETE-PLAN-NET
/REMARK &USRPW::= USER-PASSWORD (SYNTAX: 'C'.....')
/REMARK &SFILE::= PREFIX FOR THE JOURNAL SAVE FILE
/REMARK &JLSVO::= EXECUTE JOBLOG-SAVE ? YES / NO
/REMARK &DELJL::= EXECUTE DELETE-JOB-LOG ? YES / NO
/REMARK *****
/WRITE-TEXT '--> FUNKTION=&FUNKTION <--'
/SKIP-COM TO-LABEL=F&FUNKTION
/.FSJOUR REMARK &JVZDD ::=JV FOR MONITORING ZDD
/MODIFY-JV JV=&JVZDD,'IN-JRLDAT'
/.F&FUNKTION REMARK
/ADD-FILE-LINK LINK-NAME=SYSLNK,FILE-NAME=SYSLNK.AVAS.085
/ASSIGN-SYSDTA TO=*SYSCMD
/START-PROG FROM-FILE=*PHASE(LIB=SYSPRG.AVAS.085.SYSTEM,-
/ELEMENT=AVAS.SYS.LOAD.REORG.AJ)
SYSID=&SYSID
ZDD-PASSWORD=&ZDDPW
ZDL-PASSWORD=&ZDLPW
FUNKTION=&FUNKTION
NET-NAME=&NETN
DELETE-PLAN-NET=&DELPN
AVAS-USER-ID=&AVSID
USER-PASSWORD=&USRPW
JOBLOG-SAVE-OPTION=&JLSVO
DELETE-JOB-LOG=&DELJL
JOBLOG-DELETE-STATUS=&JLDELS
OUTPUT-FILE=&SFILE
END
/ASSIGN-SYSDTA TO=*PRIMARY
/.ENDE END-PROC

```

4.2 Saving the journals and logs and outputting journal listings

Saving journal records

The journal records of a production period are saved during reorganization. When this is done, the journal records of all nets which have the DELETED status are output to a SAM file (see also reorganizing files).

The format of the records corresponds to the format in the emergency journal file.

The records in the SAM file are not sorted.

In order to obtain a complete listing that is sorted by net names for a given production period, all journal backups of this production period must be merged and sorted.

Example

In the example below, four journal backups and one emergency journal file are sorted.

```
/BEGIN-PROC LOGG=CMD,PAR=YES(PROC-PAR=(&SORTO),ESC-CHAR=C'&')
/REMARK *****
/REMARK &SORTO ::= SORT OUTPUT FILE
/REMARK *****
/REMARK JOURNAL BACKUPS
/ADD-FILE-LINK LINK=SORTIN01,F-NAME=AVAS.USER.SAVE.JRNDAT.20050203.081807
/ADD-FILE-LINK LINK=SORTIN02,F-NAME=AVAS.USER.SAVE.JRNDAT.20050204.083906
/ADD-FILE-LINK LINK=SORTIN03,F-NAME=AVAS.USER.SAVE.JRNDAT.20050205.081612
/ADD-FILE-LINK LINK=SORTIN04,F-NAME=AVAS.USER.SAVE.JRNDAT.20050205.082319
/REMARK EMERGENCY JOURNAL OUTPUTS
/CREATE-FILE F-NAME=&SORTO,SUP=PUB-DISK(SPACE=RELA(PRIM-ALLOC=120,-
/SECO-ALLOC=30))
/ADD-FILE-LINK LINK=SORTOUT,F-NAME=&SORTO
/ASSIGN-SYSDTA=*SYSCMD
/START-SORT
//SORT-RECORD FIELDS=*FIELD-EXPL( -
//          POS=5,LENGTH=62,SORT-ORDER=*ASCEND,FORMAT=*CHAR)
END
/ASSIGN-SYSDTA=*PRIMARY
/.ENDE END-PROG
```

Outputting journal listings

The journal records of a production period can be edited from the backup file using the procedure AVS.LSTJRNDAT (an element in the library SYSPRC.AVAS.085) in order to produce a print file.

They require the following statements, which are entered via SYSDTA:

INPUT[-FILE]=filename

Name of a backup file or name of the merged and sorted backup of a production period.

OUTPUT[-FILE]=filename

Name of the LIST file.

If OUTPUT-FILE is omitted, the following is assumed:

AVAS.LIST.JOURNAL.yymmdd.hhmmss

HEAD[ER]=text

Header for the listings.

If HEADER is omitted, the name of the LIST file is assumed.

FUNK[TION]=

ALL

“List all nets” function.

NET

“List a net or net group” function.

The name must be predefined via NET-NAME=.

If FUNKTION is omitted, ALL is assumed.

FORM[AT]=

Output the OUTPUT-AREA.

SHORT

The OUTPUT-AREA is not included in the output.

NORMAL

The OUTPUT-AREA is included in the printout, but is not broken down into its separate parts.

LONG

All data in the OUTPUT-AREA is output with designators.

If FORMAT is omitted, SHORT is assumed.

NET[-NAME]=netname

Name of a net or a net group.

Only one net name or one net group can be specified. This statement is permitted only in conjunction with FUNK=NET. The only listings created are those of nets with the specified name.

Example

```

/BEGIN-PROC LOGG=CMD,PAR=YES(PROC-PAR=( &SVFIL=, &LIST=, &HEAD=),-
/&FUNK=ALL, &FORM=LONG), ESC-CHAR=C '&')
/REMARK *****
/REMARK &SVFIL ::= JOURNAL BACKUP
/REMARK &LIST ::= NAME OF THE OUTPUT LISTING
/REMARK &HEAD ::= HEADER OF THE JOURNAL LISTING
/REMARK &FUNK ::= FUNCTION (ALL)
/REMARK &FORM ::= FORMAT (LONG)
/REMARK *****
/ADD-FILE-LINK LINK=SYSLNK, F-NAME=SYSLNK.AVAS.085
/ASSIGN-SYSDTA TO=*SYSCMD
/START-PROG FROM-FILE=*PHASE(LIB=SYSPRG.AVAS.085.SYSTEM,-
/ELEMENT=AVAS.SYS.LOAD.LIST.JRNDAT)
INPUT=&SVFIL
OUTPUT=&LIST
HEAD=&HEAD
FUNC=&FUNC
FORM=&FORM
END
/ASSIGN-SYSDTA TO==*PRIMARY
/.ENDE END-PROC

```

The LIST file created in this manner can be printed out by issuing the following command:

```

/PRINT-DOCUMENT F-NAME=list-datei...,LINE-SPACING=*BY-EBCDIC-CONTROL

```

The saved journal records can also be evaluated by users with their own programs. The structure of the saved journal records is described on [page 194](#).

The record description of the saved journal records is defined by means of the AVASJRN macro.

Saving log data

The log entries and the log data are saved during reorganization if this is required by the control parameter. All the data of the log entries to be saved is output to a SAM file.

The structure for saving log entries is as follows:

1st record: *SYSLOG*syslog-data

2nd record: *JOBLOG*joblog-data

3rd record: *SAVLOG*element-name

4th record: *function*function-data (only if present)

5th to nth record: log data

n+1 record: *SYSLOG*\$ENDE\$ syslog-data

or *SYSLOG*\$ERROR\$ syslog-data (only in the event of an error)

element-name

Name of the log in the D division (see [section “Storing the logs” on page 139](#)).

syslog-data

Data of the *SYSLOG* record in the element of the S division (see [page 140](#)).

joblog-data

Data of the *JOBLOG* record in the element of the S division (see [page 141](#)).

function-data

Data of the *function* record in the element of the D division (see [page 141](#)).

4.3 Data structures of the backup journal and the emergency journal file

Record structure

The structure of the records in the journal record backup file is described in order to enable the user to create his own analysis programs.

Records in the backup file have the same structure as those in the emergency journal file.

The AVASJRN macro is provided to interpret the contents of records.

Access method: SAM

Addressing the records:

SYSTEM-ID	(L=7)
User group	(L=5)
Net name	(L=32)
Date	(L=8)
Time of day	(L=6)
Sequence number	(L=2)

Maximum record length: 520 bytes.

Maximum data length: 408 bytes.

Format of the records:

Bytes	0 and	1	Record length (H) including record length field
Bytes	2 and	3	Reserved (X'0000')
Bytes	4 thru	43	Header for backup journal
Bytes	44 thru	111	Header for journal record
Bytes	112 thru	427	Data record

The structure of the journal records is described by means of the AVASJRN macro. Each record consists of a fixed portion (112 bytes) and a variable-length data portion (up to 316 bytes).

Fixed portion

Macro call: F3N AVASJRN &PRE,VERSION=085[,EQU=YES]

The record length field is aligned on a word boundary. The remainder is padded with blanks.

Data area

The data area is identified by means of the record key.

For those record keys which have been assigned complex data there are record descriptions which can be called via the following macro call:

Srk AVASJRN &pre

rk Record key

&pre Prefix

Note

The journal records are still transferred in the format “output date without the century” in the CC exit AVEX0001.

The fixed portion must therefore be generated with VERSION=020.

Macro call: F3N AVASJRN &PRE,VERSION=020[,EQU=YES]

Record key

The record key (RK) determines which data is output in the data area, and defines its structure and scope.

RK	Output in data area		Macro call
01	No data	---	
08	User information (#AVA#)		
09	MSG-7 error message	MSG7 text	
11	Net data (CR-PL-NET)	S11	AVASJRN
12	Structure element data (CR-PL-NET)	S12	AVASJRN
14	Net data (MOD-PL-NET)	S14	AVASJRN
15	Format name, format text	S15	AVASJRN
16	Job name, library name	S16	AVASJRN
21	USER-PARAM-FILE/system variables data	S21	AVASJRN
22	Assign mask	statement	JCLLIB
23	Call element	statement	JCLLIB
25	Modify parameter	statement	
31	Net data (SUBMIT-NET)	S31	AVASJRN
32	BS2000 job / S procedure / Subnet data (SUBMIT-NET)	S32	AVASJRN
51	Net data (ABLDAT E2 record)	S51	AVASJRN
52	Structure element data (ABLDAT E3 record)	S52	AVASJRN
53	Data on structure element with FU=C and TYPE=JVA	S53	AVASJRN
54	Restart point data	S54	AVASJRN
55	Net data (MOD-SUBM-NET)	S55	AVASJRN
56	BS200 job / S procedure parameters	S56	AVASJRN
57	Data on structure element with FU=C / A / M / D / W	S57	AVASJRN
58	Restart data (P-O-E, P-O-R)	S58	AVASJRN
59	Data on structure element with FU=S	S59	AVASJRN
60	Data on structure element with FU=F	S60	AVASJRN
65	JCL statement	statement	

Record sequence number

The record sequence number (RSN) describes the status of the output data. It describes whether the data involved is input or output data, and whether or not it has been updated.

RSN

00	Start of action or execution
01	Existing data (INPUT)
02	Created (updated) data (OUTPUT)
03	Interrupt (CONDWAIT or HOSTWAIT)
04	Deletion of data
05	Error in existing data (INPUT)
06	Error in created (updated) data (OUTPUT)
07	End of action (C, S or E)
08	Abort action (I, R or error)
09	Error message regarding record key (text)

Journal records of the statements

Below is a summary of those journal records which are output by the individual statements. Please note that, for each processing operation, only a subset of the journal records listed is actually output, and also that a journal record may be output more than once for different objects (e.g. jobs).

Notes

- Output of the journal records marked with an x can be suppressed in the CC exits AVEX0001 and AVEX0002.
- The scope of the logging in the CREATE-PROD-NET statement can be controlled by means of the PARAM-JOURNAL-OUTPUT generation parameter.
- Depending on the generation parameter, the following journal records are output:

PARAM-JOURNAL- OUTPUT=	Journal output with journal record	
	S25-	S21-
STD	25-01 25-02	no
LIST	no	21-01
ALL	25-01 *) 25-02 *)	21-01
NO	no	no

*) The user can check using CC exit AVEX0001 or AVEX0002 whether parameters from the net or job masks or from the USER-PARAM-FILE are being modified, and can suppress the logging from the USER-PARAM-FILE.

RK-RSN	Statement	Output in data area	
	CREATE-ORDER		
01-00	Start of function	---	
01-07	End of function	---	
01-08	Abort function	---	
	CREATE-PLAN-NET		
11-00	Start of planning	S11	AVASJRN
12-00	Output structure element	S12	AVASJRN x
12-04	Delete structure element	S12	AVASJRN x
11-07	End of planning	S11	AVASJRN
11-08	Abort planning	S11	AVASJRN
09-00	Error message	MSG7	text
	MODIFY-PLAN-NET		
14-01	Modification	S14	AVASJRN
14-07	Modification	S14	AVASJRN
14-08	Modification	S14	AVASJRN
09-00	Error message	MSG7	text
	DELETE-PLAN-NET		
01-00	Start of deletion	---	
16-04	Delete job	S16	AVASJRN
16-05	Error during deletion	S16	AVASJRN
01-07	End of function	---	
01-08	Abort function	---	
09-00	Error message	MSG7	text
	COLLECT-NET-PAR		
01-00	Start of function	---	
15-00	Execute mask	S15	AVASJRN
15-05	Error mask	S15	AVASJRN
01-07	End of function	---	
01-08	Abort function	---	
09-00	Error message	MSG7	text

RK-RSN	Statement	Output in data area		
	CREATE--PROD-NET			
01-00	Start net modification	---		
21-01	USER-PAR-FILE data	S21	AVASJRN	x
21-01	System variable	S21	AVASJRN	x
21-05	USER-PAR-FILE error	S21	AVASJRN	
16-00	Start modification FU=J/P	S16	AVASJRN	
21-01	USER-PAR-FILE data	S21	AVASJRN	x
21-05	USER-PAR-FILE error	S21	AVASJRN	
22-00	Assign mask	statement	JCLLIB	x
23-00	Call JCL element	statement	JCLLIB	x
23-00	Call external element	statement	JCLLIB	x
23-01	Data for external element	statement	element	x
25-01	Modify parameter	statement	JCLLIB	x
25-02	Modify parameter	statement	JCLLIB	x
25-04	Empty record	---		
16-07	End modification FU=J/P	S16	AVASJRN	
16-08	Abort modification FU=J/P	S16	AVASJRN	
01-07	End of function	---		
01-08	Abort function	---		
09-08	Error message	MSG7	text	
	EDIT--PROD-JOB			
16-00	Start of function	S16	AVASJRN	
65-01	JCL statement (old)	statement	JMDLIB	x
65-02	JCL statement (new)	statement	JMDLIB	x
65-04	JCL statement (deleted)	statement	JMDLIB	x
16-07	End of function	S16	AVASJRN	
16-08	Abort function	S16	AVASJRN	
09-00	Error message	MSG7	text	
	MODIFY--PROD-NET			
01-00	Start of deletion	---		
16-04	Delete job	S16	AVASJRN	
16-05	Error during deletion	S16	AVASJRN	
01-07	End of function	---		
01-08	Abort function	---		
09-00	Error message	MSG7	text	
	DELETE--PROD-NET			
01-00	Start of deletion	---		
16-00	Delete job	S16	AVASJRN	
16-05	Error during deletion	S16	AVASJRN	
01-07	End of function	---		
01-08	Abort function	---		
09-00	Error message	MSG7	text	

RK-RSN	Statement	Output in data area	
	SUBMIT-NET		
31-00	Start net release	S31	AVASJRN
32-00	Output structure element with FU=J/P	S32	AVASJRN x
32-04	Delete structure element with FU=J/P	S32	AVASJRN x
32-05	Output structure element with FU=J/P (error)	S32	AVASJRN x
25-01	JCL statement (old)	statement	JMDLIB x
25-02	JCL statement (new)	statement	EX7102 x
25-04	JCL statement (delete)	statement	JMDLIB x
25-05	JCL statement (error)	statement	EX7102 x
25-09	JCL statement (EXIT error)	statement	EX7102 x
54-00	Output structure element with FU=A/C/D/M/W/F	S54	AVASJRN x
54-04	Delete structure element with FU=A/C/D/M/W/F	S54	AVASJRN x
31-02	Switch RUN-CONTROL-SYSTEM	S31	AVASJRN
31-07	End net release	S31	AVASJRN
31-08	Abort function	S31	AVASJRN
09-05	Error message	MSG7	text
09-06	Error message	MSG7	text
	REPEAT-NET		
31-00	Start net release	S31	AVASJRN
32-00	Output structure element with FU=J/P	S32	AVASJRN x
32-05	Output structure element with FU=J/P Error	S32	AVASJRN x
25-01	JCL statement (old)	statement	JMDLIB x
25-02	JCL statement (new)	statement	EX7102 x
25-04	JCL statement (delete)	statement	JMDLIB x
25-05	JCL statement (error)	statement	EX7102 x
25-09	JCL statement (ERROR-EXIT)	statement	EX7102 x
54-00	Output structure element with FU=A/C/D/M/W/F	S54	AVASJRN x
54-04	Delete structure element with FU=A/C/D/M/W/F	S54	AVASJRN x
31-02	Switch RUN-CONTROL-SYSTEM	S31	AVASJRN
31-07	End net release	S31	AVASJRN
31-08	Abort function	S31	AVASJRN
09-05	Error message	MSG7	text
09-06	Error message	MSG7	text

RK-RSN	Statement	Output in data area	
	NET-CONTROL		
52-04	CANCEL FU=J/P/F with #73	S52	AVASJRN
	CANCEL-NET		
51-00	Net status before function	S51	AVASJRN
51-07	Net status after function	S51	AVASJRN
51-08	Abort function	S51	AVASJRN
09-06	Error message	MSG7	text
	HOLD-NET		
51-00	Net status before function	S51	AVASJRN
52-02	Structure element status after function	S52	AVASJRN
52-09	Structure element/structure element status incorrect	S52	AVASJRN
51-07	Net status after function	S51	AVASJRN
51-08	Abort function	S51	AVASJRN
09-06	Error message	MSG7	text
	RESUME-NET		
51-00	Net status before function	S51	AVASJRN
52-02	Structure element status after function	S52	AVASJRN
52-09	Structure element/structure element status incorrect	S52	AVASJRN
51-07	Net status after function	S51	AVASJRN
51-08	Abort function	S51	AVASJRN
09-06	Error message	MSG7	text
	START-NET		
51-00	Net status before function	S51	AVASJRN
51-07	Net status after function	S51	AVASJRN
51-08	Abort function	S51	AVASJRN
09-06	Error message	MSG7	text
	RESTART-NET		
51-00	Net status before function	S51	AVASJRN
52-02	Structure element status after function	S52	AVASJRN
52-09	Structure element/structure element status incorrect	S52	AVASJRN
58-02	Restart point, restart job name	S58	AVASJRN
51-07	Net status after function	S51	AVASJRN
51-08	Abort function	S51	AVASJRN
09-06	Error message	MSG7	text

RK-RSN	Statement	Output in data area	
	MODIFY-SUBMIT-NET		
55-00	Start modification	S55	AVASJRN
55-02	Switch RUN-CONTROL-SYSTEM	S55	AVASJRN
54-01	Restart variants (old)	S54	AVASJRN
54-02	Restart variants (new)	S54	AVASJRN
54-04	Delete structure element	S54	AVASJRN
56-01	Parameter FU=J/P (old)	S56	AVASJRN
56-02	Parameter FU=J/P (new)	S56	AVASJRN
53-01	Parameter FU=C TYPE=JVA (old)	S53	AVASJRN
53-02	Parameter FU=C TYPE=JVA (new)	S53	AVASJRN
57-01	Parameter FU=C/A/M/D/W (old)	S57	AVASJRN
57-02	Parameter FU=C/A/M/D/W (new)	S57	AVASJRN
60-01	Parameter FU=F (old)	S60	AVASJRN
60-02	Parameter FU=F (new)	S60	AVASJRN
55-07	End of function	S55	AVASJRN
55-08	Abort function	S55	AVASJRN
09-00	Error message	MSG7 txt	
	MODIFY-SUBMIT-JOB		
52-00	Start modification of BS2000 job/S procedure	S52	AVASJRN
65-01	JCL statement (old)	statement ABLDAT	x
65-02	JCL statement (new)	statement ABLDAT	x
65-04	JCL statement (delete)	statement ABLDAT	x
52-07	End of function	S52	AVASJRN
52-08	Abort function	S52	AVASJRN
09-00	Error message	MSG7 text	
	AVAK run control		
51-00	Start of a net	S51	AVASJRN
51-02	End of a net while the run control system was not active	S51	AQVASJRN
51-03	Start after HOLD,CONDWAIT or HOSTWAIT	S51	AVASJRN
51-03	Restart a net	S51	AVASJRN
52-00	Start U=J/P/F	S52	AVASJRN
52-04	CANCEL FU=J/P/F because of CANCEL-NET KILL-JOBS=YES	S52	AVASJRN
52-07	Normal termination of FU=J/P/F	S52	AVASJRN
52-08	Abnormal termination of FU=J/P/F	S52	AVASJRN
59-00	Start of FU=S	S59	AVASJRN
59-07	Normal termination of FU=S	S59	AVASJRN
59-08	Abnormal termination of FU=S	S59	AVASJRN

57-03	CONDWAIT due to FU=C/W	S57	AVASJRN
57-03	HOSTWAIT due to FU=J/P	S57	AVASJRN
57-07	FU=C/W satisfied	S57	AVASJRN
57-07	FU=A/D/M executed	S57	AVASJRN
57-08	Error in FU=C/A/D/M/W	S57	AVASJRN
53-03	CONDWAIT due to FU=C TYPE=JVA	S53	AVASJRN
53-07	FU=C TYPE=JVA satisfied	S53	AVASJRN
53-08	Error after FU=C TYPE=JVA	S53	AVASJRN
54-07	DELAY-SOLUTION=IGNORE	S54	AVASJRN
	FU=C/J/P/S/F		
54-07	DELAY-SOLUTION=START FU=C	S54	AVASJRN
54-08	DELAY-SOLUTION=CANCEL	S54	AVASJRN
	FU=C/J/P/S/F		
56-02	ENTER parameter after	S56	AVASJRN
	AVEX0401 or when corrections		
	are made using		
	CHANGE-NET-DESCRIPTION		
51-07	End of a net	S51	AVASJRN
51-08	Abort net	S51	AVASJRN
09-00	Error message	MSG7 text	
08-01	User information from #AVA#\$J/\$M text		
08-02	User information from #AVA#\$H text		

Record structure of the fixed portion of journal records

```

F3N      AVASJRN N3,VERSION=080
*****
*        RECORD DESCRIPTION  BACKUP JOURNAL / EMERGENCY JOURNAL FILE  *
*****
N3JRNS  DS    OF      - RECORD DESCR. JOURNAL RECORDS
N3SNLN  DS    H       - RECORD LENGTH FIELD
          DS    H       - RESERVED
N3SYSID DS    CL7     - SYSTEM ID
N3UGR   DS    CL5     - USER GROUP
N3NNAM  DS    CL32    - NET NAME
N3NEND  DS    OF
N3HNLN  EQU   *-N3SYSID - LENGTH OF HEADER
          ORG   N3NEND-4
F3S      AVASJRN N3,VERSION=080,EQU=YES
*****
*        RECORD DESCRIPTION  JOURNAL RECORDS F3      *
*****
N3JOURN DS    OF      - RECORD DESCR. JOURNAL RECORDS
N3SLEN  DS    H       - RECORD LENGTH FIELD
          DS    OH     - RESERVED
N3F4SK  DS    X       - X'F4' WITH CENTURY FORMAT

```

N3SSTA	DS	X	- RECORD STATUS	
N3RF4S	DS	CL2	- BLANK IF CENTURY FORMAT USED	
N3DTZN	DS	OCL14	- DATE+TIME	CTYYMDDHHMMSS
N3DATN	DS	OCL8	- NEW DATE	CTYYMMDD
N3DTJH	DS	CL2	- CENTURY	CT
N3DATUM	DS	CL6	- DATE	YYMMDD
N3UHRZ	DS	CL6	- TIME OF DAY	HHMMSS
N3LNUM	DS	XL2	- CURRENT NUMBER (BY TIME)	
N3FUNK	DS	X	- AVAS FUNCTION (CMD)	
FUNK	AVASEQU N3,VERSION=080			
N3ERSC	EQU	04	.	RUN-CONTROL
N3EERJ	EQU	57	.	EDIT-PROD-JOB
N3ECLN	EQU	61	.	CREATE-PLAN-NET
N3EMLN	EQU	62	.	MODIFY-PLAN-NET
N3EDLN	EQU	63	.	DELETE-PLAN-NET
N3ECNP	EQU	66	.	COLLECT-NET-PARAMS
N3ECRN	EQU	68	.	CREATE-PROD-NET
N3EMRN	EQU	69	.	MODIFY-PROD-NET
N3EDRN	EQU	70	.	DELETE-PROD-NET
N3ESMN	EQU	71	.	SUBMIT-NET
N3EMSN	EQU	72	.	MODIFY-SUBMIT-NET
N3ECNN	EQU	73	.	CANCEL-NET
N3EHDN	EQU	75	.	HOLD-NET
N3ERMN	EQU	76	.	RESUME-NET
N3ERSN	EQU	77	.	RESTART-NET
N3EMSJ	EQU	78	.	MODIFY-SUBMIT-JOB
N3ESTN	EQU	79	.	START-NET
N3ERPNI	EQU	80	.	REPEAT-NET
*				
N3AKTN	DS	X	- ACTION PERFORMED	-CMD(EQU=YES)
AKTN	AVASEQU N3,VERSION=080			
N3ETCR	EQU	9	.	TOCREATE
N3EPTY	EQU	10	.	PARTIALLY
N3ECRD	EQU	11	.	CREATED
N3ENTC	EQU	12	.	NOTTOCREATE
N3ESMD	EQU	14	.	SUBMITTED
N3EAED	EQU	20	.	ABENDED
N3EEDD	EQU	21	.	ENDED
N3EERR	EQU	22	.	ERROR
N3EHOL	EQU	23	.	HOLD
N3ERUN	EQU	24	.	RUNNING
N3EWTG	EQU	25	.	WAITING
N3ECNW	EQU	26	.	CONDWAIT
N3ERSD	EQU	27	.	RESTARTED
N3ERMD	EQU	28	.	RESUMED
N3ERSM	EQU	29	.	RESUME
N3EIGN	EQU	63	.	IGNORED
N3ECAN	EQU	65	.	CANCEL

```

N3EEXD EQU 66 . EXECUTED
N3ESTR EQU 68 . START
N3ERST EQU 73 . RESTART
N3EPLD EQU 123 . PLANNED
N3ENPL EQU 124 . NO-PLAN
N3EDLD EQU 125 . DELETED
N3ENDL EQU 126 . NO DELETE
N3ESVD EQU 127 . SAVED
N3EUPD EQU 129 . UPDATED
N3ENUP EQU 130 . NO UPDATE
N3ECND EQU 131 . CANCELLED
N3ENCN EQU 132 . NO CANCEL
N3ENSB EQU 133 . NO SUBMIT
N3ENCH EQU 136 . NO CHANGE
N3ECHD EQU 137 . CHANGED
N3ESH D EQU 154 . SHIFTED
N3EOCD EQU 155 . OCCURED
N3ENOC EQU 156 . NO-OCCURE
N3EHTW EQU 173 . HOSTWAIT
*
N3USER DS CL8 - USER <AVUSER> -003101
N3IND DS CL3 - INDEX -CMD
*
N3JCNK DS C - FU (A,C,D,F,J,M,N,P,W) -CMD(EQU=YES)
JCNK AVASEQU N3,VERSION=080
*
FUNCTION
N3EFUA EQU C'A' . ADD
N3EFUC EQU C'C' . COMPARE
N3EFUD EQU C'D' . DELETE
N3EFUJ EQU C'J' . START-JOB
N3EFUM EQU C'M' . MODIFY
N3EFUN EQU C'N' . NET
N3EFUP EQU C'P' . START-PROCEDURE
N3EFUS EQU C'S' . START
N3EFUW EQU C'W' . WAIT
N3EFUX EQU C'X' . SERVER-JOB
*
N3NAME DS CL32 - NAME NET/JOB/CONDITION -CMD
N3DAT DS OCL412 - DATA RECORD MAX-L=320 -CMD
N3DSSL DS CL2 - DATA RECORD KEY -CMD
N3DFNR DS CL2 - DATA RECORD SEQ. NUMBER -CMD
N3FLEN EQU *-N3JOURN - LENGTH OF FIXED PART
N3INH DS CL408 - DATA REC. CONTENTS MAX-L=408 -CMD(003101)
N3LEN EQU *-N3JOURN - MAX LENGTH OF ENTRY

```

Record structure of the variable-length portion of journal records

Note that definitions for the variable-length portion must be superimposed on the INH field (N3INH in this example).

All data fields defined with TEXT and CODE already contain the converted texts.

Definitions: record key 11

```

          S11      AVASJRN S11
*****
*          RECORD DESCRIPTION  JOURNAL RECORD  RECORD KEY = 11      *
*****
S11S11A  DS      OC          - RECORD DESCRIPTION  REC 11
S11SYMD  DS      CL20       - SYMDAT
S11STUR  DS      CL1        - SELECT-TURNUS
S11EARL  DS      CL12       - EARLIEST START
S11LATS  DS      CL7        - LATEST START
S11RUCS  DS      CL8        - RUN-CONT-SYS
S11NXTX  DS      CL120     - NET-TEXT
S11NDSL  DS      OCL6       - NET-DELAY-SOLUTION - TEXT
S11NDSL  DS      X          - NET-DELAY-SOLUTION - CODE
          DS      CL5
S11LIFT  DS      CL7        - NET-LIFE-TIME (COND-NET)
S11SPLT  DS      CL4        - SELECT-PLAN-TYPE(NWRK,WORK)

S11CALN  DS      CL20       - CALENDAR-NAME
S11BHYP  DS      CL1        - *BY-HYPERNET - Y OR EMPTY
S11S11L  EQU     *-S11S11A  - LENGTH OF DATA

```

Definitions: record key 12

```

          ORG      N3INH
S12      AVASJRN S12
*****
*          RECORD DESCRIPTION  JOURNAL RECORD  RECORD KEY = 12 --- V2.0 *
*****
*
S12S12A  DS      OC          - RECORD DESCRIPTION  REC 12
S12TYPT  DS      OCL3       - TYPE-TEXT (STD,MOD,NET,JOB,RES,VAL,EXT,JVA)
S12TYP   DS      X          - TYPE-CODE
          DS      CL2
          DS      C          - RESERVED
S12SYMD  DS      CL20       - SYMDAT
          DS      C          - RESERVED
S12LATS  DS      CL7        - LATEST-START/-OCCURE, OCCURE-TIME
S12DSL  DS      OCL6       - DELAY-SOLUTION - TEXT
S12DSL  DS      X          - DELAY-SOLUTION - CODE

```

	DS	CL5	
S12LIFT	DS	CL7	- LIFE-TIME
S12JSYN	DS	CL3	- SYNC-INDEX
S12JRSP	DS	3CL48	- RESTART POINTS
	ORG	S12JRSP	
S12RSV1	DS	OCL48	- RESTART VARIANT 1
S12RSI1	DS	CL3	- RESTART INDEX V1
S12RSJ1	DS	CL30	- RESTART JOB V1
	DS	CL2	- RESERVED
S12RST1T	DS	OCL10	- RESTART TYPE V1 - TEXT
S12RST1	DS	X	- RESTART TYPE V1 - CODE
	DS	CL9	
S12RSA1T	DS	CL3	- AUTOMATIC YES /NO - TEXT
S12RSV2	DS	OCL48	- RESTART VARIANT 2
S12RSI2	DS	CL3	- RESTART INDEX V2
S12RSJ2	DS	CL30	- RESTART JOB V2
	DS	CL2	- RESERVED
S12RST2T	DS	OCL10	- RESTART TYPE V2 - TEXT
S12RST2	DS	X	- RESTART TYPE V2 - CODE
	DS	CL9	
S12RSA2T	DS	CL3	- AUTOMATIC YES /NO - TEXT
S12RSV3	DS	OCL48	- RESTART VARIANT 3
S12RSI3	DS	CL3	- RESTART INDEX V3
S12RSJ3	DS	CL30	- RESTART JOB V3
	DS	CL2	- RESERVED
S12RST3T	DS	OCL10	- RESTART TYPE V3 - TEXT
S12RST3	DS	X	- RESTART TYPE V3 - CODE
	DS	CL9	
S12RSA3T	DS	CL3	- AUTOMATIC YES /NO - TEXT
	DS	CL3	- RESERVED
S12SELRV	DS	C	- SELECT-RESTART-VARIANT
S12S12L	EQU	*-S12S12A	- LENGTH OF THE DATA
S12LEN	EQU	*-N3JRNS	

Definitions: record key 14

```

          ORG   N3INH
S14      AVASJRN S14
*****
*          RECORD DESCRIPTION  JOURNAL RECORD   RECORD KEY = 14          *
*****
*
S14S14A DS    OC      - RECORD DESCRIPTION   REC 14
S14EARL DS    CL12   - EARLIEST START
S14LATS DS    CL7    - LATEST START
S14LIFT DS    CL7    - LIFE-TIME
S14RUCS DS    CL8    - RUN-CONT-SYS
S14NDSL1 DS   OCL6   - NET-DELAY-SOLUTION - TEXT
S14NDSL  DS    X     - NET-DELAY-SOLUTION - CODE
          DS    CL5
S14NTYP DS    CL1    - NET-TYPE
S14BHYP DS    CL1    - *BY-HYPERNET - Y OR EMPTY
S14S14L EQU  *-S14S14A - LENGTH OF DATA
S14LEN  EQU   *-N3JRNS

```

Definitions: record key 15

```

          ORG   N3INH
S15      AVASJRN S15
*****
*          RECORD DESCRIPTION  JOURNAL RECORD   RECORD KEY = 15          *
*****
*
S15S15A DS    OC      - RECORD DESCRIPTION   REC 15
S15FNAM DS    CL8    - FORMAT-NAME
S15FTXT DS    CL40   - FORMAT-TEXT
S15S15L EQU  *-S15S15A - LENGTH OF DATA
S15LEN  EQU   *-N3JRNS

```

Definitions: record key 16

```

          ORG   N3INH
S16      AVASJRN S16
*****
*          RECORD DESCRIPTION  JOURNAL RECORD   RECORD KEY = 16          *
*****
*
S16S16A  DS    OC          - RECORD DESCRIPTION   REC 16
S16JNAM  DS    CL64       - JOB-NAME
S16JLIB  DS    CL6        - JOB-LIB (JMDLIB, JMDSYS, JCLLIB, JCLSYS)
S16S16L  EQU   *-S16S16A  - LENGTH OF DATA
S16LEN   EQU   *-N3JRNS

```

Definitions: record key 21

```

          ORG   N3INH
S21      AVASJRN S21
*****
*          RECORD DESCRIPTION  JOURNAL RECORD   RECORD KEY = 21          *
*****
*
S21S21A  DS    OC          - RECORD DESCRIPTION   REC 21
S21UPAF  DS    CL54       - FILE-NAME
S21UPAR  DS    CL124     - PARAMETER RECORD
S21S21L  EQU   *-S21S21A  - LENGTH OF DATA
S21LEN   EQU   *-N3JRNS

```

Definitions: record key 31

```

          ORG   N3INH
S31      AVASJRN S31
*****
*          RECORD DESCRIPTION  JOURNAL RECORD   RECORD KEY = 31          *
*****
*
S31S31A DS    OC      - RECORD DESCRIPTION  REC 31
S31EARL DS    CL12   - EARLIEST START
S31LTSI DS    CL7    - LATEST START INPUT NPRLIB
S31L TSA DS    CL12   - LATEST START OUTPUT ABLDAT
S31LFTI DS    CL7    - LIFE-TIME INPUT NPRLIB
S31LFTA DS    CL12   - LIFE-TIME OUTPUT ABLDAT
S31RUCS DS    CL8    - RUN-CONT-SYS
S31NDSL T DS    OCL6  - NET-DELAY-SOLUTION - TEXT
S31NDSL DS     X      - NET-DELAY-SOLUTION - CODE
          DS    CL5
S31NTYP DS    CL1    - NET-TYPE
S31NCAT DS    CL4    - NET-CAT
S31NCJV DS    CL54   - NET-CAT  JV
S31OPST T DS    OCL3  - OPERATOR-START      TEXT
S31OPST DS     X      - OPERATOR-START      CODE
          DS    CL2
S31SYMD DS    CL20   - SYMDAT
S31NSRV DS    CL8    - NET-SERVER
S31BHYP DS    CL1    - *BY-HYPERNET - Y OR EMPTY
S31S31L EQU   *-S31S31A - LENGTH OF DATA
S31LEN  EQU   *-N3JRNS

```

Definitions: record key 32

```

          ORG   N3INH
S32      AVASJRN S32
*****
*          RECORD DESCRIPTION  JOURNAL RECORD   RECORD KEY = 32          *
*****
*
S32S32A DS    OC      - RECORD DESCRIPTION  REC 32
S32JNAM DS    CL64   - JOB-NAME
S32JLIB DS    CL6    - JOB-LIB (JMDLIB, JMDSYS, JCLLIB, JCLSYS)
S32JCAT DS    CL4    - JOB-CAT
S32JCJV DS    CL54   - JOB-CAT  JV
S32S32L EQU   *-S32S32A - LENGTH OF DATA
S32LEN  EQU   *-N3JRNS

```

Definitions: record key 51

```

          ORG   N3INH
S51      AVASJRN S51
*****
*          JOURNAL RECORD   RECORD KEY = 51   NET ENTRY IN E2 (E3)      *
*          REPLACES RECORD KEY = 41   FROM VERSION 02.0A ON           *
*****
*
S51S51A  DS    OC      - RECORD DESCRIPTION   REC 51
S51RCSN  DS    CL8     - RUN-CONTROL-SYSTEM-NAME
S51NST1T DS    OCL10   - NET STATUS 1 - TEXT
S51NST1  DS    X       - NET STATUS 1 - CODE
          DS    CL9
S51NST2T DS    OCL10   - NET STATUS 2 - TEXT
S51NST2  DS    X       - NET STATUS 2 - CODE
          DS    CL9
S51NST3T DS    OCL10   - INTERNAL SYSTEM VALUE (NET STATUS 3 TEXT)
S51NST3  DS    X       - INTERNAL SYSTEM VALUE (NET STATUS 3 CODE)
          DS    CL9
S51RVAR  DS    CL1     - RESTART VARIANTS 1, 2, 3
S51EARL  DS    CL12    - EARLIEST-START
S51LTSA  DS    CL12    - LATEST-START
S51BHYP  DS    CL1     - *BY-HYPERNET - Y OR EMPTY
          DS    CL10    - RESERVE
S51NSHTW DS    CL1     - NETSTATUS CALLED FOR HOSTWAIT
S51NTYP  DS    CL1     - NET-TYPE
S51NDSL  DS    OCL6    - NET-DELAY-SOLUTION - TEXT
S51NDSL  DS    X       - NET-DELAY-SOLUTION - CODE
          DS    CL5
S51NOPST DS    OCL3    - NET-OPERATOR-START - TEXT
S51NOPS  DS    X       - NET-OPERATOR-START - CODE
          DS    CL2
S51NCDCT DS    OCL3    - NET-CONDITION CREATED - TEXT
S51NCDC  DS    X       - NET-CONDITION CREATED - CODE
          DS    CL2
          DS    CL7     - RESERVE
S51NSERR DS    X       - NET STATUS CALLED FOR ERROR
S51NSRST DS    X       - NET STATUS CALLED FOR RESTART
S51NSCWT DS    X       - NET STATUS CALLED FOR CONDWAIT
S51NSHLD DS    X       - NET STATUS CALLED FOR HOLD
S51NSRSU DS    X       - NET STATUS CALLED FOR RESUME
S51NISON EQU   X'80'   - NET STATUS CALLED FOR IS SET (TM)
S51NSDA  DS    CL21    - NET STATUS ACCORDING TO DIALOG DISPLAY
S51S51L  EQU   *-S51S51A - LENGTH OF DATA
S51LEN   EQU   *-N3JRNS

```

Definitions: record key 52

```

          ORG   N3INH
S52      AVASJRN S42
*****
*          JOURNAL RECORD   RECORD KEY = 52   JOB PARAMETER ABLAT   *
*          REPLACES RECORD KEY = 42   FROM VERSION 02.0A ON       *
*****
*
S52S42A  DS    OC          - RECORD DESCRIPTION   REC 52
S52JCTYT DS    OCL3       - TYPE - TEXT
S52JCTY  DS    X          - TYPE - CODE
          DS    CL2
S52JSTI  DS    CL3       - START-INDEX
S52JSYN  DS    CL3       - SYNC-INDEX
S52JST1T DS    OCL10     - STATUS-1 - TEXT
S52JST1  DS    X        - STATUS-1 - CODE
          DS    CL9
S52JST2T DS    OCL10     - STATUS-2 - TEXT
S52JST2  DS    X        - STATUS-2 - CODE
          DS    CL9
S52JST3T DS    OCL10     - STATUS-3 - TEXT
S52JST3  DS    X        - STATUS-3 - CODE
          DS    CL9
S52JBEN  DS    CL2       - REASON FOR TERMINATION
S52JCKZ  DS    CL1       - FUNCTION
S52ACON  DS    XL2       - NUMBER OF DEPENDENCIES EXISTING
S52ECON  DS    XL2       - NUMBER OF DEPENDENCIES SATISFIED
S52S52L  EQU   *-S52S52A  - NORMAL LENGTH OF DATA
S52JVTSN DS    CL4       - BS2000 TSN FROM MONJV
S52JVTXT DS    OCL128    - TEXT FROM MONJV
          DS    CL118
S52FTID  DS    CL10     - FILE TRANSFER ID
S52JSCAT DS    CL4       - START CATALOG ID OF THE JOB
S52JSDAT DS    CL8       - START DATE OF THE JOB
S52S52LS EQU   *-S52S52A  - SPECIAL LENGTH OF DATA
S52LEN   EQU   *-N3JRNS

```

Definitions: record key 53

```

          ORG      N3INH
S53      AVASJRN S53
*****
*          JOURNAL RECORD   RECORD KEY = 53   ABLDAT CONDITION          *
*          REPLACES        RECORD KEY = 13   FROM VERSION 02.0A 0          *
*****
*
S53S53A  DS      OC          - RECORD DESCRIPTION   REC 53
S53CTYPT DS      OCL3       - COND-TYPE (JVA) - TEXT
S53CTYP  DS      X          - COND-TYPE (JVA) - CODE
          DS      CL2
S53CSYN  DS      CL3       - SYNC-INDEX
S53CNAM  DS      CL54      - JVA-NAME
S53JVAP  DS      CL3       - JVA-POSITION
S53JVAL  DS      CL3       - JVA-LENGTH
S53JVAV  DS      CL250     - JVA-VALUE
S53CJVO  DS      X          - COND-JVA-OPTION
LGOP          AVASEQU S53,EQUAME=STD,VERSION=STD,PE=C
S53CJVEQ EQU      0          ..EQ.          B'0000 xxxx'
S53CJVSP EQU      C' '       .BLANK        B'0100 0000'
S53CJVNE EQU      X'F0'      ..NE.         B'1111 xxxx'
S53CJVLT EQU      X'70'      ..LT.         B'0111 xxxx'
S53CJVLE EQU      X'B0'      ..LE.         B'1011 xxxx'
S53CJVGE EQU      X'D0'      ..GE.         B'1101 xxxx'
S53CJVGT EQU      X'E0'      ..GT.         B'1110 xxxx'
S5353L   EQU      *-S53S53A  - LENGTH OF THE DATA
S53LEN   EQU      *-N3JRNS

```

Definitions: record key 54

```

          ORG   N3INH
S54      AVASJRN S54
*****
*          JOURNAL RECORD  RECORD KEY = 54 STR-ELEMENT  ABLDAT          *
*          REPLACES RECORD KEY = 44 AS FROM VERSION 02.0A ON          *
*****
S54S54A  DS    OC      - RECORD DESCRIPTION REC 54
S54TYPT  DS    OCL3    - TYPE - TEXT
S54TYP   DS    X       - TYPE - CODE
          DS    CL2
S54SRTI  DS    CL3     - START-INDEX
S54SYNI  DS    CL3     - SYNC-INDEX
S54LATS  DS    CL12    - LATEST-START /-OCCURE / OCCURE-TIME
S54DSL   DS    OCL6    - DELAY-SOLUTION - TEXT
S54DSL   DS    X       - DELAY-SOLUTION - CODE
          DS    CL5
S54JRSP  DS    3CL48   - RESTART POINTS
          ORG   S54JRSP
S54RSV1  DS    OCL48   - RESTART-VARIANT-1
S54RSI1  DS    CL3     - RESTART-INDEX   V1
S54RSJ1  DS    CL30    - RESTART-NAME    V1
          DS    CL2     RESERVE
S54RST1T DS    OCL10   - RESTART-TYPE    V1 - TEXT
S54RST1  DS    X       - RESTART-TYPE    V1 - CODE
          DS    CL9
S54RSA1T DS    CL3     - AUTOMATIC YES /NO - TEXT
S54RSV2  DS    OCL48   - RESTART-VARIANT-2
S54RSI2  DS    CL3     - RESTART-INDEX   V2
S54RSJ2  DS    CL30    - RESTART-NAME    V2
          DS    CL2     RESERVE
S54RST2T DS    OCL10   - RESTART-TYPE    V2 - TEXT
S54RST2  DS    X       - RESTART-TYPE    V2 - CODE
          DS    CL9
S54RSA2T DS    CL3     - AUTOMATIC YES /NO - TEXT
S54RSV3  DS    OCL48   - RESTART-VARIANT-3
S54RSI3  DS    CL3     - RESTART-INDEX   V3
S54RSJ3  DS    CL30    - RESTART-NAME    V3
          DS    CL2     RESERVE
S54RST3T DS    OCL10   - RESTART-TYPE    V3 - TEXT
S54RST3  DS    X       - RESTART-TYPE    V3 - CODE
          DS    CL9
S54RSA3T DS    CL3     - AUTOMATIC YES /NO - TEXT
          DS    CL3     RESERVE
S54SELRV DS    C       -SELECT RESTART-VARIANT
S54S54L  EQU   *-S54S54A - LENGTH OF DATA
S54LEN   EQU   *-N3JRNS

```

Definitions: record key 55

```

          ORG   N3INH
S55      AVASJRN S55
*****
*          RECORD DESCRIPTION  JOURNAL RECORD   RECORD KEY = 55          *
*          REPLACES RECORD KEY = 63 AS FROM VERSION 02.0A ON            *
*****
*
S55S55A  DS    OC      - RECORD DESCRIPTION REC 55
S55RUCS  DS    CL8     - RUN-CONTROL-SYSTEM-NAME
S55EARL  DS    CL12    - EARLIEST START
S55LATS  DS    CL12    - LATEST START
S55BHYP  DS    CL1     - *BY-HYPERNET - Y OR EMPTY
          DS    CL11    - RESERVE
S55NDSL  DS    OCL6    - NET-DELAY-SOLUTION - TEXT
S55NDSL  DS    X       - NET-DELAY-SOLUTION - CODE
          DS    CL5
S55NTYP  DS    CL1     - NET-TYPE
S55NUSR  DS    CL8     - NET-USER
S55NACC  DS    CL8     - NET-ACCOUNT
S55NCLA  DS    CL8     - NET-CLASS
S55NLGM  DS    CL8     - NET-LOG
S55NCAT  DS    CL4     - NET-CAT
S55OPSTT DS    OCL3    - OPERATOR-START   TEXT
S55OPST  DS    X       - OPERATOR-START   CODE
          DS    CL2
S55NPAR  DS    CL128   - NET-PARAMETER
S55S55L  EQU   *-S55S55A - LENGTH OF DATA
S55LEN   EQU   *-N3JRNS

```


Definitions: record key 56

```

          ORG   N3INH
S56      AVASJRN S56
*****
*          RECORD DESCRIPTION  JOURNAL RECORD   RECORD KEY = 56          *
*          REPLACES RECORD KEY = 64 AS FROM VERSION 02.0A ON             *
*****
*
S56S56A  DS    OC      - RECORD DESCRIPTION REC 56
S56TYPT  DS    OCL3    - TYPE - TEXT
S56TYP   DS    X       - TYPE - CODE
          DS    CL2
          DS    CL3      RESERVED
S56JSYN  DS    CL3     - SYNC-INDEX
S56JENTT DS    OCL5    - ENTER-PARAMS - NET, LOGON - TEXT
S56JENT  DS    X       - ENTER-PARAMS - NET, LOGON - CODE
          DS    CL4
S56JUSR  DS    CL8     - JOB-USER
S56JACC  DS    CL8     - JOB-ACCOUNT
S56JCLA  DS    CL8     - JOB-CLASS
S56JLGM  DS    CL8     - JOB-LOG
S56JCAT  DS    CL4     - JOB-CAT
S56JPAR  DS    CL128   - JOB-PARAMETER
S56JFIL  DS    CL54    - ENTER-/SERVER-FILE-NAME
          DS    CL2     - RESERVED
S56SXID  DS    CL8     - SERVER-PA-NAME
S56S56L  EQU   *-S56S56A - LENGTH OF DATA
S56LEN   EQU   *-N3JRNS

```

Definitions record: key 57

```

          ORG   N3INH
S57      AVASJRN S57
*****
*          JOURNAL RECORD  RECORD KEY = 57  CONDITION  ABLDAT*
*                               FUNCTION = A,C,D,M,W                               *
*****
*
S57S57A  DS    OC      - RECORD DESCRIPTION REC 57
S57CTYPT DS    OCL3    - COND-TYPE (NET,JOB,RES,VAL,TIM) - TEXT
S57CTYP  DS    X       - COND-TYPE   - CODE
          DS    CL2
          DS    CL3      RESERVED
S57CSYI  DS    CL3     - SYNC-INDEX
S57CVAL  DS    OCL256  - CONDITION-VALUE
S57COCV  DS    CL128   - OCCURE-VALUE
S57CERV  DS    CL128   - ERROR-VALUE
S57CRBN  DS    CL32    - CREATED-BY NET-NAME
S57CRBI  DS    CL3     - CREATED-BY INDEX
*
S57CSTA  DS    OCL10   - COND-STATUS - TEXT
S57CSTC  DS    X       - COND-STATUS - CODE
          DS    CL9
S57S57L  EQU   *-S57S57A - LENGTH OF DATA
S57LEN   EQU   *-N3JRNS

```

Definitions: record key 58

```

          ORG   N3INH
S58      AVASJRN S58
*****
*          RECORD DESCRIPTION  JOURNAL RECORD   RECORD KEY = 58          *
*          REPLACES RECORD KEY = 68 AS FROM VERSION 02.0A ON              *
*****
*
S58S58A  DS    OC      - RECORD DESCRIPTION REC 58
S58TYPT  DS    OCL3    - TYPE - TEXT
S58TYP   DS    X       - TYPE - CODE
          DS    CL2
S58ESTI  DS    CL3     - START-INDEX
S58ESYI  DS    CL3     - SYNC-INDEX
          DS    C       RESERVE
S58EIND  DS    CL3     - ERROR-INDEX
S58EJOB  DS    CL30    - ERROR-NAME
          DS    CL2     RESERVED
S58RVAR  DS    CL1     - RESTART-VARIANT
S58RIND  DS    CL3     - RESTART-INDEX
S58RJOB  DS    CL30    - RESTART-NAME
          DS    CL2     RESERVE
S58RTYPT DS    OCL10   - RESTART-TYPE - TEXT
S58RTYP  DS    X       - RESTART-TYPE - CODE
          DS    CL9
S58RTYAT DS    CL3     - AUTOMATIC YES / NO
S58S58L  EQU   *-S58S58A - LENGTH OF DATA
S58LEN   EQU   *-N3JRNS

```

Definitions: record key 59

```

          ORG          N3INH
S59      AVASJRN      S59
*****
*          JOURNAL RECORD   RECORD KEY = 59 START PARAMETER  ABLDAT          *
*                                                                                               *
*****
*
S59S59A  DS      OC      - RECORD DESCRIPTION REC 59
S59SNTYT DS      OCL3    - TYPE - TEXT
S59SNTY  DS      X       - TYPE - CODE
          DS      CL2
S59SSTI  DS      CL3     - START-INDEX
S59SSYN  DS      CL3     - SYNC-INDEX
S59SST1T DS      OCL10   - STATUS-1 - TEXT
S59SST1  DS      X       - STATUS-1 - CODE
          DS      CL9
S59SST2T DS      OCL10   - STATUS-2 - TEXT
S59SST2  DS      X       - STATUS-2 - CODE
          DS      CL9
S59SST3T DS      OCL10   - STATUS-3 - TEXT
S59SST3  DS      X       - STATUS-3 - CODE
          DS      CL9
S59SBEN  DS      CL2     - REASON FOR TERMINATION
S59SCKZ  DS      CL1     - FUNCTION
S59ACON  DS      XL2     - NUMBER OF DEPENDENCIES EXISTING
S59ECON  DS      XL2     - NUMBER OF DEPENDENCIES SATISFIED
S59S59L  EQU     *-S59S59A - NORMAL LENGTH OF DATA
          DS      CL4     - RESERVED
S59SVTXT DS      CL128   - TEXT FROM SUBNET-JVA
S59SSCAT DS      CL4     - CATID SUBNET-JVA
S59SSDAT DS      CL8     - START DATE OF THE SUBNET
S59S59LS EQU     *-S59S59A - SPECIAL LENGTH OF DATA

```

Definition: record key 60

```

          ORG   N3INH
S60      AVASJRN S60
*****
*          RECORD DESCRIPTION  JOURNAL RECORD  RECORD KEY = 60          *
*****
*
S60S60A  DS    OC          - RECORD DESCRIPTION REC 60
S60CTYPT DS    OCL3       - TYPE - TEXT
S60CTYP  DS    X          - TYPE - CODE
          DS    CL2
S60FSYN  DS    CL3       - SYNC-INDEX
S60FDIRT DS    OCL4       - DIRECTION - TEXT : TO, FROM
S60FDIR  DS    X          CODE
          DS    XL3
S60FREMT DS    OCL7       - FT-REMOTE - TEXT : *BS2000, *MSP, *ANY
S60FREM  DS    X          CODE
          DS    XL6
S60FRFA  DS    CL67      - FT-REMOTE-FTAC
S60FPTN  DS    CL8       - FT-PARTNER
S60FLFN  DS    CL54      - FT-LOCAL-FILE-NAME
S60FRFN  DS    CL54      - FT-REMOTE-FILE-NAME
S60FPAR  DS    CL192     - FT-PARAMETER
S60S60L  EQU   *-S60S60A - LENGTH OF DATA

```

4.4 Structure of the ISAM journal log file

The entries in the journal file can also be logged in an ISAM file which is opened in SHARUPD mode and can consequently be continuously updated, for example using database applications. This file is requested by means of a start parameter when the UPAM-ZD is started up (see [“JRNDAT-ISAM-NAME=” on page 110](#)). It is created with variable record length, BLKSIZE=(STD,2), KEYLEN=8 and KEYPOS=5.

Structure of the records

ISAM-KEY is the STCK value at output time.

This is followed by the user data (without record length field) from the F3N record of the journal file (see [section “Record structure of the fixed portion of journal records” on page 204](#)), then the user data (without record length field, N3F4SK and N3SSTA) from the F3S record (with N3FUNC and N3AKTN being edited for printing, as explained below) and the data from the variable part of the journal entry (described in [section “Record structure of the variable-length portion of journal records” on page 207](#)).

The following fields are displayed in readable form:

- N3FUNC The 1-byte long function code from the journal file is converted in decimal form to a 4-byte long printable format. For example the value x'49' (corresponding to d'73' = N3ERM) is mapped to c'0073'. All values which occur are defined in the F3S table starting at the label FUNC; x'00' is converted to c'****'.
- N3AKTN The 1-byte long action code from the journal file is converted to 12-byte long text as it is assigned in the comment field, starting from label AKTN, to the action codes. For example the value x'84' (corresponding to d'132' = N3EN-CN) is mapped to C'NO CANCEL '. x'00' is converted to 'PROTOCOL '.

Record structure of the ISAM journal log

Field name	Content	Length
SLF	Record length field	4
KEY	ISAM-KEY: STCK output	8
N3SYSID	SYSTEM ID	7
N3BKR	User group	5
N3NNAM	Net name	32
N3RF4S	BLANK	2
N3DTJH	Century JH	2
N3DATUM	Date YYMMTT	6
N3UHRZ	Time HHMMSS	6
N3LNUM	Sequence number	2
N3FUNC	AVAS function printable	4
N3AKTN	Executed action as text	12
N3USER	AVAS user ID	8
N3IND	Synchronization index	3
N3JCNK	FU (A,C,D,F,J,M,N,P,W)	1
N3NAME	Name Net/Job/Condition	32
N3DSSL	Journal record key	2
N3DFNR	Journal record sequence number	2
N3INH	Variable part of the journal record	max. 408

4.5 Structure of the history file

In addition to the AVAS journal, AVAS also writes a compressed file containing the following:

- A compressed record for each schedule variant of all nets (KEY=01).
- A compressed record for each job under a net (KEY=02).
- A record for each net that has expired (KEY=03).
- A record for each job that was run under a net (KEY=04).

The evaluation function HISTORY can be used for nets, jobs in nets and for jobs.

Data records with the following keys and compression levels are saved:

KEY	\$bk	net-n	YYYYMMDD	hhmmss	SYMD	Index	\$bk_job-n	FU	TYP
01	X	X	–	–	X	–	–	N	SUM
02	X	X	–	–	X	X	X	J,P,F	SUM
03	X	X	X	X	X	–	–	N	NET
04	X	X	X	X	X	X	X	J,P,F	JOB

X value exists

– no value exists (blanks)

The data is saved in an ISAM file with the key fields KEY, USER-GROUP, NET-NAME, PLAN-START-DATE, PLAN-START-TIME, SYMDAT-NAME, INDEX, JOB-NAME, FU and TYP.

The file is updated when the journal file is reorganized (function \$JOUR) with the HISTORY-SAVE-OPTION parameter.

The data is evaluated with the SHOW-HISTORY operation in the NET-CONTROL statement.

Data records are deleted in the course of a reorganization with the HISTORY function. Besides NET-NAME, the HISTORY-DELETE-OPTION, HISTORY-DELETE-DATE and HISTORY-KEEP-RECORDS parameters are also allowed in the HISTORY function to delete records from the history file.

A maximum of 64 runtimes and the wait times of the last 64 runs are allowed in the compressed records (KEY=01, 02). The mean value, range and standard difference are calculated from the maximum of 64 stored here.

In the history file, only the records with KEY 03 and 04 can be reorganized (deleted) with HISTORY-DELETE-DATE.

The compressed records with KEY 01 and 02 are deleted if no runtimes are contained in the records any more (Number=0).

Keys of the history file (POS=5, LEN=90)

Name	Value	Length	from pos.	to pos.
KEY	01, 02, 03, 04	2	5	6
USER-GROUP	\$bk	5	7	11
NET-NAME	net-n	12	12	23
PLAN-START-DATE	YYYYMMDD	8	24	31
PLAN-START-TIME	hhmmss	6	32	37
SYMDAT-NAME	SYMD	20	38	57
INDEX	index	3	58	60
JOB-NAME	\$bk_job-n	30	61	90
FU	N, J, P, F	1	91	91
TYP	SUM / NET / JOB	3	92	94

The user can read the records in the history file with own programs. To do so, the file must be opened with the parameters SHARUPD=YES and OPEN mode INPUT. The record descriptions are called via the AVASJRN macro call.

The user can save runtime data such as the CPU time used and the I/O rate in the data records of the activated jobs (KEY=04). If the data is to be saved in the history file the user must transfer them to AVAS using the AVAS statement #AVA#\$H01. The run control system enters the values with record keys 08–02 in the journal file. When the journal files are saved, they are transferred to the history record with KEY=04 (field H04UINF=USER-INFO of the #AVA#\$H statement).

The file should not be accessed while the reorganization is being performed since this might cause inconsistencies and access to individual records might be locked.

The record descriptions are called via the AVASJRN macro call:

```
Hkk  AVASJRN    &pre
kk    record key 01, 02, 03 or 04
      or "KY" for the definitions of the ISAM key
&pre  prefix
```

Structure and contents of the history records

KEY=01 – history net (compressed)

Column headings	Contents	Length
Record key	KEY=01	2
User group	\$bk	5
Net name	netname	12
Scheduled start date		8
Scheduled start time		6
Symbolic start time	SYMDAT-NAME	20
Index		3
Job name		30
Function	FU= N	1
Type	TYPE=SUM	3
Last start time	dd.mm.yyyy-hh:mm:ss	19
Last runtime	hhh:mm:ss	9
Last end time	dd.mm.yyyy-hh:mm:ss	19
Number of runs	nnnn	4
Mean runtime of net	hhh:mm:ss	9
Maximum runtime of net	hhh:mm:ss	9
Minimum runtime of net	hhh:mm:ss	9
Range	hhh:mm:ss	9
Standard difference	hhh:mm:ss	9
Mean runtime of job	hhh:mm:ss	9
Maximum runtime of job	hhh:mm:ss	9
Mean wait time of conditions	hhh:mm:ss	9
Maximum wait time of conditions	hhh:mm:ss	9
Mean wait time for ERROR	hhh:mm:ss	9
Maximum wait time for ERROR	hhh:mm:ss	9
Average number of ERRORS	000,00...999,99	6
Number of times	nn	2
Scheduled start date/time	YYYYMMDDhhmmss	14
Runtime for net – 1	hhh:mm:ss	9

continued ►

Column headings	Contents	Length
Runtime for job – 1	hhh:mm:ss	9
Wait time for conditions – 1	hhh:mm:ss	9
Wait time for ERROR – 1	hhh:mm:ss	9
Number of ERRORS – 1	nnn	3
Scheduled start date/time	YYYYMMDDhhmmss	14
Runtime net – 64	hhh:mm:ss	9
Runtime jobs – 64	hhh:mm:ss	9
Wait time conditions – 64	hhh:mm:ss	9
Wait time for ERROR – 64	hhh:mm:ss	9
Number of ERRORS – 64	nnn	3

KEY=02 – history job (compressed)

Column headings	Contents	Length
Record key	KEY=02	2
User group	\$bk	5
Net name	netname	12
Scheduled start date		8
Scheduled start time		6
Symbolic start time	SYMDAT-NAME	20
Index	IND	3
Job name	\$bk_jobname	30
Function	FU= { J / P / F }	1
Type	TYPE= SUM	3
Last start time	dd.mm.yyyy-hh:mm:ss	19
Last run time	hhh:mm:ss	9
Last end time	dd.mm.yyyy-hh:mm:ss	19
Number of runs	nnnn	4
Mean runtime of job	hhh:mm:ss	9
Maximum runtime of job	hhh:mm:ss	9
Minimum runtime of job	hhh:mm:ss	9
Range	hhh:mm:ss	9
Standard difference	hhh:mm:ss	9
Mean wait time for ERROR	hhh:mm:ss	9
Maximum wait time for ERROR	hhh:mm:ss	9
Average number of ERRORS	000,00...999,99	6
Number of times	nn	2
Scheduled start date/time	YYYYMMDDhhmmss	14
Runtime for jobs – 1	hhh:mm:ss	9
Wait time for ERROR – 1	hhh:mm:ss	9
Number of ERRORS – 1	nnn	3
...		
Scheduled start date/time	YYYYMMDDhhmmss	14
Runtime for jobs – 64	hhh:mm:ss	9
Wait time for ERROR – 64	hhh:mm:ss	9
Number of ERRORS – 64	nnn	3

KEY=03 – net run

Column headings	Contents	Length
Record key	KEY=03	2
User group	\$bk	5
Net name	netname	12
Scheduled start date	YYYYMMDD	8
Scheduled start time	hhmmss	6
Symbolic start time	SYMDAT-NAME	20
Index		3
Job name		30
Function	FU= N	1
Type	TYPE=NET	3
Start time	dd.mm.yyyy-hh:mm:ss	19
Runtime	hhh:mm:ss	9
End time	dd.mm.yyyy-hh:mm:ss	19
Runtime of jobs	hhh:mm:ss	9
Wait time for conditions	hhh:mm:ss	9
Wait time for ERROR	hhh:mm:ss	9
Number of ERRORS	01...99	6
JRNDAT-SAVE-FILE	filename	54
JOBLOG-SAVE-FILE	filename	54

KEY=04 – job run

Column headings	Contents	Length
Record key	KEY=04	2
User group	\$bk	5
Net name	netname	12
Scheduled start date	YYYYMMDD	8
Scheduled start time	hhmmss	6
Symbolic start time	SYMDAT-NAME	20
Index	IND	3
Job name	\$bk_jobname	30

continued ➡

Column headings	Contents	Length
Function	FU= { J / P / F }	1
Type	TYPE= {STD / MOD / EXT}	3
Start time	dd.mm.yyyy-hh:mm:ss	19
Runtime	hhh:mm:ss	9
End time	dd.mm.yyyy-hh:mm:ss	19
Number of errors	nnnnnn	6
Data from #AVA#\$H01 statement		
Statement key	\$H	2
Record number	01	2
Operating system	system-name	12
Operating system version	Vnn.nAmmmm	10
HOST-(BCA-) name	host-name	8
BS2000 version	Vnn.nAmmmm	10
IP address Internet Protocol Version 4	nnn.nnn.nnn.nnn	15
CPU time used	nnnnnn.nnnn	11
Number of I/Os	nnnnnnnnnnnnnnnn	16
IP address Internet Protocol Version 6	nnnn:nnnn:nnnn:nnnn:nnnn: nnnn:nnnn:nnnn	39
User information	text 1...128	128
Wait time for ERROR	hhh:mm:ss	9
Number of ERRORS	001...999	3
Number of times	00...10	2
ERROR time – 1	dd.mm.yyyy-hh:mm:ss	19
Wait time	hhh:mm:ss	9
RESTART time	dd.mm.yyyy-hh:mm:ss	19
MONJV contents for ERROR	text 129-256	128
.....
ERROR time – 10

Entering and deleting runs

KEY=01 – history net and KEY=02 – history job

In a reorganization, new runs are entered in the total records (SJOUR) or deleted from the total records (HISTORY). The following cases should be distinguished:

1. Funktion=SJOUR
Number smaller than 64; PLAN-START larger than max old;
The run is appended to the existing data.
2. Funktion=SJOUR
Number equals 64; PLAN-START larger than max old;
The first entry is deleted.
The run is appended to the existing data.
3. Funktion=SJOUR
Number equals 64; PLAN-START larger than min, smaller than max;
The first entry is deleted.
The run is appended to the existing data according to PLAN-START.
4. Funktion=SJOUR
Number smaller than 64; PLAN-START larger than min, smaller than max;
The run is appended to the existing data according to PLAN-START.
5. Funktion=SJOUR
Number equals 64; PLAN-START smaller than min;
The run is not considered.
6. Funktion=HISTORY
Number equals 64; PLAN-START larger than min, smaller than max with regard to HISTORY-DELETE-DATE;
The entry is deleted.
7. Funktion=HISTORY
Number smaller than 64; PLAN-START larger than or equals min, smaller than or equals max with regard to HISTORY-DELETE-DATE;
The entry is deleted.
8. Funktion=HISTORY
Number smaller than or equals 64 and HISTORY-KEEP-RECORDS smaller than number;
The first entry is deleted in each case until the number equals HISTORY-KEEP-RECORDS.
9. Funktion=HISTORY
Number equals 0;
The compressed record (KEY =01, 02) is deleted.

Note

All new entries must be appended according to PLAN-START because otherwise the PLAN-START of all entries must be checked to find the entry with the lowest PLAN-START when deleting.

Structure of history records**Definitions KEY=01, TYP=SUM, FU=N**

```

H01      AVASJRN H01
1 *****
1 *      RECORD DESCRIPTION  HISTORY KEY=01  SINCE VERS. 6.0A  H01 *
1 *****
1 *
1 H01H01  DS    OF          - RECORD DESCRIPTION HISTORY KEY=01
1 H01SLEN DS    H          - RECORD LENGTH FIELD
1        DS    H          - RESERVED
1 HKY     AVASJRN H01,EQU=NO
2 *****
2 *      RECORD DESCRIPTION  HISTORY ISAM-KEY  SINCE VERS. 6.0A  *
2 *****
2 *
2 H01KEY  DS    CL2        - RECORD KEY 01
2 H01BKR  DS    CL5        - USER GROUP
2 H01NETN DS    CL12       - NET-NAME
2 H01DTPL DS    0CL8      - SCHEDULED DATE  CCYYMMDD
2 H01DTJH DS    CL2        - CENTURY          CC
2 H01PLSD DS    CL6        - PLAN-START-DATE YYMMDD
2 H01PLST DS    CL6        - PLAN-START-TIME HMMSS
2 H01SYMD DS    CL20       - SYMDAT-NAME
2 H01IND  DS    CL3        - INDEX
2 H01JOBN DS    CL30       - JOB-NAME
2 H01FUNK DS    C          - FU = (N,J,P,X,F)
2 H01STYP DS    CL3        - TYP = (SUM,NET,JOB)
2 H01KLEN EQU  *-H01KEY   - KEYLEN
1 H01HLEN EQU  *-H01H01   - LENGTH OF HEADER
1        DS    CL10       - RESERVED
1 H01LSTT DS    CL19       - LAST START-TIME
1 H01LRNT DS    CL9        - LAST RUN-TIME
1 H01LENT DS    CL19       - LAST END-TIME
1 H01ANZA DS    CL4        - NUMBER OF RUNS

```


1	H01DURT	DS	CL9	-	MEAN RUN-TIME	
1	H01MART	DS	CL9	-	MAX RUN-TIME	
1	H01MIRT	DS	CL9	-	MIN RUN-TIME	
1	H01RANG	DS	CL9	-	RANGE	
1	H01STDA	DS	CL9	-	STANDARD DIFFERENCE	
1	H01DRTJ	DS	CL9	-	MEAN RUN-TIME JOBS	
1	H01MRTJ	DS	CL9	-	MAX RUN-TIME JOBS	
1	H01DWTC	DS	CL9	-	MEAN WAIT-TIME CONDITIONS	
1	H01MWTC	DS	CL9	-	MAX WAIT-TIME CONDITIONS	
1	H01DWTE	DS	CL9	-	MEAN WAIT-TIME ERRORS	
1	H01MWTE	DS	CL9	-	MAX WAIT-TIME ERRORS	
1	H01DAZE	DS	CL6	-	AVERAGE NUMBER OF ERRORS	
1	H01DAZE	DS	CL6	-	AVERAGE NUMBER OF ERRORS	
1		DS	CL250	-	RESERVED	
1	H01ARWTDSOC			-	ADR RUN-/WAIT-TIMES	
1		DS	64CL55	-	RUN-/WAIT-TIME 1-64	
1	H01ERWT	DS	OC	-	END RUN-/WAIT-TIMES	
1	H01SLEN	EQU	*-H01H01	-	LENGTH OF RECORD H01	
1		ORG	H01ARWT			
1	H01PST1	DS	CL14	-	PLAN-START	- 1
1	H01RTN1	DS	CL9	-	RUN-TIME NET	- 1
1	H01RTJ1	DS	CL9	-	RUN-TIME JOBS	- 1
1	H01WTC1	DS	CL9	-	WAIT-TIME COND	- 1
1	H01WTE1	DS	CL9	-	WAIT-TIME ERROR	- 1
1	H01AZE1	DS	CL3	-	NUMBER OF ERRORS	- 1
1		DS	CL2	-	RESERVED	- 1
1		ORG	H01ERWT			

Definitions KEY=02, TYP=SUM, FU=J / P / F

```

H02      AVASJRN H02
1 *****
1 *      RECORD DESCRIPTION  HISTORY KEY=02  SINCE VERS. 6.0A  H02  *
1 *****
1 *
1 H02H02  DS    OF      - RECORD DESCRIPTION HISTORY KEY=02
1 H02SLEN DS    H      - RECORD LENGTH FIELD
1         DS    H      - RESERVED
1 HKY     AVASJRN H02,EQU=NO
2 *****
2 *      RECORD DESCRIPTION  HISTORY ISAM-KEY  SINCE VERS. 6.0A  *
2 *****
2 *
2 H02KEY  DS    CL2      - RECORD KEY 02
2 H02BKR  DS    CL5      - USER GROUP
2 H02NETN DS    CL12     - NET-NAME
2 H02DTPL DS    OCL8     - SCHEDULED DATE  CCYYMMDD
2 H02DTJH DS    CL2      - CENTURY          CC
2 H02PLSD DS    CL6      - PLAN-START-DATE YYMMDD
2 H02PLST DS    CL6      - PLAN-START-TIME HHMMSS
2 H02SYMD DS    CL20     - SYMDAT-NAME
2 H02IND  DS    CL3      - INDEX
2 H02JOBN DS    CL30     - JOB-NAME
2 H02FUNK DS    C        - FU = (N,J,P,X,F)
2 H02STYP DS    CL3      - TYP = (SUM,NET,JOB)
2 H02KLEN EQU  *-H02KEY  - KEYLEN
1 H02HLEN EQU  *-H02H02  - LENGTH OF HEADER
1         DS    CL10     - RESERVED
1 H02LSTT DS    CL19     - LAST START-TIME
1 H02LRNT DS    CL9      - LAST RUN-TIME
1 H02LENT DS    CL19     - LAST END-TIME
1 H02ANZA DS    CL4      - NUMBER OF RUNS
1 H02DURT DS    CL9      - MEAN RUN-TIME
1 H02MART DS    CL9      - MAX  RUN-TIME
1 H02MIRT DS    CL9      - MIN  RUN-TIME
1 H02RANG DS    CL9      - RANGE
1 H02STDA DS    CL9      - STANDARD DIFFERENCE
1         DS    CL9      -
1         DS    CL9      -
1         DS    CL9      -
1         DS    CL9      -
1 H02DWTE DS    CL9      - MEAN WAIT-TIME ERRORS
1 H02MWTE DS    CL9      - MAX  WAIT-TIME ERRORS
1 H02DAZE DS    CL6      - AVERAGE NUMBER OF ERRORS
1         DS    CL250    - RESERVED
1 H02ANZS DS    CL2      - NUMBER OF STORED TIMES

```

```

1 H02ARWT DS OC - ADR RUN-/WAIT-TIMES
1 DS 64CL55 - RUN-/WAIT-TIME 1-64
1 H02ERWT DS OC - END RUN-/WAIT-TIMES
1 H02SLEN EQU *-H02H02 - LENGTH OF RECORD H02
1 ORG H02ARWT
1 H02PST1 DS CL14 - PLAN-START - 1
1 DS CL9 -
1 H02RTJ1 DS CL9 - RUN-TIME JOB - 1
1 DS CL9 -
1 H02WTE1 DS CL9 - WAIT-TIME ERROR - 1
1 H02AZE1 DS CL3 - NUMBER OF ERRORS - 1
1 DS CL2 - RESERVED - 1
1 ORG H02ERWT

```

Definitions KEY=03, TYP=NET, FU=N

```

H03 AVASJRN H03
1 *****
1 * RECORD DESCRIPTION HISTORY KEY=03 SINCE VERS. 6.0A H03 *
1 *****
1 *
1 H03H03 DS OF - RECORD DESCRIPTION HISTORY KEY=03
1 H03SLEN DS H - RECORD LENGTH FIELD
1 DS H - RESERVED
1 HKY AVASJRN H03,EQU=NO
2 *****
2 * RECORD DESCRIPTION HISTORY ISAM-KEY SINCE VERS. 6.0A *
2 *****
2 *
2 H03KEY DS CL2 - RECORD KEY 03
2 H03BKR DS CL5 - USER GROUP
2 H03NETN DS CL12 - NET-NAME
2 H03DTPL DS OCL8 - SCHEDULED DATE CCYYMMDD
2 H03DTJH DS CL2 - CENTURY CC
2 H03PLSD DS CL6 - PLAN-START-DATE YYMMDD
2 H03PLST DS CL6 - PLAN-START-TIME HHMMSS
2 H03SYMD DS CL20 - SYMDAT-NAME
2 H03IND DS CL3 - INDEX
2 H03JOBND DS CL30 - JOB-NAME
2 H03FUNK DS C - FU = (N,J,P,X,F)
2 H03STYP DS CL3 - TYP = (SUM,NET,JOB)
1 H03KLEN EQU *-H03KEY - KEYLEN
1 H03HLEN EQU *-H03H03 - LENGTH OF HEADER
1 DS CL10 - RESERVED
1 H03STT DS CL19 - START-TIME
1 H03RNT DS CL9 - RUN-TIME
1 H03ENT DS CL19 - END-TIME

```

1	DS	CL4	-	
1	DS	CL9	-	
1	DS	CL9	-	
1	DS	CL9	-	
1	DS	CL9	-	
1	DS	CL9	-	
1	H03RTJ	DS	CL9	- RUN-TIME JOBS
1		DS	CL9	-
1	H03WTC	DS	CL9	- WAIT-TIME CONDITIONS
1		DS	CL9	-
1	H03WTE	DS	CL9	- WAIT-TIME ERRORS
1		DS	CL9	-
1	H03AZE	DS	CL6	- NUMBER OF ERRORS
1	H03JRNS	DS	CL54	- JRNDAT-SAVE-FILE
1	H03LOGS	DS	CL54	- JOBLOG-SAVE-FILE
1		DS	CL626	-
1		DS	CL3044	-
1	H03SLEN	EQU	*-H03H03	- LENGTH OF KEY 03

Definitions KEY=04, TYP=JOB, FU=J / P / F

```

H04      AVASJRN H04
1 *****
1 *      RECORD DESCRIPTION  HISTORY KEY=04  SINCE VERS. 6.0A  H04  *
1 *****
1 *
1 H04H04  DS    OF      - RECORD DESCRIPTION HISTORY KEY=04
1 H04SLEN DS    H      - RECORD LENGTH FIELD
1        DS    H      - RESERVED
1 HKY     AVASJRN H04,EQU=NO
2 *****
2 *      RECORD DESCRIPTION  HISTORY ISAM-KEY  SINCE VERS. 6.0A  *
2 *****
2 *
2 H04KEY  DS    CL2      - RECORD KEY 04
2 H04BKR  DS    CL5      - USER GROUP
2 H04NETN DS    CL12     - NET-NAME
2 H04DTPL DS    OCL8     - SCHEDULED DATE  CCYYMMDD
2 H04DTJH DS    CL2      - CENTURY          CC
2 H04PLSD DS    CL6      - PLAN-START-DATE YYMMDD
2 H04PLST DS    CL6      - PLAN-START-TIME HHMMSS
2 H04SYMD DS    CL20     - SYMDAT-NAME
2 H04IND  DS    CL3      - INDEX
2 H04JOBN DS    CL30     - JOB-NAME
2 H04FUNK DS    C        - FU = (N,J,P,X,F)
2 H04STYP DS    CL3      - TYP = (SUM,NET,JOB)
2 H04KLEN EQU    *-H04KEY - KEYLEN
1 H04HLEN EQU    *-H04H04 - LENGTH OF HEADER
1        DS    CL10     - RESERVED
1 H04STT  DS    CL19     - START-TIME
1 H04RNT  DS    CL9      - RUN-TIME
1 H04ENT  DS    CL19     - END-TIME
1 H04AZE  DS    CL6      - NUMBER OF ERRORS
1        DS    CL6      -
1        DS    CL6      -
1        DS    CL341    - RESERVED
1 H04UINF DS    OCL256   - INFO from #AVA#$H statement
1 H04USKY DS    CL2      - SYSTEM KEY $H
1 H04URKY DS    CL2      - USER KEY 51-99
1 H04URDT DS    CL252    - KEY DATA
1        ORG    H04URKY
1 H04U1KY DS    CL2      - KEY 01
1 H04U1BS DS    CL12     - NAME OF OPERATING SYSTEM
1 H04U1BV DS    CL10     - VERSION OF OPERATING SYSTEM
1 H04U1HN DS    CL8      - HOST- (BCAM-) NAME
1 H04U1IP DS    CL15     - IPv4-ADDRESS
1 H04U1CV DS    CL11     - CPU CONSUME

```

1	H04U1IO	DS	CL16	- NUMBER I/O
1	H04U1IP6	DS	CL39	- IPv6-ADDRESS
1		DS	CL13	- RESERVED
1	H04U1UD	DS	CL128	- USER DATA AREA 01
1		DS	4CL256	- RESERVED
1		DS	CL31	- RESERVED
1	H04WTER	DS	CL9	- WAIT-TIME-ERROR
1	H04ANZS	DS	CL2	- NUMBER OF TABLE SEGMENTS
1	H04AEWT	DS	0C	- ANF-ADR ERROR-WAIT-TIMES
1		DS	10CL220	- START-/RUN-TIME 1-10
1	H04EEWT	DS	0C	- END-ADR ERROR-WAIT-TIMES
1	H04LREC	EQU	*-H04H04	- LENGTH OF RECORD H04
1		ORG	H04AEWT	
1	H04ERRT	DS	CL19	- ERROR-TIME
1	H04WAIT	DS	CL9	- WAIT-TIME
1	H04RSTT	DS	CL19	- RESTART-TIME
1	H04JVVE	DS	CL128	- MONJV CONTENT IN ERROR CASE
1		DS	CL45	- RESERVED
1	H04LSEG	EQU	*-H04ERRT	- LENGTH OF SEGMENT

Notes

- In nets that were released for processing under via REPEAT-NET, the parameter SYMDAT-NAME is not available. Therefore scheduling is assumed without the symbolic start time in these nets.
- The symbolic start time with which the net is planned is saved in records with KEY=02 and 04 (FU=J / P / F, TYP=SUM / JOB).
- The values which the user transferred to AVAS with the AVAS statement #AVA#\$H01 are not checked by AVAS. The values are transferred to the history record with KEY=04 (field H04UINF=INFO from #AVA#\$H statement).
- All times (runtimes, wait times, etc.) are determined from the time stamps in the journal records. The time stamps are issued by the AVAS run control system for:
 - the start and the end of nets and jobs,
 - waiting for a condition to be fulfilled and
 - a condition being fulfilled.
- If the run control system is terminated abnormally while a job is running, the runtime will not be realistic, because the journal record with the job end time is issued only when the run control system restarts, while the job will already have reached the BS2000 status \$T (or \$A) before.

5 AVAS-QUER utility routine

The AVAS-QUER utility routine reads the AVAS data stock in BS2000 and selects data for further processing in relational databases.

AVAS-QUER writes the data either to an output file containing the data in INSERT format (i.e. in SQL format), or to output files containing the data in "LOAD2" or "CSV format". CSV (Comma Separated Value) is a format which can be processed by many database systems (e.g. by Microsoft Access).

Users must transfer the output file(s) created by AVAS-QUER to their target system. They then execute the files or import the data to the database system.

On the target system, the user can employ suitable database queries to obtain derived information such as the following:

1. In which nets is a particular job used?
2. In which nets is a particular symbolic date (symdat) used?
3. On which days is a particular symbolic date set?
4. Dependency structures in the form:
 - Which net depends on another specific net?
 - Which net depends on the contents of a particular job variable?
 - Which net depends on a job in another net?
 - Which net tests a particular resource?
 - Which net depends on a particular value?
5. In which objects is a particular document used?
6. Which jobs are running on a particular pubset, or which job variable contains the pubset?
7. Which user groups use a particular calendar?
8. In which orders is a particular SELECT-TURNUS used?
9. For which orders or nets is there a document in the DOCLIB?

Recommended procedure

The following table shows the steps required to obtain the information listed above. Detailed descriptions of the steps and the results can be found in the sections listed in the table.

System platform	Procedure	Result
BS2000	Install AVAS (AVAS-QUER is automatically installed as well)	AVAS-QUER is available
Target system with database	Call the AVAS-QUER utility routine (see page 242), log on to AVAS (SIGNON), create a format (CREATE-FORMAT), select objects (SELECT-OBJECT) (see page 247 and page 250)	AVAS data is read
	Terminate the AVAS-QUER utility routine (see page 247)	Output files in SAM format created by AVAS-QUER are available. They contain the selected data, which can be imported into a relational database. (see page 248)
	Transfer the output file(s) created by AVAS-QUER via file transfer	Output file(s) are available on target system.
	The database scheme must be created	Database and its structure (tables) are created (initialization) (see page 260 and page 262)
	Call the output file(s) created by AVAS-QUER (see page 259) or import data with database-specific statements	Supply the database with values (tables are filled in)
	Formulate database queries in SQL (see page 267) or use the user interface for evaluation	Information on the relationships of the AVAS objects: in which nets is a particular job/symdat used?

5.1 Working with AVAS-QUER

AVAS-QUÉR is integrated in AVAS. The following prerequisites must be satisfied in order to work with AVAS-QUER.

5.1.1 Prerequisites for starting AVAS-QUER

- The file link name SYSLNK must be assigned to the file SYSLNK.AVAS.085
- In the system parameters of the AVAS system, the user must have COPY ELEMENT authorization for the objects which he/she may access.
- An SDF syntax file containing the statements for AVAS-QUER must be active.
- The AVAS access processes (ZDD, ZDL) must be started.
- To create SQL statements, AVAS-QUER requires output files in SAM format, which must be assigned via specific file link names. Existing output files are overwritten. Depending on the desired format, AVAS-QUER expects the following file link names:
 - INSERT FORMAT: file link name \$AVSQUER
 - LOAD FORMAT:

Name of the table	File link name
caltab	\$AVSCAL
caldaytab	\$AVSCALD
caldaysymtab	\$AVSCDAY
nettab	\$AVSNET
netdoktab	\$AVSNETD
netformtab	\$AVSNETF
netpartab	\$AVSNETP
netsymtab	\$AVSNETS
netttexttab	\$AVSNETT
ordertab	\$AVSORD
orderdoktab	\$AVSORDD
orderpartab	\$AVSORDP
ordersymtab	\$AVSORDS
ordersturmtab	\$AVSORDT
ordertexttab	\$AVSORDX

5.1.2 Starting AVAS-QUER

The AVAS-QUER utility routine can be called in either of two ways:

- via the procedure AVS-QUER:

```
/CALL-PROCEDURE LIB=$AVAS.SYSPRC.AVAS.085,ELEM=AVS.QUER
```

- via the program AVAS.SYS.LOAD.QUER:

```
/START-PROGRAM FROM-FILE=*PHASE(LIBRARY=SYSPRG.AVAS.085.SYSTEM,
ELEMENT=AVAS.SYS.LOAD.QUER)
```

If AVAS-QUER is called via the procedure, the link names are automatically assigned.

Structure of the procedure

```
/BEGIN-PROC LOG=N,PAR=YES(PROC-PAR=(&SYSVER=085,&EXEC=START, -
/ &SYSCMD=*PRIMARY,&SPPR=258,&SPSE=129, -
/ &SYSLNK=SYSLNK.AVAS., -
/ &AVUSERID=, -
/ &FORMAT=, -
/ &AUSGABE=, -
/ &CAL=, -
/ &CDAYSYM=, -
/ &CDAYTAB=, -
/ &NET=, -
/ &NETDOK=, -
/ &NETFORM=, -
/ &NETPAR=, -
/ &NETSYM=, -
/ &NETTXT=, -
/ &ORDER=, -
/ &ORDERDOK=, -
/ &ORDERPAR=, -
/ &ORDERSYM=, -
/ &ORDERTUR=, -
/ &ORDERTXT=, -
/ &SYS=, -
/ &TSTRUC=STD,
/ &SYSPAR=),ESC-CHAR=C'&')
/REMARK OUTPUT FORMAT (LOAD / INSERT): &FORMAT (1)
/SKIP-COM TO-LABEL=FA&FORMAT
/.FAINSERT REMARK
/.FAI REMARK
/REMARK NAME OF OUTPUT FILE: &OUTPUT (2)
/SET-FILE-LINK LINK=$AVSQUER,F-NAME=&OUTPUT
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&OUTPUT,SUPPORT=*ANY-DISK(SPACE= -
/ *RELATIVE(PRIMARY-ALLOCATION=&SPPR,SECONDARY-ALLOCATION=&SPSE))
```

```

/SKIP-COM TO-LABEL=FA#ENDE
/.FALOAD REMARK
/.FAL REMARK
/REMARK NAME OF OUTPUT FILE FOR CALDAYSMTAB: &CDAYSYM
/SET-FILE-LINK LINK=$AVSCDAY,F-NAME=&CDAYSYM
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&CDAYSYM,SUPPORT=*ANY-DISK(SPACE= -
/ *RELATIVE(PRIMARY-ALLOCATION=&SPPR,SECONDARY-ALLOCATION=&SPSE))
/REMARK NAME OF OUTPUT FILE FOR NETTAB: &NET
/SET-FILE-LINK LINK=$AVSNET,F-NAME=&NET
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&NET,SUPPORT=*ANY-DISK(SPACE= -
/ *RELATIVE(PRIMARY-ALLOCATION=&SPPR,SECONDARY-ALLOCATION=&SPSE))
/REMARK NAME OF OUTPUT FILE FOR NETDOKTAB: &NETDOK
/SET-FILE-LINK LINK=$AVSNETD,F-NAME=&NETDOK
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&NETDOK,SUPPORT=*ANY-DISK(SPACE= -
/ *RELATIVE(PRIMARY-ALLOCATION=&SPPR,SECONDARY-ALLOCATION=&SPSE))
/REMARK NAME OF OUTPUT FILE FOR NETSYMTAB: &NETSYM
/SET-FILE-LINK LINK=$AVSNETS,F-NAME=&NETSYM
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&NETSYM,SUPPORT=*ANY-DISK(SPACE= -
/ *RELATIVE(PRIMARY-ALLOCATION=&SPPR,SECONDARY-ALLOCATION=&SPSE))
/REMARK NAME OF OUTPUT FILE FOR NETTEXTTAB: &NETTXT
/SET-FILE-LINK LINK=$AVSNETT,F-NAME=&NETTXT
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&NETTXT,SUPPORT=*ANY-DISK(SPACE= -
/ *RELATIVE(PRIMARY-ALLOCATION=&SPPR,SECONDARY-ALLOCATION=&SPSE))
/REMARK NAME OF OUTPUT FILE FOR ORDERTAB: &ORDER
/SET-FILE-LINK LINK=$AVSORD,F-NAME=&ORDER
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&ORDER,SUPPORT=*ANY-DISK(SPACE= -
/ *RELATIVE(PRIMARY-ALLOCATION=&SPPR,SECONDARY-ALLOCATION=&SPSE))
/REMARK NAME OF OUTPUT FILE FOR ORDERDOKTAB: &ORDERDOK
/SET-FILE-LINK LINK=$AVSORDD,F-NAME=&ORDERDOK
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&ORDERDOK,SUPPORT=*ANY-DISK(SPACE= -
/ *RELATIVE(PRIMARY-ALLOCATION=&SPPR,SECONDARY-ALLOCATION=&SPSE))
/REMARK NAME OF OUTPUT FILE FOR ORDERSYMTAB: &ORDERSYM
/SET-FILE-LINK LINK=$AVSORDS,F-NAME=&ORDERSYM
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&ORDERSYM,SUPPORT=*ANY-DISK(SPACE= -
/ *RELATIVE(PRIMARY-ALLOCATION=&SPPR,SECONDARY-ALLOCATION=&SPSE))
/REMARK NAME OF OUTPUT FILE FOR ORDERSTURNTAB: &ORDERTUR
/SET-FILE-LINK LINK=$AVSORDT,F-NAME=&ORDERTUR
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&ORDERTUR,SUPPORT=*ANY-DISK(SPACE= -
/ *RELATIVE(PRIMARY-ALLOCATION=&SPPR,SECONDARY-ALLOCATION=&SPSE))
/REMARK NAME OF OUTPUT FILE FOR ORDETEXTTAB: &ORDERTXT
/SET-FILE-LINK LINK=$AVSORDX,F-NAME=&ORDERTXT
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&ORDERTXT,SUPPORT=*ANY-DISK(SPACE= -
/ *RELATIVE(PRIMARY-ALLOCATION=&SPPR,SECONDARY-ALLOCATION=&SPSE))
/REMARK TABLE-STRUCTURE (STD/EXTENDED): &TSTRUC _____ (3)
/SKIP-COM TO-LABEL=FAT&STRUC
/.FATEXTENDED REMARK
/.FATEXT REMARK
/.FATE REMARK

```

```

/REMARK NAME OF OUTPUT FILE FOR: &CAL
/SET-FILE-LINK LINK=$AVSCAL,F-NAME=&CAL
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&CAL,SUPPORT=*ANY-DISK(SPACE= -
/ *RELATIVE(PRIMARY-ALLOCATION=&SPPR,SECONDARY-ALLOCATION=&SPSE))
/REMARK NAME OF OUTPUT FILE FOR CALDAYTAB: &CDAYTAB
/SET-FILE-LINK LINK=$AVSCALD,F-NAME=&CDAYTAB
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&CDAYTAB,SUPPORT=*ANY-DISK(SPACE= -
/ *RELATIVE(PRIMARY-ALLOCATION=&SPPR,SECONDARY-ALLOCATION=&SPSE))
/REMARK NAME OF OUTPUT FILE FOR NETFORMTAB: &NETFORM
/SET-FILE-LINK LINK=$AVSNETF,F-NAME=&NETFORM
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&NETFORM,SUPPORT=*ANY-DISK(SPACE= -
/ *RELATIVE(PRIMARY-ALLOCATION=&SPPR,SECONDARY-ALLOCATION=&SPSE))
/REMARK NAME OF OUTPUT FILE FOR NETFORMMTAB: &NETFORM
/SET-FILE-LINK LINK=$AVSNETP,F-NAME=&NETPAR
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&NETPAR,SUPPORT=*ANY-DISK(SPACE= -
/ *RELATIVE(PRIMARY-ALLOCATION=&SPPR,SECONDARY-ALLOCATION=&SPSE))
/REMARK NAME OF OUTPUT FILE FOR ORDERPARTAB: &ORDERPAR
/SET-FILE-LINK LINK=$AVSORDP,F-NAME=&ORDERPAR
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&ORDERPAR,SUPPORT=*ANY-DISK(SPACE= -
/ *RELATIVE(PRIMARY-ALLOCATION=&SPPR,SECONDARY-ALLOCATION=&SPSE))
/SKIP-COM TO-LABEL=FA#ENDE
/.FATSTD REMARK
/.FATS REMARK
/REMARK PROCESS SYSTEM PARAMETERS (YES/NO): &SYS _____ (4)
/SKIP-COM TO-LABEL=FAS&SYS
/.FASYES REMARK
/.FASY REMARK
/REMARK NAME OF OUTPUT FILE FOR CALTAB: &CAL
/SET-FILE-LINK LINK=$AVSCAL,F-NAME=&CAL
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&CAL,SUPPORT=*ANY-DISK(SPACE= -
/ *RELATIVE(PRIMARY-ALLOCATION=&SPPR,SECONDARY-ALLOCATION=&SPSE))
/.FASNO REMARK
/.FASN REMARK
/.FA#ENDE REMARK
/REMARK USERID, UNDER WHICH AVAS-QUER IS INSTALLED: &AVUSERID _____ (5)
/MODIFY-TERMINAL-OPTIONS OVERFLOW-CONTROL=NO
/SET-JOB-STEP
/SET-FILE-LINK F-NAME=&AVUSERID..&SYSLNK.&SYSVER,LINK=SYSLNK
/ASSIGN-SYSDTA TO-FILE=&SYSCMD
/&EXEC-PROG FROM-FILE=*PHASE(ELEM=AVAS.SYS.LOAD.QUER, -
/ LIB=$&AVUSERID..SYSPRG.AVAS.&SYSVER..SYSTEM) _____ (6)
/SET-JOB-STEP
/REMOVE-FILE-LINK SYSLNK
/ASSIGN-SYSDTA TO-FILE=*PRIMARY
/SET-JOB-STEP
/SKIP-COM TO-LABEL=FE&FORMAT
/.FEINSERT REMARK
/.FEI REMARK

```

```
/REMOVE-FILE-LINK $AVSQUER
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&OUTPUT,SUPPORT=*ANY-DISK(SPACE= -
/ *RELEASE(NUMBER-OF-PAGES=9999))
/SKIP-COM TO-LABEL=FE&ENDE
/.FELOAD REMARK
/.FEL REMARK
/REMOVE-FILE-LINK $AVSCDAY
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&CDAYSYM,SUPPORT=*ANY-DISK(SPACE= -
/ *RELEASE(NUMBER-OF-PAGES=9999))
/REMOVE-FILE-LINK $AVSNET
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&NET,SUPPORT=*ANY-DISK(SPACE= -
/ *RELEASE(NUMBER-OF-PAGES=9999))
/REMOVE-FILE-LINK $AVSNETD
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&NETDOK,SUPPORT=*ANY-DISK(SPACE= -
/ *RELEASE(NUMBER-OF-PAGES=9999))
/REMOVE-FILE-LINK $AVSNETS
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&NETSYM,SUPPORT=*ANY-DISK(SPACE= -
/ *RELEASE(NUMBER-OF-PAGES=9999))
/REMOVE-FILE-LINK $AVSNETT
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&NETTXT,SUPPORT=*ANY-DISK(SPACE= -
/ *RELEASE(NUMBER-OF-PAGES=9999))
/REMOVE-FILE-LINK $AVSORD
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&ORDER,SUPPORT=*ANY-DISK(SPACE= -
/ *RELEASE(NUMBER-OF-PAGES=9999))
/REMOVE-FILE-LINK $AVSORDD
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&ORDERDOK,SUPPORT=*ANY-DISK(SPACE= -
/ *RELEASE(NUMBER-OF-PAGES=9999))
/REMOVE-FILE-LINK $AVSORDS
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&ORDERSYM,SUPPORT=*ANY-DISK(SPACE= -
/ *RELEASE(NUMBER-OF-PAGES=9999))
/REMOVE-FILE-LINK $AVSORT
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&ORDERTUR,SUPPORT=*ANY-DISK(SPACE= -
/ *RELEASE(NUMBER-OF-PAGES=9999))
/REMOVE-FILE-LINK $AVSORDX
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&ORDERTXT,SUPPORT=*ANY-DISK(SPACE= -
/ *RELEASE(NUMBER-OF-PAGES=9999))
/REMARK SYSTEM PARAMETERS PROCESSED (YES/NO): &SYS
/SKIP-COM TO-LABEL=FET&STRUC
/.FETEXTENDED REMARK
/.FETEXT REMARK
/.FETE REMARK
/REMOVE-FILE-LINK $AVSCAL
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&CAL,SUPPORT=*ANY-DISK(SPACE= -
/ *RELEASE(NUMBER-OF-PAGES=9999))
/REMOVE-FILE-LINK $AVSCALD
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&CDAYTAB,SUPPORT=*ANY-DISK(SPACE= -
/ *RELEASE(NUMBER-OF-PAGES=9999))
/REMOVE-FILE-LINK $AVSNETF
```

```

/MODIFY-FILE-ATTRIBUTES FILE-NAME=&NETFORM,SUPPORT=*ANY-DISK(SPACE= -
/  *RELEASE(NUMBER-OF-PAGES=9999))
/REMOVE-FILE-LINK $AVSNETP
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&NETPAR,SUPPORT=*ANY-DISK(SPACE= -
/  *RELEASE(NUMBER-OF-PAGES=9999))
/REMOVE-FILE-LINK $AVSORDP
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&ORDERPAR,SUPPORT=*ANY-DISK(SPACE= -
/  *RELEASE(NUMBER-OF-PAGES=9999))
/SKIP-COM TO-LABEL=FE#ENDE
/.FETSTD REMARK
/.FETS REMARK
/SKIP-COM TO-LABEL=FES&SYS
/.FESYES REMARK
/.FESY REMARK
/REMOVE-FILE-LINK $AVSCAL
/MODIFY-FILE-ATTRIBUTES FILE-NAME=&CAL,SUPPORT=*ANY-DISK(SPACE= -
/  *RELEASE(NUMBER-OF-PAGES=9999))
/.FESNO REMARK
/.FESN REMARK
/.FE#ENDE REMARK
/SET-JOB-STEP
/MODIFY-TERMINAL-OPTIONS OVERFLOW-CONTROL=USER
/END-PROC

```

- (1) Specifies the format in which the output file(s) will be created. The following specifications are possible:
 "l" or "load": The output files are created in LOAD2 format.
 "i" or "insert": The output file is created in INSERT format.
- (2) Depending on the format specification, the system now queries either the name of the output file (in INSERT format) or the names of the output files for the individual database tables (in LOAD format). The assignments to the file link names are established.
- (3) Specifies whether the extended table structure is to be generated. This is the case when the TABLE-STRUCTURE=EXTENDED parameter is specified in the CREATE-FORMAT statement. The STD or EXT(ENDED) entry can be specified. The caldaytab, netformtab, netpartab and orderpartab tables are also created for the extended table structure.
- (4) Specifies whether the system parameters are to be processed. This is the case when the parameter SYSTEM-PARAMS=YES is specified in the SELECT-OBJECT statement. YES or NO can be specified.

- (5) Specifies the user ID under which AVAS is installed.
- (6) Starts AVAS.SYS.LOAD.QUER.

5.1.3 Entering statements

Once the utility routine has been started, the AVAS-QUER statement SIGNON and the standard SDF statements can be entered.

After SIGNON logs the user in, the CREATE-FORMAT statement is expected. This statement determines the format of the output file(s). If CREATE-FORMAT is not specified, the output data is created by default in "LOAD2 format" with the standard table structure.

The user can then access the AVAS data with the AVAS-QUER statement SELECT-OBJECT to select the desired AVAS objects.

It is possible to enter the statements SIGNON and SELECT-OBJECT several times.

The statements, their syntax and operands are described from [page 250](#) onward.

A sample file with AVAS-QUER statements is contained in the SYSPRC.AVAS.085 library in the form of the AVS.QUER.SYSCMD member.

5.1.4 Terminating AVAS-QUER

The AVAS-QUER utility routine is terminated with the statement //END.

5.1.5 Structure of the output file(s) created by AVAS-QUER

An AVAS-QUER run produces a selected subset of data from the AVAS data, which can be further processed in relational databases. This data is written to output file(s), which must be assigned by means of specific file link names (see [section “Prerequisites for starting AVAS-QUER” on page 241](#)).

The scope of the data which is output and consequently the structure of the tables can be set using the TABLE-STRUCTURE operand in the CREATE-FORMAT statement (see [page 252](#)):

- *STD (default value) generates the database structure without extensions.
- *EXTENDED supplies the extended database which contains all information required for defining the net and calendar structure in the AVAS-DIALOG statements CREATE-NET-DESCRIPTION and CREATE-CALENDAR.

AVAS-QUER takes the following into account:

- No records are written to the output file(s) for data fields without values (*NONE, blanks). This is, for example, the case when no symdat is assigned to a calendar day.
- If the document name specified for the net description is *STD, the following names will be entered in the “netdoktab” table:
 - bkdok: user group of the net
 - dokname: net name
- If the document name specified for the order description is *STD, the following names will be entered in “orderdoktab” table:
 - bkdok: user group of the net
 - dokname: netname.ordername
- All attributes of a database table are always specified in the output files. Blanks are assigned to attributes for which there are no values in the AVAS data (for example, the attribute “enterfile” in the table “ordertab” for a condition).

The user must transfer the output file(s) created in this way to the target system. There, he/she executes them or imports the data into the database. The data in the output file(s) provides values for the individual fields of the database (tables).

It is advisable to name the output files(s) with the extension “.sql”.

With the AVAS-QUER statement CREATE-FORMAT, the user determines the format of the data in the output file.

The following formats are available:

- **INSERT FORMAT:**

Creates an output file with SQL statements with the syntax:

```
INSERT INTO tablename (attributename1,...)
VALUES ("value for attribute1",...)
```

- **LOAD2 format:**

A separate file is created for each database table. The output file records are structured as follows:

```
"attributevalue1";"attributevalue2";"...."
```

In this format, the data can be imported directly into relational databases (such as Microsoft Access) by means of the appropriate database-specific statements.

The CREATE-HEADER-LINE operand can be used to request a header line in the following format in each table:

```
attributename1,attributename2,....
```

The default values of `attributenamei` correspond to the field names described below. The following applies for the attribute names:

- With the extended table structure (TABLE-STRUCTURE=*EXTENDED) the names are freely selectable. The maximum length is 32 characters.
- With the INSERT format the attribute name entries are always present. However, the names are also freely selectable if CREATE-HEADER-LINE=*YES(...) is specified for the extended table structure.

For example, the header for the `nettab` table in the default format is:

```
bknet;netname;pvs
```

Indicator showing availability of an assigned document in the DOCLIB

In order to determine whether a document is available in the DOCLIB, AVAS-QUER attempts to create an index for all user groups. If the user has not been given *-authorization for COPY-ELEMENT, AVAS-QUER will create this index only for the user's own user group, and will output message AVSQ028.

For nets or orders to which a document has been assigned from the system user group, it is therefore impossible to determine whether the document is available. In these cases "dokvorh" (the document availability indicator in the `netdoktab` and `orderdoktab` tables) will be set to "N".

It should be noted that in AVAS the *-authorization is only valid for libraries assigned to the user.

5.2 AVAS-QUER statements

This section describes the AVAS-QUER statements, their syntax and operands. They are described in the order in which AVAS-QUER expects them. An example follows the description.

Standard SDF statements

The following standard SDF statements can be entered in the course of an AVAS-QUER run. These statements are described in the manual "Introductory Guide to the SDF Dialog Interface" [9].

Standard SDF statement	Meaning
END	Terminate the utility routine
EXECUTE-SYSTEM-CMD	Execute a command during the program run
HELP-MSG-INFORMATION	Output a system message to SYSOUT
HOLD-PROGRAM	Interrupt a program to enter commands
MODIFY-SDF-OPTIONS	Activate or deactivate the user syntax file and change the SDF settings
REMARK	Comment on statement sequences
RESET-INPUT-DEFAULTS	Reset the task-specific default values
RESTORE-SDF-INPUT	Display previous entry
SHOW-INPUT-DEFAULTS	Display the task-specific default values
SHOW-INPUT-HISTORY	Output saved inputs to SYSOUT
SHOW-SDF-OPTIONS	Display the SDF settings
SHOW-STMT	Output the syntax description of a statement
STEP	Define a reentry point
WRITE-TEXT	Output a particular test to SYSOUT or SYSLST

SIGNON – Log in to AVAS system

The SIGNON statement is used to log in to the AVAS system. SIGNON must be the first statement.

SIGNON
AVAS-USER-ID=avuser ,PASSWORD=password ,AVAS-SYSTEM-ID=string

AVAS-USER-ID= avuser

Identifier of the AVAS user (maximum 8 characters).

PASSWORD=password

Password of the AVAS user (maximum 8 characters).

In guided dialog, the password is not shown on the screen.

AVAS-SYSTEM-ID=string

Name of the AVAS system (7 characters) with which the user wants to work.

CREATE-FORMAT – Select output file format

With the CREATE-FORMAT statement, the user selects the format for the output file(s) to be created. CREATE-FORMAT is expected as the second statement (after SIGNON). If specified after SELECT-OBJECT, CREATE-FORMAT is ignored.

If CREATE-FORMAT is not specified, the format LOAD2 is used by default.

```
CREATE-FORMAT
```

```
FORMAT-NAME=INSERT / LOAD2
```

```
,TABLE-STRUCTURE=*STD / EXTENDED
```

```
,CREATE-HEADER-LINE=*NO / *YES(...)
```

```
*YES(...)
```

```
  HEADER-CALTAB='bkcal;calname;first_cal_date;last_cal_date;system_symdat
```

```
           ;type_mon;type_tue;type_wed;type_thu;type_fri;type_sat;type_sun
```

```
           ;every_day;work_day;day_of_month;last_work' / <text 1..544>
```

```
,HEADER-CALDAYTAB='calname;datum;weekday;typ;to_last_work;work_wd_cnt;work_wd_nr
```

```
           ;work_count' / <text 1..256>
```

```
,HEADER-CALDAYSYMTAB='calname;datum;weekday;typ;symdat' / <text 1..160>
```

```
,HEADER-NETTAB='bknet;netname;pvs;calname;selectturnus;selplantyp;nettyp;run_control_system
```

```
           ;net_user;net_account;net_password;net_class;net_log;net_parameter' /
```

```
           <text 1..448>
```

```
,HEADER-NETDOKTAB='bknet;netname;bkdok;dokname;dokvorh' / <text 1..160>
```

```
,HEADER-NETFORMTAB='bknet;netname;format_name;format_text' / <text 1..128>
```

```
,HEADER-NETPARTAB='bknet;netname;user_par_type;user_par_file' / <text 1..128>
```

```
,HEADER-NETSYMTAB='bknet;netname;symdat;starttime;lateststart;delay_solution;lifetime' /
```

```
           <text 1..224>
```

```
,HEADER-NETTEXTTAB='bknet;netname;nettext' / <text 1..96>
```

```
,HEADER-ORDERTAB='bknet;netname;indexnr;bkorder;ordename;function;typ
```

```
           ;pvs;jvname;enterfile;cond_index;cond_bknet;cond_netname;sync_indexnr
```

```
           ;rest_indexnr1;rest_name1;rest_typ1;rest_automatic1
```

```
           ;rest_indexnr2;rest_name2;rest_typ2;rest_automatic2
```

```
           ;rest_indexnr3;rest_name3;rest_typ3;rest_automatic3
```

```
           ;sel_rest_variant;enter_params;bs2_userid;bs2_account;bs2_password
```

```
           ;bs2_class;bs2_log;bs2_parameter;server_name;server_password
```

```
           ;value_position;value_length ;java_password;value_typ;value;value_logic
```

```
           ;error_typ;error_value;direction;partner_name;remote;local_file;remote_file
```

```
           ;remote_server_adm;ft_parameter' / <text 1..1632>
```

```
,HEADER-ORDERDOKTAB='bknet;netname;indexnr;bkorder;ordername;function;typ
;bkdok;dokname;dokvorh' / <text 1..320>
,HEADER-ORDERPARTAB='bknet;netname;indexnr;bkorder;ordername;function;typ;
;user_par_type;user_par_file' / <text 1..288>
,HEADER-ORDERSYMTAB='bknet;netname;indexnr;bkorder;ordername;function;typ
;symdat;lateststart;delay_solution;lifetime' / <text 1..352>
,HEADER-ODERSTURNTAB='bknet;netname;indexnr;bkorder;ordername;function;typ
;selectturnus' / <text 1..256>
,HEADER-ODERTEXTTAB='bknet;netname;indexnr;bkorder;ordername;function;typ
;ordertext' / <text 1..256>
```

FORMAT-NAME=

Specifies the format of the output file (see [page 248](#)).

FORMAT-NAME=INSERT

The output file contains the AVAS data in SQL INSERT statements.

FORMAT-NAME=LOAD2

The output files contains the AVAS data in the general LOAD2 format.

TABLE-STRUCTURE=

Determines the number and scope of the tables.

TABLE-STRUCTURE=*STD

The standard table structure (see [page 262](#)) is generated (format compatible with AVAS < V8.0A).

TABLE-STRUCTURE=EXTENDED

The extended table structure (see [page 267](#)) is generated. This contains all net and calendar data which is required by AVAS-DIALOG to generate the corresponding objects.

CREATE-HEADER-LINE=

Determines whether a header line with the attribute names is output for tables in LOAD2 format.

In INSERT format a header line with the attribute names is always generated. However, the attribute names can be modified using CREATE-HEADER_LINE=*YES(...).

CREATE-HEADER-LINE=*NO

Only in LOAD2 format are the tables generated without a header line.

In INSERT format the header lines contain the default names (as in AVAS < V8.0A).

CREATE-HEADER-LINE=*YES(...)

The tables are generated with header lines. Which attribute names are to be assigned in the header line is defined for each table in the HEADER-<table-name> operand. The default names are always predefined. Depending on the table structure generated, you must take note of the following:

- The following applies for TABLE-STRUCTURE=*EXTENDED:
The attribute names can be selected freely in both LOAD2 and INSERT format.
- The following applies for TABLE-STRUCTURE=*STD:
The names specified by the caller are ignored and only the predefined default names are entered.

Overview of the operands with the associated predefined default names:

Operand	Predefined default names
HEADER-CALTAB	'bkcal;calname;first_cal_date;last_cal_date;system_symdats ;type_mon;type_tue;type_wed;type_thu;type_fri;type_sat ;type_sun;every_day;work_day;day_of_month;work_month ;last_work'
HEADER-CALDAYTAB	'calname;datum;weekday;typ;to_last_work;work_wd_cnt ;work_wd_nr;work_count'
HEADER-CALDAYSYMTAB	'calname;datum;weekday;typ;symdat'
HEADER-NETTAB	'bknet;netname;pvs;calname;selectturnus;selplantyp;nettyp ;run_control_system;net_user;net_account;net_password ;net_class;net_log;net_parameter'
HEADER-NETDOKTAB	'bknet;netname;bkdok;dokname;dokvorh'
HEADER-NETFORMTAB	'bknet;netname;format_name;format_text'
HEADER-NETPARTAB	'bknet;netname;user_par_type;user_par_file'
HEADER-NETSYMTAB	'bknet;netname;symdat;starttime;lateststart;delay_solution ;lifetime'
HEADER-NETTEXTTAB	'bknet;netname;nettext'
HEADER-ORDERTAB	'bknet;netname;indexnr;bkorder;ordename;function;typ;pvs ;jvname;enterfile;cond_index;cond_bknet;cond_netname ;sync_indexnr;rest_indexnr1;rest_name1;rest_typ1 ;rest_automatic1;rest_indexnr2;rest_name2;rest_typ2 ;rest_automatic2;rest_indexnr3;rest_name3;rest_typ3 ;rest_automatic3;sel_rest_variant;enter_params;bs2_userid ;bs2_account;bs2_password;bs2_class;bs2_log;bs2_parameter ;server_name;server_password;value_position;value_length ;jva_password;value_typ;value;value_logic;error_typ;error_value ;direction;partner_name;remote;local_file;remote_file ;remote_server_adm;ft_parameter'

Operand	Predefined default names
HEADER-ORDERDOKTAB	'bknet;netname;indexnr;bkorder;ordername;function;typ;bkdok;dokname;dokvorh'
HEADER-ORDERPARTAB	'bknet;netname;indexnr;bkorder;ordername;function;typ;user_par_type;user_par_file'
HEADER-ORDERSYMTAB	'bknet;netname;indexnr;bkorder;ordername;function;typ;symdat;lateststart;delay_solution;lifetime'
HEADER-ODERSTURNTAB	'bknet;netname;indexnr;bkorder;ordername;function;typ;selecturnus'
HEADER-ODERTEXTTAB	'bknet;netname;indexnr;bkorder;ordername;function;typ;ordertext'

Attribute names can be selected freely in the various tables within the following value ranges:

Operand	Value range
HEADER-CALTAB	<text 1..544>
HEADER-CALDAYTAB	<text 1..256>
HEADER-CALDAYSYMTAB	<text 1..160>
HEADER-NETTAB	<text 1..448>
HEADER-NETDOKTAB	<text 1..160>
HEADER-NETFORMTAB	<text 1..128>
HEADER-NETPARTAB	<text 1..128>
HEADER-NETSYMTAB	<text 1..224>
HEADER-NETTEXTTAB	<text 1..96>
HEADER-ORDERTAB	<text 1..1632>
HEADER-ORDERDOKTAB	<text 1..320>
HEADER-ORDERPARTAB	<text 1..288>
HEADER-ORDERSYMTAB	<text 1..352>
HEADER-ODERSTURNTAB	<text 1..256>
HEADER-ODERTEXTTAB	<text 1..256>

SELECT-OBJECTS – Select AVAS objects

With the SELECT-OBJECT statement, the user selects AVAS objects for processing. The user must have COPY ELEMENT authorization for these objects.

Note

If an AVAS element is selected several times (either by direct specification of its name or by partially qualified specifications), it will be written to the output file several times. This is also true for system parameters, if SYSTEM-PARAMS=[*]YES is specified.

SELECT-OBJECT
NET-NAME= <u>*NO</u> / netname ,CALENDAR-NAME= <u>*NO</u> / calendar ,SYSTEM-PARAMS=[*] <u>NO</u> / [*]YES

NET-NAME=

Specifies whether AVAS nets are to be selected.

NET-NAME=*NO

No AVAS nets are to be selected.

If all nets ending with NO are to be output, NET-NAME must be specified with the user group (NET-NAME=\$bk_*NO).

NET-NAME=netname

Name of the AVAS net (written as [\$ug_]netname) used to select the objects. The net description is entered in the net library under this name. The net name can also be specified in partially qualified form.

CALENDAR-NAME=

Specifies whether AVAS calendars are to be selected.

CALENDAR-NAME=*NO

No AVAS calendars are to be selected.

CALENDAR-NAME=calendar

Name of the AVAS calendar to be selected. The calendar description is entered in the calendar library under this name. The calendar name can also be specified in partially qualified form.

SYSTEM-PARAMS=

Specifies whether the assignment of calendars to user groups is to be selected from the AVAS system parameters.

SYSTEM-PARAMS=*NO

The assignment of calendars to user groups is not to be copied.

SYSTEM-PARAMS=*YES

The assignment of calendars to user groups is to be copied from the system parameter file into the output file.

The following applies for the extended table structure (see [page 267](#)):

When a calendar is also selected via the CALENDAR-NAME, it is initially copied into the output file **without** any reference to the user group. However, on account of this specification an additional entry **with** a reference to the user group is made.

Example

The AVAS-QUER utility routine is called via the procedure. The output file "output" is to be created in INSERT format. It is then to be made available in the UNIX system with the name "insert.sql".

```

/ CALL-PROCEDURE LIB=$AVASS.SYSPRC.AVAS.085.ELE=AVS.QUER _____ (1)
/ REMARK OUTPUT FORMAT (LOAD / INSERT): &FORMAT
&FORMAT=insert _____ (2)
/ REMARK NAME OF THE OUTPUT FILE: %OUTPUT
&OUTPUT=avas.quer.output _____ (3)
/ REMARK USERID, UNDER WHICH AVAS-QUER IS INSTALLED: &AVUSERID
&AVUSERID=AVAS _____ (4)
% SSM3034 GIVEN SYSTEM FILE ALREADY 'PRIMARY'. PROCESSING CONTINUED

% BLS0500 PROGRAM 'AVASQUER', VERSION 'V8.5A' OF '2010-12-20' LOADED
% BLS0552 COPYRIGHT (C) FUJITSU TECHNOLOGY SOLUTIONS GMBH 2010. ALL
RIGHTS RESERVED _____ (5)
% AVS2201 START 'AVASQUER': 'V08.5A00': '2010-12-20/12:34:04' _____ (6)
% AVS2800 PLEASE ENTER SIGNON DATA _____ (7)
//signon avas-user-id=test,password=test,avas-system-id=sysidsy _____ (8)
//create-format format-name=insert _____ (9)
//select-object net-name=net001,calendar-name=examp*, -
//      system-params=yes _____ (10)
/end _____ (11)
% AVS2310 SYSTEM 'AVASQUER': 'V08.5A00' TERMINATED NORMALLY
% SSM3034 GIVEN SYSTEM FILE ALREADY 'PRIMARY'. PROCESSING CONTINUED

```

in the target system (here UNIX platform:)

```

$ ncopy <partner computer>!output insert.sql <userid,acc,password> _____ (12)
ncopy: Received job insert.sql(07997) started
ncopy: Received job insert.sql(07997) executed successful
$ isql _____ (13)

```

- (1) Starts the procedure
- (2) Specifies the desired format
- (3) Name of the output file to be created
- (4) Specifies the user ID
- (5) Calls AVAS.SYS.LOAD.QUER
- (6) Start message from AVAS.SYS.LOAD.QUER
- (7) Message prompting entry of signon data
- (8) Log in to AVAS-QUER via SIGNON
- (9) Specify the format for the output file
- (10) Select objects
- (11) Terminate AVAS-QUER
- (12) In the target system:
Fetch and simultaneously rename the output file "output" created by AVAS-QUER. The partner computer and the access authorizations for the partner computer must be entered in the ncopy command. The password is entered without quotes. After the ncopy command is terminated, the file is available in the target system with the name "insert.sql". INFORMIX requires the extension ".sql" as a criterion for file handling.
- (13) The INFORMIX database program is called with "isql".
The database-specific commands and statements can then be entered:
 - INFORMIX-SQL: ... SQL-Dialog ...
Conduct an interactive dialog using SQL statements.
 - SELECTION DATABASE >> avsqwer
Select a database or enter names. Continue with RETURN.
 - SQL-Dialog: ... File .
Load, save or delete a statement file.
 - FILE: Load ...
Read in a dialog statement from a statement file.
 - SELECTION >> insert
Select a statement file or enter the name. Continue with RETURN.
 - SQL-Dialog: . START ...
Execute a dialog statement.
 - SQL-Dialog: New ...
Enter a new dialog statement.
 - SQL-Dialog: . START ...
Execute a dialog statement.
 - ...

5.3 Data storage for AVAS-QUER

5.3.1 Creating the database

The database used to create these statements is called “avsquer”. This name can be modified by the user. The name of the database is contained in the first two lines of the file:

```
CREATE DATABASE avsquer; DATABASE avsquer;
```

The user cannot modify the subsequent statements for creating the database tables.

The SQL statements can be executed, for example, with INFORMIX isql (SQL dialog menu, items File and START). The item File prompts isql to list the files with an “.sql” extension, for the user to make a selection.

5.3.2 Importing the data

The user must transfer the output file(s) created by AVAS-QUER to the desired target system via file transfer.

Importing the data into the INFORMIX database

INSERT FORMAT:

The SQL statements can be executed, for example, with INFORMIX isql (SQL dialog menu items, File, Load and START).

Importing the data into other databases

This concerns the output files created in LOAD2 format.

Depending on the database system used, it may be necessary to define the database tables beforehand. The data is entered in the database with the database-specific statements. The user must modify the output files created by AVAS-QUER if his/her database system supports another format.

5.3.3 Structure of the database

The [figure 5](#) illustrates the standard structure of the database with its tables; the extended structure is illustrated in [figure 6](#). The AVAS objects and their relationships (cross-references) are stored in these tables.

The structural elements of a net are called “order”.

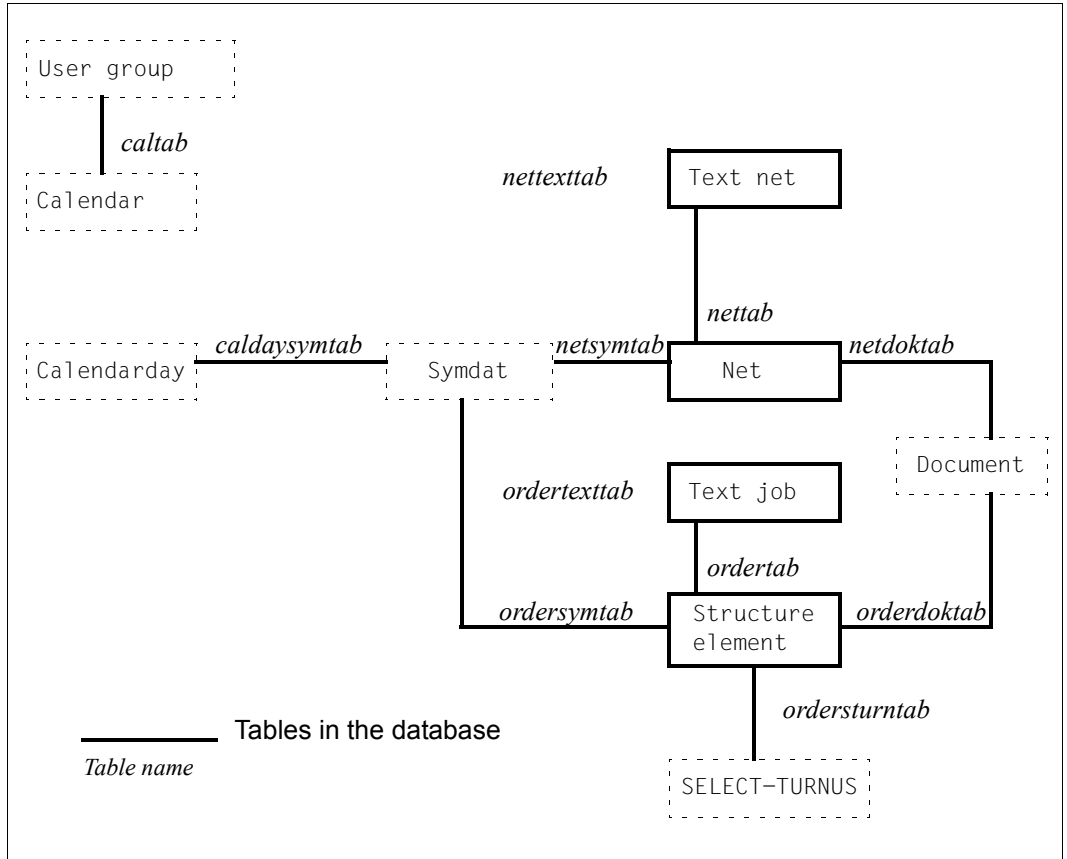


Figure 5: Standard structure of the database in the target system

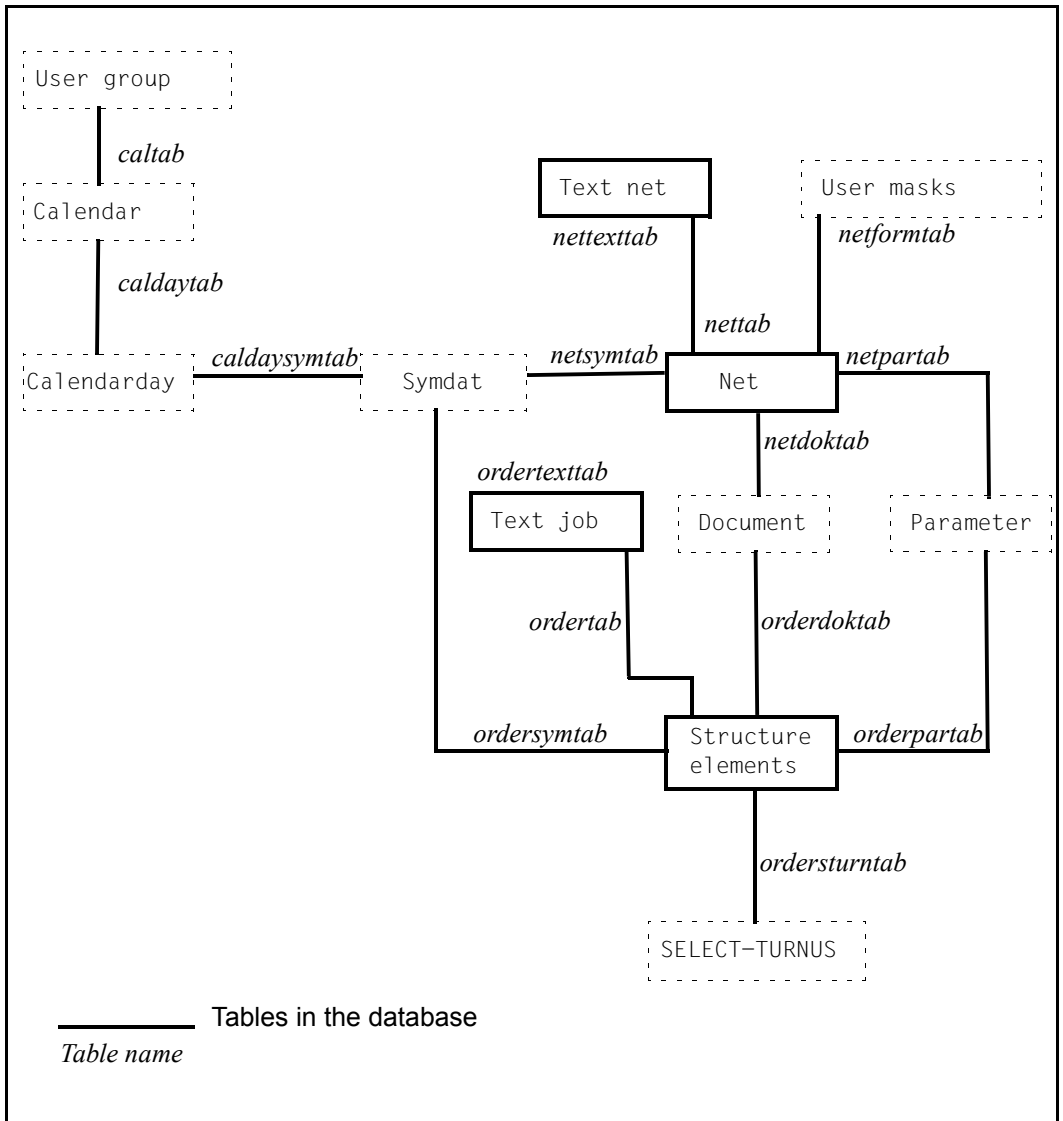


Figure 6: Extended structure of the database in the target system

5.3.4 Structure of the database tables

5.3.4.1 Default format

caltab

The caltab table contains the assignment of calendars to user groups.

bkcal	char(4)	Name of user group
calname	char(20)	Calendar name

caldaysymtab

The caldaysymtab table contains the assignment of symdat to calendar days.

calname	char(20)	Calendar name
datum	char(8)	Date of the calendar day (YYYYMMDD)
symdat	char(20)	Name of the symbolic date (symdat)
weekday	char(3)	Name of the weekday
typ	char(4)	Typ of the calendar day (WORK, NWRK, FREE)

nettab

The nettab table contains the assignment of pubsets to nets.

bknet	char(4)	User group of the net
netname	char(12)	Net name
pvs	char(54)	Assigned pubset or job variable name

netdoktab

The netdoktab table contains the assignment of documents to nets.

bknet	char(4)	User group of the net
netname	char(12)	Net name
bkdok	char(4)	User group of the document
dokname	char(37)	Document name
dokvorh	char(1)	Indicator (Y/N), showing whether the document is available in the DOCLIB.

netsymtab

The netsymtab table contains the assignment of symdats to nets.

bknet	char(4)	User group of the net
netname	char(12)	Net name
symdat	char(20)	Name of the symbolic date (Symdat) with relative specifications (+/- nn: 1<nn<99)
starttime	char(6)	Symbolic name of the start time
lateststart	char(7)	Symbolic name of the latest start time

netttextab

The netttextab table contains the assignment of texts for the net descriptions to the nets.

bknet	char(4)	User group of the net
netname	char(12)	Net name
netttext	char(120)	Net description text

ordertab

The ordertab table contains the assignment of pubsets to jobs and the assignment of jobs to nets, plus the dependencies on any nets, jobs, condition values, resources or job variables.

bknet	char(4)	User group of the net
netname	char(12)	Net name
indexnr	char(3)	Index level
bkorder	char(4)	User group of the structure element. If no user group is specified for a JOB or NET condition, the user group of the net is entered.
ordername	char(24)	Structure element name.
function	char(1)	Structure element function (FU): J (job), P (S procedure), S (Start), C (compare), W (wait), M (modify), D (delete), A (add)
typ	char(3)	Structure element type: Job: MOD, STD, EXT, EXX Condition: NET, JOB, VAL, RES, JVA, WAIT, TIM Start job: NET

pvs	char(54)	Pubset assigned to the job, or job variable containing the pubset. Pubset is written in quotes to distinguish between pubset and JV (for function=J/P)
jvname	char(54)	Name of the job variable for type=JVA
enterfile	char(54)	Name of the enter file for type=EXT
cond_index	char(3)	Index level of the job on which there is a dependency (for type=JOB)
cond_bknet	char(4)	User group of the net on which there is a dependency (for type=JOB or NET). If no user group is specified, that of the net itself will be recorded here.
cond_netname	char(12)	Name of the net on which there is a dependency
sync_indexnr	char(3)	SYNC-INDEX
rest_indexnr1	char(3)	RESTART-INDEX from restart variant 1
rest_name1	char(30)	RESTART-NAME from restart variant 1
rest_typ1	char(10)	RESTART-TYPE from restart variant 1
rest_automatic1	char(3)	AUTOMATIC from restart variant 1
rest_indexnr2	char(3)	RESTART-INDEX from restart variant 2
rest_name2	char(30)	RESTART-NAME from restart variant 2
rest_typ2	char(10)	RESTART-TYPE from restart variant 2
rest_automatic2	char(3)	AUTOMATIC from restart variant 2
rest_indexnr3	char(3)	RESTART-INDEX from restart variant 3
rest_name3	char(30)	RESTART-NAME from restart variant 3
rest_typ3	char(10)	RESTART-TYPE from restart variant 3
rest_automatic3	char(3)	AUTOMATIC from restart variant 3
bs2_userid	char(8)	Name of an ID in BS2000 In the case of enter-params=NET and typ=STD / MOD / EXT / EXX from the net description
bs2_account	char(8)	Account number in BS2000 In the case of enter-params=NET and typ=STD / MOD / EXT / EXX from the net description
value_typ	char(1)	Value type C = text of a COND-VALUE O = text of a OCCURE-VALUE E = text of a ERROR-VALUE

value	char(256)	Depending on value_typ: Contents of a job variable for typ=JVA Contents of the description for typ=VAL Status text(s) for typ=NET / JOB / RES If there are several status values, they are output with commas as separators. If OCCURE-VALUE and ERROR-VALUE both exist, two records are output to the ordertab.
value_logic	char(2)	Logical operand for the condition which the value of the job variable must satisfy. Possible operand values: =, <, >, <=, >=, <>

orderdoktab

The orderdoktab table contains the assignment of documents to structure elements.

bknet	char(4)	User group of the net
netname	char(12)	Net name
indexnr	char(3)	Index level
bkorder	char(4)	User group of the order
ordername	char(24)	Name of the structure element
bkdok	char(4)	User group of the document
dokname	char(37)	Document name
dokvorh	char(1)	Indicator (J/N), showing whether the document is present in the DOCLIB.

ordersymtab

The ordersymtab table contains the assignment of symdat to structure elements.

bknet	char(4)	User group of the net
netname	char(12)	Net name
indexnr	char(3)	Index level
bkorder	char(4)	User group of the structure element
ordername	char(24)	Name of the structure element
symdat	char(20)	Name of the symbolic date (symdat)

ordersturmtab

The ordersturmtab table contains the assignment of SELECT-TURNUS to structure elements.

bknet	char(4)	User group of the net
netname	char(12)	Net name
indexnr	char(3)	Index level
bkorder	char(4)	User group of the structure element
ordername	char(24)	Name of the structure element
selectturnus	char(1)	SELECT-TURNUS

ordertexttab

The ordertexttab table contains the assignment of net description texts to the structure elements of the nets.

bknet	char(4)	User group of the net
netname	char(12)	Net name
indexnr	char(3)	Index level
bkorder	char(4)	User group of the structure element
ordername	char(24)	Name of the structure element
ordertext	char(120)	Order description text
function	char(1)	Function (FU) of the structure element
typ	char(3)	Type of the structure element

5.3.4.2 Extended format

caltab

The caltab table contains the assignment of the calendars to user groups and general calendar data.

bkcal	char(4)	Name of user group
calname	char(20)	CALENDAR-NAME
first_cal_date	char(8)	FIRST-CALENDAR-DATE (YYYYMMDD)
last_cal_date	char(8)	LAST-CALENDAR-DATE (YYYYMMDD)
system_symdats	char(5)	SYMDAT-NAME (*NONE, *STD, *ALL)
type_mon	char(4)	Type Monday (WORK, NWRK, FREE)
type_tue	char(4)	Type Tuesday (WORK, NWRK, FREE)
type_wed	char(4)	Type Wednesday (WORK, NWRK, FREE)
type_thu	char(4)	Type Thursday (WORK, NWRK, FREE)
type_fri	char(4)	Type Friday (WORK, NWRK, FREE)
type_sat	char(4)	Type Saturday (WORK, NWRK, FREE)
type_sun	char(4)	Type Sunday (WORK, NWRK, FREE)
every_day	char(8)	Name for "daily" (TGL)
work_day	char(8)	Name for "work day" (WT)
day_of_month	char(4)	Name for "current day of month" (K)
work_month	char(4)	Name for "current work day of month" (A)
last_work	char(8)	Name for "last work day of month" (ULTIMO)

caldaytab

The caldaytab table contains the descriptive data of the symdat's for calendar days.

calname	char(20)	CALENDAR-NAME
datum	char(8)	Date of the calendar day (YYYYMMDD)
symdat	char(20)	Name of the symbolic date (symdat)
weekday	char(3)	Name of the weekday (MON, TUE, WED, THU, FRI, SAT, SUN)
typ	char(4)	Type of calendar day (WORK, NWRK, FREE)
to_last_work	char(2)	Distance to the last work day (5 or less)
work_wd_cnt	char(1)	Seq. No. of work day with this name in the month
work_wd_nr	char(1)	Number of work days with this name in the month
work_count	char(2)	Seq. No. of this work day in the month

caldaysymtab

The caldaysymtab table contains the assignment of the symdat's to calendar days.

calname	char(20)	CALENDAR-NAME
datum	char(8)	Date of the calendar day (YYYYMMDD)
weekday	char(3)	Name of the weekday (MON, TUE, WED, THU, FRI, SAT, SUN)
typ	char(4)	Type of calendar day (WORK, NWRK, FREE)
symdat	char(20)	Name of the symbolic date (symdat)

nettab

The nettab table contains the general net description data of the dialog level OBJECT=NET.

bknet	char(4)	User group of the net
netname	char(12)	NET-NAME
pvs	char(54)	NET-CAT
calname	char(20)	CALENDAR-NAME
selectturnus	char(1)	SELECT-TURNUS (1,2, ... 9)
selplantyp	char(4)	SELECT-PLAN-TYPE of the net (WORK, NWRK)
nettyp	char(1)	NET-TYPE (1, 2, 3)
run_control_system	char(8)	RUN-CONTROL-SYSTEM
net_user	char(8)	NET-USER
net_account	char(8)	NET-ACCOUNT
net_password	char(19)	NET-PASSWORD
net_class	char(8)	NET-CLASS
net_log	char(8)	NET-LOG
net_parameter	char(128)	NET-PARAMETER

netdoktab

The netdoktab table contains the assignment of documents to nets.

bknet	char(4)	User group of the net
netname	char(12)	NET-NAME
bkdok	char(4)	User group of the document
dokname	char(37)	NET-DOC
dokvorh	char(1)	Indicator showing whether the document is available in the DOCLIB (Y, N)

netformtab

The netformtab table contains the assignment of the user masks to nets.

bknet	char(4)	User group of the net
netname	char(12)	NET-NAME
format_name	char(8)	FORMAT-NAME
format_text	char(40)	FORMAT-TEXT

netpartab

The netpartab table contains the assignment of the parameter file to nets and the parameter file type.

bknet	char(4)	User group of the net
netname	char(12)	NET-NAME
user_par_type	char(3)	Parameter file type (LIB/SAM)
user_par_file	char(54)	USER-PAR-FILE

netsymtab

The netsymtab table contains the assignment of symdats to nets.

bknet	char(4)	User group of the net
netname	char(12)	NET-NAME
symdat	char(20)	Name of the symbolic date (symdat) with relative specifications (+/- nn: 1<nn<99)
starttime	char(8)	Start time of the net (hh.mm.ss)
lateststart	char(9)	LATEST-START (nnn.hh.mm)
delay_solution	char(8)	DELAY-SOLUTION (WAIT, START, IGNORE, CANCEL)
lifetime	char(9)	LIFE-TIME (nnn.hh.mm)

netttextab

The netttextab table contains the assignment of texts for the net description to nets.

bknet	char(4)	User group of the net
netname	char(12)	NET-NAME
netttext	char(120)	NET-TEXT

ordertab

The ordertab table contains the assignment of pubsets to jobs and the assignment of jobs to nets, plus the dependencies on any net, job, condition value, resource or job variable.

bknet	char(4)	User group of the net
netname	char(12)	NET-NAME
indexnr	char(3)	IND (001, ..., 999)
bkorder	char(4)	User group of the structure element. If no user group is specified for a JOB or NET condition, the user group of the net is entered.
ordername	char(24)	NAME
function	char(1)	FU J (job), P (S procedure), S (Start), C (compare), W (wait), M (modify), D (delete), A (add)
typ	char(3)	TYPE Job: MOD, STD, EXT, EXX Condition: NET, JOB, VAL, RES, JVA, WAIT, TIM Start job: NET
pvs	char(54)	JOB-CAT
jvname	char(54)	Name of the job variable for type=JVA
enterfile	char(54)	Name of the enter file for type=EXT
cond_index	char(3)	Index level of the job on which there is a dependency (for type=JOB)
cond_bknet	char(4)	User group of the net on which there is a dependency (for type=JOB or NET). If no user group is specified, that of the net itself will be recorded here.
cond_netname	char(12)	Name of the net on which there is a dependency.
sync_indexnr	char(3)	SYNC-INDEX
rest_indexnr1	char(3)	RESTART-INDEX from restart variant 1
rest_name1	char(30)	RESTART-NAME from restart variant 1
rest_typ1	char(10)	RESTART-TYPE from restart variant 1 (RESTART, NORMAL)
rest_automatic1	char(3)	AUTOMATIC from restart variant 1 (YES, NO)
rest_indexnr2	char(3)	RESTART-INDEX from restart variant 2
rest_name2	char(30)	RESTART-NAME from restart variant 2

rest_typ2	char(10)	RESTART-TYPE from restart variant 2 (RESTART, NORMAL)
rest_automatic2	char(3)	AUTOMATIC from restart variant 2 (YES, NO)
rest_indexnr3	char(3)	RESTART-INDEX from restart variant 3
rest_name3	char(3)	RESTART-NAME from restart variant 3
rest_typ3	char(10)	RESTART-TYPE from restart variant 3 (RESTART, NORMAL)
rest_automatic3	char(3)	AUTOMATIC from restart variant 3 (YES, NO)
sel_rest_variant	char(1)	SELECT-RESTART-VARIANT
enter_params	char(5)	ENTER-PARAMS (NET/LOGON)
bs2_userid	char(8)	USER
bs2_account	char(8)	JOB-ACCOUNT
bs2_password	char(19)	PASSWORD
bs2_class	char(8)	JOB-CLASS
bs2_log	char(8)	LOG
bs2_parameter	char(128)	JOB-PARAMETER
server_name	char(8)	SERVER-NAME
server_password	char(8)	SERVER-PASSWORD
value_position	char(3)	JVA-POSITION
value_length	char(3)	JVA-LENGTH
jva_password	char(11)	JVA-PASSWORD
value_typ	char(1)	Value type C = Text of a COND-VALUE O = Text of a OCCURE-VALUE
value	char(256)	Depending on value_typ: Contents of a job variable for typ=JVA (value_typ C) Contents of the description for typ=VAL (value_typ O) Status text(s) for typ=NET / JOB / RES (value_typ O) If there are several status values, they are output with commas as separators.
value_logic	char(2)	Logical operand for the condition which the value of the job variable must satisfy. Possible operand values: =, <, >, <=, >=, <>
error	char(1)	E = ERROR-VALUE is available
error_value	char(128)	ERROR-VALUE

direction	char(4)	TO/FROM: Transfer direction for FT request
partner-name	char(9)	FT request: Remote host
remote	char(7)	*BS2000/*MSP/*ANY: Typ des fernen Rechners
local_file	char(54)	FT request: Local file name
remote_file	char(54)	FT request: Remote file name
remote_transfer_adm	char(67)	FTAC access authorization or *PAR(user-id,account,'password')
ft_parameter	char(192)	Free parameters of the TRANSFER-FILE command

orderdoktab

The orderdoktab table contains the assignment of documents to structure elements.

bknet	char(4)	User group of the net
netname	char(12)	NET-NAME
indexnr	char(3)	IND
bkorder	char(4)	User group of the order
ordername	char(24)	Name of the structure element
function	char(1)	FU
typ	char(3)	TYPE
bkdok	char(4)	User group of the document
dokname	char(37)	JOB-DOC
dokvorh	char(1)	Indicator das anzeigt, showing whether the document is present in the. (A, N)

orderpartab

The orderpartab table contains the assignment of the parameter file to structure elements and the type of the parameter file.

bknet	char(4)	User group of the net
netname	char(12)	NET-NAME
indexnr	char(3)	IND
bkorder	char(4)	User group of the order
ordername	char(24)	Name of the structure element
function	char(1)	FU
typ	char(3)	TYPE
user_par_type	char(3)	Type of the parameter file (LIB / SAM)
user_par_file	char(54)	USER-PAR-FILE

ordersymtab

The ordersymtab table contains the assignment of symdat to structure elements.

bknet	char(4)	User group of the net
netname	char(12)	NET-NAME
indexnr	char(3)	IND
bkorder	char(4)	User group of the structure element
ordername	char(24)	Name of the structure element
function	char(1)	FU
typ	char(3)	TYPE
symdat	char(20)	Name of the symbolic date (symdat)
lateststart	char(7)	Symbolic name of the latest start time
delay_solution	char(8)	DELAY-SOLUTION
lifetime	char(7)	LIFE-TIME

ordersturntab

The ordersturntab table contains the assignment of SELECT-TURNUS to structure elements.

bknet	char(4)	User group of the net
netname	char(12)	NET-NAME
indexnr	char(3)	IND
bkorder	char(4)	User group of the structure element
ordername	char(24)	Name of the structure element
function	char(1)	FU
typ	char(3)	TYPE
selectturnus	char(1)	SELECT-TURNUS

ordertexttab

The ordertexttab table contains the assignment of net description texts to the structure elements of the nets.

bknet	char(4)	User group of the net
netname	char(12)	NET-NAME
indexnr	char(3)	IND
bkorder	char(4)	User group of the structure element
ordername	char(24)	Name of the structure element
function	char(1)	FU
typ	char(3)	TYPE
ordertext	char(120)	Text of the job description

5.3.5 Sample database queries

The following examples illustrate which SQL instructions the user must issue in the target system in order to obtain the information listed on [page 239](#).

1. In which nets is a particular job used?

Examples

- In which nets is the job “\$QUER_JOB1” used?

```
SELECT bknet,netname FROM ordertab
WHERE (ordename="JOB1" AND bkorder="QUER" AND function="J")
```

- In which net descriptions is the symbolic date “SYMDATE10” used?

```
SELECT bknet,netname FROM netsymtab WHERE symdat="SYMDATE10"
```

- In which orders is the symbolic date “SYMDATE1” used?

```
SELECT bknet,netname,ordename FROM ordersymtab
WHERE symdat="SYMDATE1"
```

- In which jobs is the symbolic date “SYMDATE1” used?

```
SELECT DISTINCT os.bknet,os.netname,os.ordename
FROM ordersymtab os, ordertab o
WHERE (os.symdat="SYMDATE1" AND (o.bknet=os.bknet AND
o.netname=os.netname AND o.ordename=os.ordename AND o.function="J"))
```

2. In which nets is a particular symbolic date used?

Example

- In which nets is the symbolic date “SYMDATE10” used?

```
SELECT bknet,netname FROM netsymtab WHERE symdat="SYMDATE10" UNION
SELECT bknet,netname FROM ordersymtab WHERE symdat="SYMDATE10"
```

3. On which days is a particular symbolic date set?

Example

- On which days is the symbolic date “SYM1” set?

```
SELECT calname,date FROM caldaysymtab WHERE (symdat="SYM1")
```

4. Which net depends on another specific net or on the contents of a particular job variable (dependency structures)?

Examples

- Which nets depend on the net “\$QUER_NET1”?

```
SELECT bknet,netname FROM ordertab
  WHERE (ordename="NET1" AND bkorder="QUER" AND typ="NET" AND
  function="C")
```

- Which nets depend on the job variable “JVA1”?

```
SELECT bknet,netname FROM ordertab
  WHERE (ordename="JVA1" AND typ="JVA" AND function="C")
```

- In which nets is there a dependency on “JOB1” in the net “\$QUER-NET1”?

```
SELECT bknet,netname FROM ordertab
  WHERE (ordename="JOB1" AND typ="JOB" AND function="C" AND
  cond_bknet="QUER" AND cond_netname="NET1")
```

- In which nets is there a dependency on the resource “RES10”?

```
SELECT bknet,netname FROM ordertab
  WHERE (ordename="RES10" AND typ="RES" AND function="C")
```

- Which nets modify the entry for the resource “RES1”?

```
SELECT bknet,netname FROM ordertab
  WHERE (ordename="RES1" AND function="M" AND typ="RES")
```

- To which net descriptions is the document “\$QUER_DOCUMENT1” assigned?

```
SELECT bknet,netname FROM netdoktab
  WHERE (bkdok="QUER" AND dokname="DOCUMENT1")
```

5. In which objects is a particular document used?

Examples

- To which nets is the document “\$QUER_DOCUMENT1” assigned?

```
SELECT bknet,netname FROM netdoktab
  WHERE (bkdok="QUER" AND dokname="DOCUMENT1") UNION
SELECT bknet,netname FROM orderdoktab
  WHERE (bkdok="QUER" AND dokname="DOCUMENT1")
```

- In which nets is the subset “ABCD” used?

```
SELECT bknet,netname FROM nettab WHERE pvs='ABCD' UNION
SELECT bknet,netname FROM ordertab WHERE pvs='ABCD'
```

- In which jobs is the pubset “ABCD” used?

```
SELECT bknet,netname,ordername FROM ordertab WHERE pvs="'ABCD' "
```

6. Which jobs are running on a particular pubset, or which job variable contains the pubset?

Example

Which jobs are running on the pubset “ABCD”?

```
SELECT DISTINCT o.bknet,o.netname,o.ordername
FROM ordertab o, nettab n
WHERE (o.pvs="'ABCD' " OR (o.bknet=n.bknet AND o.netname=n.netname
AND n.pvs="'ABCD' " AND o.function="J"))
```

7. Which user group is using a particular calendar?

Example

Which user groups are using the calendar “EXAMPLECALENDAR”?

```
SELECT bkcal FROM caltab WHERE (calname="EXAMPLECALENDAR")
```

8. In which orders is a particular selectturnus used?

Example

In which orders is SELECT-TURNUS 7 used?

```
SELECT bknet,netname,ordername FROM ordersturntab
WHERE (selectturnus="7")
```

9. For which orders or nets is there a document in the DOCLIB and what is its name?

Example

```
SELECT bknet,netname,bkorder,bkdoc,dokname FROM orderdoctab
WHERE (docvorh="y")
```

5.4 Error handling

Successful login with SIGNON is necessary in order to process the SELECT-OBJECT statement. If this is not the case, all SELECT-OBJECT statements will be ignored until the next SIGNON or END statement.

If an error occurs when output file(s) are opened, the program is terminated.

If it is not possible to write to the output file(s), processing of the SELECT-OBJECT statement is aborted and the program is terminated.

If the user accesses an object locked by AVAS while processing AVAS objects, a message is issued and processing continues with the next object.

The user is informed of all errors by messages and the corresponding error information.

6 Coupling AVAS with MAREN

With the aid of the MARENAV module it is possible to couple the two self-contained software products AVAS and MAREN as of Version 9.0. This coupling is performed by calling MARENAV at the CC exit AVEX7102 of the AVAS component “release for production”. This module contains, among other things, the MAREN program interface and carries out the following actions, which are explained further in the course of this chapter:

- perform volume checks
- enter VSNs in the JCL
- create volume listings.

You will find further information on MAREN in the manual “MAREN Volume 1” [8].

6.1 Performing volume checks

All magnetic tape devices and cartridges required by a net (referred to below as “volumes” or “tapes”) are checked by MARENAV as to their availability. MARENAV fetches the VSNs of these volumes from various sources:

- MAREN catalog
- system catalog (TSOSCAT)
- DMS commands issued in the individual jobs of a net. In the latter case, the following commands are involved:
 - CREATE-FILE
 - CREATE-FILE-GENERATION
 - IMPORT-FILE
 - MODIFY-FILE-ATTRIBUTES
 - MODIFY-FILE-GENERATION-SUPPORT
 - ADD-FILE-LINK

Whether a volume is available is determined on the basis of the following criteria:

1. VSN contained in the MAREN catalog?
2. If so, is it a reserved volume (RESERVED or PRIVATE)?

If desired, checking these criteria can be avoided by means of the MAREN parameter FOREIGN-TAPE-CHECK=NO. If this parameter is set to YES and if one of the aforementioned checks is negative, the release for production of MARENAV is aborted. If the volume is available, i.e. reserved and entered in the MAREN catalog, two further conditions are checked:

3. The device type from the MAREN archive entry must match the device type specified in the DMS command or the one taken from the file catalog entry.
4. The volume must not be exported.

If one of these checks is negative, the release for production is not aborted; instead, a note to this effect for the VSN in question is included in the two volume listings created by MARENAV.

Other circumstances which can cause the release for production to abort:

- A file consists of more than 255 volumes.
- With a multivolume file, not all of the follow-up reels have been entered in the MAREN catalog.
- Not all VSNs of a multivolume file are assigned to the same ID in the MAREN catalog.
- The MAREN system is not operable (e.g. control program not loaded, MAREN catalog closed).
- On the basis of the CLOSE indicator in the archive entry of a volume MARENAV determines that the associated file was not (or not properly) closed at creation time. If this is the case, the file involved is an input file since the wildcard (#VOL#) was specified in the DMS command.

As a rule, the VSNs of a tape file are only checked once within a net, namely at the first command that addresses the volume in question. If a volume is used once again in a subsequent job of the same net, no further MAREN checks are performed.

If the volumes reserved by MAREN are output tapes or cartridges, none of the aforementioned checks is required. MAREN has already carried out all the checks during reservation.

It is only possible to carry out the volume checks in a meaningful way if the archive entries in the MAREN catalog are up to date, i.e. reflect the actual file status of the associated tapes. This is no longer the case, for instance, if the tapes were exported and written in a different computer center, or if tape processing ran temporarily without the MAREN system and the necessary updates were not entered afterwards in the MAREN catalog.

Whenever entries from the file catalog (TSOSCAT) are used for purposes of checking, it is always assumed that the current catalog entry also exists in exactly the same form at the time the net is executed. Thus, if a VSN not contained in the MAREN catalog is entered in a system catalog entry of a file and the MAREN parameter FOREIGN-TAPE-CHECK is set to YES, release for production will be aborted.

The catalog entry of a file will not be evaluated if

- the entry was created with the IMPORT-FILE command (file from private volume)
- the VOLUME-LIST operand has been used in the ADD-FILE-LINK command
- one of the variables (#VOL#) or (#SCR#) has been specified in the file assignment with the VOLUME operand
- the file has already been addressed in the net.
(In this case, the file may have been written to and any catalog entry existing at the time of SUBMIT will no longer be up to date at execution time.)

If a volume is contained in the file catalog entry but has not yet been written to (SHOW-FILE-ATTRIBUTES displays the VSN in parentheses), this fact is noted in the volume lists.

Before the file catalog entry is read, the specified file name is checked for the presence of a catalog ID. If a catalog ID has been specified, this will be used for reading; if not, the file name will be extended by the catalog ID passed by AVAS at CC exit AVEX7102.

Example

The catalog ID :2: is passed to MARENAV at the CC exit. A job in the net contains the command

```
/ADD-FILE-LINK LINK=LINK01,FILE-NAME=$UG01.FIL1.
```

The catalog entry reads as follows:

```
/SHOW-FILE-ATTRIBUTES :2:$UG01.FIL1 .
```

In this connection it should be noted that this extension of the file name may be errored if an MPVS system is set up on the target processor.

A different catalog ID may be entered for user ID UG01 on the target processor than the ID passed by AVAS at the CC exit. Consequently, on release of the net the catalog entry of a different file will be read than that which is actually referenced on execution of the net.

Generally speaking, a net is not released under the system administrator ID TSOS. Therefore it is not possible to ascertain the default catalog ID of the target processor by reading the user entries. A correct extension algorithm for the file names cannot therefore always be guaranteed.

If DMS reports one of the following errors on reading the catalog entry, release of the net is aborted with the message CATALOG (&00) NOT ACCESSIBLE.

DMS0501 REQUESTED CATALOG NOT ACCESSIBLE. COMMAND REJECTED

DMS0502 REQUESTED CATALOG IN QUIET OR HOLD MODE. WAITING TIME IS OVER. COMMAND NOT PROCESSED

DMS0503 MRSCAT CONTAINS INCORRECT INFORMATION. COMMAND REJECTED

DMS0504 CATALOG MANAGEMENT SYSTEM ERROR. COMMAND TERMINATED

DMS0505 CMS RETURNED ERROR IN THE MRS COMMUNICATION FACILITY TO THE COMMAND PROCESSING

DMS0512 REQUESTED CATALOG CANNOT BE FOUND. COMMAND TERMINATED

6.2 Entering VSNs in the JCL

The following wildcards can be used in the BS2000 commands (see [page 281](#)) to enter VSNs in the JCL of the jobs:

- (#VOL#) or (#VOL#,n) for input tapes
- (#SCR#) or (#SCR#,n) for output tapes

These wildcards may not be used as comments in the DMS commands evaluated by MARENAV. When entered, the VSNs are always enclosed in parentheses, even if there is only one VSN.

If MARENAV is to be used to good effect in AVAS with multiprocessor capability, particularly when wildcards are employed, the following requirement must be met: All processors on which the jobs referenced by a net release are to be executed must be configured into a MAREN network, i.e. using the same MAREN catalog.

Otherwise, volumes are reserved by (#SCR#), for example, at the time of the net release which are not then contained in the MAREN catalog of the target processor upon subsequent execution of the net. This will result in the MAREN system aborting commands.

Note

When MARENAV enters VSNs in the JCL, the tape is reserved for the execution ID of the job. For this reason, it is no longer permitted to modify this ID by means of the AVAS statement MODIFY-SUBMIT-NET (NET-USER or USER operand).

6.2.1 Entering VSNs for input tapes

In the command

```
/IMPORT-FILE SUPPORT=TAPE(VOLUME=(#VOL#),DEVICE-TYPE=device,FILE-NAME=file)
```

MARENAV substitutes a concrete VSN for the wildcard (#VOL#).

If a multivolume file is involved, this variable is replaced by a VSN list. MARENAV fetches the VSNs from the MAREN catalog. If this catalog contains a file more than once, the most recent version of the file is selected. It is also optionally possible to request an earlier version by entering (#VOL#-n), where "n" may take any value between 0 and 9999. For example, if VOLUME=(#VOL#-1) is specified, MARENAV will select the next-to-last version of the file and enter the associated VSNs in the relevant DMS command. Since, in the MAREN catalog, the same file name may occur in the archive entries of different IDs, MARENAV must be told which ID, and therefore which VSN, has to be selected in this case.

This can be done by means of the following /REMARK command:

```
/REMARK #UID#={*OWN / *ALL / userid}
```

*OWN

Default value.

MARENAV selects only those VSNs which have the specified file name in the archive entry and are assigned either to the user ID from the file name or to the execution ID. The execution ID is determined as follows:

- If the nameJENT field in the AVAS communication area has the value “nameJENTJ” (specification ENTER-PARAMS=NET in the net description), the ID specified in the nameJUSE field is used.
- However, if the nameJENT field has the value “nameJENTL” (specification ENTER-PARAMS=LOGON in the net description), the ID specified in the SET-LOGON-PARAMETERS command is used as the execution ID.

At least one of these three positions must have an ID, otherwise release for production will be aborted. As with the “userid” value, *OWN likewise enables a selection specific to a single ID. However, the *OWN value offers the advantage of job neutrality, i.e. when an ID is changed it is not necessary to change the JCL with regard to the /REMARK #UID# command.

*ALL

When VSNs are selected, the IDs under which they are entered in the MAREN catalog are not taken into account. This parameter must be selected whenever the procedures employed in a computer center allow the possibility that individual versions of one and the same file can be created under different IDs.

userid

MARENAV selects only those VSNs which have the specified file name in the archive entry and are assigned to the user ID “userid”. This parameter must be selected whenever a file name occurs under more than one ID in the MAREN catalog but different files are involved. If MARENAV cannot locate an archive entry with this file name under the specified ID, release for production is aborted.

The VSN selection made by means of /REMARK #UID# is only valid until the job has terminated.

Within a job, as many additional /REMARK #UID# commands as desired can be used. The selection remains valid until superseded by a new one made by the following /REMARK #UID# command.

6.2.2 Entering VSNs for output tapes

In the command

```
/CREATE-FILE FILE-NAME=file,SUPPORT=TAPE(VOLUME=(#SCR#),DEVICE-TYPE=device)
```

MARENAV reserves a free tape and substitutes the received VSN for the wildcard (#SCR#) in the relevant DMS command. The user ID for which the tape is reserved is determined in the same way as is possible for input tapes using /REMARK #UID#=*OWN. In other words, if the file name specified in the command does not contain an ID, a reservation takes place for the execution ID. This is determined as follows:

- If the field nameJENT in the AVAS communication area has the value "nameJENTJ", the execution ID is stored in the nameJUSE field.
- If the field nameJENT has the value nameJENTL, the execution ID corresponds to the ID specified in the /SET-LOGON-PARAMETERS command.

If the ID is lacking at all these locations, MARENAV will abort the release for production.

By default, the free volumes are reserved from the storage location defined by means of the MAREN parameter DEFAULT-HOME-LOCATION (MARENADM statement //RESERVE-FREE-VOLUME ...,HOME-LOCATION=*STD).

If a reservation from a different storage location is to be effected, this must be previously defined in the job concerned:

```
/REMARK #LOC#=storage-location
```

Note that no blanks may be entered before or after the = sign.

The definition made by means of /REMARK #LOC# always remains in force only until the end of the job. The reservation storage location can be changed as often as required in each job. If another reservation is to be made from DEFAULT-HOME-LOCATION within a job, this will be effected by means of the statement /REMARK #LOC#=*STD.

With a multivolume file, provision can also be made for two or more tapes. For example, when VOLUME=(#SCR#,3) is entered, MARENAV reserves three tapes and substitutes a VSN list (vsn1,vsn2,vsn3) for the wildcard. In this way, up to 255 volumes can be reserved at once.

The date produced by *adding* the *start time* specified by EARLIEST-START to the *default retention period* set in the MAREN parameter DEFAULT-FREE-DATE is used as the release date.

As with tape processing in BS2000 (DMS), these volumes are given the attribute USER-ACCESS=FOREIGN-READ-ONLY in the archive entry, i.e. they can only be read-accessed from other IDs. This presetting can be changed if necessary via the MAREN CC exit.

6.2.3 Handling continuation lines

If, with multivolume files, the insertion of VSNs makes the command line sent to MARENAV too long, the command or command portion will be replaced by two or more command lines without any special arrangements required on the part of the user.

6.3 Creating volume lists

During release for production, MARENAV creates the following three files for each net:

- transport list for the AV staff (file name beginning with TRS)
- tape mount listing for the operator (file name beginning with OPR)
- shift procedure for the archivist (file name beginning with PRC).

These files are SAM files, of which the first two contain print control characters (printout with `/PRINT-DOCUMENT FILE-NAME=... , LINE-SPACING= *BY-EBCDIC-CONTROL`).

Since on SUBMIT, MARENAV only receives from AVAS the catalog IDs of the target processors but the storage locations assigned to the respective target processors are to be inserted in the volume lists and specifically in the shift procedure, an assignment table must be made available to MARENAV. This table should be stored in a SAM file; the individual records in this file should have the following format:

4 bytes	Catalog ID of the target processor, padded on the right with blanks if necessary.
1 byte	Blank as separator.
8 bytes	Storage location assigned to the target processor, padded on the right with blanks if necessary. The data type of the storage location name must be "alphanumeric", i.e. it may contain only letters, digits and the special symbols \$, # and @.

For those cases in which a catalog ID is neither specified in the file name nor passed by AVAS at the CC exit, a file record with a catalog ID consisting of four blanks must be defined.

Example of individual file records

```
C3 CPU003
CI02 DA12ZE02
1 DVA001
2 DVA002
CENTRAL
```

The storage location file must be assigned with the link name MARENAVL in the procedure SYSPRC.AVAS.085 to be called by the user for initiating the AVAS dialog.

```
/ADD-FILE-LINK LINK-NAME=MARENAVL, FILE-NAME=lagerortdatei
```

At the CC exit, MARENAV uses RDTFT to ascertain the name of the file and fetches the required information from it.

If no file with the link name MARENAVL was assigned, the shift procedure is created with a variable &TEMPLOC, and the "TO-LOC" column in the transport and tape mount listings remains empty. Thus, with all AVAS installations not in a multiprocessor system environment, there is no need to create a storage location file. If in this case different catalog IDs are passed to MARENAV, a /REMARK line with the following text is included in the shift procedure:

```
* WARNING: DIFFERENT CATIDS USED DURING SUBMIT *
```

This is intended to indicate that the volumes used in the net may possibly not be processed only on one processor and so different storage locations would also need to be specified for the procedure variable &TEMPLOC.

The release for production is aborted in the following cases:

- Although the storage location file is assigned it could not be opened, or a DMS error occurred during processing of the file.
- A storage location is not of data type "alphanumeric".
- There was no entry in the storage location file for a catalog ID passed to MARENAV.

6.3.1 Transport list

This file has the name TRS.AVAS.rcs.yymmdd.hhmmss.

rcs Name of the run control system.

yymmdd.hhmmss Date and time of day of the release for production.

This file contains a list of all VSNs belonging to a net, including those volumes which are required only in the case of a restart. It enables the job scheduler to ready all volumes required for execution of a net. If a volume is used more than once in a net, it nevertheless appears only once in the transport list.

The list header of this file contains the following information:

- file name of the volume list
- date and time of day of the release for production
- name of the net
- scheduled execution time
- number of all requests for scratch tapes of a particular device type
- number of all requested private tapes of a particular device type.

If a storage location file was assigned on release of the net (by means of LINK=MARENAV), the requests for scratch and private tapes will also be output separately according to storage locations.

Following the list header is a line for each VSN addressed in the net. This line has the following structure:

- Hierarchy level (index) of the job requesting this VSN
- Job name from the SET-LOGON-PARAMETERS command
- Catalog ID of the processor on which the volume is processed
- Current storage location of the volume (FROM-LOC)
- Storage location assigned to the catalog ID (TO-LOC). The volume is requested at this location on execution of the net. If no storage location file was assigned on release of the net, this column remains empty.
- VSN (volume serial number) of the volume
- Possibly a remark that the volume's availability is restricted (e.g. the tape has been exported) or that it is only required in the case of a restart.

The transport list is sorted according to

1. current storage location
2. VSNs of the volumes

On the basis of this list, job management staff can gather the tape sets addressed in a net and, if necessary, transport them to the system.

Example illustrating the structure of a transport list

A net bearing the name \$UG01_SAL_NET_061219_160500 consists of only two jobs. The first job, with the name SALARY01, processes files from the pubset CI02. The second job, with the name SALARY02, processes files from the pubset C3.

A total of seven volumes is required, where the tape with VSN Y1C002 is processed in both jobs and tape Y1A003 is still exported at the time of release for production. Two of the volumes requested by the job SALARY02 are private tapes.

The name of the processing system is RCS003. At the CC exit, a storage location file was assigned which contains the file records listed in the example on [page 289](#).

The transport list has the file name TRS.AVAS.RCS003.061218.113012 and is stored under the user ID of the user releasing the nets for production.

```

*** FILE TRS.AVAS.RCS003.061218.113012    ***                PAGE    1
DATE/TIME OF RELEASE: 06-12-18/11:30
NET NAME: $BK01.GEH.NET.061219.160500      EARLIEST START: 06-12-19/16:05
      2 PRIVATE TAPES REQUESTED WITH DEVICE TYPE TAPE-C4    AT LOCATION CPU003
    
```

INDEX	JOBNAME	CATID	FROM-LOC	TO-LOC	VOLUME	REMARK
020	SALARY02	C3	CENTRAL	CPU003	Y1A003	VOLUME IS EXPORTED
010	SALARY01	CI02	CENTRAL	DA12ZE02	Y1B001	
010	SALARY01	CI02	DVA11	DA12ZE02	Y1C002	
010	SALARY01	CI02	DVA11	DA12ZE02	Y1D001	
020	SALARY02	C3	DVA12	CPU003	Y1B003	

6.3.2 Tape mount listing

This file has the name OPR.AVAS.rcs.yymmdd.hhmmss.

rcs Name of the run control system.

yymmdd.hhmmss Date and time of day of release for production.

Like the transport list, this file contains a list of all VSNs belonging to a net, including those volumes which are only required in the case of a restart. It tells the operator which volumes a net has requested. If a volume is requested more than once in a net, it will also appear more than once in this list. Requests for scratch and private tapes are accompanied by the device type and, in the case of private volumes, by the specified number of such volumes in the file as well. The list header and the record structure of this file are identical to those in the transport list of the job scheduler. However, sorting does not take place; instead, the VSNs and the scratch and private tape requests are listed in chronological order, i.e. in the order in which they were requested by the individual jobs in a net at execution time.

Example illustrating the structure of a tape mount listing

The tape mount listing has the file name OPR.AVAS.RCS003.061218.113012 and is stored under the user ID of the user releasing the nets for production.

This file is structured as follows, using the same basic data as in the "transport list" example above:

*** FILE OPR.AVAS.RCS003.061218.113012 *** PAGE 1

DATE/TIME OF RELEASE: 06-12-18/11:30

NET NAME: \$BK01.GEH.NET.061219.160500 EARLIEST START: 06-12-19/16:05

INDEX	JOBNAME	CATID	FROM-LOC	TO-LOC	VOLUME	REMARK
010	SALARY01	CI02	CENTRAL	DA12ZE02	Y1B001	
010	SALARY01	CI02	DVA11	DA12ZE02	Y1C002	
010	SALARY01	CI02	DVA11	DA12ZE02	Y1D001	
020	SALARY02	C3	CENTRAL	CPU003	Y1A003	VOLUME IS EXPORTED
020	SALARY02	C3		CPU003		REQUESTED FOR 2 PRIVATE TAPES
020	SALARY02	C3		CPU003		WITH DEVICE TYPE TAPE-C4
020	SALARY02	C3	DVA12	CPU003	Y1B003	
020	SALARY02	C3	DVA11	CPU003	Y1C002	

6.3.3 Shift procedure

This file has the name PRC.AVAS.rcs.yymmdd.hhmmss.

rcs Name of the run control system.

yymmdd.hhmmss Date and time of day of the release for production.

In this procedure file, the storage locations (TEMPORARY-LOCATIONS) in the MAREN archive entry are updated for all volumes required by a net. This shift procedure can be called by the work scheduler when he transports the volumes from the individual archives to the systems on which the jobs of the released nets are to run.

As in the tape mount listing of the operator, each VSN is only listed once in this procedure file.

If, for any volume, the shift triggered by the MARENADM statement //MODIFY-VOLUME-ATTRIBUTES cannot be carried out (e.g. because the volume is being processed on another system), the procedure will not be aborted, but will continue normally with the next VSN. However, it must be ensured that all volumes required to execute the net are actually available.

Example illustrating the structure of a shift procedure

The shift procedure has the file name PRC.AVAS.RCS003.061218.113012 and is stored under the user ID of the user releasing the nets for production.

This file is structured as follows, given the same basic data as in the "transport list" example above:

```
/PROC A,(&MARENADM=MARENADM)
/SYSFILE SYSDTA=(SYSCMD)
/EXEC &MARENADM
//MOD-VOL-ATTR VOL=Y1A003,TEMP-LOC=CPU003
//STEP
//MOD-VOL-ATTR VOL=Y1B001,TEMP-LOC=DA12ZE02
//STEP
//MOD-VOL-ATTR VOL=Y1C002,TEMP-LOC=DA12ZE02
//STEP
//MOD-VOL-ATTR VOL=Y1D001,TEMP-LOC=DA12ZE02
//STEP
//MOD-VOL-ATTR VOL=Y1B003,TEMP-LOC=CPU003
//END
/ENDP
```

TEMP-LOC specifies the storage location to which the volumes are to be transferred.

6.3.4 Restrictions pertaining to MARENAV

The MARENAV functions cited are activated only if VSNs occur in one of the DMS commands mentioned on [page 281](#) (CREATE-FILE etc.). In the following cases, MARENAV does not initiate any actions:

- VSNs are specified in the /SECURE-RESOURCE-ALLOCATION command.
- VSNs occur in program statements.
- VSNs are to be replaced by job variable values.
- VSNs are specified in procedure variables.
- VSNs are addressed in a procedure which is started within a job via the BS2000 command CALL-PROCEDURE.
- In the DMS command ADD-FILE-LINK, the VSNs contained in the TST entries are not checked when the TAPE-SET-NAME operand is specified.
- The START-POSITION operand is not evaluated in connection with the availability checks, i.e. all VSNs in a cataloged tape file are always checked, regardless of the selection made in the START-POSITION operand.

If a file is addressed more than once in a net, the volumes are only checked the first time it is addressed. Additional VSNs occurring in later assignments for this file are not checked, nor do they appear in the volume lists created by MARENAV.

6.3.5 Example illustrating further processing of a VSN in a net

A tape file is processed twice within a net. It is created as an output file in the first job and used as an input file in one of the subsequent jobs. If MARENAV is not used, the following actions must be performed every time before starting the net:

- Reserve tapes with MARENADM or other archive routines.
- Enter VSNs of the newly reserved tapes in the ENTER file(s).

By using MARENAV you can skip each of these preliminary steps. Since the parameter VOLUME=(#SCR#,n) or VOLUME=(#VOL#) is already entered in the file assignments when the ENTER files are created, no preliminary steps are needed before starting the production run.

6.3.6 Notes on retry runs

Certain special considerations should be noted with regard to the two different types of retry runs, using the RESTART-NET and REPEAT-NET statements:

1. Retrying an interrupted net via RESTART-NET

If tapes which were already being processed before abortion of the job or net are requested within the framework of the RESTART run, then from the standpoint of the MAREN system it is necessary before carrying out the restart to ensure that all the tapes archived in the MAREN catalog are available and that there are no restrictions impairing access:

- For all tapes written with a retention period specified, the retention period must be reset in the MAREN catalog by means of the MAREN/MARENADM statement `//MOD-VOL-ATTR VOL=vsn, EXPIRATION-DATE=0` (provided it has not already expired at the time of the restart).
- Between the abortion of the job and its restart using MAREN/MARENADM statements, no attributes which prevent renewed processing of these volumes may be applied to the tapes in the MAREN catalog (this should in any case be an improbable situation). Possible error causes would be, for example, modification of volume passwords, shifting to remote storage locations, deletion of the VSNs from the volume catalog and release or exporting of the volumes.
- No MAREN parameters which result in a basic change in the checking of volume availability should be modified in the interim (also improbable). The MAREN parameter FOREIGN-TAPE-CHECK, which indicates whether all volumes to be processed must be archived in the MAREN catalog, is particularly critical in this connection.

2. Retry run via REPEAT-NET

If a retry run is to be performed using REPEAT-NET it is necessary to ensure, as also in the case of RESTART, that the retention period for all the output tapes archived in the MAREN catalog has already expired at the time of net execution. If this is not the case, EXPIRATION-DATE must first be reset with the aid of the MAREN statement `//MODIFY-VOLUME-ATTRIBUTES`.

If a file is to be further processed in one job or distributed over two or more jobs of a net, other actions may be required before REPEAT-NET. Two cases must be distinguished here:

- further processing of an input file
- further processing of an output file

Further processing an input file

A file update occurs in a net: a tape file created in the net most recently executed is read in, updated and written back to a new tape. Selection of the VSNs is performed with the aid of the wildcards (#VOL#) and (#SCR#).

Example for this job

```

/SET-LOGON-PARAMETERS
/REMARK * ASSIGN TAPE INPUT FILE *
/IMPORT-FILE SUPPORT=TAPE(VOLUME=(#VOL#),DEVICE-TYPE=TAPE-C4,FILE-NAME=FILX)
/SET-FILE-LINK LINK-NAME=IN,FILE-NAME=FILX
/REMARK * ASSIGN DISK OUTPUT FILE *
/SET-FILE-LINK LINK-NAME=OUT,FILE-NAME=FILY
/REMARK * COPY FILX (TAPE) TO FILY (DISK)
/START-PROGRAM FROM-FILE=PROG1
/REMARK * DELETE CATALOG ENTRY OF FILX *
/REMARK * RELEASE LINK NAME *
/EXPORT-FILE FILE-NAME=FILX
/REMOVE-FILE-LINK LINK-NAME=IN
/REMOVE-FILE-LINK LINK-NAME=OUT
/REMARK * UPDATE FILX IN A SORT RUN *
/REMARK * OUTPUT FILX TO A NEW TAPE *
/SET-FILE-LINK LINK-NAME=SORTIN01,FILE-NAME=FILY
/SET-FILE-LINK LINK-NAME=SORTIN02,FILE-NAME=FILZ
/CREATE-FILE FILE-NAME=FILX,SUPPORT=TAPE(VOLUME=(#SCR#),DEVICE-TYPE=TAPE-C4)
/SET-FILE-LINK LINK-NAME=SORTOUT,FILE-NAME=FILX
/START-PROGRAM FROM-FILE=$SORT
.
.
.
/REMARK * DELETE CATALOG ENTRY OF FILX *
/EXPORT-FILE FILE-NAME=FILX
/EXIT-JOB

```

If this run needs to be repeated because the tape output file created by the sort run is incorrect (e.g. an incorrect file was assigned with SORTIN02), the following should be noted:

In REPEAT-NET, MARENAV would enter the VSN of the tape noted in the MAREN catalog for the latest version of FILX in the JCL instead of the variable (#VOL#). This means that the sort output file created in the last, errored run would be further processed and not the still correct file from the previous run (assigned here with LINK=IN). In order to prevent this the last, errored file version must be removed from the MAREN catalog. This can be done in several different ways. The simplest method consists in deleting the file name FILX in the archive entry of the most recently created output tape by means of the MAREN statement

```
//MOD-VOL-ATTR VOL=vsn, FILE=*NONE.
```

If the volume is no longer required, the volume release date (FREE-DATE) and, if necessary, also the file release date (EXPIRATION-DATE) must be reset at the same time by means of the MAREN statement

```
//MOD-VOL-ATTR VOL=vsn, FILE-NAME=*NONE, FREE-DATE=0, EXPIR-DATE=0
```

in order that the volume be released by the MAREN administrator during the next release run. If there are multiple output tapes, this action must be performed for each individual tape.

Further processing an output file

A net comprises two jobs executing in succession; in the first job a tape file is created which is used as an input file in the second job. Selection of the VSNs is again performed using the wildcards (#SCR#) (#VOL#). In this example the output may only be directed to a tape and no archive entries containing the specified file names may be present in the MAREN catalog.

Example for job 1:

```
/SET-LOGON-PARAMETERS
/REMARK * ASSIGN DISK INPUT FILE *
/SET-FILE-LINK LINK-NAME=PCIN, FILE-NAME=FILE1
/REMARK * ASSIGN TAPE OUTPUT FILE *
/CREATE-FILE FILE-NAME=FILE2, SUPPORT=TAPE(VOLUME=(#SCR#), DEVICE=TAPE-C4)
/SET-FILE-LINK LINK-NAME=PCOUT, FILE-NAME=FILE2
/REMARK * COPY SELECTED RECORDS FROM FILE1 TO FILE2 *
/START-PROGRAM FROM-FILE=$PERCON
SELECT COND=...
END
. . .
/REMARK * DELETE CATALOG ENTRY OF FILE2 *
/EXPORT-FILE FILE-NAME=FILE2
/EXIT-JOB
```

Example for job 2:

```
/SET-LOGON-PARAMETERS
/REMARK * ASSIGN TAPE FILE CREATED IN FIRST JOB *
/REMARK * ASSIGN DISK INPUT FILE *
/IMPORT-FILE SUPPORT=TAPE(VOLUME=(#VOL#), DEVICE-TYPE=TAPE-C4, FILE-NAME=FILE2)
/SET-FILE-LINK LINK-NAME=INPUT1, FILE-NAME=FILE2
/SET-FILE-LINK LINK-NAME=INPUT2, FILE-NAME=FILE3
/REMARK * ASSIGN DISK OUTPUT FILE *
/SET-FILE-LINK LINK-NAME=OUTPUT, FILE-NAME=FILE4
/START-PROGRAM FROM-FILE=PROG2
. . .
/REMARK * DELETE CATALOG ENTRY OF FILE2 *
/EXPORT-FILE FILE-NAME=FILE2
/EXIT-JOB
```

If a net constructed in this way is to be repeated using REPEAT-NET, no particular actions are required. It is advisable to release the volume created in the previous execution of the net, assuming it is no longer required, (MAREN statement //MOD-VOL-ATTR VOL=vsn, FREE-DATE=0, EXPIR-DATE=0) and thus also remove the errored version of this file from the MAREN catalog.

6.3.7 Error messages from MARENAV

In the event of an error, MARENAV passes the message code of the MAREN message for purposes of diagnosis, and possibly the VSN where the error occurred. If no precise MAREN message code exists for the cause of the error, one of the following texts is sent:

CATALOG catid NOT ACCESSIBLE

The attempt to read the TSOSCAT entry of a file was rejected with one of the DMS error codes DMS0501 through DMS0505 or DMS0512, i.e. the corresponding catalog is not available. The release for production is aborted. It cannot be performed again until all catalogs referenced by the net in question are available.

CATID x NOT CONTAINED IN LOCATION FILE

There is no entry in the assigned storage location file for the specified catalog ID. The release for production is aborted. The location file must be updated before the SUBMIT is repeated.

This message is also used if a catalog ID is neither specified in the file name nor was passed by AVAS at the SUBMIT exit and the location file contained no entry with a blank catalog ID, i.e. a catalog ID comprising simply four blanks.

LOCATION locname NOT ALPHANUMERIC

One of the storage locations contained in the assigned storage location file is not of the "alphanumeric" data type. The location name is possibly shorter than 8 bytes and padded on the right with blanks.

The release for production is aborted.

VALUE FOR OPERAND VOLUME NOT CORRECT

The wildcard contained a syntax error. For example, the parentheses are missing, or the specified file version is not separated from the character string #VOL# by the character "-".

The release for production is aborted.

NO USERID SPECIFIED

No user ID was specified either in the file name or in the BS2000 command SET-LOGON-PARAMETERS. Neither has AVAS passed a user ID at the SUBMIT exit. The release for production is aborted.

DEVICE TYPE MISSING OR INVALID

The release for production operation was aborted for one of the following reasons:

1. A wildcard was used in connection with a device type other than a tape or a cartridge.
2. An invalid device type was specified in connection with a request for scratch or private tapes.

FILENAME/VERSION NOT FOUND IN MAREN CATALOG

Either the file name could not be found (if (#VOL#) was specified), or the specified file version could not be found (if (#VOL#-nnn) was specified), or there is a discrepancy in the DEVICE specification.

The release for production is aborted.

DIFFERENT USERIDS userid1,userid2 IN MAREN CATALOG

When (#VOL#) and #UID#=*SAME were specified, tapes with different user IDs were found in the MAREN catalog for the current file name.

The release for production is aborted.

VOLUMES OF MULTI VOLUME SET ARE MISSING

When (#VOL#) is specified, it was discovered that not all VSNs of the multivolume set are entered in the MAREN catalog for the current file name.

The release for production is aborted.

VOLUMES OF MULTI VOLUME HAVE DIFFERENT USERIDS

For a multivolume file, not all VSNs are assigned to the same ID in the MAREN catalog.

The release for production is aborted.

FILE NOT CLOSED

On the basis of the CLOSE indicator in the archive entry for a volume, MARENAV discovered that when the associated file was created it was not closed, or not closed properly. If the wildcard (#VOL#) was entered in the file assignment, the file involved is an input file, and the release for production is aborted since it would be pointless to process it further.

If MARENAV cannot determine whether the file is an input or output file, this message will appear as a warning only and the release for production will proceed.

ERROR error code DURING PROCESSING OF MACRO macro name

The specified error occurred while a macro was being executed.

The release for production is aborted.

6.3.8 Warning messages from MARENAV

In the transport and mount listings created by MARENAV, the individual VSNs may be accompanied by the following warnings pointing out possible sources of errors during net execution. It is therefore advisable to check the status of the volumes indicated. The following warning messages are possible:

VOLUME EXPORTED

The specified volume has been exported and must be returned by the scheduled time of execution.

VOLUME HAS DIFFERENT DEVICE TYPE

For the volume indicated, the device type stored in the archive entry does not match the device type specified in the DMS command or in the file's TSOSCAT entry.

VOLUME CONTAINS NO VALID DATA

According to the TSOSCAT entry, no data has been written to the specified VSN, which must therefore not be used as an input tape.

VOLUME ONLY USED IN CASE OF RESTART

The volume is required only in the event of a restart.

7 AVAS reports

Just as in the AVAS interactive system, a task, the REPORT generator, can be loaded. The REPORT generator serves to create reports and to output them to a print file, the report file.

An AVAS report constitutes the evaluation of the AVAS production plan and the AVAS journal file according to predefined criteria. These criteria are specified by the user with the aid of REPORT statements read by the REPORT generator via the BS2000 system file SYSDTA.

The PLANNED-NET-MODIFICATION report lists nets that were changed after production planning.

The OUT-OF-PLAN report lists nets that have exceeded a defined delay and/or are in a selected status.

The set of nets to be logged is determined on the basis of the current production plan (NPRLIB). It can be determined more precisely by specifying the NET-NAME and/or PERIOD-NAME. The data to be logged is selected from the current journal file (JRNDAT).

PLANNED-NET-MODIFICATION report

The PLANNED-NET-MODIFICATION report comprises all nets from the specified AVAS files to which unplanned changes were made after completion of net planning by means of CREATE-PLAN-NET. The changes are logged in the journal file.

An unplanned change in the above sense is the

- modification of a net via MODIFY-PLAN-NET
- modification of a temporary task via EDIT-PROD-JOB
- deletion of tasks during SUBMIT-NET
- modification of a net via MODIFY-SUBMIT-NET
- modification of a task via MODIFY-SUBMIT-JOB

The report lists all the nets involved, all tasks within the nets that are affected by the modification, as well as the type of each modification.

The scope of the modifications and the altered parameters/statements of the tasks can be seen from the allocated journals.

OUT-OF-PLAN report

The OUT-OF-PLAN report comprises all planned nets from the specified AVAS files. The current values of the nets are determined from the journal file.

Current values of nets in the above sense are

- the current status of the nets
- the planned EARLIEST-START time of the nets
- any delay relative to EARLIEST-START
- the LATEST-START time of the nets relative to the planned start time in the net name
- any delay relative to LATEST-START
- the actual start time of the nets if they have already been started
- the time of normal termination of the nets if they have already reached the normal end
- the tasks of the nets that terminated with an error during execution, even if the net reached normal end after a restart.

For evaluation, the net data can be sorted according to different criteria.

The following fields may serve as sort criteria:

- global status of the nets
- net name
- delay relative to EARLIEST-START
- EARLIEST-START.

The set of nets to be output to a report file can be restricted by the specification of selection criteria.

Selection criteria are

- the global status of the nets and
- any exceeding of a predefined delay tolerance.

The set intersection of the nets that satisfies both selection criteria is output in each case. A given file can be multiply evaluated according to different criteria.

Each evaluation is followed by a summary showing the totals of all nets in global status; it also lists the percentage of nets in the predefined states.

All REPORT statements entered are listed in the table below. Note here that only those REPORT statements which were assigned to the executed functions have been processed. The statements not processed need not correspond to the contents of the work file or to the executed sort keys.

The following statements are processed with the individual functions:

REPORT statements	FUNKTION=		
	CREATE	SORT	PRINT
AVAS-SYSTEM-ID	yes	no	no
SYSTEM-PASSWORD	yes	no	no
AVAS-USER-ID	yes	no	no
USER-PASSWORD	yes	no	no
NET-NAME	yes	no	no
PERIOD-NAME	yes	no	no
OUTPUT-WORK-FILE	yes	yes	yes
SORT-STATE-ORDER	no	yes	no
SORT-FIELDS-ORDER	no	yes	no
SELECT-STATUS	no	no	yes
THRESHOLD	no	no	yes
OUTPUT-REPORT-FILE	no	no	yes
END	yes	yes	yes

7.1 REPORT generator

Execution of the REPORT generator

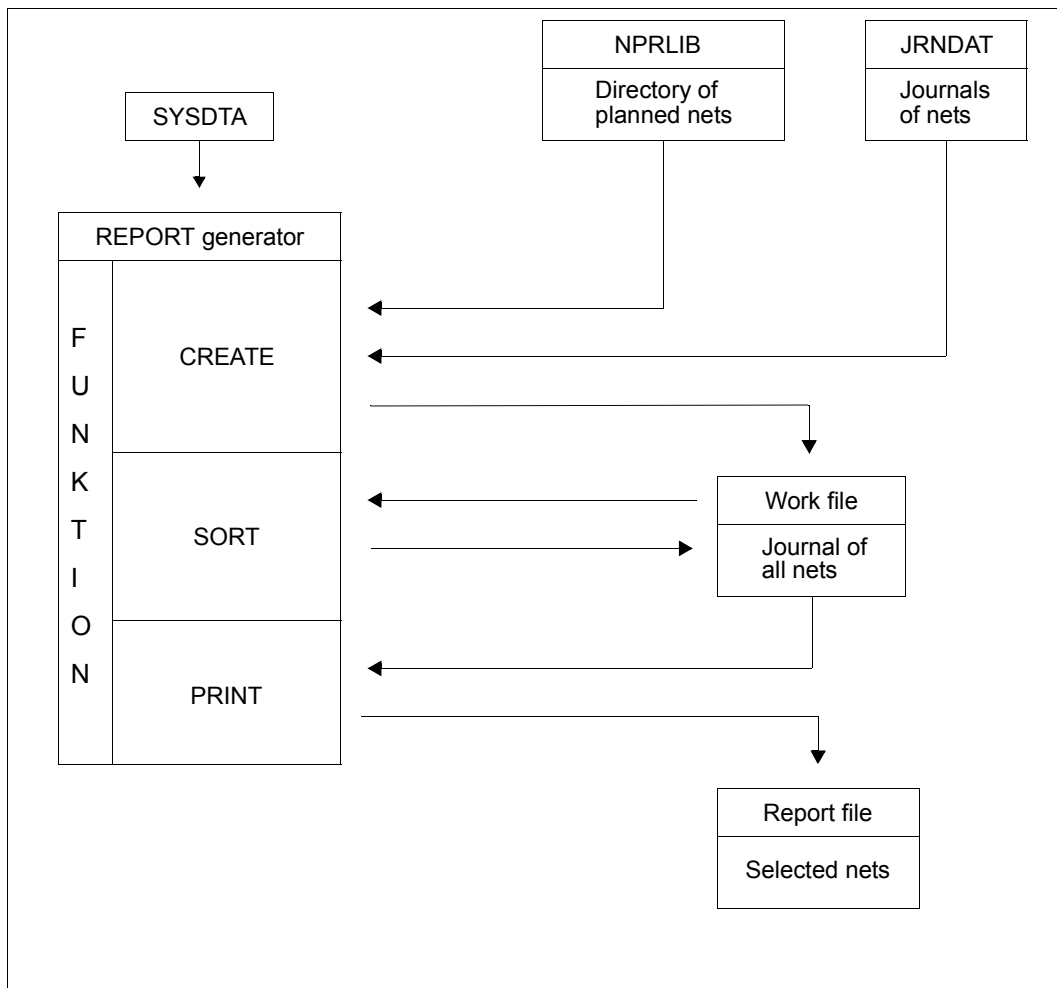


Figure 7: Execution of the REPORT generator

7.1.1 Entering REPORT statements for the REPORT generator

The statements to the REPORT generator are read via SYSDTA.

The set of requisite statements depends on the desired report (REPORT-NAME) and the selected function (FUNKTION).

REPORT statements must be entered in the following format:

statementname=value or statementname=(list)

If a REPORT statement is entered more than once, the values entered last are used.

Exception

Entries made via the REPORT statement FUNKTION are added.

The input parameters are not processed via SDF. Errored or illegal entries are logged and processing is aborted.

The error cause can be seen from the SYSOUT log.

Invalid entries cannot be corrected interactively, even if the report procedure is running in interactive mode.

Example of statement input in the procedure

```

/BEGIN-PROC LOGG=COM,PAR=YES(PROC-PAR=(&REPORT,...),ESC-CHAR=C'&')
.....
/ASSIGN-SYSDTA TO=*SYSCMD
/START-PROG FROM-FILE=*PHASE(LIB=$AVAS.SYSPRG.AVAS.085.SYSTEM,-
/ELEMENT=AVAS.SYS.LOAD.REPORT)
REPORT-NAME      =  OUT-OF-PLAN
FUNKTION         =  ALL
AVAS-SYSTEM-ID   =  SYSNAME
SYSTEM-PASSWORD  =  *STD
AVAS-USER-ID     =  USERNAME
USER-PASSWORD    =  C'XXXXXX'
NET-NAME         =  *ALL
OUTPUT-WORK-FILE =  AVAS.WORK.OOP
SORT-STATE-ORDER =  (ERROR,NO-SUBMIT,WAITING,STARTED,ENDED,ABENDED)
SORT-FIELDS-ORDER =  (STATE,NET-NAME,E-DELAY)
SELECT-STATUS    =  (ERROR,NO-SUBMIT,WAITING,STARTED)
THRESHOLD        =  (5,2)
OUTPUT-REPORT-FILE =  AVAS.REPORT.OOP
END
/ASSIGN-SYSDTA TO=*PRIMARY
.....
/.ENDE END-PROC

```

Example of statement input via a file

```
/BEGIN-PROC LOGG=COM,PAR=YES(PROC-PAR=(&REPORT,...),ESC-CHAR=C'&')
.....
/ASSIGN-SYSDTA TO=AVAS.PAR.&REPORT
/START-PROG FROM-FILE=*PHASE(LIB=$AVAS.SYSPRG.AVAS.085.SYSTEM,-
/ELEMENT=AVAS.SYS.LOAD.REPORT)
/ASSIGN-SYSDTA TO=*PRIMARY
.....
/.ENDE END-PROC
```

7.1.2 Selecting the REPORT generator functions

The REPORT statement FUNKTION enables the user to select and start the functions of the REPORT generator.

7.1.3 Allocating the production plan and the journal file

The REPORT statements AVAS-SYSTEM-ID and AVAS-USER-ID permit the user to select a specific production plan (NPRLIB), which is assigned to the REPORT generator. These two statements are only required for FUNKTION=CREATE.

Access to the production plan selected in this way is enabled by specification of the assigned passwords via the REPORT statements USER-PASSWORD and SYSTEM-PASSWORD.

The REPORT statement AVAS-SYSTEM-ID permits the user to select a specific journal file. This statement is only required for FUNKTION=CREATE.

Access to the journal file selected in this way is enabled by specification of the assigned password via the REPORT statement SYSTEM-PASSWORD.

7.1.4 Allocating the work file for the REPORT generator

The REPORT statement OUTPUT-WORK-FILE permits the user to assign a work file to the REPORT generator. This statement is mandatory for all values of FUNKTION.

When a new work file is to be created, any existing work file must first be deleted.

7.1.5 Allocating the report file for the REPORT generator

The REPORT statement OUTPUT-REPORT-FILE permits the user to assign a report file to the REPORT generator. This statement is only required for FUNKTION=PRINT.

If FUNKTION=PRINT is selected, this file is the output file for any report to be printed.

When a new report file is to be created, any existing report file must first be deleted.

7.1.6 Output of the REPORT generator to the work file

The REPORT statements AVAS-USER-ID and NET-NAME permit the user to define the set of nets to be output to the work file. The statements are only required for FUNKTION=CREATE.

7.1.7 Output of the REPORT generator to the report file

The REPORT statements SELECT-STATUS and THRESHOLD permit the user to control output of the REPORT generator to the report file in the case of the OUT-OF-PLAN report. The statements are only required for FUNKTION=PRINT.

In the case of the PLANNED-NET-MODIFICATION report, all selected nets are included in the report.

7.1.8 Allocating the input/output file for FUNKTION=SORT

In the OUT-OF-PLAN report, net data can be sorted according to various criteria. Sorting is only possible if the REPORT generator is assigned an input/output file.

The input/output file must be assigned the following file link names via the BS2000 command ADD-FILE-LINK:

LINK-NAME=SORTIN

or

LINK-NAME=SORTOUT

If FUNKTION=SORT is not called together with FUNKTION=CREATE, several input files may be assigned:

LINK-NAME=SORTIN01

LINK-NAME=SORTIN02

In this case the work files must be assigned correspondingly different names in FUNKTION=CREATE.

Example of commands for FUNKTION=ALL

```
.....  
/ADD-FILE-LINK LINK-NAME=SORTIN,FILE-NAME=AVAS.RPRT.WORK.PNM  
/ADD-FILE-LINK LINK-NAME=SORTOUT,FILE-NAME=AVAS.RPRT.WORK.PNM  
.....
```

Example of commands for FUNKTION=SORT with FUNKTION=PRINT

```
.....  
/ADD-FILE-LINK LINK-NAME=SORTIN01,FILE-NAME=AVAS.RPRT.WORK.OOP1  
/ADD-FILE-LINK LINK-NAME=SORTIN02,FILE-NAME=AVAS.RPRT.WORK.OOP2  
.....  
.....  
/ADD-FILE-LINK LINK-NAME=SORTOUT,FILE-NAME=AVAS.RPRT.WORK.OOP  
.....
```

The REPORT statements SORT-STATE-ORDER and SORT-FIELDS-ORDER permit the user to define the sort sequence in the OUT-OF-PLAN report. The statements are only required for FUNKTION=SORT.

Note

Sorting in the PLANNED-NET-MODIFICATION report is always effected by net name. No statements for the sort sequence are required.

7.2 REPORT statements

AVAS-SYSTEM-ID – Select AVAS system

This REPORT statement serves to select the AVAS system for which a report is to be generated.

The statement is only required for FUNKTION=CREATE.

AVAS-SYSTEM-ID=string

7-character name of the AVAS system for which a report is to be created.

The statement causes the central access tasks to be selected.

If either of the access tasks ZDD or ZDL is not active, the following message is output to SYSOUT:

```
AVS7521 – ZDD string NOT ACTIVE
```

AVAS-USER-ID – Select production plan

This statement serves to select the NPRLIB production plan assigned to the specified user via the AVAS system parameters.

If several production plans have been defined in the system, they must be selected one after the other for evaluation.

The statement is only required for FUNKTION=CREATE.

AVAS-USER-ID=avuser

Identification (up to 8 characters) of an AVAS user who has been defined in the system parameters and assigned the production plan to be evaluated.

The functional authorization of the user for the statement SHOW-PLAN-NET is not checked for report creation. The production plan is evaluated for all user groups specified.

END – Terminate statement sequence for REPORT generator

This statement serves to terminate a sequence of statements to the REPORT generator. Any subsequent statements are not processed, but they are not skipped either.

The statement can be specified for all FUNKTION values.

END

Terminates the stream of REPORT statements to the REPORT generator.

FUNKTION – Select REPORT generator functions

This statement serves to select and execute the REPORT generator functions.

The statement may be entered more than once if a number of functions are to be performed.

If the statement is omitted, the value ALL is assumed.

FUNKTION={ALL / CREATE / SORT / PRINT}

FUNKTION=ALL

The functions CREATE, SORT and PRINT are performed.

FUNKTION=CREATE

Creates a work file (OUTPUT-WORK-FILE) with the requisite records of all nets selected using the REPORT statements NET-NAME and/or PERIOD-NAME.

FUNKTION=SORT

Sorts the OUTPUT-WORK-FILE according to the definitions made in the REPORT statements SORT-STATE-ORDER and SORT-FIELDS-ORDER.

FUNKTION=PRINT

Generates a list of all nets from the allocated OUTPUT-WORK-FILE that were selected via the REPORT statements SELECT-STATUS and THRESHOLD, and outputs this list to the OUTPUT-REPORT-FILE assigned.

NET-NAME – Select nets using net names

This statement enables nets to be selected from the production plan by their net names and transferred to the work file OUTPUT-WORK-FILE.

NET-NAME={*OWN / *ALL / \$x* / \$ug_ / \$ug_netname}

NET-NAME=*OWN

Nets from the user's own user group are selected.

NET-NAME=*ALL

Nets from all user groups of the allocated NPRLIB are selected.

(The *ALL parameter is converted to \$*.)

NET-NAME=\$x*

The nets of all user groups beginning with the specified character are selected.

Example: NET-NAME=\$T*

NET-NAME=\$ug_

Nets belonging to the specified user group are selected.

NET-NAME=\$ug_netname[_yymmdd[_hhmmss]]

If a fully qualified net name is specified, this net only will be selected.

If the net name is entered via a partial qualification (final character *), an overview is displayed of those nets whose names begin with the partial qualification.

The user group must be specified.

The number of nets to be selected can be further limited through the PERIOD-NAME statement.

If the statement is not specified, *OWN is assumed.

If a given list of net names is to be processed, FUNKTION=CREATE can be called up several times with different net names.

The REPORT Generator collates the net data in the specified work file (OPEN=EXTEND).

When both the NET-NAME and PERIOD-NAME statements are specified, only nets that satisfy both selection criteria will be selected.

OUTPUT-REPORT-FILE – Assign BS2000 file name to report file

This statement serves to assign a BS2000 file name to the report file.
This file is the output file for any report to be printed.

The statement is only required for FUNKTION=PRINT.

OUTPUT-REPORT-FILE=filename

File name in accordance with BS2000 conventions.

If it already exists, the report file assigned via this statement is opened by the REPORT generator in OUTPUT=EXTEND mode.

The REPORT generator collects the report data in the specified report file if FUNKTION=PRINT is called more than once using the same file assignment.
This makes it possible to output different evaluations of a work file via the REPORT statements SELECT-STATUS and THRESHOLD to one report file and to print this file by means of the BS2000 command PRINT-FILE. Output is in edited format (operand CONTROL-CHARACTERS=EBCDIC).

When a new report file is to be created, any existing report file must first be deleted.

Alternatively, the statement OUTPUT-REPORT-LINK can be specified.

If neither the statement OUTPUT-REPORT-FILE nor the statement OUTPUT-REPORT-LINK is issued, the following is set:

OUT-OF-PLAN:

OUTPUT-REPORT-FILE=LST.AVAS.OOP.yymmdd.hhmmss

PLANNED-NET-MODIFICATION:

OUTPUT-REPORT-FILE=LST.AVAS.PNM.yymmdd.hhmmss

OUTPUT-REPORT-LINK – Assign report file via link name

This statement serves to assign a BS2000 file link name to the report file. This file is the output file for the report to be printed.

The statement is only required for FUNKTION=PRINT.

OUTPUT-REPORT-LINK=filename

Link name in accordance with BS2000 conventions

If it already exists, the report file assigned via this statement is opened by the REPORT generator in OUTPUT=EXTEND mode.

The REPORT generator collects the report data in the specified report file if FUNKTION=PRINT is called more than once using the same file assignment. This makes it possible to output different evaluations of a work file via the REPORT statements SELECT-STATUS and THRESHOLD to one report file and to print this file by means of the BS2000 command PRINT-DOCUMENT. Output is in edited format (LINE-SPACING=*BY-EBCDIC-CONTROL operand).

When a new report file is to be created, any existing report file must first be deleted.

The OUTPUT-REPORT-LINK statement may be specified as an alternative to the OUTPUT-REPORT-FILE statement.

The link name must be defined via the BS2000 command ADD-FILE-LINK.

If neither the statement OUTPUT-REPORT-FILE nor the statement OUTPUT-REPORT-LINK is issued, the following is set:

OUT-OF-PLAN:

OUTPUT-REPORT-FILE=LST.AVAS.OOP.yymmdd.hhmmss

PLANNED-NET-MODIFICATION:

OUTPUT-REPORT-FILE=LST.AVAS.PNM.yymmdd.hhmmss

OUTPUT-WORK-FILE – Assign BS2000 file name to work file

This statement serves to assign a BS2000 file name to the work file.

The work file is

- the output file for FUNKTION=CREATE
- the input/output file for FUNKTION=SORT
- the input file for FUNKTION=PRINT

The work file must be specified for all functions.

OUTPUT-WORK-FILE=filename

File name in accordance with BS2000 conventions

If it already exists, the work file assigned via this statement is opened by the REPORT generator in OUTPUT=EXTEND mode.

The REPORT generator collects the net data in the specified work file if FUNKTION=CREATE is called more than once using the same file assignment.

When a new work file is to be created, any existing work file must first be deleted.

Alternatively, the statement OUTPUT-WORK-LINK can be specified.

OUTPUT-WORK-LINK – Assign work file via BS2000 link name

This statement serves to assign a BS2000 file link name to the work file.

The work file is

- the output file for FUNKTION=CREATE
- the input/output file for FUNKTION=SORT
- the input file for FUNKTION=PRINT

The work file must be specified for all functions.

OUTPUT-WORK-LINK=filename

Link name in accordance with BS2000 conventions

If it already exists, the work file assigned via this statement is opened by the REPORT generator in OUTPUT=EXTEND mode.

The REPORT generator collects the net data in the specified work file if FUNKTION=CREATE is called more than once using the same file assignment.

When a new work file is to be created, any existing work file must first be deleted.

The link name must be defined using the BS2000 command ADD-FILE-LINK.

The OUTPUT-WORK-LINK statement can be specified as an alternative to the OUTPUT-WORK-FILE statement.

PERIOD-NAME – Select nets using start date PLAN-START

This statement enables nets that are to be transferred to the work file OUTPUT-WORK-FILE to be selected from the production plan using the start date PLAN-START.

PERIOD-NAME={period / dd.mm.yy/hh:mm:ss,dd.mm.yy/hh:mm:ss}

PERIOD-NAME=period

Symbolic name for a stored period.

PERIOD-NAME=dd.mm.yy/hh:mm:ss

Actual date and time data that determines the start and end times of the period (FROM-DATE, TO-DATE).

If only the TO-DATE is to be defined, it must be preceded by a comma.

If TO-DATE is omitted, the end date is assumed to be the same as the start date and the end time is set to 23:59:59.

The number of nets to be selected can be further limited through the NET-NAME statement.

If the statement is omitted, nets are selected without taking the planned start time PLAN-START into account.

When both the NET-NAME and PERIOD-NAME statements are specified, only nets that satisfy both selection criteria will be selected.

REPORT-NAME – Select reports

This statement serves to select a report.

REPORT-NAME={OUT-OF-PLAN / PLANNED-NET-MODIFICATION}

REPORT-NAME=OUT-OF-PLAN

A report of the nets not started according to plan or not executing as planned is to be created.

REPORT-NAME=PLANNED-NET-MODIFICATION

A report of the nets to which changes have been made via AVAS functions either after planning or during execution is to be created.

SELECT-STATUS – Select nets according to status

This statement serves to select those nets, on the basis of their global status, whose records are to be output to the report file.

Nets with status values other than those specified are not included in the report file.

The output scope may be further restricted by means of the REPORT statement THRESHOLD.

This statement is processed for the OUT-OF-PLAN report only.

The statement is only required for FUNKTION=PRINT.

SELECT-STATUS=(status-list)

List of status values of the nets whose records are to be output to the report file. The global status must be specified here (see GLOBAL-STATE, [page 328](#)).

ERROR	Nets in ERROR status.
NO-SUBMIT	Nets not released for processing.
WAITING	Nets released but not yet started.
STARTED	Nets started and not yet terminated, with the exception of nets listed under ERROR.
ENDED	Nets terminated normally.
ABENDED	Nets terminated abnormally.

If the statement is omitted, no nets are selected according to status.

In this case, only an evaluation page showing the net totals in the defined GLOBAL-STATE values is output.

NET-SUMMARY indicates the number of all nets found on the basis of the operand in the NET-NAME statement.

The number of nets is determined via the production plan.

SORT-FIELDS-ORDER – Define sequence of sort fields

This statement serves to define the sequence of the permissible sort fields on the basis of the field name.

This statement is processed for the OUT-OF-PLAN report only.
The statement is only required for FUNKTION=SORT.

SORT-FIELDS-ORDER=(field-list)

List of field names by which the nets are to be sorted.

NET-NAME	Sorting is performed by net names. The net name contains the user group and the PLAN-START date.
E-DELAY	Sorting is performed by delays relative to the prospective start time, EARLIEST-START of the net. Sorting is done in descending order.
EARLIEST	Sorting occurs in ascending order according to the specified values for EARLIEST-START. The current value of the net at the time of journal file evaluation is taken into account.
STATE	Sorting is based on the global status of the nets (see page 328). The field name STATE need not be specified: it is always the first sort field, even if specified at a different point in the list.

If the statement is omitted, the sort sequence is based on the internally defined value:

SORT-FIELDS-ORDER=(STATE,NET-NAME,E-DELAY,EARLIEST)

If only some of the field names are specified, the others are added in the internally defined sequence.

SORT-STATE-ORDER – Define net sequence for report file

This statement defines the sequence in which the nets are to be sorted on the basis of their current global status in the journal file.

This statement is processed for the OUT-OF-PLAN report only.
The statement is only required for FUNKTION= SORT.

SORT-STATE-ORDER=(status-list)

List of status values by which the nets are to be sorted. The global status must be specified here (see GLOBAL-STATE, [page 328](#)).

ERROR	Nets in ERROR status.
NO-SUBMIT	Nets not released for processing.
WAITING	Nets released but not yet started.
STARTED	Nets started and not yet terminated, with the exception of nets listed under ERROR.
ENDED	Nets terminated normally.
ABENDED	Nets terminated abnormally.

If the statement is omitted, the sort sequence is based on the internally defined value:

SORT-STATE-ORDER=(ERROR,NO-SUBMIT,WAITING,STARTED,ENDED,ABENDED)

If only some of the status values are specified, the others are added in the internally defined sequence.

SYSTEM-PASSWORD – Sign on to central access task

The password of the AVAS system is required for signing on to the central access task ZDD. For ZDD signon, the REPORT generator needs the password REOPW=password given during startup of the ZDD task.

This statement is only required for FUNKTION=CREATE.

SYSTEM-PASSWORD={*STD / password}

SYSTEM-PASSWORD=*STD

This value must be specified if the central access task ZDD is started without the REOPW statement or via REOPW=*STD.

SYSTEM-PASSWORD=password

Specifies the password with which the central access task for reorganization was started.

C'....' 4 characters

or

X'.....' 8 characters

THRESHOLD – Select nets according to delay

This statement serves to select those nets, by their delay, whose records are to be output to the report file.

Nets for which the specified delay has not yet been reached are not included in the report file.

The output scope may be further restricted via the REPORT statement SELECT-STATUS.

This statement is processed for the OUT-OF-PLAN report only.

The statement is only required for FUNKTION=PRINT.

THRESHOLD=(e-delay,I-delay)

e-delay Delay relative to EARLIEST-START
 000 through 999 = time in minutes

I-delay Delay relative to LATEST-START
 000 through 999 = time in minutes

Both values must be specified in each case.

If all nets are to be output in accordance with the list defined via SELECT-STATUS, the following entry is required:

THRESHOLD=(000,000)

If the statement is omitted, the following internal value is set:

THRESHOLD = (002,002)

USER-PASSWORD – Specify user password

Each user is assigned a signon password via the system parameters.

The password of the user defined with AVAS-USER-ID must be specified here.

This statement is only required for FUNKTION=CREATE.

USER-PASSWORD=password

Specifies the user password assigned via the system parameters.

C'.....' 8 characters

7.3 PLANNED-NET-MODIFICATION report

The following REPORT statements are processed for the PLANNED-NET-MODIFICATION report:

```
REPORT-NAME=PLANNED-NET-MODIFICATION
FUNKTION={ALL / CREATE / SORT / PRINT}
AVAS-SYSTEM-ID=string
SYSTEM-PASSWORD={*STD / password}
AVAS-USER-ID=avuser
USER-PASSWORD=password
NET-NAME={*OWN / *ALL / $x* / $ug_ / $ug_netname}
PERIOD-NAME=period
OUTPUT-WORK-FILE=filename
OUTPUT-WORK-LINK=filename
OUTPUT-REPORT-FILE=filename
OUTPUT-REPORT-LINK=filename
END
```

If REPORT statements are entered which are required by the OUT-OF-PLAN report only, they are ignored.

If errored or illegal statements are encountered, they are logged as invalid and processing of the input is aborted.

The following net data is shown in the list generated via FUNKTION=PRINT:

NET-NAME Complete net name with date and time extensions as formed in the AVAS function CREATE-PLAN-NET.

STATE Current status of the net at the time of report generation.

MOD-TYPE Type of modification and the AVAS function used to perform the modification.

AVAS functions:

- MODIFY-PLAN NET for modified net parameters
- EDIT-PROD-JOB for a modified task
- SUBMIT-NET for modified net parameters
- MODIFY-SUBMIT-NET for modified net parameters
- MODIFY-SUBMIT-JOB for a modified task

AVAS actions:

- CHANGED for modified parameters or JCL statements
- INSERTED for inserted JCL statements
- DELETED for a deleted task or a JCL statement
- SHIFTED for a shifted net

MOD-TIME Time of modification in the form ddmmyy/hh:mm

OBJECT Name of the modified object:
 NET-NAME
 JOB-NAME
 or
 CONDITION-NAME

USER Name of the AVAS user who initiated the modification.

7.4 OUT-OF-PLAN report

The following REPORT statements are processed for the OUT-OF-PLAN report:

```
REPORT-NAME=OUT-OF-PLAN
FUNKTION={ALL / CREATE / SORT / PRINT}
AVAS-SYSTEM-ID=string
SYSTEM-PASSWORD={*STD / password}
AVAS-USER-ID=avuser
USER-PASSWORD=password
NET-NAME={*OWN / *ALL / $x* / $ug_ / $ug_netname}
PERIOD-NAME=period
OUTPUT-WORK-FILE=filename
OUTPUT-WORK-LINK=filename
SORT-STATE-ORDER=(status-list)
SORT-FIELDS-ORDER=(field-list)
SELECT-STATUS=(status-list)
THRESHOLD=(e-delay, l-delay)
OUTPUT-REPORT-FILE=filename
OUTPUT-REPORT-LINK=filename
END
```

If errored or illegal statements are encountered, they are logged as invalid and processing of the input is aborted.

The sort sequence for the OUT-OF-PLAN report can be defined via the SORT-STATE-ORDER and SORT-FIELDS-ORDER statements.

Nets for output to the report file can be selected using the REPORT statements SELECT-STATUS and THRESHOLD.

NET-SUMMARY also lists the nets not selected.

The following net data is shown in the list generated via FUNKTION=PRINT:

GLOBAL-STATE=

Designates the global status of the nets listed on one page. The AVAS status STATE of the nets is aggregated into the global status values:

NO-SUBMIT	corresponds to	TOCREATE NOTTOCREATE PARTIALLY CREATED
WAITING	corresponds to	SUBMITTED WAITING OPWAIT START (after START-NET) HOLD (before START) RESUMED (before START)
STARTED	corresponds to	RUNNING CONDWAIT HOSTWAIT RESTARTED HOLD (after START) RESUMED (after START)
ERROR	corresponds to	ERROR
ENDED	corresponds to	ENDED IGNORED
ABENDED	corresponds to	ABENDED

The above values for GLOBAL-STATE are also the permissible entries for the REPORT statements SORT-STATE-ORDER and SELECT-STATUS.

NET-NAME	Complete net name with date and time extensions as formed in the AVAS function CREATE-PLAN-NET.
STATE	Current AVAS status of the net at the time of report generation.

L-E-DELAY	Delay of the nets relative to LATEST-START and EARLIEST-START. L-DELAY relative to LATEST-START is identified only by an asterisk "*" if the LATEST-START time has been exceeded. E-DELAY relative to LATEST-START is identified by a time specification showing hours and minutes in the form hhhh:mm
REAL-START	Start time of the net in the form ddmmyy/hh:mm This value is only shown if the net has been started. If no date appears for nets in ENDED or ABENDED status, the net status has resulted from NET-DELAY-SOLUTION=IGNORE or NET-DELAY-SOLUTION=CANCEL or from CANCEL-NET.
REAL-END	Termination time of the net in the form ddmmyy/hh:mm This value is only shown if the net has reached the normal end.
EARLIEST	Current net value for EARLIEST-START in the form ddmmyy/hh:mm If the EARLIEST-START value was changed after planning of the net, this is the value last assigned.
LATEST-START	Current net value for LATEST-START in the form ddmmyy/hh:mm If the LATEST-START value was changed after planning of the net, this is the value last assigned.
SYS-ID	Name of the AVAS system in which the net was planned for processing.
USER	Name of the AVAS user who initiated the change. If the journal record on which evaluation is based was output by AVAS execution control, the name of the RUN-CONTROL-SYSTEM is stored under USER.

If jobs have terminated abnormally in a net (JOB-STATUS=ERROR has been set), each error case triggers output of a log record in the following format:

ERROR-JOB= Job name according to entry in net structure

JV: Contents of the task job variable

or

AVSnnnn Contents of the error message

TSN= TSN of the job if the ENTER call could be executed

The following reports can be generated using the REPORT generator and
 FUNKTION=PRINT:

List L06001: OUT-OF-PLAN

AVAS-Vnn.yxmm/L06001 REPORT-Generator LIST:OUT-OF-PLAN DATE:ttmmjj TIME:hh:mm PAGE:nnnn

GLOBAL-STATE =

NET-NAME	STATE	L-E-DELAY	REAL-START	REAL-END	EARLIEST	LATEST-START	SYS-ID	USER
.	.							
ERROR-JOB=			JV:					TSN=
.	.							
.	.							

AVAS-Vnn.yxmm/L06001 REPORT-Generator LIST:OUT-OF-PLAN DATE:ttmmjj TIME:hh:mm PAGE:nnnn

NET-SUMMARY	ERROR	NO-SUBMIT	WAITING	STARTED	ENDED
nnn	nnn	nnn			
	pp%	pp%			

REPORT-NAME= OUT-OF-PLAN

FUNKTION=

AVAS-SYSTEM-ID=

AVAS-USER-ID=

NET-NAME=

PERIOD-NAME=

OUTPUT-WORK-FILE=

SORT-FIELDS-ORDER=

SORT-STATE-ORDER=

SELECT-STATUS=

THRESHOLD=

OUTPUT-REPORT-FILE=

List L06002: PLANNED-NET-MODIFICATION

AVAS-Vnn.yxmm/L06002	REPORT-Generator	LIST:PLANNED-NET-MODIFICATION	DATE:ttmmjj	TIME:hh:mm	PAGE:nnnn
NET-NAME	STATE	MOD-TYPE	MOD-TIME	OBJECT	USER
.
.
.
.
.

AVAS-Vnn.yxmm/L06002 REPORT-Generator LIST:PLANNED-NET-MODIFICATION DATE:ttmmjj TIME:hh:mm PAGE:nnnn

REPORT-NAME= PLANNED-NET-MODIFICATION

FUNKTION=

AVAS-SYSTEM-ID=

AVAS-USER-ID=

NET-NAME=

PERIOD-NAME=

OUTPUT-WORK-FILE=

SORT-FIELDS-ORDER=

OUTPUT-REPORT-FILE=

7.5 Procedures and libraries for the AVAS reports

Procedures.....:	The library SYSPRC.AVAS.085 contains the J-element AVS.REPORT for execution of the REPORT generator; being only a sample procedure, this must be tailored to suit individual requirements. It is advisable to store the statements for the reports in a SAM file and to assign this file via SYSDTA.
Libraries.....:	Execution of the REPORT generator furthermore requires the libraries SYSLNK.AVAS.085 and SYSPRG.AVAS.085.SYSTEM.

Example

```

/BEGIN-PROC LOGG=NO,PAR=YES(PROC-PAR=(&USERID=,&REPORT=,&DELETE=,&AVSYID=,-
/&AVSYPW=*STD,&AVUSID=,&AVUSPW),ESC-CHAR=C'&')
/REMARK *****
/REMARK &USERID ::= USER ID OF THE AVAS INSTALLATION
/REMARK &REPORT ::= OOP / PNM (OUT-OF-PLAN / PLAN.-NET-MODI.)
/REMARK &DELETE ::= YES / NO (DELETE EXISTING FILES ?)
/REMARK >
/REMARK &AVSYID ::= AVAS SYSTEM ID
/REMARK &AVSYPW ::= AVAS SYSTEM PASSWORD
/REMARK &AVUSID ::= AVAS USER ID
/REMARK &AVUSPW ::= AVAS USER PASSWORD (SYNTAX: C'_____')
/REMARK *****
/SKIP-COM .DEL#&DELETE
/.DEL#YES REMARK -----> DELETE FILES
/CREATE-FILE FILE-NAME=AVAS.WORK.REPORT.&REPORT
/DELETE-FILE FILE-NAME=AVAS.WORK.REPORT.&REPORT
/CREATE-FILE FILE-NAME=AVAS.FILE.REPORT.&REPORT
/DELETE-FILE FILE-NAME=AVAS.FILE.REPORT.&REPORT
/.DEL#NO REMARK -----> DO NOT DELETE FILES
/SKIP-COM .REP#&REPORT
/.REP#OOP REMARK -----> OOP REPORT
/ADD-FILE-LINK LINK-NAME=SYSLNK,FILE-NAME=$&USERID..SYSLNK.AVAS.085
/ADD-FILE-LINK LINK-NAME=SORTIN,FILE-NAME=AVAS.WORK.REPORT.&REPORT
/ADD-FILE-LINK LINK-NAME=SORTOUT,FILE-NAME=AVAS.WORK.REPORT.&REPORT
/ASSIGN-SYSDTA TO=*SYSCMD
/START-PROG FROM-FILE=*PHASE(LIB=$&USERID..SYSPRG.AVAS.085.SYSTEM,-
/ELEMENT=AVAS.SYS.LOAD.REPORT)

```

```

REPORT-NAME          = OUT-OF-PLAN
FUNKTION             = ALL
AVAS-SYSTEM-ID      = &AVSYID
SYSTEM-PASSWORD     = &AVSYPW
AVAS-USER-ID        = &AVSUSID
USER-PASSWORD       = &AVUSPW
NET-NAME            = *ALL
OUTPUT-WORK-FILE    = AVAS.WORK.REPORT.&REPORT
SORT-STATE-ORDER    = (ERROR,NO-SUBMIT,WAITING,STARTED,ENDED,ABENDED)
SORT-FIELDS-ORDER   = (STATE,NET-NAME,E-DELAY)
SELECT-STATUS       = (ERROR,NO-SUBMIT,WAITING,STARTED)
THRESHOLD           = (5,2)
OUTPUT-REPORT-FILE  = AVAS.FILE.REPORT.&REPORT
END
/ASSIGN-SYSDTA TO-FILE=*PRIMARY
/SHOW-FILE-ATTR FILE-NAME=AVAS.FILE.REPORT.&REPORT
/WRITE-TEXT TEXT='--> FILE:AVAS.FILE.REPORT.&REPORT CREATED <--'
/SKIP-COM .REP#END
/SET-JOB-STEP
/SKIP-COM .ABEND
/.REP#PNM REMARK -----> PNM REPORT
/ADD-FILE-LINK LINK-NAME=SYSLNK,FILE-NAME=$&USERID..SYSLNK.AVAS.085
/ADD-FILE-LINK LINK-NAME=SORTIN,FILE-NAME=AVAS.WORK.REPORT.&REPORT
/ADD-FILE-LINK LINK-NAME=SORTOUT,FILE-NAME=AVAS.WORK.REPORT.&REPORT
/ASSIGN-SYSDTA TO-FILE=*SYSCMD
/START-PROG FROM-FILE=*PHASE(LIB=$&USERID..SYSPRG.AVAS.085.SYSTEM,-
/ELEMENT=AVAS.SYS.LOAD.REPORT)
REPORT-NAME          = PLANNED-NET-MODIFICATION
FUNKTION             = ALL
AVAS-SYSTEM-ID      = &AVSYID
SYSTEM-PASSWORD     = &AVSYPW
AVAS-USER-ID        = &AVSUSID
USER-PASSWORD       = &AVUSPW
NET-NAME            = *ALL
OUTPUT-WORK-FILE    = AVAS.WORK.REPORT.&REPORT
SORT-FIELDS-ORDER   = (NET-NAME)
OUTPUT-REPORT-FILE  = AVAS.FILE.REPORT.&REPORT
END
/ASSIGN-SYSDTA TO-FILE=*PRIMARY
/SHOW-FILE-ATTR FILE-NAME=AVAS.FILE.REPORT.&REPORT
/WRITE-TEXT TEXT='--> FILE:AVAS.FILE.REPORT.&REPORT CREATED <--'
/SKIP-COM .REP#END
/SET-JOB-STEP
/.ABEND CANCEL-PROC
/.REP#PNM REMARK -----> END REPORT
/END-PROC

```

8 Batch functions

Frequently recurring functions of AVAS control may also be implemented by means of procedures. For this purpose, batch-oriented statements are available for a whole range of AVAS dialog functions.

As in the AVAS interactive system, a task can be loaded which reads batch statements via the BS2000 system file SYSDTA and logs the result via the system file SYSOUT.

The batch process logs in to AVAS like the dialog process (see the manual “AVAS Statements” [2]), i.e. with the SIGNON data AVAS-USER-ID, PASSWORD and AVAS-SYSTEM-ID.

All messages in connection with a batch function are identical to the corresponding interactive messages.

The execution and result of the batch functions are analogous to those of the AVAS interactive functions (qv).

A batch statement initiates immediate processing and output of a log. This corresponds to the interactive functions

- request an overview
- mark all elements with Y
- operation EXECUTE.

Marks and operations cannot be entered.

The parameter for selecting the element (NET-NAME, ELEMENT-NAME, etc.) must therefore be present in all batch statements in which the PERIOD-NAME parameter is not permitted.

Some batch statements, however, are able to accept the required parameters from selected mask fields.

Any functional deviations from the interactive functions are described under the batch statements concerned.

Specification of the RUN-CONTROL-SYSTEM parameter is omitted if the requested statement does not permit this parameter. In any subsequent statement which requires the RUN-CONTROL-SYSTEM parameter, the default value is assumed if the parameter is not defined.

The same applies to the CANCEL-TYPE parameter.

The following batch statements are supported:

ADD-JOB-LOG
CANCEL-NET
CHANGE-NET-DESCRIPTION
COPY-ELEMENT
CREATE-PERIOD
CREATE-PLAN-NET
CREATE-PROD-JOB
CREATE-PROD-NET
DELETE-DOCUMENT
DELETE-JOB
DELETE-JOB-LOG
DELETE-NET-DESCRIPTION
DELETE-PERIOD
DELETE-PLAN-NET
DELETE-PROD-JOB
END
HOLD-NET
MODIFY-COND-DESCRIPTION
REPEAT-NET
RESTART-NET
RESUME-NET
SIGNON
SUBMIT-NET

All other AVAS functions will be rejected as unknown at the batch interface.

During the processing of batch statements, all error messages are output after SYSOUT. Job switch 30 or 31 is set depending on the gravity of the error (see [page 368](#)).

8.1 Batch statements

Batch statements and their parameters must be input in separate lines, i.e. the only acceptable syntax format is:

```
statementname  
  operand1=value  
  operand2=value
```

The name of the batch statement and the operand entries may be followed by a comma.

The batch statements need not begin in the first column of an input record.

Entries that do not contain “=” are interpreted as a batch statement.

An asterisk in the first column of an input record is regarded as a comment, i.e. the record will be skipped.

ADD-JOB-LOG – Add log data

The batch statement ADD-JOB-LOG enables logs to be added where the log entry has the status CREATED, ASSIGNED, ERROR or ADDED. If only the net name is specified when the batch is called, only log entries with the status ASSIGNED and ERROR are processed.

```
ADD-JOB-LOG  
  NET-NAME=[$ug_]netname  
  [CATID=cat-id  
  TSN=tsn  
  DATE=yyyymmdd  
  JOBLOG-NAME={*NONE / filename}  
  [EXTEND={YES / NO / NEW}]  
  [INPUT-FILE=filename]]
```

For the meaning of the operands and operand values, see the dialog function description under the AVI019 mask (see manual “AVAS Statements” [2]).

The NET-NAME operand must be specified; its value can be partially qualified.

Only if the operand NET-NAME has been specified with a fully qualified net name are the other operands also permissible.

After a batch statement ADD-JOB-LOG, no further AVAS-BATCH statements can be processed under a BS2000 command START-PROG, because the JOB-LOG files are read in via SYSDTA and it is not yet possible to switch back to the file containing the AVAS-BATCH statements.

If several logs are to be read in via AVAS-BATCH statements, AVAS-BATCH should be called up the appropriate number of times.

Example

```
/ BEGIN-PROC ....  
...  
/ REMARK    TRANSFER LOG 1  
/ CALL-PROC SYSPRC.AVAS.085(AVS.BATCH), -  
/ PROC-PAR=(USERID=<userid>, CMDFILE=AVAS-BATCH.1)  
/ REMARK    TRANSFER LOG 2  
/ CALL-PROC SYSPRC.AVAS.085(AVS.BATCH), -  
/ PROC-PAR=(USERID=<userid>, CMDFILE=AVAS-BATCH.2)  
...  
/ END-PROC
```

The structure of the AVS.BATCH procedure is described in [section “Procedures and libraries for the batch functions” on page 366](#).

Processing log**The log record**

```
AVS8521 'ADD-JOB-LOG:netname RESULT:result'
```

or an appropriate error message is output for each net processed.

CANCEL-NET – Cancel or abort net because of error

The batch statement CANCEL-NET enables processing of a net to be canceled. The status the net has to have before the function is executed and the status it has afterwards are described under the dialog function.

CANCEL-NET

```
NET-NAME=[$ug_]netname  
[RUN-CONTROL-SYSTEM=avak]  
[CANCEL-TYPE={SOFT / HARD}]  
[PERIOD-NAME={period / (yymmdd/hhmmss,yymmdd/hhmmss)}]  
[KILL-JOBS={NO / YES}]
```

The NET-NAME parameter must be specified; its value can be partially qualified.

If the RUN-CONTROL-SYSTEM parameter is omitted, the name of the run control system is taken from the signon data.

If the CANCEL-TYPE parameter is omitted, the default value defined by the generation parameter is assumed.

The PERIOD-NAME parameter is only permitted in conjunction with a partially qualified net name.

If the KILL-JOBS parameter is not specified, the NO setting applies by default.

Processing log

The log record

```
AVS8521 'CANCEL-NET:netname RESULT:result'
```

or an appropriate error message is output for each net that is canceled.

CHANGE-NET-DESCRIPTION – Make global changes to nets

The batch statement CHANGE-NET-DESCRIPTION enables certain net parameters in the NETLIB user library to be modified.

The parameters to be modified and the permissible values are described under the dialog function of the same name.

```
CHANGE-NET-DESCRIPTION
NET-NAME=[$ug_]netname
[RETENTION-PERIOD=value]
[OLD-USER=value
NEW-USER=value
OLD-PASSWORD=value
NEW-PASSWORD=value]
[OLD-ACCOUNT=value
NEW-ACCOUNT=value]
[OLD-CLASS=value
NEW-CLASS=value]
[OLD-LOG=value
NEW-LOG=value]
[OLD-CAT=value
NEW-CAT=value]
```

The NET-NAME parameter must be specified; its value can be partially qualified.

The subsequent list of parameters to be modified must be entered as pairs of parameters (OLD-... and NEW-...).

The keys words *DEL and *INS must be specified for any values that are omitted:

```
e.g.: OLD-ACCOUNT=*INS
      NEW-ACCOUNT=AVAS20A
```

If one of the parameters USER and/or PASSWORD is to be modified, both parameters, with their old and new values, must be entered each time.

The parameters OLD-para= and NEW-para= correspond to the parameter fields in the AVN007 mask.

Note

Batch processing is terminated with an error message if an incorrect parameter is detected. The message AVS5051, AVS5052, AVS5054 or AVS5240 will be output.

Processing log

The log record

AVS8521 'CHANGE-NET-DESCRIPTION:netname RESULT:result'

or an appropriate error message is output for each net modified.

COPY-ELEMENT – Copy library elements

The batch statement COPY-ELEMENT copies elements of a PLAM library to an AVAS library or elements of an AVAS library to an external PLAM library.

Input/output of SAM files is not supported.

When MODE=LIBIN and AVAS-USER-LIB= NETLIB are specified, the batch function COPY-ELEMENT checks the net using the internal CHECK function, which validates the structure and restarts variants.

In addition, the parameters (STR/NET) are checked and the record sequence examined to see that it conforms to AVAS conventions. What these checks entail is described in manual “AVAS Statements” [2]. If the checks are completed successfully without detecting any errors, the message AVS5810 ELEMENT SATISFIES AVAS CONVENTIONS is output. A message is output each time an error is detected.

Messages are sent to a PRINT file, which is assigned by means of the BS2000 command /SET-FILE-LINK LINK=AVASPRT, FILE-NAME=filename. EXTEND=YES is set at the same time. If no file is assigned, the messages are output to the file LST.AVAS.ug.avuser.yymmdd.hhmmss.

The checking process will be canceled if CHECK detects an error that makes further checking pointless (consequential errors). CHECK uses a weighting system for each error it detects. If errors are detected, the code for the most serious error is returned as a return code to the batch statement and is stored as a check character in the net description (NETLIB).

Before the first error message in each net containing errors is output, a start record specifying the name of the net is output. In the case of an internal call to CHECK, the start record will contain not only the name of the library from EXTERNAL-FILE but also the OUTPUT net name.

```
AVS6050 START-CHECK <libname> <netname>
```

An end message is output if all the checks were performed (error code < 5).

```
AVS6051 END-CHECK <libname> <netname>
```

An error message is output if the checks were canceled (error code = 5).

```
AVS6052 CANCEL-CHECK <libname> <netname>
```

If an error was detected that can be assigned to a structure element, a job/cond. identification record containing the index and the structure name is output before the error message.

```
AVS6040 IDENTIFICATION: <ind> <structurename> <record type>
```

At least one record is output for each error that is detected. The user can examine the log using EDT.

COPY-ELEMENT

```
MODE={LIBOUT / LIBIN}  
AVAS-USER-LIBRARY={NETLIB / JCLLIB / DOCLIB / CALLIB}  
EXTERNAL-FILE=libname  
[USER-GROUP=$ug]  
[ELEMENT-NAME={element / group}]  
[OVERWRITE={NO / YES}]
```

The MODE parameter is only accepted with one of the above values.

The USER-GROUP parameter is only accepted in conjunction with MODE=LIBIN.

The other parameters are fields of the AVS011 mask (see dialog statement).

Processing log

The log record

```
AVS8521 'COPY-ELEMENT: elementname RESULT: result'
```

or an appropriate error message is output for each element copied.

CREATE-PERIOD – Create period

The batch statement CREATE-PERIOD creates an entry in the period file containing the specified PERIOD-NAME and the default time limits for the period.

Periods with a variable start and end date (standard periods) can be defined using this batch statement.

Wildcard characters can be used for creating variable periods when defining periods. The values of the wildcards are determined on the basis of the current date.

Wildcards can only be used in the date. The dates always relate to the current year. The time can also be specified for periods like TODAY.

Variable periods with wildcards can only be read in using the batch statement CREATE-PERIOD and deleted using the batch statement DELETE-PERIOD.

Variable periods are identified by TYPE=VAR in the display.

They cannot be modified using the dialog function MODIFY-PERIOD.

Permitted wildcards:

Day

- dd = real day
If the month is specified relatively + – , the highest permitted value is 28.
- +0 = current value from the current date
- d = current day -d days
- +d = current day +d days
- ++ = last day of the month
- . = separator for day and month

Month

- mm = real month
A real month is only permitted with a real day.
- +0 = current value from the current date
- m = current month -m months
- +m = current month +m months
m = 0, 1 or 2
- . = Separator for month and year

Year

- +0 = Current value from the date
A real year cannot be specified. If a real day and a real month are specified, the year must be specified using .+0.
- = only in relation to ++ for Day:
If the day falls on a Sunday, the previous Saturday is used.
- = only in relation to ++ for Day:
If the day falls on a Saturday or Sunday, the previous Friday is used.
- * = only in relation to +0.+0. (current date):
The current time is taken as the current date.

Day of the week in week

- *n/±w
- *n = nth. day of the week
1 = Monday
7 = Sunday
- / = Separator for day of the week and week
- /+w = current week +w weeks
- /-w = current week -w weeks
w = 0 ... 9
- *3/+0 = Wednesday of the current week

Examples

```
PERIOD-NAME=TODAY
PERIOD-START-DATE=+0.+0.
PERIOD-END-DATE=+0.+0.
```

TODAY= from 00:00 to 23:59 hrs. on the current day

```
PERIOD-NAME=TODAY.COMPLETED
PERIOD-START-DATE=+0.+0.
PERIOD-END-DATE=+0.+0.*
```

TODAY.COMPLETED= from 00:00 hrs. today to the current time on the current day

```
PERIOD-NAME=TODAY.REMAINING-WORK
PERIOD-START-DATE=+0.+0.*
PERIOD-END-DATE=+0.+0.
```

TODAY.REMAINING-WORK= from the current time on the current day to 23:59 hrs. on the day

```
PERIOD-NAME=TOMORROW
PERIOD-START-DATE=+1.+0.
PERIOD-END-DATE=+1.+0.
```

TOMORROW= **from 00:00 to 23:59 hrs. on the following day**

```
PERIOD-NAME=WEEK
PERIOD-START-DATE=*7/-1.
PERIOD-START-TIME=21:00
PERIOD-END-DATE=*7/+0.
PERIOD-END-TIME=20:59
```

WEEK= **from Sunday of last week to Sunday of the current week**

```
PERIOD-NAME=ULTIMO
PERIOD-START-DATE=++.+0.--
PERIOD-END-DATE=++.+0.--
```

ULTIMO= **the last day of the current month (from 00:00 to 23:59 hrs.)**
If the day falls on a Saturday or Sunday, the previous Friday is used.

```
PERIOD-NAME=ULTIMO.LONG
PERIOD-START-DATE=++.+0.--
PERIOD-END-DATE=++.+0.
```

ULTIMO.LONG= **the last day of the current month (from 00:00 to 23:59 hrs.)**
If the day falls on a Saturday (or Sunday), ULTIMO includes the
days Friday and Saturday (and Sunday).

Notes

- The user can define the names and the start and end times of the periods (see the description of naming conventions in the manual “AVAS Functions and Tables” [1]).
- If a variable date is specified (PERIOD-START-DATE or PERIOD-END-DATE), the other date must also be variable.
- When defining your own variable periods, it is important to ensure that PERIOD-START-DATE and PERIOD-END-DATE always describe a positive time span. Otherwise, the message AVS5188 is displayed and the current statement is aborted when you try to access the period.

Example PERIOD-NAME=ERROR.MONTH.COLPLETED
PERIOD-START-DATE=02.+0.
PERIOD-END-DATE=+0.+0.

This period causes the message AVS5188 to be displayed on the first day of the month, but can be used on all other days.

```

PERIOD-NAME=ERROR.WEEK.COMPLETED
PERIOD-START-DATE=*1/+0.
PERIOD-END-DATE=-1.+0.

```

This period causes the message AVS5188 to be displayed every Monday because START-DATE is Monday 00:00 hrs. and END-DATE is Sunday 23:59 hrs.

- **No** wildcards can be used for the PERIOD-NAME parameter.
- The calendar is not accessed in order to determine the real dates of a variable period. This is particularly important if ULTIMO falls on a holiday (not Saturday or Sunday) or if the Friday before ULTIMO is a holiday (when START-/END-DATE=++.+0.-- is used).
- The real dates of a variable period are determined at the start of an AVAS statement and are only obtained again when an parameter is modified.

The package includes the AVS.GENPERIOD procedure (an element of the library SYSPRC.AVAS.085), which can be used to create the following standard periods with variable limit dates in the PERDAT file:

PERIOD-NAME	PERIOD-START- DATE	PERIOD-START- TIME	PERIOD-END- DATE	PERIOD-END- TIME
TODAYM3	-3.+0.	00.00	-3.+0.	23.59
TODAYM2	-2.+0.	00.00	-2.+0.	23.59
YESTERDAY	-1.+0.	00.00	-1.+0.	23.59
TODAY	+0.+0.	00.00	+0.+0.	23.59
TOMORROW	+1.+0.	00.00	+1.+0.	23.59
TODAYP2	+2.+0.	00.00	+2.+0.	23.59
TODAYP3	+3.+0.	00.00	+3.+0.	23.59
*				
WEEKV	*1/-1.	00.00	*7/-1.	23.59
WEEKE	*1/+0.	00.00	+0.+0.	00.01
WOCHE	*1/+0.	00.00	*7/+0.	23.59
WEEKR	+0.+0.	00.00	*7/+0.	23.59
WEEKN	*1/+1.	00.00	*7/+1.	23.59
*				
MONTHV	01.-1.	00.00	++.-1.	23.59
MONTHE	01.+0.	00.00	+0.+0.	00.01
MONTH	01.+0.	00.00	++.+0.	23.59
MONTHR	+0.+0.	00.00	++.+0.	23.59
MONTHN	01.+1.	00.00	++.+1.	23.59
*				
YEARE	01.01.	00.00	+0.+0.	00.01
YEARR	+0.+0.	00.00	++..12.	23.59

If a period with one of the default names already exists in the period file, no standard periods are entered using this name. In this case, the AVS.GENPERIOD procedure reports 'AVAS run with warning'.

CREATE-PERIOD

```
PERIOD-NAME=period
PERIOD-START-DATE=dd.mm.yy
[PERIOD-START-TIME=hh:mm]
[PERIOD-END-DATE=dd.mm.yy]
[PERIOD-END-TIME=hh:mm]
```

The PERIOD-NAME parameter must be specified in fully qualified form.

If the PERIOD-START-TIME parameter is omitted, PERIOD-START-TIME=00:00 will be assumed.

If the PERIOD-END-DATE parameter is omitted, PERIOD-END-DATE = PERIOD-START-DATE will be assumed.

If the PERIOD-END-TIME parameter is omitted, PERIOD-END-TIME=23:59 will be assumed.

Processing log

The log record

```
AVS8521 'CREATE-PERIOD: period RESULT: result'
```

or an appropriate error message is output for each period created.

The value for `period` is the name of the period in which the period file is to be created.

CREATE-PLAN-NET – Plan net processing

The batch statement CREATE-PLAN-NET supports two different types of net processing:

- planning nets via the calendar
- planning a net without using the calendar

If the batch statement CREATE-PLAN-NET is executed with the parameter OPERATION=PRINT, the nets are not planned; instead a list of all nets selected for planning is created.

Planning nets via the calendar

All nets to be planned using this batch function must be assigned to a symbolic (Symdat) or real start date in the net description.

Planning via the calendar requires specification of the PERIOD-NAME parameter.

The RUN-CONTROL-SYSTEM parameter corresponds to the value in the parameter field ALTERN-RUN-CONT-SYS of the AVP011 mask.

The EARLIEST-START and SYMDAT-NAME parameters are not permitted. The calendar designated with CALENDAR-NAME must exist in the CALLIB.

CREATE-PLAN-NET

PERIOD-NAME={period / (date/time,date/time)}

NET-NAME=[\$ug_]netname

CALENDAR-NAME= calendar

[RUN-CONTROL-SYSTEM={*STD / avak}]

[OPERATION=PRINT]

[ALTERN-NET-NAME=[\$bk_]netname]

Planning a net without using the calendar

Calendar-independent planning requires fully qualified specification of the net name (NET-NAME parameter). The SYMDAT-NAME parameter merely serves to select the structure elements in the net; it corresponds to the value in parameter field SYMDAT-NAME of the AVP001 mask.

If the EARLIEST-START parameter is not specified, the current time of day is assumed as EARLIEST-START.

If a planned net turns out to be faulty, the date and time of day are not supplied in the processing log.

The PERIOD-NAME and RUN-CONTROL-SYSTEM parameters are not permitted.

CREATE-PLAN-NET

```
NET-NAME=[$ug_]netname
[SYMDAT-NAME=symdat]
[EARLIEST-START=(dd.mm.yy/hh:mm:ss)]
[OPERATION=PRINT]
[ALTERN-NET-NAME=[$bk_]netname]
```

Processing log

The log record

```
AVS8521 'CREATE-PLAN-NET:netname date/time RESULT:result'
```

or an appropriate error message is output for each planned net.

If an error occurs, the value for `date/time` may be omitted.

Logging processing when OPERATION=PRINT is specified.

List AVL014 (list of nets selected for planning) is output.

Output is in the file assigned via LINK=AVASPRT (see the PRINT operation in the “AVAS Statements” manual [2]).

CREATE-PROD-JOB – Create static B2000 jobs and S procedures

The batch statement CREATE-PROD-JOB is used to create static jobs and S procedures in the AVAS user library JMDLIB.

```
CREATE-PROD-JOB
  INPUT-NAME=[$ug_]jobname
  [OVERWRITE={YES / NO}]
  [USER-PAR-FILE={filename / libname(element[,type])}]
  [OUTPUT-NAME=jobname]
```

The INPUT-NAME parameter must be specified; the parameter value may be partially qualified.

If the INPUT-NAME parameter is partially qualified, OUTPUT-NAME=INPUT-NAME must be set for all created BS2000 jobs and S procedures.

If the system user group is entered as INPUT-NAME, the user group specified in the signon data is used for all created jobs.

The OVERWRITE parameter is identical to the value in the OVERWRITE parameter field in the AVM013 mask.

The USER-PAR-FILE parameter is identical to the value in the USER-PAR-FILE parameter field in the AVM013 mask.

The OUTPUT-NAME parameter is only permitted in conjunction with a fully qualified INPUT-NAME. If the OUTPUT-NAME is not specified, OUTPUT-NAME=INPUT-NAME is set.

The FORMAT-NAME parameter is not permitted.

Processing log

The log record

```
AVS8521 'CREATE-PROD-JOB:jobname RESULT:result'
```

or an appropriate error message is output for each BS2000 job or S procedure created.

CREATE-PROD-NET – Create temporary BS2000 jobs and S procedures for net

The batch statement CREATE-PROD-NET can only be used if no masks are assigned via the AVAS statement #AVM# in the BS2000 jobs and S procedures of the net.

The net parameters must previously be input via the COLLECT-NET-PARAMS statement.

This batch statement is recommended primarily for modification via USER-PARAM-FILE.

CREATE-PROD-NET

NET-NAME=[\$ug_]netname

[PERIOD-NAME=period]

[USER-PAR-FILE={*NONE / filename/ libname(element[,type])}]

The PERIOD-NAME parameter is only permitted in conjunction with a partially qualified net name.

In the case of a partially qualified net name, the USER-PAR-FILE parameter corresponds to the value in parameter field USER-PAR-FILE (*NONE) of the AVM012 mask with the net list.

In the case of a fully qualified net name, the USER-PAR-FILE parameter corresponds to the value in parameter field USER-PAR-FILE of the AVM001 mask with the description of the data of a structure element.

If the file entered under USER-PARAM-FILE is not present in the batch program, a message is output just as in the interactive session. Processing continues, however, until missing parameters are detected, whereupon a message is output and processing terminates.

Moreover, the rules for the default mechanism described under the interactive function apply.

Processing log

The log record

```
AVS8521 'CREATE-PROD-NET:netname RESULT:result'
```

or an appropriate error message is output for each net processed.

DELETE-DOCUMENT – Delete documentation elements

The batch statement DELETE-DOCUMENT is used to delete documentation elements from the AVAS user library DOCLIB.

```
DELETE-DOCUMENT  
  ELEMENT-NAME=[$ug_]element
```

The ELEMENT-NAME parameter must be specified; the parameter value may be partially qualified.

Processing log

The log record

```
AVS8521 'DELETE-DOCUMENT:elementname RESULT:result'
```

or an appropriate error message is output for each document deleted.

DELETE-JOB – Delete BS2000 jobs, S procedures and JCL elements

The batch statement DELETE-JOB can be used to delete jobs, S procedures and JCL elements from the AVAS user library JCLLIB.

```
DELETE-JOB  
  ELEMENT-NAME=[$ug_]element
```

The ELEMENT-NAME parameter must be specified; the parameter value may be partially qualified.

Processing log

The log record

```
AVS8521 'DELETE-JOB:elementname RESULT:result'
```

or an appropriate error message is output for each BS2000 job, S procedure or JCL element deleted.

DELETE-JOB-LOG – Delete logs of specific net

The batch statement DELETE-JOB-LOG can be used to delete all logs in a net.

```
DELETE-JOB-LOG  
  NET-NAME=[$ug_]netname
```

Processing log

The log record

```
AVS8521 'DELETE-JOB-LOG:netname RESULT:result'
```

or an appropriate error message is output for each net deleted.

DELETE-NET-DESCRIPTION – Delete nets

The batch statement DELETE-NET-DESCRIPTION can be used to delete nets from the AVAS user library NETLIB.

```
DELETE-NET-DESCRIPTION  
  NET-NAME=[$ug_]netname
```

The NET-NAME parameter must be specified; the parameter value may be partially qualified.

Processing log

The log record

```
AVS8521 'DELETE-NET-DESCRIPTION:netname RESULT:result'
```

or an appropriate error message is output for each net deleted.

DELETE-PERIOD – Delete period

The batch statement DELETE-PERIOD can be used to delete an entry containing the specified PERIOD-NAME in the period file.

If the PERIOD-NAME parameter is specified in partially qualified form, all periods are deleted whose names correspond to the specified partial qualification.

Note

Periods for which TYPE=VAR is specified (standard periods with a variable date) can only be deleted using the batch statement DELETE-PERIOD.

```
DELETE-PERIOD  
  PERIOD-NAME=period
```

The PERIOD-NAME parameter must be specified.
It can be specified in fully or partially qualified form (final character *).

Processing log

The log record

```
AVS8521 'DELETE-PERIOD: period RESULT: result'
```

or an appropriate error message is output for each period deleted.

The value for `period` is the name of the period to be deleted in the period file.

DELETE-PLAN-NET – Delete planned nets from production plan

The batch statement DELETE-PLAN-NET is identical to the interactive function. It is advisable to use this statement only in conjunction with the parameter NET-STATUS=SUBMITTED or REPEATED so that only processed nets are removed.

DELETE-PLAN-NET

```
NET-NAME=[$ug_]netname  
[PERIOD-NAME=period]  
[NET-STATUS={TOCREATE / PARTIALLY / CREATED / NOTTOCREATE /  
SUBMITTED / REPEATED}]
```

The PERIOD-NAME parameter is only permitted in conjunction with a partially qualified net name.

Processing log

The log record

```
AVS8521 'DELETE-PLAN-NET:netname RESULT:result'
```

or an appropriate error message is output for each net deleted.

DELETE-PROD-JOB – Delete static BS2000 jobs and S procedures

The batch statement DELETE-PROD-JOB can be used to delete static BS2000 jobs and S procedures from the AVAS user library JMDLIB.

DELETE-PROD-JOB

```
ELEMENT-NAME=[$ug_]element
```

The ELEMENT-NAME parameter must be specified; the parameter value may be partially qualified.

Processing log

The log record

```
AVS8521 'DELETE-PROD-JOB:elementname RESULT:result'
```

or an appropriate error message is output for each BS2000 job or S procedure deleted.

END – Terminate batch statement stream

The batch statement END terminates a processing sequence.

END

Processing log

AVS2310 SYSTEM. TERMINATED

HOLD-NET – Suspend nets currently being processed

The batch statement HOLD-NET enables the processing of nets to be interrupted. Only nets that currently have the status RUNNING, CONDWAIT, HOSTWAIT, WAITING, OPWAIT, RESTARTED, ERROR or START can be suspended.

If the net has the status RUNNING, its status will change to 'CALLED FOR' HOLD as a result of the statement.

If the net has the status CONDWAIT, HOSTWAIT, WAITING, OPWAIT, RESTARTED, ERROR or START, execution of the statement will give the net a status which depends on the type of processing.

In all other cases the status of the net is set to HOLD.

The net status 'CALLED FOR' HOLD is converted to HOLD status by the run control system if it has reached the index level for which the interruption was requested and no other task is still being processed.

The execution of the statement is logged in the journal.

HOLD-NET

```
NET-NAME=[$ug_]netname
[RUN-CONTROL-SYSTEM=avak]
[INDEX=index]
[PERIOD-NAME={period / (date/time,date/time)}]
```

The NET-NAME parameter must be specified; the parameter value may be partially qualified.

The INDEX parameter is only permitted in conjunction with a fully qualified net name. If the INDEX parameter is specified, the HOLD status is set for this index level.

The PERIOD-NAME parameter is only permitted in conjunction with a partially qualified net name.

If the RUN-CONTROL-SYSTEM parameter is not specified, the name of the run control system is determined via the signon data.

Processing log

The log record

```
AVS8521 'HOLD-NET:netname RESULT:result'
```

or an appropriate error message is output for each net interrupted.

MODIFY-COND-DESCRIPTION – Modify condition description

The batch statement MODIFY-COND-DESCRIPTION modifies existing condition descriptions of types NET, JOB, RES and VAL.

The COND-NAME parameter must be specified and the parameter value must be fully qualified.

The TYPE parameter must be specified because the processing depends on the condition type. The following parameters are permitted for the different condition types:

- TYPE=NET

```
MODIFY-COND-DESCRIPTION
  COND-NAME=[$ug_]condname
  TYPE=NET
  [COND-VALUE=status]
  [LIFE-TIME=[dd.mm.yy/hh:mm:ss]]
  [CREATED-BY-NET=[$ug_]netname_date_time]
```

At least one of the parameters LIFE-TIME or COND-VALUE must be specified to allow processing.

The CREATED-BY-NET parameter need only be specified if the condition cannot be uniquely identified via the COND-NAME and TYPE parameters. If CREATED-BY-NET is used, it must be fully qualified. If no user group is specified, the function prefixes the user group of COND-NAME to the net name.

- TYPE=JOB

```
MODIFY-COND-DESCRIPTION
  COND-NAME=[$ug_]condname
  TYPE=JOB
  [COND-VALUE=status]
  [LIFE-TIME=[dd.mm.yy/hh:mm:ss]]
  [CREATED-BY-NET=[$ug_]netname_date_time]
  [CREATED-BY-INDEX=index]
```

At least one of the parameters LIFE-TIME or COND-VALUE must be specified to allow processing.

The CREATED-BY-NET parameter need only be specified if the condition cannot be uniquely identified via the COND-NAME and TYPE parameters. If CREATED-BY-NET is used, it must be fully qualified. If no user group is specified, the function prefixes the user group of COND-NAME to the net name. If the CREATED-BY-NET parameter is not sufficient to identify the condition, the CREATED-BY-INDEX parameter must be specified as well.

The CREATED-BY-INDEX parameter is only permitted in conjunction with the CREATED-BY-NET parameter.

- TYPE=RES

```
MODIFY-COND-DESCRIPTION
  COND-NAME=[$ug_]condname
  TYPE=RES
  COND-VALUE=status
```

The COND-VALUE parameter must be specified to allow processing.

- TYPE=VAL

```
MODIFY-COND-DESCRIPTION
  COND-NAME=[$ug_]condname
  TYPE=VAL
  COND-VALUE=(pos,value)
```

pos = Starting position for value specification, 1...128

value = Value specification, 'c-string', C'c-string', X'x-string'

The COND-VALUE parameter must be specified to allow processing.

Up to 74 characters are available for the description (pos,value). If more than 74 character positions of COND-VALUE are to be modified, the batch statement MODIFY-COND-DESCRIPTION must be specified more than once.

When this statement is executed, the condition description is changed to the specified value as of the specified position.

Notes

- The condition description must be uniquely identifiable via the parameters, otherwise processing is rejected with a message.
- If TYPE=NET/JOB/RES is set, a permissible status value can be specified under COND-VALUE. This value is described in the AVD030 mask, parameter field COND-VALUE (see the dialog statement MODIFY-COND-DESCRIPTION in the “AVAS Statements” manual [2]).
- If TYPE=VAL is set, COND-VALUE can only be defined in the form COND-VALUE=(pos,value).
- The LIFE-TIME parameter corresponds to the value in the LIFE-TIME parameter field in mask AVD030.

Processing log

The log record

AVS8521 'MODIFY-COND-DESCRIPTION:condname type RESULT:result'

or an appropriate error message is output for the modified condition description.

REPEAT-NET – Repeat release of planned net

The batch statement REPEAT-NET duplicates a planned and already released net and releases the duplicate for processing. A new scheduled start time is required for the release. The net name of the duplicate is formed for processing from the new start time. It can be preset with the NEW-PLAN-START parameter. If no new start time is preset, the time the command was called is used.

In the production plan (NPRLIB), the duplicate is stored under its new name with the net parameters, thus ensuring there are no gaps in the documentation. The original net is retained unaltered in the production plan and can be further used as an input net for REPEAT-NET.

The RUN-CONTROL-SYSTEM parameter can be used to assign the net the name of another run control system. The RUN-CONTROL-SYSTEM parameter corresponds to the value in the parameter field RUN-CONTROL-SYSTEM of the AVF012 mask.

REPEAT-NET

```
NET-NAME=[$ug_]netname  
[NEW-PLAN-START=(dd.mm.yy/hh:mm:ss)]  
[RUN-CONTROL-SYSTEM=avak]
```

The NET-NAME parameter must be specified in fully qualified form.

If the NEW-PLAN-START parameter is omitted, the current date and time will be assumed.

If the RUN-CONTROL-SYSTEM parameter is omitted, the name of the run control system determined in the net is used.

Processing log

The log record

```
AVS8521 'REPEAT-NET:netname date/time RESULT:result'
```

or an appropriate error message is output for each net released.

In the event of errors, the value for `date/time` may be missing.

The value for `netname` is the name of the input net and the value for `date/time` is the planning date of the released net (also part of the new net name).

RESTART-NET – Restart net following error

The batch statement RESTART-NET can be used to restart nets whose processing was interrupted due to an error (the net has the status ERROR), or in which at least one structure element already has the status ERROR (the net has the status CALLED FOR ERROR).

The NET-NAME parameter must be specified; the parameter value must be fully qualified.

RESTART-NET

```
NET-NAME=[$ug_]netname  
[RUN-CONTROL-SYSTEM=avak]  
[RESTART-VARIANT={1 / 2 / 3}]  
[ERROR-INDEX=index]  
[ERROR-NAME={jobname / condname}]
```

The ERROR-INDEX parameter must be specified if there is more than one structure element in the net with a status of ERROR. If there are several elements with the status ERROR in the index level specified by ERROR-INDEX, the ERROR-NAME parameter must also be specified. It must be possible to identify the structure element with the status ERROR uniquely, otherwise the restart attempt will be rejected.

ERROR-INDEX is the index of the structure element at the POINT-OF-ERROR.

ERROR-NAME is the name of the structure element at the POINT-OF-ERROR.

Both parameters must refer to a structure element with the status ERROR.

If a structure element within the range of restart index levels (index levels 900 through 999) has the status ERROR, this must be processed before any other elements with the status ERROR.

If the RUN-CONTROL-SYSTEM parameter is omitted, the name of the run control system is taken from the signon data.

If the RESTART-VARIANT parameter is omitted, the restart is initiated using the variant set by the task job variable. If no restart variant has been set in the task job variable, the restart will be rejected with an error message.

If the RESTART-VARIANT parameter is specified, the restart will be initiated using this variant. No check is performed to see whether a different restart variant is set by the task job variable.

Processing log

The log record

```
AVS8521 'RESTART-NET:netname RESULT:result'
```

or an appropriate error message is output for each net started.

RESUME-NET – Cancel HOLD status

The batch statement RESUME-NET enables the processing of nets or structure elements of nets suspended by means of HOLD-NET to be resumed. The net resumes execution at those positions where processing was interrupted via HOLD-NET. Net processing takes place in the same way as for an uninterrupted net.

The net status required before the statement is executed is HOLD or 'CALLED FOR' HOLD (only if the run control system has not yet processed the HOLD called for).

Once RESUME-NET has been executed, the nets are in the RUNNING, CONDWAIT or HOSTWAIT status (if the run control system has already activated the net) or in the WAITING, OPWAIT, RESTART, ERROR or START status. If not all HOLD statuses have been canceled in the net, the net also has the status 'CALLED FOR' HOLD.

RESUME-NET

```
NET-NAME=[$ug_]netname[*]  
[INDEX=index]  
[PERIOD-NAME={period / (date/time,date/time)}]  
[RUN-CONTROL-SYSTEM=avak]
```

The NET-NAME parameter must be specified; the parameter value may be partially qualified (with * as wildcard character).

The INDEX parameter is only permitted in conjunction with a fully qualified net name. If the INDEX parameter is specified, the HOLD status is canceled for this index.

The PERIOD-NAME parameter is only permitted in conjunction with a partially qualified net name.

If the RUN-CONTROL-SYSTEM parameter is not specified, the name of the run control system is determined via the signon data.

Processing log

The log record

```
AVS8521 'RESUME-NET:netname RESULT:result'
```

or an appropriate error message is output for each net interrupted.

SIGNON – Start batch statement stream

The batch statement SIGNON must appear at the beginning of a processing sequence. The parameters correspond to the input fields in the AVS010 mask.

```
SIGNON
  AVAS-USER-ID=avuser
  PASSWORD=password
  AVAS-SYSTEM-ID=string
```

Processing log

```
AVS2501 PLEASE ENTER COMMAND
```

or an appropriate error message is output.

SUBMIT-NET – Release planned nets

The batch statement SUBMIT-NET does not support the OBJECT parameter. The net parameters cannot be changed upon release of the nets by means of the batch function.

```
SUBMIT-NET
  [PERIOD-NAME=period]
  [NET-NAME=[$ug_]netname]
  [EARLIEST-START=(dd.mm.yy / hh:mm:ss)]
```

The PERIOD-NAME parameter is only permitted in conjunction with a partially qualified net name.

The EARLIEST-START parameter is only permitted together with a fully qualified name.

Processing log

The log record

```
AVS8521 'SUBMIT-NET:netname RESULT:result'
```

or an appropriate error message is output for each net released.

8.2 Procedures and libraries for the batch functions

- Procedures.....: The SYSPRC.AVAS.085 library is provided for execution of the batch functions with the J element AVS.BATCH, which must be tailored to suit individual requirements.
- Libraries.....: The libraries SYSLNK.AVAS.085 and SYSPRG.AVAS.085.SYSTEM are required for execution of the batch functions.

Prozedur AVS.BATCH

```

/BEGIN-PROC LOG=N,PAR=YES(PROC-PAR=(&EXEC=START,           -
/                                                         &USERID=,           -
/                                                         &CMDFILE=),           -
/                                                         ESC-CHAR=C'&' )
/REMARK *****
/REMARK &USERID ::= USER ID OF THE AVAS INSTALLATION
/REMARK > <
/REMARK &CMDFILE ::= FILE WITH THE FUNCTION STATEMENTS FOR AVAS
/REMARK *****
/SET-FILE-LINK F-NAME=&USERID..SYSLNK.AVAS.085,LINK=SYSLNK
/ASSIGN-SYSDTA TO-FILE=&CMDFILE
/&EXEC-PROG FROM-FILE=*PHASE(LIB=&USERID..SYSPRG.AVAS.085.SYSTEM, -
/                                                         ELEM=AVAS.SYS.LOAD.BATCH)
/ASSIGN-SYSDTA TO-FILE=*PRIMARY
/REMOVE-FILE-LINK LINK-NAME=SYSLNK
/ SKIP-COM      TO-LABEL=#SW30SET,IF=JOB-SWITCHES(ON=30)
/ SKIP-COM      TO-LABEL=#SW31SET,IF=JOB-SWITCHES(ON=31)
/ SKIP-COM      TO-LABEL=ENDE
/ .#SW30SET     REMARK
/ WRITE-TEXT    '-----> AVAS RUN WITH "WARNING" <-----'
/ MODIFY-JOB-SWITCHES OFF=30
/ SKIP-COM      TO-LABEL=#SW31SET,IF=JOB-SWITCHES(ON=31)
/ SKIP-COM      TO-LABEL=ABEND
/ .#SW31SET     REMARK
/ WRITE-TEXT    '-----> AVAS RUN WITH "ERROR" <-----'
/ MODIFY-JOB-SWITCHES OFF=31
/SET-JOB-STEP
/ .ABEND CANCEL-PROC
/ .ENDE END-PROC

```

The batch statements for the procedure call are read in via the file assigned using the procedure parameter &CMDFILE.

Example

Example of batch statement for planning, producing and releasing the net
ABR.JOB.EXT.

```
SIGNON
  AVAS-USER-ID=USER2
  PASSWORD=FSC
  AVAS-SYSTEM-ID=AVAS085
*
CREATE-PLAN-NET
  NET-NAME=$A1_ABR.JOB.EXT
*
CREATE-PROD-NET
  NET-NAME=$A1_ABR*
*
SUBMIT-NET
  NET-NAME=$A1_*
*
END
```

8.3 Error information for batch functions

When processing AVAS statements via BATCH, all error messages are output to SYSOUT. An error is deemed to exist when an element cannot be processed correctly. The messages are output even if they have been suppressed in order to process an element overview in interactive mode. Messages generated within the batch functions are therefore output in the same way as messages generated when processing an element in an interactive session.

Regardless of the above, a message and RESULT is output for each element processed by a batch statement.

In the event of an error, the following sequence of error messages may occur:

```

...
AVS8301 INPUT 'CREATE-PLAN-NET'
AVS8301 INPUT 'NET-NAME=net*'
AVS8301 INPUT 'PERIOD-NAME=period'
AVS5912 NUMBER OF ELEMENTS SELECTED: nnn
AVS8521 'CREATE-PLAN-NET: element-1 RESULT:CREATED'
AVSmmmm Error message for element-2
AVS8521 'CREATE-PLAN-NET: element-2 RESULT:ERROR'
...

```

The job switch 30 (warning) or 31 (error) is set depending on the weight code assigned to the message AVSmmmm. If several messages are output, both switches can also be set.

Notes

- Which job switch is set is determined by the WEIGHT-CODE parameter in the AVAS message file.
The following applies:

WEIGHT-CODE 0 thru 91	normal information	No job switch set
WEIGHT-CODE 92 and 93	warning message	Job switch 30
WEIGHT-CODE 94 thru 99	error message	Job switch 31
- Job switches cannot be set by the user.
The user must reset the job switch on completion of the batch function if the switch is not permitted to be set at that point.

AVAS messages with WEIGHT-CODE

The message numbers and their associated weight codes, as delivered, can be found in the overview in the “AVAS Functions and Tables” [1].

9 External creation of AVAS elements

Nets, BS2000 jobs, S procedures, JCL elements, and documentation for the AVAS nets can also be created outside AVAS and read into the AVAS libraries NETLIB, JCLLIB and DOCLIB by means of the COPY-ELEMENT statement.

The external creation of nets through programs is performed using the macro definitions of the data records in the NETLIB. The externally created nets are checked by the CHECK function when they are read into the AVAS library, and in the event of errors a log of the check is output.

9.1 Creating nets externally through programs

The AVASNET macro enables user programs to create nets in an external library and then transfer them into an AVAS net library using the COPY-ELEMENT statement. Nets can therefore be created externally through programs.

The AVASNET macro is contained in SYSLIB.AVAS.AVAS.085.

The AVASNET macro defines the data structure of the nets in the AVAS net library NETLIB. The valid values for the relevant parameters are described under the CREATE-NET-DESCRIPTION statement. If parameters are to be stored in the form of keys, these are defined in the macro.

Each net consists of the data records N1, N2, N3, N4 and N5 at least, which are arranged in the sequence shown here.

Three data records must be created for each structure element.

The data record type depends on the function of the structure element which is to be defined.

FUNCTION	Data record
J (Job) P (Procedure)	J1 / J2 / J3
F (File transfer)	F1 / F2 / F3
C (Compare)	C1 / C2 / C3
A (Add)	A1 / A2 / A3
M (Modify)	M1 / M2 / M3
D (Delete)	D1 / D2 / D3
W (Wait)	W1 / W2 / W3
S (Start)	S1 / S2 / S3

The data records must be set up in the order shown (record1, record2, record3).

Data records for structure elements must be sorted in ascending INDEX order. Where there are several structure elements at an index level, the order in which the records are stored will have precedence.

General form of the macro call

Macro	Operands
AVASNET	MF = <u>C</u> / D ,PREFIX = <u>N</u> / <char (1)> ,MACID = <u>RECORD</u> / <char (3)> ,RECORD = N1/N2/N3/N4/N5 / J1/J2/J3 / C1/C2/C3 / S1/S2/S3 / A1/A2/A3 / M1/M2/M3 / D1/D2/D3 / W1/W2/W3 / F1/F2/F3

- MF The MF operand (“macro form”) controls code generation
- C The layout of the data structure (generally the parameter area) is created; in doing so, each field and each equate is named. This data structure becomes a part of the current program control/dummy section (CSECT/DSECT).
- D Creates the layout of the data structure as for MF = C; in addition, a DSECT statement is created.
- PREFIX The PREFIX operand is used to generate the names which have to be created. PREFIX, which is exactly one letter, is used as the first letter of all the names. The default value is the identifying letter of the functional unit to

which the macro belongs. To avoid identical names arising, PREFIX should be used when the same data structure is to be used repeatedly within a module.

If MACID is not specified, then PREFIX is permitted to be a two-character prefix.

- MACID** The MACID operand is used to generate the names which have to be created, and defines the character which follows PREFIX in the name. Its default value guarantees that there will be no duplication of names within the component group.
- RECORD** The second and third characters of the field name and equates are given the value specified for RECORD.
- <char(3)>** A three character string which specifies the second to fourth characters of the field name and equates generated by the Assembler.
- RECORD=** Indicates the net description record for which the definitions are to be generated.
The macro expansions which follow specify the contents of the records.
- N1/N2/N3** Defines net description records 1/2/3 with the symbolic or real scheduled start times and the assigned plan parameters.
- N4** Defines net description record 4 with the net parameters.
- N5** Defines net description record 5 with the masks for netwide modification of parameters.
- J1/C1/A1/M1/D1/W1/S1/F1**
Defines record 1 of a structure element specification, with the symbolic scheduled start time (SYMDAT) and the assigned plan parameters.
- J2/C2/A2/M2/D2/W2/S2/F2**
Defines record 2 of a structure element specification, with the restart points and the first part of the parameters for the structure element.
- J3/C3/A3/M3/D3/W3/S3/F3**
Defines record 3 of a structure element specification, with the second part of the parameters for the structure element.

Note

The AVASNET macro itself calls the MFCHK macro. At compile time the BS2000 macro library must therefore be assigned as well as the AVAS macro library, or AVASNET should be included in the BS2000 macro library.

Expansion of the AVASNET macro

AVASNET RECORD=NI

```

MFCHK MF=C, C
      PREFIX=N, C
      MACID=N1, C
      DMACID=N1D, C
      SUPPORT=(C,D)
DS OF
      *,##### PREFIX=N, MACID=N1 #####
*-----*
NN1A DS OF
NN1SLEN DS H PLAM-RECORD-LENGTH
      DS C PLAM-RESERVED PLAM - SK
      DS C PLAM-RESERVED PLAM - SA
NN1SSL DS CL2 RECORD-KEY
      DS CL1 RESERVED
      DS CL1 RESERVED
      DS CL1 RESERVED
NN1IND DS CL3 NET-INDEX = 000
NN1NAM DS CL18 NET-NAME
      DS CL14 RESERVED
*-----*
* PLAN - DESCRIPTION *
*-----*
NN1NSEL DS C NET-SELECT-TURNUS
NN1RWNW DS X RELATIV SYMDAT WORK/NWRK
NN1QNWR EQU 83 SELECT-PLAN-TYPE:NWRK
NN1QWRK EQU 84 WORK
      DS CL2 RESERVED
NN1PSSY DS 44CL42 PLAN-START
NN1PSSE DS OC
      ORG NN1PSSY
NN1SYMD DS OCL20 SYMDAT-NAME <NAME+/-NN>
NN1RLDI DS C REAL-START-DATE-IDENTIFICAT.
      NN1RLDA DS CL6 REAL-START-DATE <YYMMDD>
      DS CL13 RESERVED
      NN1STIM DS CL6 SYMDAT-START-TIME <HHMMSS>
      NN1LAST DS CL7 SYMDAT-LATEST-START <DDHMM>
      NN1LTIM DS CL7 LIFE-TIME <DDHMM>
NN1INDSL DS X NET-DELAY-SOLUTION
*-----*

```

```

*          DELAY-SOLUTION:  NET   STR  -> MEANING
NN1EDSC  EQU   65          |  +  |  +  |  . CANCEL
NN1EDSW  EQU   67          |  +  |  -  |  . WAIT
NN1EDSS  EQU   68          |  +  |  +  |  . START
NN1EDSI  EQU   69          |  +  |  +  |  . IGNORE
*          ( + ::= USED ) ( - ::= NOT USED )
*
*-----
*-----
NN1BHYP  DS    X          *BY-HYPERNET
NN1EBHP  EQU  X'E8'      . *BY-HYPERNET
NN1LMIN  EQU  *-NN1A     RECORD-LENGTH (MINIMUM)
NN1CE$L  EQU  *-NN1SYMD  SEGM.-LENGTH
          ORG  NN1PSSE
NN1CE$Z  EQU  (*-NN1PSSY)/NN1CE$L SEGM.-COUNTER
          DS  0F
NN1L     EQU  *-NN1A

```

AVASNET RECORD=N2

```

MFCHK MF=C,
      PREFIX=N,
      MACID=N2,
      DMACID=N2D,
      SUPPORT=(C,D)
DS OF
*,##### PREFIX=N, MACID=N2 #####
*-----*
NN2A DS OF
NN2SLEN DS H PLAM-RECORD-LENGTH
      DS C PLAM-RESERVED PLAM - SK
      DS C PLAM-RESERVED PLAM - SA
NN2SSL DS CL2 RECORD-KEY
      DS CL1 RESERVED
      DS CL1 RESERVED
      DS CL1 RESERVED
NN2IND DS CL3 NET-INDEX = 000
NN2NAM DS CL18 NET-NAME
      DS CL14 RESERVED
*-----*
* PLAN - DESCRIPTION *
*-----*
      DS CL4 RESERVED
NN2LMIN EQU *-NN2A RECORD-LENGTH (MINIMUM)
NN2PSSY DS 44CL42 PLAN-START
NN2PSSE DS OC
      ORG NN2PSSY
NN2SYMD DS OCL20 SYMDAT-NAME <NAME+/-NN>
NN2RLDI DS C REAL-START-DATE-IDENTIFICAT.
NN2RLDA DS CL6 REAL-START-DATE <YYMMDD>
NN2ZYK DS CL4 ZYKLUS OF REAL START-DATE <+/-DNN>
NN2PMW DS CL2
      DS CL7
      DS CL7 RESERVED
NN2STIM DS CL6 SYMDAT-START-TIME <HHMMSS>
NN2LAST DS CL7 SYMDAT-LATEST-START <DDHHMM>
NN2LTIM DS CL7 LIFE-TIME <DDHHMM>
NN2NDSL DS X NET-DELAY-SOLUTION
*
* DELAY-SOLUTION: NET STR -> MEANING
NN2EDSC EQU 65 | + | + | . CANCEL
NN2EDSW EQU 67 | + | - | . WAIT
NN2EDSS EQU 68 | + | + | . START
NN2EDSI EQU 69 | + | + | . IGNORE
*
* ( + ::= USED ) ( - ::= NOT USED )
*-----*

```

```
NN2BHYP DS X *BY-HYPERNET
NN2EBHP EQU X'E8' . *BY-HYPERNET
NN2CE$L EQU *-NN2SYMD SEGM.-LENGTH
ORG NN2PSSE
NN2CE$Z EQU (*-NN2PSSY)/NN2CE$L SEGM.-COUNTER
DS OF
NN2L EQU *-NN2A
```

AVASNET RECORD=N3,PREFIX=N,EQU=YES

```

MFCHK MF=C,
      PREFIX=N,
      MACID=N3,
      DMACID=N3D,
      SUPPORT=(C,D)
DS OF
*,##### PREFIX=N, MACID=N3 #####
    
```

```

*-----*
NN3A  DS  OF
NN3SLEN DS  H          PLAM-RECORD-LENGTH
      DS  C          PLAM-RESERVED PLAM - SK
      DS  C          PLAM-RESERVED PLAM - SA
NN3SSL DS  CL2       RECORD-KEY
      DS  CL1       RESERVED
      DS  CL1       RESERVED
      DS  CL1       RESERVED
NN3IND DS  CL3       NET-INDEX = 000
NN3NAM DS  CL18      NET-NAME
      DS  CL14      RESERVED
    
```

```

*-----*
*          PLAN - DESCRIPTION          *
*-----*
    
```

```

      DS  CL4          RESERVED
NN3LMIN EQU *-NN3A    RECORD-LENGTH (MINIMUM)
NN3PSSY DS 44CL42     PLAN-START
NN3PSSE DS 0C
      ORG NN3PSSY
NN3SYMD DS 0CL20     SYMDAT-NAME          <NAME+/-NN>
NN3RLDI DS C         REAL-START-DATE-IDENTIFICAT.
NN3RLDA DS CL6       REAL-START-DATE          <YYMMDD>
NN3ZYK  DS CL4       ZYKLUS OF REAL START-DATE <+/-DNN>
NN3PMW  DS CL2       .... +/-W
      DS CL7         RESERVED
NN3STIM DS CL6       SYMDAT-START-TIME      <HHMMSS>
NN3LAST DS CL7       SYMDAT-LATEST-START   <DDHHMM>
NN3LTIM DS CL7       LIFE-TIME              <DDHHMM>
NN3NDSL DS X         NET-DELAY-SOLUTION
    
```

```

*
*          DELAY-SOLUTION:  NET  STR  -> MEANING
NN3EDSC EQU 65          | + | + | . CANCEL
NN3EDSW EQU 67          | + | - | . WAIT
NN3EDSS EQU 68          | + | + | . START
NN3EDSI EQU 69          | + | + | . IGNORE
*
*          ( + ::= USED ) ( - ::= NOT USED )
*-----*
    
```



```
NN3BHYP DS X *BY-HYPERNET
NN3EBHP EQU X'E8' . *BY-HYPERNET
NN3CE$L EQU *-NN3SYMD SEGM.-LENGTH
ORG NN3PSSE
NN3CE$Z EQU (*-NN3PSSY)/NN3CE$L SEGM.-COUNTER
DS OF
NN3L EQU *-NN3A
```

AVASNET RECORD=N4,PREFIX=N,EQU=YES

```

MFCHK MF=C,
      PREFIX=N,
      MACID=N4,
      DMACID=N4D,
      SUPPORT=(C,D)
DS OF
*,##### PREFIX=N, MACID=N4 #####
*-----*
NN4A DS OF
NN4SLEN DS H PLAM-RECORD-LENGTH
      DS C PLAM-RESERVED PLAM - SK
      DS C PLAM-RESERVED PLAM - SA
NN4SSL DS CL2 RECORD-KEY
      DS CL1 RESERVED
      DS CL1 RESERVED
      DS CL1 RESERVED
NN4IND DS CL3 NET-INDEX = 000
NN4NAM DS CL18 NET-NAME
      DS CL14 RESERVED
*-----*
* NET - PARAMETERS *
*-----*
NN4CFST DS X CHECK: ERROR-STATE
NN4EQFN EQU X'00' . NO CHECK NET-VERS. < V-1.3A
NN4EQF0 EQU X'01' . NO ERROR
NN4EQF1 EQU X'02' . ERR-ST.1 WARNING
NN4EQF2 EQU X'04' . ERR-ST.2 WARNING BY RESTART-NET
NN4EQF3 EQU X'10' . ERR-ST.3 MODIFY BY SUBMIT-NET
NN4EQF4 EQU X'20' . ERR-ST.4 ERROR, RESULT:NO-PLAN
NN4CCMD DS X CMD OF LAST CHECK
NN4EQ00 EQU 0 . NO CHECK NET-VERS. < V-1.3A
NN4EQ41 EQU 41 . CREATE-NET-DESCRIPTION
NN4EQ42 EQU 42 . MODIFY-NET-DESCRIPTION
NN4EQ55 EQU 55 . COPY-ELEMENT
NN4EQ60 EQU 60 . ADD-PLAN-NET (ONLY NPRLIB)
NN4NTYP DS C NET-TYPE
      DS C RESERVED
NN4RCSY DS CL8 RUN-CONTROL-SYSTEM
NN4NUSE DS CL8 NET-USER (ENTER:USER-ID)
NN4NACC DS CL8 NET-ACCOUNT (ENTER:ACC-NUMBER)
NN4NCLA DS CL8 NET-CLASS (ENTER:JOB-CLASS)
NN4NLGM DS CL8 NET-LOG
NN4NPAS DS CL19 NET-PASSWORD
NN4NPAR DS CL128 NET-PARAMETER
NN4NCAT DS OCL54 NET-CAT (CATID IN JVA)
NN4NCTK DS CL1 QUOTE OF CATID

```

NN4EJCK	EQU	X'7D'	. CATID-QUOTE (JOB)
NN4ENCK	EQU	X'7D'	. CATID-QUOTE (NET)
NN4NCTI	DS	CL4	NET-CAT (STRING OF CATID)
	DS	CL49	RESERVED
	DS	CL10	RESERVED
NN4NTEX	DS	CL120	TITLE / INFORMATION
NN4TUPF	DS	CL1	TYPE OF USER-PAR-FILE (LIB/SAM)
NN4EUSM	EQU	C'S'	. SAM
NN4EULB	EQU	C'L'	. LIB
NN4EUST	EQU	C'*'	. *STD
NN4EUNN	EQU	C' '	. *NONE
NN4EUBH	EQU	C'Y'	. *BY-HYPERNET
NN4LMIN	EQU	*-NN4A	RECORD-LENGTH (MINIMUM)
NN4UPAR	DS	CL54	USER-PAR-FILE
NN4NDOC	DS	CL43	DOCUMENT-NAME
NN4CALN	DS	CL20	CALENDAR-NAME
	DS	CL3	RESERVED
	DS	OF	
NN4L	EQU	*-NN4A	

AVASNET RECORD=N5,PREFIX=N,EQU=YES

```

MFCHK MF=C,
      PREFIX=N,
      MACID=N5,
      DMACID=N5D,
      SUPPORT=(C,D)
DS OF
*,##### PREFIX=N, MACID=N5 #####
    
```

```

NN5A   DS   OF
NN5SLEN DS   H          PLAM-RECORD-LENGTH
        DS   C          PLAM-RESERVED PLAM - SK
        DS   C          PLAM-RESERVED PLAM - SA
NN5SSL DS   CL2        RECORD-KEY
        DS   CL1        RESERVED
        DS   CL1        RESERVED
        DS   CL1        RESERVED
NN5IND DS   CL3        NET-INDEX = 000
NN5NAM DS   CL18       NET-NAME
        DS   CL14       RESERVED
    
```

* LIST OF MAPS *

```

NN5LMIN EQU *-NN5A          RECORD-LENGTH (MINIMUM)
NN5NFML DS 32CL50          TAB OF MAPS
NN5NFME DS OF
        ORG NN5NFML
NN5NFNM DS CL8            MAP-NAME
NN5NFST DS CL2            RESERVED
NN5NFTX DS CL40           MAP-INFORMATION
NN5CE$L EQU *-NN5NFNM     SEGM. LENGTH
        ORG NN5NFME
NN5CE$Z EQU (*-NN5NFML)/NN5CE$L SEGM. COUNTER
        DS OF
NN5L    EQU *-NN5A
    
```

AVASNET RECORD=J1,PREFIX=N,EQU=YES

```

MFCHK MF=C,                                C
      PREFIX=N,                             C
      MACID=J1,                              C
      DMACID=J1D,                            C
      SUPPORT=(C,D)
DS    OF
      *,##### PREFIX=N, MACID=J1 #####
*-----*
NJ1A  DS    OF
NJ1SLEN DS  H          PLAM-RECORD-LENGTH
      DS    C          PLAM-RESERVED PLAM - SK
      DS    C          PLAM-RESERVED PLAM - SA
NJ1SSL DS  CL2        RECORD-KEY
NJ1FUNC DS  C          DESCRIPTION-FUNCTION-CODE
*      FUNCTION
NJ1EFUA EQU  C'A'      . ADD
NJ1EFUC EQU  C'C'      . COMPARE
NJ1EFUD EQU  C'D'      . DELETE
NJ1EFUF EQU  C'F'      . FILE-TRANSFER
NJ1EFUJ EQU  C'J'      . START-JOB
NJ1EFUM EQU  C'M'      . MODIFY
NJ1EFUN EQU  C'N'      . NET
NJ1EFUP EQU  C'P'      . START-PROCEDURE
NJ1EFUS EQU  C'S'      . START
NJ1EFUW EQU  C'W'      . WAIT
NJ1EFUX EQU  C'X'      . START-SERVER-JOB
*      TYPE
NJ1ENET EQU  33        . NET
NJ1EJVA EQU  60        . JVA
NJ1EEXT EQU  70        . EXT
NJ1EMOD EQU  71        . MOD
NJ1EEXX EQU  81        . EXX
NJ1ESTD EQU  94        . STD
NJ1EJOB EQU  112       . JOB
NJ1ERES EQU  113       . RES
NJ1EVAL EQU  114       . VAL
NJ1ETIM EQU  115       . TIM
NJ1ETRA EQU  186       . TRA
      DS    CL1        RESERVED
NJ1IND  DS    CL3        INDEX
NJ1NAM  DS    CL30       NAME
      DS    CL2        RESERVED

```

```

*-----*
*          PLAN - DESCRIPTION          *
*-----*
* FIELD USED ONLY WHEN|
*   DESCR.-TYPE=|JVA|NET  |JOB|RES   |VAL   |TIM|MOD|EXT|STD|TRA  *
*   AND         -FUNC=|C  |C D S|C D|C A M D|C A M D|W  |J P|J P|J P|F  *
*-----*
*..LST  LAST-STRT|   |   +|   |   |   |   | + +|+ +|+ +|+  *
*..LOC  LAST-OCCR|+  |+  +|+  |+  |   |   |   |   |   |   *
*..DSO  DELAY-SOL|+  |+  +|+  |+  |   |   | + +|+ +|+ +|+  *
*-----*

NJ1STL  DS    10CL1                LIST OF SELECT-TURNUS
NJ1STE  DS    0C
        ORG   NJ1STL
NJ1SEL  DS    C                    SELECT-TURNUS (0-9 OR X'40')
NJ1$L1  EQU   *-NJ1SEL             TAB.1 SEGM.-LENGTH
        ORG   NJ1STE
NJ1$Z1  EQU   (*-NJ1STL)/NJ1$L1    TAB.1 SEGM.-COUNTER
        DS    CL2                  RESERVED
*
NJ1SYL  DS    51CL36               LIST OF SYMDAT
NJ1SYE  DS    0C
        ORG   NJ1SYL
NJ1SYM  DS    CL20                 SYMDAT-NAME
NJ1LST  DS    0CL7                 LATEST-START <DDHMM>
NJ1LOC  DS    CL7                 LATEST-OCCURE <DDHMM>
NJ1LTI  DS    CL7                 LIFE-TIME <DDHMM>
NJ1DSO  DS    X                    DELAY-SOLUTION
*
*          -----
*          DELAY-SOLUTION:  NET   STR  -> MEANING
NJ1EDSC EQU   65                | + | + | . CANCEL
NJ1EDSW EQU   67                | + | - | . WAIT
NJ1EDSS EQU   68                | + | + | . START
NJ1EDSI EQU   69                | + | + | . IGNORE
*          ( + ::= USED ) ( - ::= NOT USED )
*          -----
        DS    C                    RESERVED
NJ1$L2  EQU   *-NJ1SYM             TAB.2 SEGM.-LENGTH
NJ1LMIN EQU   *-NJ1A              RECORD-LENGTH (MINIMUM)
        ORG   NJ1SYE
NJ1$Z2  EQU   (*-NJ1SYL)/NJ1$L2    TAB.2 SEGM.-COUNTER
        DS    OF
NJ1L    EQU   *-NJ1A
    
```

AVASNET RECORD=J2,PREFIX=N,EQU=YES

```

MFCHK MF=C,                                C
      PREFIX=N,                             C
      MACID=J2,                              C
      DMACID=J2D,                            C
      SUPPORT=(C,D)
DS    OF
      *,##### PREFIX=N, MACID=J2 #####
*-----*
NJ2A   DS    OF
NJ2SLEN DS    H          PLAM-RECORD-LENGTH
      DS    C          PLAM-RESERVED PLAM - SK
      DS    C          PLAM-RESERVED PLAM - SA
NJ2SSL DS    CL2       RECORD-KEY
NJ2FUNC DS    C          DESCRIPTION-FUNCTION-CODE
*      FUNCTION
NJ2EFUA EQU    C'A'     . ADD
NJ2EFUC EQU    C'C'     . COMPARE
NJ2EFUD EQU    C'D'     . DELETE
NJ2EFUF EQU    C'F'     . FILE-TRANSFER
NJ2EFUJ EQU    C'J'     . START-JOB
NJ2EFUM EQU    C'M'     . MODIFY
NJ2EFUN EQU    C'N'     . NET
NJ2EFUP EQU    C'P'     . START-PROCEDURE
NJ2EFUS EQU    C'S'     . START
NJ2EFUW EQU    C'W'     . WAIT
NJ2EFUX EQU    C'X'     . START-SERVER-JOB
NJ2TYPE DS    X          DESCRIPTION-TYPE-CODE
*      TYPE
NJ2ENET EQU    33       . NET
NJ2EJVA EQU    60       . JVA
NJ2EEXT EQU    70       . EXT
NJ2EMOD EQU    71       . MOD
NJ2EEXX EQU    81       . EXX
NJ2ESTD EQU    94       . STD
NJ2EJOB EQU    112      . JOB
NJ2ERES EQU    113      . RES
NJ2EVAL EQU    114      . VAL
NJ2ETIM EQU    115      . TIM
NJ2ETRA EQU    186      . TRA
      DS    CL1       RESERVED
NJ2IND  DS    CL3       INDEX
NJ2NAM  DS    CL30      NAME
      DS    CL2       RESERVED

```

```

*-----*
*          START AND RESTART - DESCRIPTION          *
*-----*
* FIELD USED ONLY WHEN |
*   DESCR.-TYPE=|JVA|NET |JOB|RES |VAL |TIM|MOD|EXT|STD|TRA *
*   AND         -FUNC=|C |C D S|C D|C A M D|C A M D|W |J P|J P|J P|F *
*-----*
*..CIX COND-IND | | | | + +| | | | | | | | *
*..CJN C-JOB-NA | | | | + +| | | | | | | | *
*..CNN C-NET-NA | | + +| + +| | | | | | | | *
*..ENT ENT-PARA | | | | | | | | + +| + +| + +| *
*..FCN FULL-C-N | + | | | + +| | | | | | | | *
*..FN  FULL-NAM | | | | | | | | | | + +| | | *
*..VPOS VAL-POS | + | | | | | | | | | | | | *
*..VLEN VAL-LEN | + | | | | | | | | | | | | *
*..VPAS JVA-PSWD | + | | | | | | | | | | | | *
*..DIR  DIRECTIO | | | | | | | | | | | | + | *
*..REM  REMOTE   | | | | | | | | | | | | + | *
*..PTN  PARTNER  | | | | | | | | | | | | + | *
*..RFA  REM-FTAC | | | | | | | | | | | | + | *
*-----*

NJ2SYN DS CL3 SYNC-INDEX
NJ2ENT DS X ENTER-PARAMS
* ENTER-PARAMS
NJ2EEPN EQU 33 . NET
NJ2EEPL EQU 157 . LOGON
NJ2DOC DS CL43 DOCUMENT-NAME
DS CL21 RESERVED
NJ2TEX DS CL120 TITLE / INFORMATION
*
NJ2RSP DS 3CL34 RESTART-POINT (1-3)
NJ2RSE DS 0C
ORG NJ2RSP
NJ2RSIN DS CL3 RESTART-INDEX
NJ2RSTY DS X RESTART-TYPE
* RESTART-TYPE
NJ2ERNM EQU 72 . NORMAL
NJ2ERST EQU 73 . RESTART
NJ2ERSA EQU 78 . RESTART-AUTOMATIC
NJ2ERNA EQU 79 . NORMAL-AUTOMATIC
NJ2RSJN DS CL30 RESTART-NAME
NJ2E$L EQU *-NJ2RSIN SEGM.-LENGTH
ORG NJ2RSE
NJ2E$Z EQU (*-NJ2RSP)/NJ2E$L SEGM.-COUNTER
DS CL2 RESERVED
NJ2LMIN EQU *-NJ2A RECORD-LENGTH (MINIMUM)
NJ2VA DS CL77 VARIABLE-AREA
ORG NJ2VA
    
```


NJ2FN	DS	CL57	/ FULL-NAME
	ORG	NJ2VA NJ2FCN	DS CL54 / FULL-COND-NAME
NJ2VPOS	DS	CL3	+ VALUE-POSITION
NJ2VLEN	DS	CL3	+ VALUE-LENGTH
NJ2VPAS	DS	CL11	+ JVA-PASSWORD
	ORG	NJ2VA	
NJ2CJN	DS	CL30	/ COND-JOB-NAME
NJ2CNN	DS	CL32	+ /COND-NET-NAME
NJ2CIX	DS	CL3	+ /COND-INDEX
	ORG	NJ2VA	
NJ2DIR	DS	X	/DIRECTION
*		DIRECTION	
NJ2EFDT	EQU	C'T'	. TO
NJ2EFDF	EQU	C'F	. FROM
NJ2REM	DS	X	+ /REMOTE
*		REMOTE	
NJ2EFRB	EQU	C'B'	. *BS2000
NJ2EFRM	EQU	C'N'	. *MSP
NJ2EFRA	EQU	C'A'	. *ANY
NJ2PTN	DS	CL8	+ /PARTNER
NJ2RFA	DS	CL67	+ /REMOTE-FTAC
	ORG	NJ2VA+L'NJ2VA	
	DS	CL17	RESERVED
	DS	OF	
NJ2L	EQU	*-NJ2A	

AVASNET RECORD=J3,PREFIX=N

```

MFCHK MF=C,
      PREFIX=N,
      MACID=J3,
      DMACID=J3D,
      SUPPORT=(C,D)
DS    OF
      *,##### PREFIX=N, MACID=J3 #####
    
```

```

NJ3A   DS    OF
NJ3SLEN DS    H          PLAM-RECORD-LENGTH
        DS    C          PLAM-RESERVED PLAM - SK
        DS    C          PLAM-RESERVED PLAM - SA
NJ3SSL DS    CL2        RECORD-KEY
NJ3FUNC DS    C          DESCRIPTION-FUNCTION-CODE
NJ3TYPE DS    X          DESCRIPTION-TYPE-CODE
        DS    CL1        RESERVED
NJ3IND  DS    CL3        INDEX
NJ3NAM  DS    CL30       NAME
        DS    CL2        RESERVED
    
```

```

*          PARAMETERS
*
*          FUNCTION=J, TYPE=(STD/MOD/EXT)
*          =P,          =(STD/MOD/EXT/EXX)
*          =X,          =(STD/MOD/EXT)
*-----*
    
```

```

NJ3JUSE DS    CL8          JOB-USER (ENTER:USER-ID)
NJ3JACC DS    CL8          JOB-ACCOUNT (ENTER:ACC-NUMBER)
NJ3JCLA DS    CL8          JOB-CLASS (ENTER:JOB-CLASS)
NJ3JLGM DS    CL8          JOB-LOG
NJ3JPAS DS    CL19         JOB-PASSWORD
        DS    CL5          RESERVED
NJ3JPAR DS    CL128        JOB-PARAMETER
NJ3JCAT DS    OCL54        JOB-CAT (CATID IN JVA)
NJ3JCTK DS    CL1          QUOTE OF CATID
NJ3EJCK EQU    X'7D'       . CATID-QUOTE (JOB)
NJ3ENCK EQU    X'7D'       . CATID-QUOTE (NET)
NJ3JCTI DS    CL4          JOB-CAT (STRING OF CATID)
        DS    CL49         RESERVED
        DS    CL10         RESERVED
    
```

```

*-----*
* FIELDS USED ONLY WHEN |                                     *
*   DESCRIPTION-TYPE= | MOD | STD | EXT | EXX |               *
* AND   -FUNCTION= |J P X|J P X|J P X| P |                   *
*-----*
NJ3LMIN EQU *-NJ3A RECORD-LENGTH (MINIMUM)
NJ3JFIL DS OCL54 NAME OF ENTER-FILE
NJ3SXFN DS OCL54 SERVER FILENAME
NJ3JUPF DS CL54 NAME OF USER-PAR-FILE
NJ3TUPF DS C TYPE OF USER-PAR-FILE (LIB/SAM)
NJ3EUSM EQU C'S' . SAM
NJ3EULB EQU C'L' . LIB
NJ3EUST EQU C'*' . *STD
NJ3EUNN EQU C' ' . *NONE
          DS C RESERVED
NJ3SXID DS CL8 SERVER-NAME
NJ3FPSW DS OCL11 PASSWORD OF ENTER-FILE
NJ3SXPW DS CL11 PASSWORD OF SERVER-NAME
          DS C RESERVED
          DS OF
NJ3L EQU *-NJ3A

```

AVASNET RECORD=C3,PREFIX=N

```

MFCHK MF=C,
      PREFIX=N,
      MACID=C3,
      DMACID=C3D,
      SUPPORT=(C,D)
DS    OF
      *,##### PREFIX=N, MACID=C3 #####
*-----*
NC3A  DS    OF
NC3SLEN DS    H          PLAM-RECORD-LENGTH
      DS    C          PLAM-RESERVED PLAM - SK
      DS    C          PLAM-RESERVED PLAM - SA
NC3SSL DS    CL2        RECORD-KEY
NC3FUNC DS    C          DESCRIPTION-FUNCTION-CODE
NC3TYPE DS    X          DESCRIPTION-TYPE-CODE
      DS    CL1        RESERVED
NC3IND  DS    CL3        INDEX
NC3NAM  DS    CL30       NAME
      DS    CL2        RESERVED
*-----*
*          PARAMETERS
*          FUNCTION=C, TYPE=(JVA/NET/JOB/RES/VAL )
*-----*
NC3CVAL DS    0CL256     COND-VALUE
NC3COST DS    CL128     OCCURE-VALUE STRING
NC3CEST DS    CL128     ERROR-VALUE STRING
NC3COCO DS    12XL1     OCCURE-VALUE (CODE)
*-----*
*          DESCRIPTION-TYPE: NET JOB RES    -> MEANING
NC30ABE EQU    20       | + | + | - | | . ABENDED
NC30END EQU    21       | + | + | - | | . ENDED
NC30ERR EQU    22       | - | + | - | | . ERROR
NC30SKI EQU    62       | - | + | - | | . SKIPPED
NC30IGN EQU    63       | + | + | - | | . IGNORED
NC30FRE EQU   107       | - | - | + | | . FREE
NC30USH EQU   118       | - | - | + | | . SHARE
NC30NPL EQU   124       | - | + | - | | . NO-PLAN
NC30DEL EQU   125       | - | + | - | | . DELETED
NC30NSM EQU   133       | - | + | - | | . NO-SUBMIT
*          ( + ::= USED ) ( - ::= NOT USED )
*-----*
NC3CECO DS    12XL1     ERROR-VALUE (CODE)
*-----*

```

*	DESCRIPTION-TYPE:	NET	JOB	RES	-> MEANING
NC3ECRE	EQU 11	-	- +		. CREATED
NC3EABE	EQU 20	+	+ -		. ABENDED
NC3EEND	EQU 21	+	+ -		. ENDED
NC3EERR	EQU 22	-	+ +		. ERROR
NC3ESKI	EQU 62	-	+ -		. SKIPPED
NC3EIGN	EQU 63	+	+ -		. IGNORED
NC3EMIS	EQU 64	+	+ +		. MISSING
NC3EFRE	EQU 107	-	- +		. FREE
NC3EUSH	EQU 118	-	- +		. SHARE
NC3EUEX	EQU 119	-	- +		. EXCLUSIVE
NC3ENPL	EQU 124	-	+ -		. NO-PLAN
NC3EDEL	EQU 125	-	+ -		. DELETED
NC3ENSM	EQU 133	-	+ -		. NO-SUBMIT
*		(+ ::= USED) (- ::= NOT USED)			

NC3CSRV	DS C	SELECT-RESTART-VARIANT			
NC3CVLG	DS C	COND-VALUE: LOGIK			
	DS CL2	RESERVED			
NC3LMIN	EQU *-NC3A	RECORD-LENGTH (MINIMUM)			
	DS OF				
NC3L	EQU *-NC3A				

AVASNET RECORD=A3,PREFIX=N

```

MFCHK MF=C,
      PREFIX=N,
      MACID=A3,
      DMACID=A3D,
      SUPPORT=(C,D)
DS    OF
      *,##### PREFIX=N, MACID=A3 #####
*-----*
NA3A   DS    OF
NA3SLEN DS    H          PLAM-RECORD-LENGTH
      DS    C          PLAM-RESERVED PLAM - SK
      DS    C          PLAM-RESERVED PLAM - SA
NA3SSL DS    CL2        RECORD-KEY
NA3FUNC DS    C          DESCRIPTION-FUNCTION-CODE
NA3TYPE DS    X          DESCRIPTION-TYPE-CODE
      DS    CL1        RESERVED
NA3IND  DS    CL3        INDEX
NA3NAM  DS    CL30       NAME
      DS    CL2        RESERVED
*-----*
*          PARAMETERS
*          FUNCTION=A, TYPE=(RES/VAL)
*          =M          =(RES/VAL)
*-----*
NA3AVAL DS    CL128      VALUE
NA3AVCO DS    XL1        VALUE-CODE
*-----*
*          DESCRIPTION-TYPE      : RES
*          DESCRIPTION-FUNCTION: |A M| | | | -> MEANING
NA3ECRE EQU    11          |+ -| | | | . CREATED
NA3EERR EQU    22          |+ +| | | | . ERROR
NA3EFRE EQU   107          |+ +| | | | . FREE
NA3EUSH EQU   118          |+ -| | | | . SHARE
NA3EUEX EQU   119          |+ -| | | | . EXCLUSIVE
*          ( + ::= USED ) ( - ::= NOT USED )
*-----*
      DS    CL3          RESERVED
NA3LMIN EQU    *-NA3A    RECORD-LENGTH (MINIMUM)
      DS    OF
NA3L    EQU    *-NA3A
    
```

AVASNET RECORD=M3,PREFIX=N

```

MFCHK MF=C,
      PREFIX=N,
      MACID=M3,
      DMACID=M3D,
      SUPPORT=(C,D)
DS OF
*,##### PREFIX=N, MACID=M3 #####
*-----*
NM3A DS OF
NM3SLEN DS H PLAM-RECORD-LENGTH
      DS C PLAM-RESERVED PLAM - SK
      DS C PLAM-RESERVED PLAM - SA
NM3SSL DS CL2 RECORD-KEY
NM3FUNC DS C DESCRIPTION-FUNCTION-CODE
NM3TYPE DS X DESCRIPTION-TYPE-CODE
      DS CL1 RESERVED
NM3IND DS CL3 INDEX
NM3NAM DS CL30 NAME
      DS CL2 RESERVED
*-----*
* PARAMETERS *
* FUNCTION=A, TYPE=(RES/VAL) *
* =M =(RES/VAL) *
*-----*
NM3AVAL DS CL128 VALUE
NM3AVCO DS XL1 VALUE-CODE
*-----*
* DESCRIPTION-TYPE : RES
* DESCRIPTION-FUNCTION: |A M| | | |-> MEANING
NM3ECRE EQU 11 |+ -| | | | . CREATED
NM3EERR EQU 22 |+ +| | | | . ERROR
NM3EFRE EQU 107 |+ +| | | | . FREE
NM3EUSH EQU 118 |+ -| | | | . SHARE
NM3EUEX EQU 119 |+ -| | | | . EXCLUSIVE
* ( + ::= USED ) ( - ::= NOT USED )
*-----*
NM3LMIN DS CL3 RESERVED
EQU *-NM3A RECORD-LENGTH (MINIMUM)
DS OF
NM3L EQU *-NM3A

```

AVASNET RECORD=D3,PREFIX=N

```

MFCHK MF=C,
      PREFIX=N,
      MACID=D3,
      DMACID=D3D,
      SUPPORT=(C,D)
DS    OF
      *,##### PREFIX=N, MACID=D3 #####
*-----*
ND3A  DS    OF
ND3SLEN DS    H          PLAM-RECORD-LENGTH
      DS    C          PLAM-RESERVED PLAM - SK
      DS    C          PLAM-RESERVED PLAM - SA
ND3SSL DS    CL2        RECORD-KEY
ND3FUNC DS    C          DESCRIPTION-FUNCTION-CODE
ND3TYPE DS    X          DESCRIPTION-TYPE-CODE
      DS    CL1        RESERVED
ND3IND  DS    CL3        INDEX
ND3NAM  DS    CL30       NAME
      DS    CL2        RESERVED
*-----*
*          PARAMETERS
*          FUNCTION=W, TYPE=TIM
*          FUNCTION=D, TYPE=(NET/JOB/RES/VAL)
*-----*
ND3LMIN DS    CL4        RESERVED
      EQU  *-ND3A       RECORD-LENGTH (MINIMUM)
      DS    OF
ND3L    EQU  *-ND3A
    
```


AVASNET RECORD=W3,PREFIX=N

```

MFCHK MF=C,
      PREFIX=N,
      MACID=W3,
      DMACID=W3D,
      SUPPORT=(C,D)
DS OF
*,##### PREFIX=N, MACID=W3 #####
*-----*
NW3A DS OF
NW3SLEN DS H PLAM-RECORD-LENGTH
      DS C PLAM-RESERVED PLAM - SK
      DS C PLAM-RESERVED PLAM - SA
NW3SSL DS CL2 RECORD-KEY
NW3FUNC DS C DESCRIPTION-FUNCTION-CODE
NW3TYPE DS X DESCRIPTION-TYPE-CODE
      DS CL1 RESERVED
NW3IND DS CL3 INDEX
NW3NAM DS CL30 NAME
      DS CL2 RESERVED
*-----*
* PARAMETERS *
* FUNCTION=W, TYPE=TIM *
* FUNCTION=D, TYPE=(NET/JOB/RES/VAL) *
*-----*
NW3LMIN DS CL4 RESERVED
      EQU *-NW3A RECORD-LENGTH (MINIMUM)
      DS OF
NW3L EQU *-NW3A

```

AVASNET RECORD=S3,PREFIX=N

```

MFCHK MF=C,
      PREFIX=N,
      MACID=S3,
      DMACID=S3D,
      SUPPORT=(C,D)
DS    OF
      *,##### PREFIX=N, MACID=S3 #####
*-----*
NS3A  DS    OF
NS3SLEN DS   H          PLAM-RECORD-LENGTH
      DS    C          PLAM-RESERVED PLAM - SK
      DS    C          PLAM-RESERVED PLAM - SA
NS3SSL DS   CL2        RECORD-KEY
NS3FUNC DS   C          DESCRIPTION-FUNCTION-CODE
NS3TYPE DS   X          DESCRIPTION-TYPE-CODE
      DS    CL1        RESERVED
NS3IND DS   CL3        INDEX
NS3NAM DS   CL30       NAME
      DS    CL2        RESERVED
*-----*
*          PARAMETERS
*          FUNCTION=S, TYPE=(NET)
*
*-----*
NS3NUSE DS   CL8        NET-USER (ENTER:USER-ID)
NS3NACC DS   CL8        NET-ACCOUNT (ENTER:ACC-NUMBER)
NS3NCLA DS   CL8        NET-CLASS (ENTER:NET-CLASS)
NS3NLGM DS   CL8        NET-LOG
NS3NPAS DS   CL19       NET-PASSWORD
      DS    CL5        RESERVED
NS3NPAR DS   CL128      NET-PARAMETER
NS3NCAT DS   OCL54      NET-CAT (CATID IN JVA)
NS3NCTK DS   CL1        QUOTE OF CATID
*          QUOTE OF CATID
NS3ENSO EQU  X'4D'      . SERVER-BRACKET-OPEN (NET)
NS3ENSC EQU  X'5D'      . SERVER-BRACKET-CLOSE (NET)
NS3EJCK EQU  X'7D'      . CATID-QUOTE (JOB)
NS3ENCK EQU  X'7D'      . CATID-QUOTE (NET)
NS3NCTI DS   CL4        NET-CAT (STRING OF CATID)
      DS    CL49       RESERVED
      DS    CL10       RESERVED
NS3LMIN EQU  *-NS3A     RECORD-LENGTH (MINIMUM)
NS3NUPF DS   CL54       NAME OF USER-PAR-FILE
NS3TUPF DS   C          TYPE OF USER-PAR-FILE (LIB/SAM)
*          TYPE OF USER-PAR-FILE
NS3EUSM EQU  C'S'      . SAM
    
```

```
NS3EULB EQU C'L'          . LIB
NS3EUST EQU C'*'          . *STD
NS3EUNN EQU C' '          . *NONE
          DS CL21          RESERVED
          DS OF
NS3L     EQU *-NS3A
```

AVASNET RECORD=F3,PREFIX=N

```

MFCHK MF=C,
      PREFIX=N,
      MACID=F3,
      DMACID=F3D,
      SUPPORT=(C,D)
DS    OF
      *,##### PREFIX=N, MACID=F3 #####
*-----*
NF3S  DS    OF
NF3SLEN DS    H          PLAM-RECORD-LENGTH
      DS    C          PLAM-RESERVED PLAM - SK
      DS    C          PLAM-RESERVED PLAM - SA
NF3SSL DS    CL2        RECORD-KEY
NF3FUNC DS    C          DESCRIPTION-FUNCTION-CODE
NF3TYPE DS    X          DESCRIPTION-TYPE-CODE
      DS    CL1        RESERVED
NF3IND DS    CL3        INDEX
NF3NAM DS    CL30       NAME
      DS    CL2        RESERVED
*-----*
*          PARAMETERS
*          FUNCTION=F, TYPE=(TRA)
*-----*
NF3LFIL DS    CL54       LOCAL-FILE-NAME
NF3RFIL DS    CL54       REMOTE-FILE-NAME
NF3LMIN EQU    *-NF3A    RECORD-LENGTH (MINIMUM)
NF3FPAR DS    CL192      FT-PARAMETER
      DS    OF
NF3L    EQU    *-NF3A
    
```

9.2 External creation of tasks

BS2000 jobs, S procedures and JCL elements can be created with job generators and transferred to the AVAS library JCLLIB by means of COPY-ELEMENT.

The jobs can contain AVAS statements and variables in accordance with the AVAS rules.

Note that the AVAS statements are not checked for syntax errors. In particular, with S procedures, the separator string for separating statements and parameters must be generated as defined by the system parameters (see DEFAULT-PROPCAR-STRING on [page 48](#)).

When generating BS2000 jobs and S procedures, comments can be largely omitted in order to reduce the load on the system files.

The documentation on the BS2000 jobs and S procedures can be stored within AVAS in the DOCLIB.

9.3 External creation of documents

Within AVAS, the documentation elements can be assigned to the nets and their structure elements.

These documentation elements can also be created externally (outside the AVAS system) via suitable editors and can be transferred to the AVAS library DOCLIB via COPY-ELEMENT.

Note that the documentation elements are displayed via EDT following the DOCUMENT operation. The record length restrictions valid for EDT must therefore be taken into account.

Naming conventions must be observed when assigning the documents to the nets and their structure elements (see also the DOCUMENT operation and the CREATE-NET-DESCRIPTION statement in the “AVAS Statements” manual [\[2\]](#)).

10 AVAS program interface

The AVAS program interface enables selected AVAS functions to be invoked from within a user program, and also allows processing to be executed under control of the program.

When the user program calls up the program interface, the general syntax rules of the programming language concerned apply as for any other subprogram.

In the descriptions of the statements which are permitted for the program interface, a note is made of the operands which are required.

The user program uses the communication areas to pass across the statement and its operands, and in the same areas it receives back processing data and AVAS messages.

For the programming languages which are supported (Assembler and Cobol), program interfaces are provided in the form of macros or COPY elements, to permit problem-free communication between the user program and AVAS.

10.1 AVAS program interface statements

In this version of AVAS it is possible to enter the following statements via the program interface:

- CREATE-PLAN-NET
- CREATE-PROD-NET
- END
- MODIFY-COND-DESCRIPTION
- RESTART-NET
- SIGNON
- SUBMIT-NET

These statements correspond to the batch statements of the same name which are described in [chapter "Batch functions" on page 335](#).

In addition, the READ-AVAS-LIBRARY statement is available for administration purposes.

The statement operands should be passed in the following fields in the communication area:

```

statement-name
  operand=value                (def-name)
  operand=value                (def-name)
(def-name) specifies the name which is used for the program interface in
the definitions.

CREATE-PLAN-NET
  PERIOD-NAME=...             (AVSCPERN)
  NET-NAME=...                (AVSCNETN)
  [RUN-CONTROL-SYSTEM=...]   (AVSCRCSN)

CREATE-PLAN-NET
  NET-NAME=...                (AVSCNETN)
  [SYMDAT-NAME=...]          (AVSCSYDN)
  [EARLIEST-START=...]       (AVSCEAS )

CREATE-PROD-NET
  NET-NAME=...                (AVSCNETN)
  [PERIOD-NAME=...]          (AVSCPERN)
  [CALENDAR-NAME=...]        (AVSCCALN)
  [USER-PAR-FILE=...]        (AVSCUSPF)

MODIFY-COND-DESCRIPTION
  CONDITION-NAME=...          (AVSCCDNM)
  CONDITION-TYPE=NET          (AVSCCDNT)
  [CREATED-BY-NET=...]        (AVSCCDNN)
  [LIFE-TIME=...]             (AVSCLTI)
  [CONDITION-VALUE=...]       (AVSCCDVL)

MODIFY-COND-DESCRIPTION
  CONDITION-NAME=...          (AVSCCDNM)
  CONDITION-TYPE=JOB          (AVSCCDNT)
  [CREATED-BY-NET=...]        (AVSCCDNN)
  [CREATED-BY-INDEX=...]      (AVSCCDNI)
  [LIFE-TIME=...]             (AVSCLTI)
  [CONDITION-VALUE=...]       (AVSCCDVL)

MODIFY-COND-DESCRIPTION
  CONDITION-NAME=...          (AVSCCDNM)
  CONDITION-TYPE=RES          (AVSCCDNT)
  CONDITION-VALUE=...         (AVSCCDVL)

```


MODIFY-COND-DESCRIPTION	
CONDITION-NAME=...	(AVSCCDNM)
CONDITION-TYPE=VAL	(AVSCCDNT)
CONDITION-VALUE=...	(AVSCCDVL)
READ-AVAS-LIBRARY	
AVAS-USER-LIBRARY=JRNDAT	(AVSCUSLB)
NET-NAME={netname/group}	(AVSCNETN)
PERIOD-NAME=period	(AVSCPERN)
RUN-CONTROL-SYSTEM=avak	(AVSCRCSN)
READ-AVAS-LIBRARY	
AVAS-USER-LIBRARY=ABLDAT	(AVSCUSLB)
NET-NAME={netname/group}	(AVSCNETN)
PERIOD-NAME=period	(AVSCPERN)
RUN-CONTROL-SYSTEM=avak	(AVSCRCSN)
NET-STATUS=status	(AVSCNST1)
READ-AVAS-LIBRARY	
AVAS-USER-LIBRARY=ABLDAT	(AVSCUSLB)
CONDITION-NAME={condname/group}	(AVSCCDNM)
CONDITION-TYPE=type	(AVSCCDNT)
CREATED-BY-NET=netname	(AVSCDNN)
CREATED-BY-INDEX=index	(AVSCCDNI)
RESTART-NET	
NET-NAME=...	(AVSCNETN)
[RUN-CONTROL-SYSTEM=...]	(AVSCRCSN)
[RESTART-VARIANT=...]	(AVSCRSVT)
[ERROR-INDEX=...]	(AVSCERIN)
[ERROR-NAME=...]	(AVSCERJN)
SIGNON	
AVAS-USER-ID=...	(AVSCAUID)
PASSWORD=...	(AVSCAUPW)
AVAS-SYSTEM-ID=...	(AVSCASID)
SUBMIT-NET	
[PERIOD-NAME=...]	(AVSCPERN)
[NET-NAME=...]	(AVSCNETN)

The fields assigned to statement operands should be cleared for operands which are not to be used.

Specification of the RUN-CONTROL-SYSTEM operand is omitted if the requested statement does not permit this operand. In any subsequent statement which requires the RUN-CONTROL-SYSTEM operand, the default value is assumed if the operand is not defined.

The same applies to the NET-STATUS operand.

READ-AVAS-LIBRARY – Enable read access to journal and runtime files

This statement enables user programs to read the AVAS journal file and the run control file. The program can request the following information:

From the journal file:

- directory of the nets in a net group

READ-AVAS-LIBRARY

AVAS-USER-LIBRARY=JRNDAT	(AVSCUSLB)
NET-NAME=group	(AVSCNETN)
RUN-CONTROL-SYSTEM=avak	(AVSCRCNS)
PERIOD-NAME=period	(AVSCPERN)

- all the journals for a net

READ-AVAS-LIBRARY

AVAS-USER-LIBRARY=JRNDAT	(AVSCUSLB)
NET-NAME=netname	(AVSCNETN)
RUN-CONTROL-SYSTEM=avak	(AVSCRCNS)

From the run control file for nets:

- overview of the nets in a net group

READ-AVAS-LIBRARY

AVAS-USER-LIBRARY=ABLDAT	(AVSCUSLB)
NET-NAME=group	(AVSCNETN)
RUN-CONTROL-SYSTEM=avak	(AVSCRCNS)
NET-STATUS=status	(AVSCNST1)
PERIOD-NAME=period	(AVSCPERN)

- structure data for a net

READ-AVAS-LIBRARY

AVAS-USER-LIBRARY=ABLDAT	(AVSCUSLB)
NET-NAME=netname	(AVSCNETN)
RUN-CONTROL-SYSTEM=avak	(AVSCRCNS)
NET-STATUS=status	(AVSCNST1)

From the run control file for events:

- overview of the condition descriptions
- data for a condition description

READ-AVAS-LIBRARY

AVAS-USER-LIBRARY=ABLDAT	(AVSCUSLB)
CONDITION-NAME={condname / group}	(AVSCCDNM)
CONDITION-TYPE=type	(AVSCCDNT)
CREATED-BY-NET=netname	(AVSCCDNN)
CREATED-BY-INDEX=index	(AVSCCDNI)

Names in parentheses identify the fields in the communication area into which the operands must be written (see [section "Structure of the communication areas" on page 406](#)).

Combinations of operands other than those shown above are not permitted.

AVAS-USER-LIBRARY={ABLDAT / JRNDAT}

ABLDAT Data items are to be read from the run control file.

JRNDAT Data items are to be read from the journal file.

This operand must always be specified.

CONDITION-NAME={condname / group}

Name of the condition description.

If this is specified in fully qualified form, the function supplies data on the condition parameters and the user of the condition.

If it is specified in partially qualified form, the function supplies a list of the condition descriptions which match the partial qualification.

If the operand is not specified, then CONDITION-NAME=\$ug_* is assumed.

CONDITION-TYPE=type

Specifies the condition type. The possible condition types can be found in the manual "AVAS Statements" [2] (under the ADD-CONDITION-DESCRIPTION statement).

If the data for a completely specific condition description is required, this operand must be specified, because the CONDITION-NAME alone is not a unique identifier.

This is an optional operand.

- CREATED-BY-NET**=[\$ug__{netname}[_date[_time]]]
 Restricts the selection of condition descriptions to those created by a particular net.
 If \$ug_ is omitted, the function will insert the user's own group before the net name.
 If _date or _time is omitted, the function supplies data for the last condition description recorded.
 This operand may only be specified for the condition types NET and JOB.
- CREATED-BY-INDEX**=index
 Restricts the selection of records to those created at a particular job index.
 This operand may only be specified for the condition type JOB.
- NET-NAME**={netname / group}
 Name of the net.
 If this is specified in fully qualified form, the function supplies data on a net (structure data or journals).
 If it is specified in partially qualified form, the function supplies a directory of the nets which match the partial qualification.
- NET-STATUS**=status
 Restricts the selection to nets with the specified status. The possible status values can be found in the "AVAS Statements" manual [2] (under the SHOW-NET-STATUS statement).
 This operand may only be specified with partially qualified net names.
- PERIOD-NAME**={period / [dd.mm.yy/hh:mm:ss,dd.mm.yy/hh:mm:ss]}
 Specification of a period.
 All nets with a start time in this period are to be selected.
 The selection is made in the run control file (ABLDAT) using the EARLIEST-START parameter and in the journal file (JRNDAT) using the PLAN-START parameter from the net names since EARLIEST-START is not available here.
- period Symbolic name of a period.
- dd.mm.yy/hh:mm:ss,dd.mm.yy/hh:mm:ss
 Real date and time which determine the start and end date/time.
 If no end date and end time are specified, the end date is set to the start date and the end time is set to 23:59:59.
 If the start time is not specified, 00:00:00 is used.

RUN-CONTROL-SYSTEM=avak

Name of the run control system.

If this operand is not specified, the run control system which is assigned to the user is assumed.

The requested information is output to the result area (OUTAREA, see [section “Structure of the communication areas” on page 406](#)) or into the SAM file assigned to the area.

The structure of the data is specified in the following Assembler macros or Cobol COPY elements, as applicable:

ASS	Cobol	
AVSASSJD	AVSCOBJD	Directory entry of the journal file
AVSASSJN	AVSCOBJN	Journal record header
AVASJRN	¹⁾	Key-specific part of the journal records (see section “Data structures of the backup journal and the emergency journal file” on page 194)
AVSASSAD	AVSCOBAD	Net directory entry for the run control file
AVSASSAN	AVSCOBAN	Net structure data in the run control file
AVSASSCD	AVSCOBCE	Directory entry for the condition descriptions
AVSASSCE	AVSCOBCE	Condition descriptions in the run control file

1) There is no COBOL COPY element for these data items.

Note

If the READ-AVAS-LIBRARY statement is used in a status monitor to output the current journals, the monitor should from time to time write 'IN-JRLDAT' in the UPAM-ZD job variable, in order to merge any emergency journals into the journal file.

10.2 Structure of the communication areas

Three areas are used in the exchange of information at the AVAS program interface:

- the communication area (COMAREA)
- the result area (OUTAREA)
- the work area (WRKAREA)

The communication area (COMAREA)

The communication area specifies all the parameters required for the statement, together with the return information necessary for evaluating the processing carried out by AVAS. The definitions for this area are set down in the AVSASSBC macro (see [page 426](#)) or in the COPY element AVSCOBBC (see [page 452](#)), as applicable. Specifically, the data items are exchanged using the following symbolic names:

AVSCASID	AVAS-SYSTEM-ID operand
AVSCAUID	AVAS-USER-ID operand
AVSCAUPW	PASSWORD operand
AVSCCDNM	CONDITION-NAME operand
AVSCCDNT	CONDITION-TYPE operand
AVSCCDNN	CREATED-BY-NET operand
AVSCCDNI	CREATED-BY-INDEX operand
AVSCCDVL	CONDITION-VALUE operand
AVSCEAS	EARLIEST-START/NEW-PLAN-START, date and time operand
AVSCERIN	ERROR-INDEX operand
AVSCERJN	ERROR-NAME operand
AVSCLTI	LIFE-TIME, date and time operand
AVSCMDC	Statement code, statement key (see the AVSASSBC macro on page 426 or COPY element AVSCOBBC on page 452 , as applicable)
AVSCNETN	NET-NAME operand
AVSCNST1	NET-STATUS operand
AVSCPERN	PERIOD-NAME operand
AVSCRCSN	RUN-CONTROL-SYSTEM operand
AVSCRSVT	RESTART-VARIANT operand

AVSCSYDN	SYMDAT-NAME operand
AVSCUSLB	AVAS-USER-LIBRARY operand
AVSCUSPF	USER-PAR-FILE operand
AVSRCMDT	Return data: statement text, name of the AVAS statement
AVSRFDBK	Return data: the return code, specifying how successful the AVAS processing was
AVSRDMSN	Return data: DMS message number
AVSRMSGN	Return data: AVAS message number
AVSRMSG1	Return data: 1st value inserted in the message text
AVSRMSG2	Return data: 2nd value inserted in the message text
AVSRMSG3	Return data: 3rd value inserted in the message text
AVSRNECT	Return data: the number of elements selected for processing
AVSRNEER	Return data: the number of elements which could not be processed
AVSRNEOK	Return data: the number of elements which were processed without errors
AVSRRLTC	Return data: RESULT code, specifying error-free processing of the element

Details of the data to be supplied, and the return data provided for evaluation, will be found in the description of each individual statement. To make them easier to use, the symbolic names are quoted in parentheses. For programming purposes, further details are specified in the macro or COPY element, as applicable, by definitions which are required particularly in the case of fields which are encoded (e.g. AVSCMDC, AVSRFDBK).

The SIGNON statement is the first statement which must be sent to the AVAS program interface.

To execute a normal signoff from processing with the ZDs, END must be specified as the last statement. If it is not, signoff from the ZDs will not take place until the end of the program or task, when it will be forced.

Note

When a program run terminates, AVAS informs the operating system if an AVAS routine is to be activated (this is also the case at the end of tasks). If the user program corrupts or modifies this communication by calls of its own (with the STXIT Assembler macro), an error-free signoff from the ZDs will not be effected.

A possible result of this is that a signon remains in effect with the ZDs for which there is no longer a user. This can lead in turn to the number of permitted batch users of the ZDs being reduced, possibly even to zero.

Result area (OUTAREA)

The result area is where the user program receives the results from the interface call. This area consists of an area header and a results table. The size and structure of the table entries are determined by the statement concerned.

The contents of the area header are written by the Assembler macro AVSASSBO or by the COBOL COPY element AVSCOBBO. It contains the following entries:

- AVSOFIL** The name of a SAM file in which the results are to be stored, record by record. The new data items are always added to the end of the file. The existing contents are not overwritten
To guarantee separation of the individual results for each statement, each result is prefixed by a record which, as of position 1 contains the statement name in the general length for statements, and as of position 33 the date the statement was issued in the form yymmddhhmmss
(e.g. READ-AVAS-LIBRARY010108112856)..
- AVSONECT** The number of elements (nets or events) for the results of which there is space in the output area.
If AVSONECT contains the value 0, the result will be written into the SAM file. If AVSONECT contains a non-zero value, then AVSOFIL is ignored and the results of the processed nets are entered in the results table, insofar as there is sufficient space for them. If too little space was reserved, the fields AVSRNECT, AVSRNEER and AVSRNEOK can be used to determine whether or not all the elements were processed without error. However, it may sometimes be impossible to determine which element led to an error, e.g. when it was not possible to write the result of the errored net into the result table. Consequently, it is safer to request output to the SAM file when the number of elements to be processed cannot be calculated.
- AVSOTAB** Start of the results table
The available space from this point on must be at least equal to the product of the value in AVSONECT and the length of each result. The subsequent macros or COPY elements, as appropriate, must be called at this point to write a table entry.

The statements produce the following results (for each element):

SIGNON and END

These statements produce no result in this area.

CREATE-PLAN-NET, CREATE-PROD-NET, MODIFY-COND-DESCRIPTION, SUBMIT-NET and RESTART-NET.

The result from any of these statements is written out using the AVSASSRT macro or the COPY element AVSCOBRT, as appropriate:

AVSECDNM	Name of the condition description
AVSEEASD	EARLIEST-START/NEW-PLAN-START date parameter
AVSEEAET	EARLIEST-START/NEW-PLAN-START time parameter
AVSENAME	Name of the element
AVSENETN	Name of the net
AVSERLTC	Code for the RESULT text (positive result; see also the AVSRRLTC field in the communication area)
AVSERLTT	RESULT text
AVSERSVT	Restart variant
AVSESYND	SYMDAT-NAME parameter which led to the selection.

The way that certain fields are encoded (e.g. AVSERLTC) is specified in the AVSASSRT macro or in the COPY element AVSCOBCQ.

READ-AVAS-LIBRARY

Different results are supplied, depending on the NET-NAME and CONDITION-NAME operands and on AVAS-USER-LIBRARY.

- Parameter values: AVAS-USER-LIBRARY=ABLDAT and NET-NAME=group
An entry is stored in the result table for each net, in accordance with the AVSASSAD macro or the COPY element AVSCOBAD.
- Parameter values: AVAS-USER-LIBRARY=ABLDAT and NET-NAME=netname
An entry is stored in the result table for each net structure element, in accordance with the AVSASSAN macro or the COPY element AVSCOBAN.
- Parameter values: AVAS-USER-LIBRARY=ABLDAT and CONDITION-NAME=group
An entry is stored in the result table for each event, in accordance with the AVSASSCD macro or the COPY element AVSCOBAD.

- Parameter values: AVAS-USER-LIBRARY=ABLDAT and CONDITION-NAME=condname
An entry is stored in the result table for each event, in accordance with the AVSASSCE macro or the COPY element AVSCOBCE.
- Parameter values: AVAS-USER-LIBRARY=JRNDAT and NET-NAME=group
An entry is stored in the result table for each net, in accordance with the AVSASSJD macro or the COPY element AVSCOBJD.
- Parameter values: AVAS-USER-LIBRARY=JRNDAT and NET-NAME=netname
An entry is stored in the result table for each journal record, in accordance with the AVSASSJN macro or the COPY element AVSCOBJN. The structure of the area AVPSOUTP (output data items) is specified by the AVASJRN macro.
A COBOL COPY element for the different types of journal record is not supplied with the system.

The work area (WRKAREA)

The work area is used to hold all the messages which are output by AVAS during the processing of a statement. This output corresponds to the log which is created for the AVAS batch functions. Using the AVSASSBW macro or the COPY element AVSCOBW, as appropriate, an area header and a message table are generated. The number of table entries is controlled by means of the COUNT parameter.

The following fields in the work area are used for the exchange of data:

AVSWFILE	Name of a file in which the messages are to be stored, instead of in the work area. The messages are always added to the end of the file. The existing contents of the file are thus retained.
AVSWGCT	Maximum number of messages which can be held in the area. The value zero indicates that no area is available, and the messages are instead to be written to the file. If AVSWGCT contains a zero and AVSWFILE contains blanks, the user program will receive no messages. Nevertheless, processing will be carried out.
AVSWGCR	Number of messages which were output by AVAS during processing of the statement. If this is greater than the value in the AVSWGCT field, and if AVSWGCT contains a non-zero value, the last messages will be unavailable to the program.

The following fields are output for every message:

AVSWMSGK	AVAS message code
AVSWMSGN	AVAS message number
AVSWMSG	If this field contains the code AVSWQNTL, the next table entry does not contain a new message, but rather the continuation of the current one (longer than 125 characters).
AVSWMSGT	AVAS message text

Further details required for programming purposes are specified in the macro or COPY element, as appropriate.

The output of messages can be suppressed by clearing the AVSWFILE field (the file name) and setting AVSWGCT (number) to zero.

10.3 Assembler Interface

1. Calling the Assembler interface

For Assembler programs, the following register usage is required:

Register 1 Address of a parameter area containing the following three addresses:

1. Address of the communication area (COMAREA)
2. Address of the results area (OUTAREA)
3. Address of the work area (WRKAREA)

Register 13 Address of an 18-word register save area.

Register 14 Address for the return from AVAS to the user program.

Register 15 Entry address AVASBASS of the AVAS linkage module AVSBCALL. When processing is finished, the two rightmost bytes of register 15 contain the value of the AVSRFDBK field.

The Assembler call is performed in the user program:

```
LA 1,<parameterarea>
LA 13,<savearea>
L 15,=V(AVASBASS)
BALR 14,15
```

Registers 13 to 15 and the register save area are used by AVAS to check on the orderly termination of the programs. If the user program undertakes its own checking here, it is possible that errors may occur in conjunction with the AVAS central access routines (ZDs).

2. Program compilation

In order to compile an Assembler program with AVAS parameter structures, the macro library must be assigned. This assignment can be made with the following BS2000 command:

```
/ADD-FILE-LINK LINK-NAME=OLDLIB,FILE-NAME=SYSLIB.AVAS.085
```

3. Program linkage

The AVAS interface module AVSBCALL must be permanently linked to the main program from the library SYSLNK.AVAS.085.

4. Starting the program

When the first call is made from the user program to AVAS, the AVAS interface module loads additional modules. This load call specifies SYSLNK.AVAS.085 as the library name.

If the AVAS modules are not available in the library with this name, or if the AVAS library is held on another pubset, then before the program is started the following BS2000 command must be used to make an assignment:

```
/ADD-FILE-LINK LINK-NAME=SYSLNK, -
/ FILE-NAME=:<catid>:$<userid>.SYSLNK.AVAS.085
```

10.3.1 Programming example in Assembler

This example shows how the statements SIGNON, SUBMIT-NET and END are handled from a programming point of view. Error handling measures are only outlined.

```
***** EXAMPLE PROGRAM: AVSASSEM *****
AVSASSEM CSECT
*
R1      EQU    1
R2      EQU    2
R3      EQU    3
R4      EQU    4
R5      EQU    5
R10     EQU    10
R13     EQU    13
R14     EQU    14
R15     EQU    15
*
      BALR   R10,0
      USING *,R10
* CALL      LA      R13,SAVEAREA
              AVAS              S I G N O N
              BAL   R14,DEL#BCOM  DELETE BCOM-AREA
              MVI  AVSCMDC,AVSQSIGN SET   CMD:SIGNON
              MVC  AVSCAUID,CS#USRID AVAS-USER-ID
              MVC  AVSCAUPW,CS#USRPW AVAS-USER-PASSWORD
              MVC  AVSCASID,CS#SYSID AVAS-SYSTEM-ID
*
      LA     R1,PARAM
      L      R15,=V(AVASBASS)
      BALR  R14,R15
      CLI   AVSRFDB1,AVSQ10K
      BNE   ERR#SGN
* CALL      LA      R13,SAVEAREA
              AVAS              S U B M I T - N E T
              BAL   R14,DEL#BCOM  DELETE BCOM-AREA
              MVI  AVSCMDC,AVSQSUNE SET  CMD:SUBMIT-NET
              MVC  AVSCNETN,CS#NETNM OPR:NET-NAME
              MVC  AVSCPERN,CS#PERNM PERIOD-NAME
              MVC  AVSONECT,ANZ#RSLT SET  MAX. RESULT-COUNTER
*
```

```

        LA    R1,PARAM
        L     R15,=V(AVASBASS)
        BALR R14,R15
        CLI  AVSRFDB1,AVSQ10K
        BNE  ERR#SUN
*
*          SUBMIT WITHOUT ERROR
*
ERR#SUN EQU  *
* CALL    AVAS                      E N D
        BAL R14,DEL#BCOM            DELETE BCOM-AREA
        MVI AVSCMDC,AVSQEND        SET CMD:END
*
        LA    R1,PARAM
        L     R15,=V(AVASBASS)
        BALR R14,R15
        CLI  AVSRFDB1,AVSQ10K
        BNE  ERR#END
*
ERR#SGN EQU  *
ERR#END EQU  *
        TERM
*
*          S U B - R O U T I N E
DEL#BCOM EQU  *                      DELETE BCOM-AREA
        LA    R2,AVASBCOM
        LA    R3,AVSR#QCL
        LA    R4,CS#BLANK
        LA    R5,L'CS#BLANK
        ICM  R5,B'1000',CS#BLANK
        MVCL R2,R4
        BR   R14
*
*          S A V E - A R E A
SAVEAREA DS  18F
*          P A R A M E T E R - L I S T
PARAM    DS  0F
PADRBCOM DC  A(AVASBCOM)
PADRBOUT DC  A(AVASBOUT)
PADRBWRK DC  A(AVASBWRK)
*          DC  - DEFINITION
ANZ#RSLT DC  YL2(MAX#RSLT)
CS#BLANK DC  CL4' '                      BLANKS
CS#USRID DC  CL8'TEST'                   AVAS-USER-ID
CS#USRPW DC  CL8'TEST'                   AVAS-USER-PASSWORD
CS#SYSID DC  CL7'SYSIDAV'                 AVAS-SYSTEM-ID
CS#NETNM DC  CL32'TEST.NET_'             NET-NAME
CS#PERNM DC  CL37' '                     PERIOD-NAME
        LTORG

```

```

*          A V A S - DEFINITION
AVASBCOM AVSASSBC
AVASBOUT AVSASSBO
          AVSASSRT
          DS    19CL(L'AVSERSLT)
MAX#RSLT EQU    (*-AVSERSLT)/L'AVSERSLT
AVASBWRK AVSASSBW COUNT=0
          END

```

10.3.2 Macros for defining the communication areas

AVSASSAD – Define net contents directory entry for run control file

Macro call for addressing the entries in the net contents directory:

Macro	Operands
AVSASSAD	MF = <u>C</u> / D ,PREFIX = <u>A</u> / <char (1)> ,MACID = <u>VSD</u> / <char (3)> ,EQU = <u>NO</u> / YES

- MF** The MF operand (“Macro Form”) controls code generation.
- C The layout of the data structure (generally the parameter area) is created; in doing so, each field and each equate is named. This data structure becomes a part of the current program control/dummy section (CSECT/DSECT).
- D Creates the layout of the data structure as for MF = C; in addition, a DSECT statement is created.
- PREFIX** The PREFIX operand is used to generate the names which have to be created. PREFIX, which is exactly one letter, is used as the first letter of all the names. The default value is the identifying letter of the functional unit to which the macro belongs. To avoid identical names arising, PREFIX should be used when the same data structure is to be used repeatedly within a module.
- MACID** The MACID operand is used to generate the names which have to be created. This three-character string specifies the second to fourth character of the name.
 The default value, VSD, guarantees that names are unique within the component group.

EQU Controls the generation of equates for the key fields.

NO No equates are generated.

YES Equates are generated.

Note

The AVSASSAD macro generates an extension of the output area. Consequently it must be called after the AVSASSBO macro.

Expansion of the AVSASSAD macro

```

AVSASSAD
MFCHK MF=C,
    PREFIX=A,
    MACID=VSD,
    DMACID=VSD,
    SUPPORT=(C,D)
DS    OF
    *,##### PREFIX=A, MACID=VSD #####
*-----*
*      ABLDAT: NET DIRECTORY DESCRIPTION
AVSDADIR DS    OCL96      ABLDAT DIRECTORY
AVSDNETN DS    CL32      BK_NETNAME_PLANSTARTDATE_PLANSTARTTIME
AVSDNST1 DS    X         STATE 1
AVSDNST2 DS    X         STATE 2
AVSDNST3 DS    X         STATE 3
*
AVSDNERR DS    X         STATE OF ERROR
AVSDNRST DS    X         STATE OF RESTART
AVSDNCWT DS    X         STATE OF CONDWAIT
AVSDQSON EQU  X'80'     . 0 = NET STATE ON; 1 = NET STATE OFF
AVSDNHLD DS    X         STATE OF HOLD
AVSDNES  DS    OCL12     EARLIEST-START
AVSDNESD DS    CL6      EARLIEST-START-DATE YYMMDD
AVSDNEST DS    CL6      EARLIEST-START-TIME HHMMSS
AVSDNLS  DS    OCL12     LATEST-START
AVSDNLSD DS    CL6      LATEST-START-DATE YYMMDD
AVSDNLST DS    CL6      LATEST-START-TIME HHMMSS
AVSDNTYP DS    CL1      NET-TYPE
AVSDNDSL DS    X         NET-DELAY-SOLUTION
AVSDOPST DS    X         OPERATOR-START YES/NO
AVSDQSN  EQU  92         . NO
AVSDQSY  EQU  93         . YES
AVSDNHTW DS    X         STATE OF HOSTWAIT
          DS    CL29     RESERVED
*-----*
          DS    OC

```


Explanation of the status displays

NST1	X	Current processing status of the net in the run control file. The meaning of the keywords is described under the dialog functions (see SHOW-NET-STATUS etc.)
	X'14'	ABENDED
	X'15'	ENDED
	X'16'	ERROR
	X'17'	HOLD
	X'18'	RUNNING
	X'19'	WAITING
	X'1A'	CONDWAIT
	X'1B'	RESTARTED
	X'1C'	RESUMED
	X'3D'	OPWAIT
	X'3F'	IGNORED
	X'44'	START
	X'9A'	SHIFTED
	X'AD'	HOSTWAIT
NST2	X	Requested action of a dialog function (also batch and program interface). The action is implemented by the run control system if the net status and the processing state allow it.
	X'00'	LOW No action has been requested.
	X'17'	HOLD
	X'1C'	RESUMED
	X'41'	CANCEL
NST3	X	Saving of the current processing status NST1, if net processing is interrupted (NST1 HOLD or NST1 RESUMED).
	X'00'	LOW No status has been saved.
	X'16'	ERROR
	X'18'	RUNNING
	X'19'	WAITING
	X'1A'	CONDWAIT
	X'1B'	RESTARTED
	X'3D'	OPWAIT
	X'44'	START
	X'AD'	HOSTWAIT

Example

```
NST1  NST2  NST3
X'18' X'17' X'00'  RUNNING  HOLD  LOW
```

The net is running and HOLD-NET has been requested via the dialog function.

```
NST1  NST2  NST3
X'17' X'00' X'18'  HOLD      LOW      RUNNING
```

No job is running and no structure element can be processed because HOLD is set.

Whether or not there are also any structure elements with the status ERROR, NO-OCCURE (CONDWAIT) or HOSTWAIT, or a restart was initiated, can be determined from the net state switch.

NET-STATE-SWITCH

NERR	X	Indicates whether there are any structure elements with the status ERROR.
NRST	X	Indicates whether a restart has been initiated for structure elements.
NCWT	X	Indicates whether there are any structure elements with the status NO-OCCURE.
NHTW	X	Indicates whether there are any structure elements with the status HOSTWAIT.
NHLD	X	Indicates whether the HOLD request has been set for structure elements.
	X'00'	LOW The status is not set.
	X'80'	2**7=1 The status is set.

Example

```
NST1  NST2  NST3  NERR  NRST  NCWT  NHLD
X'17' X'00' X'18' X'80' X'00' X'80' X'80'
```

Net processing was interrupted by HOLD-NET (NST1 = HOLD).

At the time of the interruption, the net was running (NST3 = RUNNING). The net contains structure elements for which the ERROR status is set (bit 2**7 set if NERR), structure elements for which the NO-OCCURE status is set (bit 2**7 set if NCWT) and structure elements for which the HOLD status is set (bit 2**7 set if NHLD).

If the HOLD status is canceled, the net reverts to the NST3 status (here RUNNING).

Furthermore, processing of a part of the net has come to a standstill because an error occurred during processing of a structure element. In further operation, the ERROR status will occur if no restart is initiated beforehand (NERR is set).

In addition, processing of a part of the net has come to a standstill because a condition is not satisfied (NCWT is set). In further operation, the COND-WAIT status will come into effect if the condition cannot be satisfied.

AVSASSAN – Define net structure data

Macro	Operands
AVSASSAN	MF = <u>C</u> / D ,PREFIX = <u>A</u> / <char (1)> ,MACID = <u>VSN</u> / <char (3)> ,EQU = <u>NO</u> / YES

For descriptions of the MF and PREFIX operands, see AVSASSAD on [page 415](#).

MACID The MACID operand is used to generate the names which have to be created. This three-character string specifies the second to fourth character of the name.

Default value: VSN

EQU Controls the generation of equates for the key fields.

NO No equates are generated.

YES Equates are generated.

Note

The AVSASSAN macro generates an extension of the output area. Consequently it must be called after the AVSASSBO macro.

Expansion of the AVSASSAN macro

```

AVSASSAN
MFCHK MF=C,
        PREFIX=A,
        MACID=VSN,
        DMACID=VSN,
        SUPPORT=(C,D)
DS      OF
        *,##### PREFIX=A, MACID=VSN #####
*-----*
*      ABLDAT: NET DESCRIPTION
AVSNANET DS      OCL612
AVSNFNAM DS      OCL30      FUNCTION NAME
AVSNCNAM DS      OCL32      COND-NAME
AVSNSNM DS      OCL32      SUBNET-NAME
AVSNJNAM DS      CL30      JOB-NAME
DS      CL2
AVSNIND DS      CL3      FUNCTION INDEX
AVSNSIND DS      CL3      START-INDEX
AVSNSYN DS      CL3      FUNCTION SYNC-INDEX

```

```

AVSNSTT DS X FUNCTION STATE 1
AVSNST2 DS X FUNCTION STATE 2
AVSNST3 DS X STATE: NO-PLAN/NO-SUBMIT/IGNORE/DELETED
AVSNSTTS DS X SAVE STATE FOR HOLD IN STT
AVSNENDG DS CL2 END-CONDITION JOB/PROC
AVSNFUNC DS CL1 FUNCTION
AVSNATYPE DS X TYPE
AVSNRSRV DS CL1 RESTART-VARIANT
AVSNHPVS DS CL4 HOME-PVS MONJV
AVSNSRVN DS OCL8 SERVER/SINIX-ID
          DS CL4
AVSNCATI DS CL4 CATID JOB/PROC
    
```

```

*-----*
*                               USED FIELDS ABOUT TYPE AND FUNCTION                               *
*-----*
    
```

```

* FIELD          TYPE: JVA!NET !JOB!RES !VAL !TIM!MOD!EXT!STD!TRA *
*                FUNC: C !C D S!C D!C A M D!C A M D!W !J P!J P!J P!F *
*-----*
*...RSRV SEL-RS-VL! !+ !+ !+ !+ ! !+ +!+ +!+ +! *
*                ! ! ! ! ! ! ! ! ! ! *
*...LST. LST-STRT ! ! +! ! ! ! !+ +!+ +!+ +!+ *
*...LTS. LST-STRT ! ! +! ! ! ! !+ +!+ +!+ +!+ *
*                .DATE/TIME ! ! ! ! ! *
*...LOC. LST-OCC !+ !+ !+ !+ !+ !+ ! ! ! ! *
*...LTO. LST-OCC !+ !+ !+ !+ !+ !+ ! ! ! ! *
*                .DATE/TIME ! ! ! ! ! *
*...DSO DEL-SOLU !+ !+ +!+ !+ !+ !+ !+ +!+ +!+ +!+ *
*                ! ! ! ! ! ! ! ! ! ! *
*...VA VALUE ! ! ! ! ! + + ! ! ! ! ! *
*...VC VAL-CODE ! ! ! ! + + ! ! ! ! ! ! *
*...OS OCC-V-STR! ! ! ! ! !+ ! ! ! ! ! *
*...OC OCC-V-COD! !+ !+ !+ ! ! ! ! ! ! *
*...ES ERR-V-STR! ! ! ! ! !+ ! ! ! ! ! *
*...EC ERR-V-COD! !+ !+ !+ ! ! ! ! ! ! *
*                ! ! ! ! ! ! ! ! ! ! *
*...CN CREA-N-N ! !+ + !+ +! ! ! ! ! ! *
*...CI CREA-IDX ! ! !+ +! ! ! ! ! ! ! *
*
*... PREFIX=AVSN
* . FU
* . TYPE
*-----*
    
```

```

*
*                               RESTART-POINTS
AVSNRSI1 DS CL3 RESTART 1:INDEX
AVSNRST1 DS X RESTART 1:TYP
AVSNRSJ1 DS CL30 RESTART 1:JOBNAME
AVSNRSI2 DS CL3 RESTART 2:INDEX
AVSNRST2 DS X RESTART 2:TYP
    
```

```

AVSNRSJ2 DS    CL30      RESTART 2:JOBNAME
AVSNRSI3 DS    CL3       RESTART 3:INDEX
AVSNRST3 DS    X         RESTART 3:TYP
AVSNRSJ3 DS    CL30      RESTART 3:JOBNAME
*
AVSNLTS  DS    0CL14     LATEST-START-DATE(8) AND TIME(6)
AVSNLTS  DS    0CL8      LATEST-START-DATE YYYYMMDD
AVSNLTS  DS    CL2       LATEST-START-DATE YY..
AVSNLST  DS    0CL12     LATEST-START DATE(6) AND TIME(6)
AVSNLSTD DS    CL6       LATEST-START-DATE YYMMDD
AVSNLSTT DS    CL6       LATEST-START-TIME HHMMSS
      ORG    AVSNLTS
AVSNLTO  DS    0CL14     LATEST-OCCURE-DATE(8) AND TIME(6)
AVSNLTOD DS    0CL8      LATEST-OCCURE-DATE YYYYMMDD
AVSNLTOH DS    CL2       LATEST-OCCURE-DATE YY..
AVSNLOC  DS    0CL12     LATEST-OCCURE DATE(6) AND TIME(6)
AVSNLOCD DS    CL6       LATEST-OCCURE-DATE YYMMDD
AVSNLOCT DS    CL6       LATEST-OCCURE-TIME HHMMSS
AVSNDSO  DS    X         DELAY-SOLUTION
      DS    CL17         RESERVED
AVSNLMIN EQU    *-AVSNANET
*
*-----*
*                               VARIABLE AREA ABOUT TYPE AND FUNCTION   *
*-----*
AVSNVANF DS    CL420
*-----*
* FUNCTION=C ,TYPE=JVA *
      ORG    AVSNVANF
AVSNFCN  DS    CL54      JVA-NAME
AVSNVPOS DS    CL3       JVA-POSITION
AVSNVLEN DS    CL3       JVA-LENGTH
AVSNCVAL DS    CL256     JVA-VALUE
*-----*
* FUNCTION=C ,TYPE=NET *
      ORG    AVSNVANF
AVSNCNOC DS    12XL1     OCCURE-VALUE CODE-KEYWORD
AVSNCNEC DS    12XL1     ERROR-VALUE CODE-KEYWORD
AVSNCNCN DS    CL32      COND-DESCR CREATED BY NET-NAME
*-----*
* FUNCTION=C ,TYPE=JOB *
      ORG    AVSNVANF
AVSNCJOC DS    12XL1     OCCURE-VALUE CODE-KEYWORD
AVSNCJEC DS    12XL1     ERROR-VALUE CODE-KEYWORD
AVSNCJCN DS    CL32      COND-DESCR CREATED BY NET-NAME
AVSNCJCI DS    CL3       COND-DESCR CREATED BY INDEX
*-----*
* FUNCTION=C ,TYPE=RES *
      ORG    AVSNVANF

```

```

AVSNCROC DS    12XL1      OCCURE-VALUE CODE-KEYWORD
AVSNCREC DS    12XL1      ERROR-VALUE CODE-KEYWORD
*-----*
* FUNCTION=C   ,TYPE=VAL   *
      ORG     AVSNVANF
AVSNCVOS DS    CL128      OCCURE-VALUE STRING
AVSNCVES DS    CL128      ERROR-VALUE STRING
*-----*
* FUNCTION=W   ,TYPE=TIM   *
      ORG     AVSNVANF
*-----*
* FUNCTION=A   ,TYPE=RES   *
      ORG     AVSNVANF
AVSNARVC DS    XL1        VALUE-CODE
*-----*
* FUNCTION=A   ,TYPE=VAL   *
      ORG     AVSNVANF
AVSNAVVA DS    CL128      VALUE
*-----*
* FUNCTION=M   ,TYPE=RES   *
      ORG     AVSNVANF
AVSNMRVC DS    XL1        VALUE-CODE
*-----*
* FUNCTION=M   ,TYPE=VAL   *
      ORG     AVSNVANF
AVSNMVVA DS    CL128      VALUE
*-----*
* FUNCTION=D   ,TYPE=RES   *
*             ,TYPE=VAL   *
      ORG     AVSNVANF
*-----*
* FUNCTION=D   ,TYPE=NET   *
*-----*
      ORG     AVSNVANF
AVSNDNCN DS    CL32      COND-DESCR CREATED BY NET-NAME
*-----*
* FUNCTION=D   ,TYPE=JOB   *
*-----*
      ORG     AVSNVANF
AVSNDJCN DS    CL32      COND-DESCR CREATED BY NET-NAME
AVSNDJCI DS    CL3       COND-DESCR CREATED BY INDEX
*-----*
* FUNCTION=J   ,TYPE=STD   *
*             ,TYPE=MOD   *
*             ,TYPE=EXT   *
* FUNCTION=P   ,TYPE=STD   *
*             ,TYPE=MOD   *
*             ,TYPE=EXT   *

```

```

*           ,TYPE=EXX                                     *
* FUNCTION=X ,TYPE=STD                                     *
*           ,TYPE=MOD                                     *
*           ,TYPE=EXT                                     *
          ORG AVSNVANF
AVSNCJLD DS CL8      DATE JOB-START (YYYYMMDD)
AVSNTSN  DS CL4      TSN  JOB-START
AVSNCJCR DS X        CONDITION JOB CREATION STATE
AVSNJENT DS X        ENTER-PARAMS
AVSNJUSE DS CL8      JOB-USER
AVSNJACC DS CL8      JOB-ACCOUNT
AVSNJCAT DS OCL6     JOB-CATID
AVSNJCTK DS C                CATID-PREFIX
AVSNJCTI DS CL4                CATID-VALUE
          DS CL1                CATID-SUFFIX
AVSNJCLA DS CL8      JOB-CLASS
AVSNJLGM DS CL8      JOB-LOG
AVSNJPAR DS CL128    JOB-PARAMETER
AVSNJFIL DS CL54     JOB-ENTER-FILE-NAME
*-----*
* FUNCTION=S ,TYPE=NET                                     *
          ORG AVSNVANF
AVSNSNLD DS CL8      DATE SUBNET-START (YYYYMMDD)
          DS CL6
AVSNSUSE DS CL8      SUBNET-USER
AVSNSACC DS CL8      SUBNET-ACCOUNT
AVSNSSRV DS OCL8     SUBNET-SERVER
AVSNSCAT DS OCL6     SUBNET-CATID
AVSNSCTK DS C                CATID-PREFIX
AVSNSCTI DS CL4                CATID-VALUE
          DS CL1                CATID-SUFFIX
AVSNSCLA DS CL8      SUBNET-CLASS
AVSNSLGM DS CL8      SUBNET-LOG
AVSNSPAR DS CL128    SUBNET-PARAMETER
*-----*
* FUNCTION=F ,TYPE=TRA                                     *
          ORG AVSNVANF
AVSNFNLD DS CL8      DATE FT-START (YYYYMMDD)
          DS CL6
AVSNFTID DS CL10     FT-TRANSFER-ID
AVSNCFCR DS X        CONDITION FT-JOB CREATION STATUS
AVSNFDIR DS X        FT-DIRECTION
AVSNFREM DS X        FT-REMOTE
AVSNFRFA DS CL67     FT-REMOTE-TRANSFER-ADMISSION
AVSNFLFN DS CL54     FT-LOCAL-FILE-NAME
AVSNFRFN DS CL54     FT-REMOTE-FILE-NAME
AVSNFPAR DS CL192    FT-PARAMETER
*-----*

```

```

ORG   AVSNVANF+L'AVSNVANF
DS    OC

```

Explanation of the status displays

NSTT	X	<p>Current processing status of the structure element if it is being processing normally.</p> <p>The status of structure elements which do not get processed (NO-PLAN, NO-SUBMIT, DELETED and IGNORED) is stored in NST3. If HOLD is set in NSTT, the processing status is saved in NSTTS. The meaning of the keywords is described under the dialog functions (SHOW-NET-STATUS, etc.).</p>
		<pre> X'0B' CREATED X'14' ABENDED X'15' ENDED X'16' ERROR X'17' HOLD X'18' RUNNING X'19' WAITING X'3E' SKIPPED X'42' EXECUTED X'9B' OCCURRED X'9C' NO-OCCURE X'AD' HOSTWAIT </pre>
NST2	X	<p>Requested action of a dialog function (also batch and program interface). The action is implemented by the run control system when the structure element gets processed.</p>
		<pre> X'00' LOW No action has been requested. X'17' HOLD X'1C' RESUMED </pre>
NST3	X	<p>Status of structure elements which do not get processed.</p>
		<pre> X'00' LOW The status is not set. X'3F' IGNORED X'7C' NO-PLAN X'7D' DELETED X'85' NO-SUBMIT </pre>
NSTTS	X	<p>Saving of the current processing status NSTT if net processing is interrupted (NSTT HOLD or NSTT RESUMED).</p>
		<pre> X'00' LOW No status has been saved. X'16' ERROR X'19' WAITING X'9C' NO-OCCURE X'AD' HOSTWAIT </pre>

Example

INDEX	NSTT	NST2	NST3	NSTTS	
010	X'9B'	X'00'	X'00'	X'00'	OCCURRED (FU= C / W)
020	X'15'	X'00'	X'00'	X'00'	ENDED (FU= J / P)
020	X'42'	X'00'	X'00'	X'00'	EXECUTED (FU= A / M / D)
030	X'15'	X'00'	X'7C'	X'00'	NO-PLAN (processed)
040	X'17'	X'00'	X'00'	X'19'	HOLD (on WAITING)
050	X'19'	X'00'	X'85'	X'00'	NO-SUBMIT (not processed)
060	X'19'	X'17'	X'7D'	X'00'	DELETED; HOLD requested (not yet processed)

The indices 010, 020 and 030 were processed by the run control system.

The structure element at index level 030 was not planned for processing.

The net processing was interrupted by HOLD-NET at index 040.

The structure element at index level 050 was not released at SUBMIT-NET; it has not yet been processed by the run control system because HOLD is set at index level 040.

The structure element at index level 060 was deleted with MODIFY-SUBMIT-NET; it has not yet been processed by the run control system because HOLD is set at index level 040. If index 060 is processed, HOLD is set.

AVSASSBC – Define communication area

Macro	Operands
AVSASSBC	MF = <u>C</u> / D ,PREFIX = <u>A</u> / <char (1)> ,MACID = <u>VSR</u> / <char (3)>

For descriptions of the MF and PREFIX operands see AVSASSAD on [page 415](#).

MACID The MACID operand is used to generate the names which have to be created. This three-character string specifies the second to fourth character of the name. Depending on the type, the fourth character will be set to R, C or Q. The default value ensures the uniqueness of names within the component group.

Expansion of the AVSASSBC macro

```

AVSASSBC
MFCHK MF=C,
        PREFIX=A,
        MACID=VSR,
        DMACID=VSR,
        SUPPORT=(C,D)
DS      OF
        *,##### PREFIX=A, MACID=VSR #####
*-----*
* AVAS BATCH-PROGRAM-INTERFACE      COMMUNICATION AREA
*-----*
*----- RETURN-AREA -----*
AVSRAREA DS      OCL192
AVSRCMDT DS      CL24      CMD-TEXT
*
AVSRFDBK DS      OAL2      RETURN CODE
*----- FEEDBACK-CODE1 -----*
AVSRFDB1 DS      AL1      FEEDBACK 1
AVSQ10K EQU      X'00'    . NO ERROR
AVSQ1FCT EQU      X'04'    . ERROR: FUNCTION
AVSQ1OUT EQU      X'08'    . ERROR: OUT-AREA
AVSQ1WRK EQU      X'0C'    . ERROR: WRK-AREA
AVSQ1SYS EQU      X'12'    . ERROR: AVAS-SYSTEM
*----- FEEDBACK-CODE2 -----*
AVSRFDB2 DS      AL1      FEEDBACK 2
*      EQUATES ARE VALID IF OK IS SET IN FEEDBACK 1
AVSQ2NOK EQU      X'04'    . FUNCTION WITH WARNING/RESUL
*      EQUATES ARE VALID IF FCT IS SET IN FEEDBACK 1
AVSQ20K EQU      X'00'    . NO ERROR

```

```

AVSQ2CMD EQU  X'01'      . ERROR: CMD
AVSQ2OPR EQU  X'02'      . ERROR: OPR
AVSQ2PRM EQU  X'03'      . ERROR: PARAMS
AVSQ2FCT EQU  X'04'      . ERROR: FUNCTION
AVSQ2TMO EQU  X'05'      . ERROR: TOO MANY OPERANDS
*          EQUATES ARE VALID IF OUT IS SET IN FEEDBACK 1
*          EQUATES ARE VALID IF WRK IS SET IN FEEDBACK 1
*          EQUATES ARE VALID IF SYS IS SET IN FEEDBACK 1
AVSQ2ERR EQU  X'01'      . ERROR
AVSQ2DMS EQU  X'02'      . ERROR: DMS
*-----*
          DS      AL2      RESERVED
          DS      CL4      RESERVED
AVSRNECT DS      H        COUNTER NET/ELEMENT FOUND
AVSRNEOK DS      H        .          .          PROCESSED
AVSRNEER DS      H        .          .          INVALID
AVSRRLTC DS      X        RESULT-CODE: NET/ELEMENT OK
          DS      X        RESERVED
AVSRDMSN DS      CL4      DMS MESSAGE NUMBER
AVSRMSGN DS      CL4      AVAS MESSAGE NUMBER
AVSRMSG1 DS      CL30     .          .          INSERT 1
AVSRMSG2 DS      CL30     .          .          INSERT 2
AVSRMSG3 DS      CL30     .          .          INSERT 3
          DS      CL30     RESERVED
*          RESULT FIELD: OPERAND ERROR
AVSRNETN DS      X        .          NET-NAME
AVSRRC SN DS      X        .          RUN-CONTROL-SYSTEM
AVSRPERN DS      X        .          PERIOD-NAME
AVRSYDN  DS      X        .          SYMDAT-NAME
AVSREAS  DS      X        .          EARLIEST-/NEW-PLAN-START
AVSRUSPF DS      X        .          USER-PAR-FILE
AVSRERJN DS      X        .          ERROR-JOBNAME
AVSRERIN DS      X        .          ERROR-INDEX
AVSRRSVT DS      X        .          RESTART-VARIANT
AVSRUSLB DS      X        .          AVAS-USER-LIBRARY
AVSRNST1 DS      X        .          NET-STATUS
AVSRC DNT DS      X        .          CONDITION-TYPE
AVSRC DNM DS      X        .          CONDITION-NAME
AVSRC DNN DS      X        .          CONDITION CREATED BY NET
AVSRC DVL DS      X        .          CONDITION-VALUE
AVSRLTI  DS      X        .          LIFE-TIME
          DS      CL7      RESERVED
*-----* COMMAND-AREA -----*
AVSCAREA DS      OC
AVSCMDC  DS      X        AVAS-CMD-CODE
*
AVSQSIGN EQU      11      SIGNON
AVSQEND  EQU      19      END
    
```

```

AVSQRDAL EQU 51      READ-AVAS-LIBRARY
AVSQCPLN EQU 61      CREATE-PLAN-NET
AVSQCPRN EQU 68      CREATE-PROD-NET
AVQSUNE  EQU 71      SUBMIT-NET
AVSQRENE EQU 77      RESTART-NET
AVSQMCCD EQU 88      MODIFY-COND-DESCRIPTION
*-----*
          DS      CL3
*
          PARAMETER-AREA 1
AVSCPAR1 DS      0CL92
AVSCAUID DS      CL8      AVAS-USER-ID
AVSCAUPW DS      CL8      AVAS-USER-PASSWORD
AVSCASID DS      CL7      AVAS-SYSTEM-ID
          DS      CL69
          ORG     AVSCPAR1
*
AVSCNETN DS      CL32     NET-NAME
AVSCRCNS DS      CL8      RUN-CONTROL-SYSTEM
AVSCPERN DS      CL37     PERIOD-NAME
          DS      CL15
*
          PARAMETER-AREA 2
AVSCPAR2 DS      0CL96
AVSCSYDN DS      CL20     SYMDAT-NAME
AVSCEAS  DS      CL19     EARLIEST-START / NEW-PLAN-STAR
AVSCCALN DS      CL20     CALENDAR-NAME
          DS      CL37
          ORG     AVSCPAR2
*
AVSCUSPF DS      CL54     USER-PAR-FILE
          DS      CL42
          ORG     AVSCPAR2
*
AVSCERJN DS      CL30     ERROR-JOBNAME
AVSCERIN DS      CL3      ERROR-INDEX
AVSCRSVT DS      CL1      RESTART-VARIANT
          DS      CL62
          ORG     AVSCPAR2
*
AVSCUSLB DS      CL6      AVAS-USER-LIBRARY
AVSCNST1 DS      CL12     NET-STATUS
AVSCCDNT DS      CL3      CONDITION-TYPE
AVSCCDNM DS      CL30     CONDITION-NAME
AVSCCDNN DS      CL32     CONDITION CREATED BY NET-NAME
AVSCCDNI DS      CL3      CONDITION CREATED BY INDEX
          DS      CL10

```

```
*                PARAMETER-AREA 3
AVSCPAR3 DS      0CL96
AVSCLTI  DS      CL19      LIFE-TIME
AVSCCDVL DS      CL74      CONDITION-VALUE
                DS      CL3
*
AVSR#QCL EQU    *-AVSRAREA      LNG COMMUNICATION AREA
*-----*
```

AVSASSBO – Define result area header

Macro	Operands
AVSASSBO	MF = <u>C</u> / D ,PREFIX = <u>A</u> / <char (1)> ,MACID = <u>VSO</u> / <char (3)>

For descriptions of the MF and PREFIX operands see AVSASSAD on [page 415](#).

MACID The MACID operand is used to generate the names which have to be created. This three-character string specifies the second to fourth character of the name.
Default value: VSO

Expansion of the AVSASSBO macro

```

AVSASSBO
MFCHK MF=C,
      PREFIX=A,
      MACID=VSO,
      DMACID=VSO,
      SUPPORT=(C,D)
DS    OF
      *,##### PREFIX=A, MACID=VSO #####
*-----
* AVAS BATCH-PROGRAM-INTERFACE      OUTPUT-AREA
*-----
AVSOAREA DS    OC
AVSONECT DS    H          COUNTER OF MAX. TAB-OUTPUT
          DS    CL10      RESERVED
AVSOFILE DS    CL54      FILENAME OF TAB-OUTPUT
          DS    CL14      RESERVED
AVSOTAB  DS    OC

```

AVSASSBW – Define work area

Macro	Operands
AVSASSBW	MF = <u>C</u> / D ,PREFIX = <u>A</u> / <char (1)> ,MACID = <u>VSW</u> / <char (3)> ,COUNT = <u>0</u> / <integer 0..999> ,MSGTAB = <u>YES</u> / NO / ONLY

For descriptions of the MF and PREFIX operands see AVSASSAD on [page 415](#).

MACID The MACID operand is used to generate the names which have to be created. This three-character string specifies the second to fourth character of the name.

Default value: VSW

COUNT Specifies the size of a table. This is the area where messages issued during the processing of a statement are stored in tabular form.

<integer 0..999>

Number of segments in the table. COUNT=0 indicates that no area is to be defined (reserved).

MSGTAB Controls the scope of the macro expansion.

YES The table header, the area for table entries and the specification of the table entries are generated.

NO Only the table header and the area for table entries are generated.

ONLY Only a table entry is generated.

Expansion of the AVSASSBW macro

```

AVSASSBW
MFCHK MF=C,
      PREFIX=A,
      MACID=VSW,
      DMACID=VSW,
      SUPPORT=(C,D)
DS    OF
      *,##### PREFIX=A, MACID=VSW #####
*-----*
* AVAS BATCH-PROGRAM-INTERFACE          WORK-AREA
*-----*
*----- WRK - AREA -----*
AVSWAREA DS    OC
AVSWGCT  DC    H'0'          COUNTER MAX. MESSAGE-TAB
AVSWGCR  DS    H            RETURN MESSAGE-TAB COUNTER
AVSWFILE DS    CL54         FILENAME (MESSAGE OUTPUT FILE)
          DS    CL22         RESERVED
AVSWMTAB DS    OCL136
AVSW#QWL EQU   *-AVSWAREA   LNG WORK AREA
          ORG   AVSWMTAB
*----- MSG-TAB -----*
AVSWMPAR DS    OCL136
          DS    H            RESERVED
          DS    X            RESERVED
AVSWMSG  DS    C            CONTINUATION LINE
AVSWQNTL EQU   C'+ '        . NEXT LINE (AVAS-MESSAGE)
AVSWMSG  DS    OCL7         AVAS-MESSAGE
AVSWMSGK DS    CL3          -KEY
AVSWMSGN DS    CL4          -NUMBER
AVSWMSGT DS    CL125        AVAS-MESSAGE-TEXT
*-----*

```


AVSASSCD – Define directory entry for condition descriptions in run control file

Macro	Operands
AVSASSCD	MF = <u>C</u> / D ,PREFIX = <u>A</u> / <char (1)> ,MACID = <u>VSI</u> / <char (3)> ,EQU = <u>NO</u> / YES

For descriptions of the MF and PREFIX operands see AVSASSAD on [page 415](#).

MACID The MACID operand is used to generate the names which have to be created. This three-character string specifies the second to fourth character of the name.

Default value: VSI

EQU Controls the generation of equates for the key fields.

NO No equates are generated.

 YES Equates are generated.

Note

The AVSASSCD macro generates an extension of the output area. Consequently it must be called after the AVSASSBO macro.

Expansion of the AVSASSCD macro

```

AVSASSCD
MFCHK MF=C,
      PREFIX=A,
      MACID=VSI,
      DMACID=VSI,
      SUPPORT=(C,D)
DS    OF
      *,##### PREFIX=A, MACID=VSI #####
*-----*
*      ABLDAT: CONDITION DIRECTORY DESCRIPTION
AVSICDIR DS    OCL96      CONDITION DIRECTORY
AVSIFCNM DS    OCL68      FULL-CONDITION-DESCRIPTION-NAME
AVSICTYS DS    CL3        CONDITION-TYPE-STRING  NET/JOB/RES/VAL
AVSICNAM DS    CL30       BK_CONDITIONNAME
AVSICNM DS    CL32       CREATED BY BK_NET-NAME
AVSICIND DS    CL3        CREATED BY INDEX
AVSICTYP DS    X          COND-TYPE-CODE
AVSICST1 DS    X          STATE 1
AVSICLTI DS    OCL12     LIFE-TIME
AVSICLTD DS    CL6       LIFE-DATE YYMMTT
AVSICLTT DS    CL6       LIFE-TIME HHMMSS
*
*      TYPE NET LIFE-TIME NET + NET-PLAN
*      TYPE JOB LIFE-TIME JOB + NET-PLAN
      DS    CL1          RESERVED
AVSICMUS DS    X          MAX-USING-SHARE
AVSICUWT DS    H          WAITING NET COUNT
AVSICUUS DS    H          USING NET COUNT
      DS    CL8          RESERVED
*-----*
      DS    OC

```

AVSASSCE – Define condition description in run control file

Macro	Operands
AVSASSCE	MF = <u>C</u> / D ,PREFIX = <u>A</u> / <char (1)> ,MACID = <u>VSB</u> / <char (3)> ,EQU = <u>NO</u> / YES

For descriptions of the MF and PREFIX operands see AVSASSAD on [page 415](#).

MACID The MACID operand is used to generate the names which have to be created. This three-character string specifies the second to fourth character of the name.

Default value: VSB

EQU Controls the generation of equates for the key fields.

NO No equates are generated.

YES Equates are generated for the keywords.

Note

The AVSASSCE macro generates an extension of the output area. Consequently it must be called after the AVSASSBO macro.

Expansion of the AVSASSCE macro

```

AVSASSCE
MFCHK MF=C,
      PREFIX=A,
      MACID=VSB,
      DMACID=VSB,
      SUPPORT=(C,D)
DS    OF
      *,##### PREFIX=A, MACID=VSB #####
*-----*
*      ABLDAT:  CONDITION DESCRIPTION
AVSBACON DS    0CL192
DS      CL2    RESERVED
AVSBCCR  DS    0CL12  CREATION-DATE
AVSBCCRD DS    CL6    CREATION-DATE-DATE  YYMMDD
AVSBCCRT DS    CL6    CREATION-DATE-TIME  HHMMSS
AVSBCTXT DS    CL120  CONDITION-TEXT
AVSBCDOC DS    CL43   DOCUMENT-ELEMENTNAME
AVSBCLU  DS    0CL12  LAST UPDATE
AVSBCLUD DS    CL6    LAST UPDATE DATE  YYMMDD
AVSBCLUT DS    CL6    LAST UPDATE TIME  HHMMSS
DS      CL2    RESERVED
ORG     AVSBACON
AVSBCVAL DS    XL128  VALUE X-STRING BY TYP=VAL
DS      CL64
ORG     AVSBACON
AVSBNNAM DS    CL32   BK_NETNAME_PLANSTARTDATE_PLANSTARTTIME
AVSBCIND DS    CL3    CONDITION-INDEX
AVSBCWUD DS    CL6    WAITING OR USING DATE  YYMMDD
AVSBCWUT DS    CL6    WAITING OR USING TIME  HHMMSS
AVSBSTUS DS    X      STATE WAITING / USING / OCCURRED / FREE
AVSBENWT EQU  25     NET IS WAITING      NET / JOB / VAL /
AVSBENCO EQU  155    CONDITION OCCURRED  NET / JOB / VAL
AVSBENUF EQU  117    NET IS NOT USING RESOURCE
AVSBENUS EQU  118    NET IS USING RESOURCE SHARE
AVSBENUE EQU  119    NET IS USING RESOURCE EXCLUSIV
AVSBCOCO DS    12XL1  WAIT FOR CONDITION STATE      NET/JOB/
AVSBCOST DS    CL128  WAIT FOR CONDITION VALUE
DS      CL2    RESERVED
*-----*
DS      OC

```

AVSASSJD – Define journal directory

Macro	Operands
AVSASSJD	MF = <u>C</u> / D ,PREFIX = <u>A</u> / <char (1)> ,MACID = <u>VSJ</u> / <char (3)> ,EQU = <u>NO</u> / YES

For descriptions of the MF and PREFIX operands see AVSASSAD on [page 415](#).

MACID The MACID operand is used to generate the names which have to be created. This three-character string specifies the second to fourth character of the name.

Default value: VSJ

EQU Controls the generation of equates for the key fields.

NO No equates are generated.

YES Equates are generated for the keywords.

Note

The AVSASSJD macro generates an extension of the output area. Consequently it must be called after the AVSASSBO macro.

Expansion of the AVSASSJD macro

```

AVSASSJD
MFCHK MF=C,
      PREFIX=A,
      MACID=VSJ,
      DMACID=VSJ,
      SUPPORT=(C,D)
DS    OF
      *,##### PREFIX=A, MACID=VSJ #####
*-----*
*          JRNDAT: NET DIRECTORY DESCRIPTION
AVSJDIR DS    0CL64  JOURNAL DIRECTORY DESCRIPTION
AVSJNETN DS    CL32  BK_NETNAME_PLANSTARTDATE_PLANSTARTTIME
AVSJNST1 DS    X    - STATE 1 - NET-SUBMITTED
AVSJNST2 DS    X    - STATE 2 - NET-STATE
*
AVSJNST3 DS    X    - STATE 3 - OUTPUT-PAVED
AVSJNST4 DS    X    - STATE 4 - OUTPUT-ERROR
*-----*
          DS    CL28  RESERVED
*-----*
          DS    OC

```

Explanation of the status displays

NST1	X	Status of the net release (SUBMIT-NET)	
X'00'	LOW	The net has not yet been released. The current status shows NST2.	
X'0E'	SUBMITTED	The net was released. The current status shows NST2.	
X'AE'	NETWAIT	The net has been released and runs as a subnet under the control of the hypernet. The current status shows NST2.	
X'19'	WAITING	The net has been released as a subnet and does not run under the control of the hypernet. The current status shows NST2.	
X'3D'	OPWAIT	The net has been released as a subnet und does not run under the control of the hypernet. The current status shows NST2.	

NST2	X	Current processing status of the net in the journal file	
			The meaning of the keywords is described under the dialog functions (see SHOW-PLAN-NET, SHOW-NET-STATUS, etc.).
	X'09'	TOCREATE	
	X'0A'	PARTIALLY	
	X'0B'	CREATED	
	X'0C'	NOTTOCREATE	
	X'0E'	SUBMITTED	
	X'14'	ABENDED	
	X'15'	ENDED	
	X'16'	ERROR	
	X'17'	HOLD	
	X'18'	RUNNING	
	X'19'	WAITING	
	X'1A'	CONDWAIT	
	X'1B'	RESTARTED	
	X'1C'	RESUMED	
	X'3F'	IGNORED	
	X'7C'	NOT-PLANNED	
	X'7D'	DELETED	
	X'7E'	NOT-DELETED	
	X'81'	UPDATED	
	X'9A'	SHIFTED	
	X'AD'	HOSTWAIT	
NST3	X	Status of the journal save	
	X'00'	LOW	The journal records have not yet been saved.
	X'7F'	SAVED	The journal records were saved during the reorganization.
NST4	X	OUTPUT-ERROR status	
	X'00'	LOW	All output journal records are in the journal file.
	X'16'	ERROR	Journal records of this net were output to the emergency journal file due to an error.

AVSASSJN – Define journal record header

Macro	Operands
AVSASSJN	MF = <u>C</u> / D ,PREFIX = <u>A</u> / <char (1)> ,MACID = <u>VSP</u> / <char (3)> ,EQU = <u>NO</u> / YES

For descriptions of the MF and PREFIX operands see AVSASSAD on [page 415](#).

MACID The MACID operand is used to generate the names which have to be created. This three-character string specifies the second to fourth character of the name.

Default value: VSP

EQU Controls the generation of equates for the key fields.

NO No equates are generated.

 YES Equates are generated for the keywords.

Notes

- The AVSASSJN macro generates an extension of the output area. Consequently it must be called after the AVSASSBO macro.
- The key-dependent parts of the journal records are generated by the AVASJRN macro. This must be called immediately after AVSASSJN. This macro is described in [section “Data structures of the backup journal and the emergency journal file” on page 194](#).

Expansion of the AVSASSJN macro

```

AVSASSJN
MFCHK MF=C,
        PREFIX=A,
        MACID=VSP,
        DMACID=VSP,
        SUPPORT=(C,D)
DS      OF
        *,##### PREFIX=A, MACID=VSP #####
*-----*
*          JRNDAT: NET JOURNAL DESCRIPTION
AVSPJNET DS      0CL384
AVSPULEN DS      H          - USED LENGTH
          DS      H          - RESERVED
AVSPDATE DS      CL6        - OUTPUT-DATE
AVSPTIME DS      CL6        - OUTPUT-TIME
AVSPNUMS DS      XL2        - NUMBER AFTER TIME
AVSPACMD DS      X          - AVAS-COMMAND
*
AVSPAKTN DS      X          - EXECUTED ACTION
*
AVSPUSER DS      CL8        - AVAS-USER
AVSPINDX DS      CL3        - INDEX
AVSPFUNC DS      C          - FUNCTION
*
AVSPNAME DS      CL32       - NAME OF ELEMENT
AVSPRECA DS      0CL320     - RECORD-AREA WITH KEY MAX-L=320
AVSPRSSL DS      CL2        - RECORD-KEY
AVSPRFNR DS      CL2        - RECORD-FUNCTION-KEY
AVSPOUTD DS      CL316     - OUTPUT-DATA  MAX-L=316
*-----*
*          FOR OUTPUT-DATA - DSECT  USE MACRO  AVASJRN
*          ORG      &P.OUTD
*  SSL      AVASJRN  &PRE
*-----*
*-----*
DS      OC
    
```

AVSASSRT – Define result area

Macro	Operands
AVSASSRT	MF = <u>C</u> / D ,PREFIX = <u>A</u> / <char (1)> ,MACID = <u>VSE</u> / <char (3)>

For descriptions of the MF and PREFIX operands see AVSASSAD on [page 415](#).

MACID The MACID operand is used to generate the names which have to be created. This three-character string specifies the second to fourth characters of the name.
 Default value: VSE

Expansion of the AVSASSRT macro

```

AVSASSRT
MFCHK MF=C,
        PREFIX=A,
        MACID=VSE,
        DMACID=VSE,
        SUPPORT=(C,D)
DS      OF
        *,##### PREFIX=A, MACID=VSE #####
*-----*
*          RESULT-DESCRIPTION
AVSERSLT DS      OCL96
AVSEOPAR DS      CL64
          ORG     AVSEOPAR
AVSENETN DS      CL32          NET-NAME
AVSEEAS  DS      OCL12          EARLIEST-START / NEW-PLAN-START
AVSEEASD DS      CL6          DATE
AVSEEAST DS      CL6          TIME
AVSESYND DS      CL20          SYMDAT-NAME
          DS      CL3          RESERVED
          ORG     AVSEOPAR
          DS      CL32
AVSERSVT DS      C          RESTART-VARIANT
          DS      CL31
          ORG     AVSEOPAR
AVSENAME DS      CL43          ELEMENT-NAME
          DS      CL21
          ORG     AVSEOPAR
AVSECDNM DS      CL30          CONDITION-NAME
          DS      CL34
*-----*
```

```

                DS      CL3          RESERVED
*-----*
AVSERLT DS      0CL13
AVSERLTC DS      X          RESULT-CODE
AVSQPART EQU    10          . PARTIALLY
AVSQCREA EQU    11          . CREATED
AVSQSUBM EQU    14          . SUBMITTED
AVSQERR EQU     22          . ERROR
AVSQREST EQU    27          . RESTARTED
AVSQPLAN EQU   123          . PLANNED
AVSQNOPL EQU   124          . NO PLAN
AVSQSAVE EQU   127          . SAVED
AVSQUPDT EQU   129          . UPDATED
AVSQNOUP EQU   130          . NO UPDATE
AVSQNOSM EQU   133          . NO SUBMIT
AVSQNOFO EQU   138          . NOT FOUND
AVSQLCKD EQU   139          . LOCKED
AVSERLTT DS      CL12        RESULT-TEXT
*-----*
                DS      CL16        RESERVED
*-----*
                DS      0C

```

10.4 Cobol Interface

1. Calling the Cobol interface

The Cobol call is performed in the user program:

```
CALL AVASBCOB USING AVASBCOM,   (Communication area)
                    AVASBOUT,   (Result area)
                    AVASBWRK.   (Work area)
```

2. Program compilation

In order to compile a Cobol program with AVAS parameter structures, the library with the COPY elements must be assigned. This assignment can be made with the following BS2000 command:

```
/ADD-FILE-LINK LINK-NAME=COBLIB,FILE-NAME=SYSSRC.AVAS.085
```

3. Program linkage

The AVAS interface module AVSBCALL from the library SYSLNK.AVAS.085 must be permanently linked to the main program.

4. Starting the program

When the first call is made from the user program to AVAS, the AVAS interface module loads additional modules. This load call specifies SYSLNK.AVAS.085 as the library name.

If the AVAS modules are not available in the library with this name, or if the AVAS library is held on another pubset, then before the program is started the following BS2000 command must be used to make an assignment:

```
/ADD-FILE-LINK LINK-NAME=SYSLNK,-
/                FILE-NAME=:<catid>:$<userid>.SYSLNK.AVAS.085
```

5. Interface areas

The areas for the interfaces are specified by COPY elements:

AVSCOBQB	Key values
AVSCOBBC	Communication area
AVSCOBBO	Result area
AVSCOBRT	Function-specific result area
AVSCOB BW	Work area
AVSCOB JD	Net directory entry in journal file
AVSCOB JN	Net entry for the journal

AVSCOBAD	Net directory entry in run control file
AVSCOBAN	Net description entry
AVSCOB CD	Event directory entry
AVSCOBCE	Event description

The COPY element fields have the same names as for the Assembler definitions. They are described in [section "Structure of the communication areas" on page 406](#).

10.4.1 Programming example in Cobol

This example shows how the statements SIGNON, SUBMIT-NET and END are handled from a programming point of view. Error handling measures are only outlined.

```

IDENTIFICATION DIVISION.
PROGRAM-ID. AVSCOBOL.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SPECIAL-NAMES.
    SYMBOLIC CHARACTERS
        COPY AVSCOBQ.

*      .
*      -----> POINT FOR END OF SPECIAL-NAMES.

DATA DIVISION.
WORKING-STORAGE SECTION.
77 ASSIGN-SIGNON      PICTURE X      VALUE AVSA-SIGNON.
77 ASSIGN-END        PICTURE X      VALUE AVSA-END.
77 ASSIGN-SUBMIT     PICTURE X      VALUE AVSA-SUBMIT-NET.
77 USER-ID           PICTURE X(08)  VALUE "TEST  ".
77 PASSWORD          PICTURE X(08)  VALUE "TEST  ".
77 SYSTEM-ID         PICTURE X(07)  VALUE "SYSIDAV".
77 NET-NAME          PICTURE X(32)  VALUE "TEST.NET_".
77 MAX-RESULT-COUNTER PICTURE S9(4)  COMPUTATIONAL VALUE 20.
    COPY AVSCOBBC.
    COPY AVSCOBBO.
    06 FILLER                OCCURS 20 TIMES.
    COPY AVSCOBRT.
    COPY AVSCOBWB                REPLACING COUNT BY 0.

PROCEDURE DIVISION.
* CALL AVAS                                FOR S I G N O N
    MOVE LOW-VALUE                    TO AVSCAREA.
    MOVE AVASA-SIGNON                  TO AVSCMDC.
    MOVE USER-ID                       TO AVSCAUD.
    MOVE PASSWORD                      TO AVSCAUPW.
    MOVE SYSTEM-ID                     TO AVSCASID.
    CALL "AVASBCOB" USING AVASBCOM AVASBOUT AVASBWRK.

```

```
        IF    AVSQOK                THEN  NEXT SENTENCE
        ELSE  GO TO ERROR.
* CALL AVAS                        FOR    S U B M I T - N E T
  MOVE  LOW-VALUE                  TO    AVSCAREA.
  MOVE  ASSIGN-SUBMIT              TO    AVSCMDC.
  MOVE  NET-NAME                   TO    AVSCNETN.
  MOVE  MAX-RESULT-COUNTER        TO    AVSONECT.
  CALL  "AVASBCOB" USING AVASBCOM AVASBOUT AVASBWRK.
  IF    AVSQOK                THEN  NEXT SENTENCE
  ELSE  GO TO ERROR.
ENDE.
* CALL AVAS                        FOR    E N D
  MOVE  ASSIGN-END                TO    AVSCMDC.
  CALL  "AVASBCOB" USING AVASBCOM AVASBOUT AVASBWRK.
  STOP RUN.
ERROR.
.
.
.
END PROGRAM  AVSCOBOL.
```

10.4.2 COPY elements for statements

AVSCOBAD – Define net directory for run control file

The COPY element AVSCOBAD defines an entry for the run control file net directory.

Call: COPY AVSCOBAD

Expansion of the COPY element AVSCOBAD

```

COPY AVSCOBAD.
* AVSCOBAD          004          110309  47105171 AVAS          U
*****
*
* COPYRIGHT (C) FUJITSU TECHNOLOGY SOLUTIONS 2011          *
*                   ALL RIGHTS RESERVED                    *
*
*****
* ABLDAT: NET DIRECTORY DESCRIPTION
  07 AVSDADIR.
    10 AVSDNETN          PICTURE X(32).
    10 AVSDNST1          PICTURE X.
    10 AVSDNST2          PICTURE X.
    10 AVSDNST3          PICTURE X.
    10 AVSDNERR          PICTURE X.
    10 AVSDNRST          PICTURE X.
    10 AVSDNCWT          PICTURE X.
    10 AVSDNHLD          PICTURE X.
    10 AVSDNES.
      15 AVSDNESD          PICTURE 9(06).
      15 AVSDNEST          PICTURE 9(06).
    10 AVSDNLS.
      15 AVSDNLSD          PICTURE 9(06).
      15 AVSDNLST          PICTURE 9(06).
    10 AVSDNTYP          PICTURE X.
    10 AVSDNDSL          PICTURE X.
    10 AVSDOPST          PICTURE X.
    10 AVSDNHTW          PICTURE X.
    10 FILLER            PICTURE X(29).
***                          END OF COPY ELEMENT AVSCOBAD          ***

```

AVSCOBAN – Define net structure data

The COPY element AVSCOBAN defines the description of a net structure element.

Call: COPY AVSCOBAN

Expansion of the COPY element AVSCOBAN

```

COPY AVSCOBAN.
* AVSCOBAN          009          110309  47105172 AVAS          U
*****
*
* COPYRIGHT (C) FUJITSU TECHNOLOGY SOLUTIONS 2011          *
*                   ALL RIGHTS RESERVED                    *
*                                                           *
*****
* ABLDAT: NET DESCRIPTION
  07 AVSNANET.
    10 AVSNCNAM          PICTURE X(32).
    10 FILLER REDEFINES AVSNCNAM.
       15 AVSNSNM          PICTURE X(32).
    10 FILLER REDEFINES AVSNCNAM.
       15 AVSNFNAM          PICTURE X(30).
       15 FILLER          PICTURE X(02).
    10 FILLER REDEFINES AVSNCNAM.
       15 AVSNJNAM          PICTURE X(30).
       15 FILLER          PICTURE X(02).
    10 AVSNIND          PICTURE 9(03).
    10 AVSNSIND          PICTURE 9(03).
    10 AVSNSYN          PICTURE X(03).
    10 AVSNSTT          PICTURE X.
    10 AVSNST2          PICTURE X.
    10 AVSNST3          PICTURE X.
    10 AVSNSTTS          PICTURE X.
    10 AVSNENDG          PICTURE X(02).
    10 AVSNFUNC          PICTURE X.
    10 AVSNTYPE          PICTURE X.
    10 AVSNRSRV          PICTURE X.
    10 AVSNHPVS          PICTURE X(04).
    10 AVSNSRVN          PICTURE X(08).
    10 FILLER REDEFINES AVSNSRVN.
       15 AVSNCATI          PICTURE X(04).
       15 FILLER          PICTURE X(04).
    10 AVSNRSI1          PICTURE X(03).
    10 AVSNRST1          PICTURE X.
    10 AVSNRSJ1          PICTURE X(30).
    10 AVSNRSI2          PICTURE X(03).
    10 AVSNRST2          PICTURE X.

```



```

10 AVSNRSJ2          PICTURE X(30).
10 AVSNRSI3          PICTURE X(03).
10 AVSNRST3          PICTURE X.
10 AVSNRSJ3          PICTURE X(30).
10 AVSNLTS.
15 AVSNLTS.
    17 AVSNLTS.
    17 AVSNLTS.
15 AVSNLSTT          PICTURE 9(06).
10 FILLER            REDEFINES AVSNLTS.
15 FILLER            PICTURE 9(02).
15 AVSNLST           PICTURE 9(12).
10 AVSNLTO           REDEFINES AVSNLTS.
15 AVSNLTO.
    17 AVSNLTOH        PICTURE 9(02).
    17 AVSNLOCD        PICTURE 9(06).
15 AVSNLOCT          PICTURE 9(06).
10 FILLER            REDEFINES AVSNLTS.
15 FILLER            PICTURE 9(02).
15 AVSNLOC           PICTURE 9(12).
10 AVSNDSO           PICTURE X.
10 FILLER            PICTURE X(13).
*-----*
* VARIABLE AREA ABOUT TYPE AND FUNKTION *
10 AVSNVANF          PICTURE X(420).
*-----*
* FUNCTION=C ,TYPE=JVA *
10 FILLER            REDEFINES AVSNVANF.
15 AVSNFCN           PICTURE X(54).
15 AVSNVPOS          PICTURE 9(03).
15 AVSNVLEN          PICTURE 9(03).
15 AVSNCVAL          PICTURE X(256).
*-----*
* FUNCTION=C ,TYPE=NET *
10 FILLER            REDEFINES AVSNVANF.
15 AVSNCNOC          PICTURE X OCCURS 12 TIMES.
15 AVSNCNEC          PICTURE X OCCURS 12 TIMES.
15 AVSNCNCN          PICTURE X(32).
*-----*
* FUNCTION=C ,TYPE=JOB *
10 FILLER            REDEFINES AVSNVANF.
15 AVSNCJOC          PICTURE X OCCURS 12 TIMES.
15 AVSNCJEC          PICTURE X OCCURS 12 TIMES.
15 AVSNCJCN          PICTURE X(32).
15 AVSNCJCI          PICTURE 9(03).
*-----*
* FUNCTION=C ,TYPE=RES *
*-----*

```

```

          10  FILLER      REDEFINES      AVSNVANF.
              15  AVSNCROC      PICTURE X      OCCURS 12 TIMES.
              15  AVSNCREC      PICTURE X      OCCURS 12 TIMES.
*-----*
* FUNCTION=C ,TYPE=VAL *
          10  FILLER      REDEFINES      AVSNVANF.
              15  AVSNCVOS      PICTURE X(128).
              15  AVSNCVES      PICTURE X(128).
*-----*
* FUNCTION=W ,TYPE=TIM *
*-----*
* FUNCTION=A ,TYPE=RES *
          10  FILLER      REDEFINES      AVSNVANF.
              15  AVSNARVC      PICTURE X.
*-----*
* FUNCTION=A ,TYPE=VAL *
          10  FILLER      REDEFINES      AVSNVANF.
              15  AVSNAVVA      PICTURE X(128).
*-----*
* FUNCTION=M ,TYPE=RES *
          10  FILLER      REDEFINES      AVSNVANF.
              15  AVSNMRVC      PICTURE X.
*-----*
* FUNCTION=M ,TYPE=VAL *
          10  FILLER      REDEFINES      AVSNVANF.
              15  AVSNMVVA      PICTURE X(128).
*-----*
* FUNCTION=D ,TYPE=RES *
*                ,TYPE=VAL *
*-----*
* FUNCTION=D ,TYPE=NET *
          10  FILLER      REDEFINES      AVSNVANF.
              15  AVSNDNCN      PICTURE X(32).
*-----*
* FUNCTION=D ,TYPE=JOB *
          10  FILLER      REDEFINES      AVSNVANF.
              15  AVSNDJCN      PICTURE X(32).
              15  AVSNDJCI      PICTURE X(03).
*-----*
* FUNCTION=J ,TYPE=STD / MOD / EXT *
* FUNCTION=P ,TYPE=STD / MOD / EXT / EXX *
* FUNCTION=X ,TYPE=STD / MOD / EXT *
          10  FILLER      REDEFINES      AVSNVANF.
              15  AVSNCJLD      PICTURE X(08).
              15  AVSNTSN      PICTURE X(04).
              15  AVSNCJCR      PICTURE X.
              15  AVSNJENT      PICTURE X.
              15  AVSNJUSE      PICTURE X(08).

```

```

15 AVSNJACC      PICTURE X(08).
15 AVSNJSRV     PICTURE X(08).
15 AVSNJCAT     REDEFINES  AVSNJSRV.
    17 AVSNJCTK  PICTURE X.
    17 AVSNJCTI  PICTURE X(04).
    17 FILLER    PICTURE X.
    17 FILLER    PICTURE X(02).
15 AVSNJCLA     PICTURE X(08).
15 AVSNJLGM     PICTURE X(08).
15 AVSNJPAR     PICTURE X(128).
15 AVSNJFIL     PICTURE X(54).
*-----*
* FUNCTION=S ,TYPE=NET *
    10 FILLER    REDEFINES  AVSNVANF.
    15 AVSNSNLD  PICTURE X(08).
    15 FILLER    PICTURE X(06).
    15 AVSNSUSE  PICTURE X(08).
    15 AVSNSACC  PICTURE X(08).
    15 AVSNSSRV  PICTURE X(08).
    15 AVSNSCAT  REDEFINES  AVSNSSRV.
        17 AVSNSCTK  PICTURE X.
        17 AVSNSCTI  PICTURE X(04).
        17 FILLER    PICTURE X.
        17 FILLER    PICTURE X(02).
    15 AVSNSCLA  PICTURE X(08).
    15 AVSNSLGM  PICTURE X(08).
    15 AVSNSPAR  PICTURE X(128).
*-----*
* FUNCTION=F ,TYPE=TRA *
    10 FILLER    REDEFINES  AVSNVANF.
    15 AVSNFTLD  PICTURE X(08).
    15 FILLER    PICTURE X(06).
    15 AVSNFTID  PICTURE X(10).
    15 AVSNCFCR  PICTURE X.
    15 AVSNFDIR  PICTURE X.
    15 AVSNFREM  PICTURE X.
    15 AVSNFRFA  PICTURE X(67).
    15 AVSNFPTN  PICTURE X(08).
    15 AVSNFLFN  PICTURE X(54).
    15 AVSNFRFN  PICTURE X(54).
    15 AVSNFPAR  PICTURE X(192).
***      END OF COPY ELEMENT AVSCOBAN      ***

```

AVSCOBBC – Define communication area

The COPY element AVSCOBBC defines the communication area for all batch functions.

Call: COPY AVSCOBBC

Expansion of the COPY element AVSCOBBC

```

COPY AVSCOBAN.
* AVSCOBAN          009          110309  47105172 AVAS          U
*****
*
* COPYRIGHT (C) FUJITSU TECHNOLOGY SOLUTIONS 2011          *
*                   ALL RIGHTS RESERVED                    *
*                                                           *
*****
* ABLDAT: NET DESCRIPTION
  07 AVSNANET.
    10 AVSNCNAM          PICTURE X(32).
    10 FILLER REDEFINES AVSNCNAM.
      15 AVSNSNM          PICTURE X(32).
    10 FILLER REDEFINES AVSNCNAM.
      15 AVSNFNAM          PICTURE X(30).
      15 FILLER          PICTURE X(02).
    10 FILLER REDEFINES AVSNCNAM.
      15 AVSNJNAM          PICTURE X(30).
      15 FILLER          PICTURE X(02).
    10 AVSNIND          PICTURE 9(03).
    10 AVSNSIND          PICTURE 9(03).
    10 AVSNSYN          PICTURE X(03).
    10 AVSNSTT          PICTURE X.
    10 AVSNST2          PICTURE X.
    10 AVSNST3          PICTURE X.
    10 AVSNSTTS          PICTURE X.
    10 AVSNENDG          PICTURE X(02).
    10 AVSNFUNC          PICTURE X.
    10 AVSNTYPE          PICTURE X.
    10 AVSNRSRV          PICTURE X.
    10 AVSNHPVS          PICTURE X(04).
    10 AVSNSRVN          PICTURE X(08).
    10 FILLER REDEFINES AVSNSRVN.
      15 AVSNCATI          PICTURE X(04).
      15 FILLER          PICTURE X(04).
    10 AVSNRSI1          PICTURE X(03).
    10 AVSNRST1          PICTURE X.
    10 AVSNRSJ1          PICTURE X(30).
    10 AVSNRSI2          PICTURE X(03).
    10 AVSNRST2          PICTURE X.

```

```

10 AVSNRSJ2          PICTURE X(30).
10 AVSNRSI3          PICTURE X(03).
10 AVSNRST3          PICTURE X.
10 AVSNRSJ3          PICTURE X(30).
10 AVSNLTS.
15 AVSNLTS.
    17 AVSNLTS.
    17 AVSNLST.
15 AVSNLST.
10 FILLER            REDEFINES AVSNLTS.
15 FILLER            PICTURE 9(02).
15 AVSNLST.
10 AVSNLTO.
15 AVSNLTO.
    17 AVSNLTO.
    17 AVSNLOCD.
15 AVSNLOCT.
10 FILLER            REDEFINES AVSNLTS.
15 FILLER            PICTURE 9(02).
15 AVSNLOC.
10 AVSNDSO.
10 FILLER            PICTURE X(13).
*-----*
* VARIABLE AREA ABOUT TYPE AND FUNKTION *
10 AVSNVANF          PICTURE X(420).
*-----*
* FUNCTION=C ,TYPE=JVA *
10 FILLER            REDEFINES AVSNVANF.
15 AVSNFCN.
15 AVSNVPOS.
15 AVSNVLEN.
15 AVSNCVAL.
*-----*
* FUNCTION=C ,TYPE=NET *
10 FILLER            REDEFINES AVSNVANF.
15 AVSNCNOC.
15 AVSNCNEC.
15 AVSNCNCN.
*-----*
* FUNCTION=C ,TYPE=JOB *
10 FILLER            REDEFINES AVSNVANF.
15 AVSNCJOC.
15 AVSNCJEC.
15 AVSNCJCN.
15 AVSNCJCI.
*-----*
* FUNCTION=C ,TYPE=RES *
*-----*

```

```

          10  FILLER      REDEFINES      AVSNVANF.
              15  AVSNCROC      PICTURE X      OCCURS 12 TIMES.
              15  AVSNCREC      PICTURE X      OCCURS 12 TIMES.
*-----*
* FUNCTION=C ,TYPE=VAL *
          10  FILLER      REDEFINES      AVSNVANF.
              15  AVSNCVOS      PICTURE X(128).
              15  AVSNCVES      PICTURE X(128).
*-----*
* FUNCTION=W ,TYPE=TIM *
*-----*
* FUNCTION=A ,TYPE=RES *
          10  FILLER      REDEFINES      AVSNVANF.
              15  AVSNARVC      PICTURE X.
*-----*
* FUNCTION=A ,TYPE=VAL *
          10  FILLER      REDEFINES      AVSNVANF.
              15  AVSNAVVA      PICTURE X(128).
*-----*
* FUNCTION=M ,TYPE=RES *
          10  FILLER      REDEFINES      AVSNVANF.
              15  AVSNMRVC      PICTURE X.
*-----*
* FUNCTION=M ,TYPE=VAL *
          10  FILLER      REDEFINES      AVSNVANF.
              15  AVSNMVVA      PICTURE X(128).
*-----*
* FUNCTION=D ,TYPE=RES *
* ,TYPE=VAL *
*-----*
* FUNCTION=D ,TYPE=NET *
          10  FILLER      REDEFINES      AVSNVANF.
              15  AVSNDNCN      PICTURE X(32).
*-----*
* FUNCTION=D ,TYPE=JOB *
          10  FILLER      REDEFINES      AVSNVANF.
              15  AVSNDJCN      PICTURE X(32).
              15  AVSNDJCI      PICTURE X(03).
*-----*
* FUNCTION=J ,TYPE=STD / MOD / EXT *
* FUNCTION=P ,TYPE=STD / MOD / EXT / EXX *
* FUNCTION=X ,TYPE=STD / MOD / EXT *
          10  FILLER      REDEFINES      AVSNVANF.
              15  AVSNCJLD      PICTURE X(08).
              15  AVSNTSN      PICTURE X(04).
              15  AVSNCJCR      PICTURE X.
              15  AVSNJENT      PICTURE X.
              15  AVSNJUSE      PICTURE X(08).

```

```

15 AVSNJACC      PICTURE X(08).
15 AVSNJSRV     PICTURE X(08).
15 AVSNJCAT     REDEFINES    AVSNJSRV.
    17 AVSNJCTK  PICTURE X.
    17 AVSNJCTI  PICTURE X(04).
    17 FILLER    PICTURE X.
    17 FILLER    PICTURE X(02).
15 AVSNJCLA     PICTURE X(08).
15 AVSNJLGM     PICTURE X(08).
15 AVSNJPAR     PICTURE X(128).
15 AVSNJFIL     PICTURE X(54).
*-----*
* FUNCTION=S    ,TYPE=NET                                     *
    10 FILLER    REDEFINES    AVSNVANF.
    15 AVSNSNLD  PICTURE X(08).
    15 FILLER    PICTURE X(06).
    15 AVSNSUSE  PICTURE X(08).
    15 AVSNSACC  PICTURE X(08).
    15 AVSNSSRV  PICTURE X(08).
    15 AVSNSCAT  REDEFINES    AVSNSSRV.
        17 AVSNSCTK  PICTURE X.
        17 AVSNSCTI  PICTURE X(04).
        17 FILLER    PICTURE X.
        17 FILLER    PICTURE X(02).
    15 AVSNSCLA  PICTURE X(08).
    15 AVSNSLGM  PICTURE X(08).
    15 AVSNSPAR  PICTURE X(128).
*-----*
* FUNCTION=F    ,TYPE=TRA                                     *
    10 FILLER    REDEFINES    AVSNVANF.
    15 AVSNFTLD  PICTURE X(08).
    15 FILLER    PICTURE X(06).
    15 AVSNFTID  PICTURE X(10).
    15 AVSNCFCR  PICTURE X.
    15 AVSNFDIR  PICTURE X.
    15 AVSNFREM  PICTURE X.
    15 AVSNFRFA  PICTURE X(67).
    15 AVSNFPTN  PICTURE X(08).
    15 AVSNFLFN  PICTURE X(54).
    15 AVSNFRFN  PICTURE X(54).
    15 AVSNFPAR  PICTURE X(192).
***          END OF COPY ELEMENT AVSCOBAN          ***

```

AVSCOBBO – Define result area header

The COPY element AVSCOBBO defines the header of the result area. The function-specific areas must follow adjacent to this area.

Call: COPY AVSCOBBO

Expansion of the COPY element AVSCOBBO

```

COPY AVSCOBBO.
  * AVSCOBBO          003          110309  47100687 AVAS          U
  * *****
  *
  * COPYRIGHT (C) FUJITSU TECHNOLOGY SOLUTIONS 2011          *
  *                   ALL RIGHTS RESERVED                    *
  * *****
  *-----*
  * AVAS BATCH-PROGRAM-INTERFACE          OUTPUT-AREA          *
  *-----*
01 AVASBOUT.
  03 AVSOAREA.
    05 AVSONECT          PICTURE S9(4) COMPUTATIONAL.
    05 FILLER          PICTURE X(10).
    05 AVSOFILE          PICTURE X(54).
    05 FILLER          PICTURE X(14).
  03 AVSOTAB.
***          END OF COPY ELEMENT AVSCOBBO          ***

```


AVSCOBQBQ – Define key for AVSCMDC and AVSERLTC

The COPY element defines the keys for the function and result codes of the interface.

Call: COPY AVSCOBQBQ

Example, with expansion of the COPY element AVSCOBQBQ

```

SPECIAL-NAMES.
  TERMINAL IS USER-INOUT
  SYMBOLIC CHARACTERS
    COPY AVSCOBQBQ.
  * AVSCOBQBQ 004 110309 47100689 AVAS U
  *****
  *
  * COPYRIGHT (C) FUJITSU TECHNOLOGY SOLUTIONS 2011 *
  * ALL RIGHTS RESERVED *
  *
  *****
  ***** AVAS-ASSIGNMENT *****
    AVSA-SIGNON IS 12
    AVSA-END IS 20
    AVSA-READ-AVAS-LIB IS 52
    AVSA-CREA-PLAN-NET IS 62
    AVSA-CREA-PROD-NET IS 69
    AVSA-SUBMIT-NET IS 72
    AVSA-RESTART-NET IS 78
    AVSA-MOD-COND-DES IS 89
  ***** AVAS-RESULT/STATE *****
    AVSR-PARTIALLY IS 11
    AVSR-CREATED IS 12
    AVSR-SUBMITTED IS 15
    AVSR-ERROR IS 23
    AVSR-RESTARTED IS 28
    AVSR-PLANNED IS 124
    AVSR-NOPLAN IS 125
    AVSR-MAILED IS 128
    AVSR-UPDATED IS 130
    AVSR-NOUPDATE IS 131
    AVSR-NOSUBMIT IS 134
    AVSR-NOTFOUND IS 139
    AVSR-LOCKED IS 140
  ***
  *** END OF COPY ELEMENT AVSCOBQBQ ***
  .
  * ^ -----> PUNKT FUER SPECIAL-NAMES ERFORDERLICH.
  
```

AVSCOBW – Define work area

Call: COPY AVSCOBW REPLACING COUNT BY <integer 0..999>

COUNT Specifies the size of a table. This is the area where messages issued during the processing of a batch statement are stored in tabular form.

<integer 0..999>

Number of segments in the table. COUNT=0 indicates that no area is to be created (reserved).

Expansion of the COPY element AVSCOBW

```

COPY AVSCOBW REPLACING COUNT BY 3.
* AVSCOBW          003          110309   47100688 AVAS          U
*****
*
* COPYRIGHT (C) FUJITSU TECHNOLOGY SOLUTIONS 2011          *
*              ALL RIGHTS RESERVED                          *
*
*****
*-----*
* AVAS BATCH-PROGRAM-INTERFACE          WORK-AREA          *
*-----*

01 AVASBWRK.
   05 AVSWAREA.
      10 AVSWGCT          PICTURE S9(4) COMPUTATIONAL
                                VALUE IS COUNT.
      10 AVSWGCR          PICTURE S9(4) COMPUTATIONAL.
      10 AVSWFILE        PICTURE X(54).
      10 FILLER           PICTURE X(22).
   05 AVSWMTAB          OCCURS COUNT TIMES.
      10 AVSWMPAR.
          15 FILLER       PICTURE S9(4) COMPUTATIONAL.
          15 FILLER       PICTURE X.
          15 AVSWMSG      PICTURE X.
          15 AVSWMSG.
              17 AVSWMSGK PICTURE X(03).
              17 AVSWMSGN PICTURE X(04).
          15 AVSWMSGT     PICTURE X(125).
***
                                END OF COPY ELEMENT AVSCOBW          ***
    
```

Note

The REPLACE clause modifies the value of AVSWGCT (VALUE IS 3) and the OCCURS clause at AVSWMTAB.

AVSCOB CD – Define directory entry for condition descriptions in run control file

The COPY element AVSCOB CD specifies an entry in the directory of condition descriptions.

Call: COPY AVSCOB CD

Expansion of the COPY element AVSCOB CD

```

COPY AVSCOB CD.
* AVSCOB CD          003          110309  47105173 AVAS          U
*****
*
* COPYRIGHT (C) FUJITSU TECHNOLOGY SOLUTIONS 2011          *
*                   ALL RIGHTS RESERVED                    *
*
*****
* ABLDAT: CONDITION DIRECTORY DESCRIPTION
  07 AVSICDIR.
    10 AVSIFCNM.
      15 AVSICTYS          PICTURE X(3).
      15 AVSICNAM          PICTURE X(30).
      15 AVSICNM           PICTURE X(32).
      15 AVSICIND          PICTURE 9(3).
    10 AVSICTYP            PICTURE X.
    10 AVSICST1            PICTURE X.
    10 AVSICLTI.
      15 AVSICLTD          PICTURE X(6).
      15 AVSICLTT          PICTURE X(6).
    10 FILLER              PICTURE X(14).
***                      END OF COPY ELEMENT AVSCOB CD          ***

```

AVSCOBCE – Define condition description in run control file

The COPY element AVSCOBCE defines the specification of a condition description.

Call: COPY AVSCOBCE

Expansion of the COPY element AVSCOBCE

```

COPY AVSCOBCE.
* AVSCOBCE          003          110309  47105174 AVAS          U
*****
*
* COPYRIGHT (C) FUJITSU TECHNOLOGY SOLUTIONS 2011          *
*                   ALL RIGHTS RESERVED                    *
*                                                           *
*****
* ABLDAT: CONDITION DESCRIPTION
  07 AVSBACON.
    10 FILLER          PICTURE X(2).
    10 AVSBCCR.
      15 AVSBCCRD      PICTURE X(6).
      15 AVSBCCRT      PICTURE X(6).
    10 AVSBCTXT        PICTURE X(120).
    10 AVSBCDOC        PICTURE X(43).
    10 AVSBCLU.
      15 AVSBCLUD      PICTURE X(6).
      15 AVSBCLUT      PICTURE X(6).
    10 FILLER          PICTURE X(3).
  07 FILLER REDEFINES AVSBACON.
    10 AVSBCVAL        PICTURE X(128).
    10 FILLER          PICTURE X(64).
  07 FILLER REDEFINES AVSBACON.
    10 AVSBNNAM        PICTURE X(32).
    10 AVSBCIND        PICTURE X(3).
    10 AVSBCWUD        PICTURE X(6).
    10 AVSBCWUT        PICTURE X(6).
    10 AVSBSTUS        PICTURE X.
    10 FILLER          OCCURS 12 TIMES.
      15 AVSBCOCO      PICTURE X.
    10 AVSBCOST        PICTURE X(128).
    10 FILLER          PICTURE X(4).
***                               END OF COPY ELEMENT AVSCOBCE                               ***

```

AVSCOBJD – Define net directory specification (journal file)

Call: COPY AVSCOBJD

Expansion of the COPY element AVSCOBJD

```

COPY AVSCOBJD.
* AVSCOBJD          004          110309  47105175 AVAS          U
*****
*
* COPYRIGHT (C) FUJITSU TECHNOLOGY SOLUTIONS 2011          *
*                   ALL RIGHTS RESERVED                    *
*
*****
* JRNDAT: NET DIRECTORY DESCRIPTION
  07 AVSJDIR.
    10 AVSJNETN          PICTURE X(32).
    10 AVSJNST1          PICTURE X.
    10 AVSJNST2          PICTURE X.
    10 AVSJNST3          PICTURE X.
    10 AVSJNST4          PICTURE X.
    10 FILLER            PICTURE X(28).
***                          END OF COPY ELEMENT AVSCOBJD          ***
    
```

AVSCOBJN – Define journal record header

The COPY element AVSCOBJN specifies the header for journal records for a net.

Call: COPY AVSCOBJN

Expansion of the COPY element AVSCOBJN

```

COPY AVSCOBJN.
* AVSCOBJN          004          110309  47105176 AVAS          U
*****
*
* COPYRIGHT (C) FUJITSU TECHNOLOGY SOLUTIONS 2011          *
*                   ALL RIGHTS RESERVED                    *
*                                                           *
*****
* JRNDAT: NET JOURNAL DESCRIPTION
  07 AVSPJNET.
    10 FILLER          PICTURE X(02).
    10 AVSPDATF.
      15 FILLER          PICTURE X(02).
      15 AVSPDATE       PICTURE X(06).
    10 AVSPTIME        PICTURE X(06).
    10 AVSPNUMS        PICTURE X(02).
    10 AVSPACMD        PICTURE X.
    10 AVSPAKTN        PICTURE X.
    10 AVSPUSER        PICTURE X(08).
    10 AVSPINDX        PICTURE X(03).
    10 AVSPFUNC        PICTURE X.
    10 AVSPNAME        PICTURE X(32).
    10 AVSPRECA.
      15 AVSPRSSL        PICTURE X(02).
      15 AVSPRFNR        PICTURE X(02).
      15 AVSPOUTD        PICTURE X(316).
*-----
* FOR OUTPUT-DATA, PLEASE DEFINE YOUR OWN DEFINITION WITH
*                   15 FILLER          REDEFINES AVSPOUTD.
*                   17 FIELD          PICTURE ...
*-----
***                               END OF COPY ELEMENT AVSCOBJN                               ***
    
```

AVSCOBRT – Define result area for net functions

The COPY element AVSCOBRT defines the result area for the functions CREATE-PLAN-NET, CREATE-PROD-NET, MODIFY-COND-DESCRIPTION, SUBMIT-NET and RESTART-NET.

Call: COPY AVSCOBRT

Expansion of the COPY element AVSCOBRT

```

COPY AVSCOBRT.
* AVSCOBRT          004          110309   47100678 AVAS          U
*****
*
* COPYRIGHT (C) FUJITSU TECHNOLOGY SOLUTIONS 2011          *
*                   ALL RIGHTS RESERVED                    *
*                                                           *
*****
* RESULT DESCRIPTION
  07 AVSERSLT.
    10 AVSEOPAR      PICTURE X(64).
    10 FILLER        REDEFINES  AVSEOPAR.
      15 AVSENETN    PICTURE X(32).
      15 AVSEEAS.
        17 AVSEEASD  PICTURE 9(06).
        17 AVSEEAST  PICTURE 9(06).
      15 AVSESYND    PICTURE X(20).
    10 FILLER        REDEFINES  AVSEOPAR.
      15 FILLER      PICTURE X(32).
      15 AVSERSVT    PICTURE 9.
      15 FILLER      PICTURE X(31).
    10 FILLER        REDEFINES  AVSEOPAR.
      15 AVSENAME    PICTURE X(43).
      15 FILLER      PICTURE X(21).
    10 FILLER        REDEFINES  AVSEOPAR.
      15 AVSECDNM    PICTURE X(30).
      15 FILLER      PICTURE X(34).
    10 FILLER        PICTURE X(03).
    10 AVSERLT.
      15 AVSERLTC    PICTURE X.
        88 AVSQPART  VALUE  AVSR-PARTIALLY.
        88 AVSQCREA  VALUE  AVSR-CREATED.
        88 AVSQSUBM  VALUE  AVSR-SUBMITTED.
        88 AVSQERR   VALUE  AVSR-ERROR.
        88 AVSQREST  VALUE  AVSR-RESTARTED.
        88 AVSQPLAN  VALUE  AVSR-PLANNED.
        88 AVSQNOPL  VALUE  AVSR-NOPLAN.
        88 AVSQSAVE  VALUE  AVSR-MAVED.

```

```

      88 AVSQPDT  VALUE  AVSR-UPDATED.
      88 AVSQNOUP VALUE  AVSR-NOUPDATE.
      88 AVSQNOSM VALUE  AVSR-NOSUBMIT.
      88 AVSQNOFO VALUE  AVSR-NOTFOUND.
      88 AVSQLCKD VALUE  AVSR-LOCKED.
15  AVSERLTT  PICTURE X(12).
10  FILLER    PICTURE X(16).
***      END OF COPY ELEMENT AVSCOBRT      ***
```

11 AVAS-SV-BS2 - Job control for remote BS2000 systems

AVAS offers two interfaces for starting and monitoring runs on server systems:

1. a server interface with an AVAS server in a remote BS2000 system
2. a JV interface

A BS2000 procedure is used to start and to monitor a job in the server system. The result is returned to the AVAS run control system via the JV interface ([section “AVAS interface for external servers” on page 519](#)).

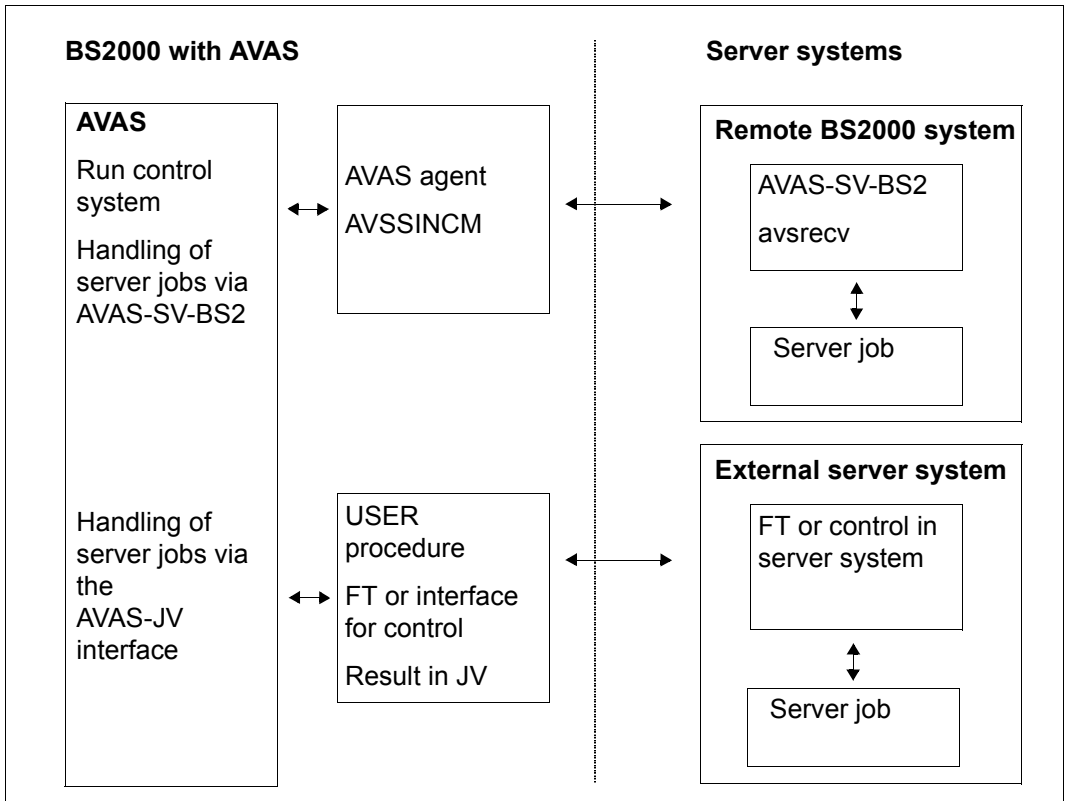


Figure 8: Overview of the AVAS server interfaces

Run control and monitoring for server jobs with an AVAS server on a BS2000 system

Jobs on the remote BS2000 system are started via structure elements using `FUNCTION=J/P` if a server name is entered in the net description for the CATID of the net or job.

In this case the BS2000 server jobs can, as normal, be managed via the AVAS libraries (with `TYPE=STD` or `MOD`) or be provided on the server system (with `TYPE=EXT`).

The run control system starts an AVAS agent (AVSSINCM) on the local AVAS server under BS2000 which provides the following services:

- Setup of the connection to the AVAS server on the remote BS2000 system via the sockets interface
- Transfer of the server job script (`TYPE=STD` or `MOD`) to the AVAS server
- Initiation of the job on the server system via the AVAS server

- Acceptance of the result of the job run (ENDED or ERROR status) from the AVAS server

Note

Log files do not need to be taken over from AVSSINCM. The SIGNAL and TRANSFER programs started in the BS2000 server jobs communicate directly with the DCAM application CENTRAL which takes over the log files from the BS2000 server system.

- Transfer of the result to the AVAS run control system via the controlling task job variable

The components required for the AVAS server for BS2000 are contained in the product AVAS-SV-BS2. The SIGNAL and TRANSFER programs are also made available for use on a remote BS2000 system in the context of the product AVAS-SV-BS2.

The general procedure is described in the [section “Run control and monitoring for server jobs” on page 470](#).

On every BS2000 server system on which AVAS is to handle jobs it must be possible to load the server AVSRECV. The address and port number of the computer must be entered in a configuration file under the AVAS ID in BS2000. The access authorization to AVSRECV is checked using the password specified in the configuration file under

<comm-pw>.

AVSRECV starts the jobs in the same way as the run control system in the local system. For the started jobs the runtime information is stored in a separate data maintenance system until it is transferred to the local AVAS. This ensures data security even in the event of a potential net failure or loss of connection.

The current job status is ascertained when a connection is set up again with RESTART-NET. Jobs that have already been ended receive the relevant result. In the case of jobs that have not been completed the end of the job is waited for.

The jobs can be redirected unmodified from the local BS2000 system to the remote BS2000 server system. All that is needed is for the new target system to be specified by means of an entry in the CATID for the net or the job in the net description or, if it has already been released, in the runtime file.

Run control and monitoring for jobs with an external server

For starting and monitoring runs on any external server systems, the AVAS-JV interface is provided.

Runs which are to be monitored via an AVAS job variable are started via structure elements with FUNCTION=P and TYPE=EXX. This starts an external procedure of the user, which is given the name of the monitoring AVAS job variable via PROCEDURE-PARAMETERS.

The run control system must be informed of the normal or abnormal termination of the run started in this way via the defined contents of the AVAS job variable.

This interface enables control to be passed to any task in BS2000.

This task can either

- itself handle the transport to the server system, the start of a job on the server system and the transfer of the result to BS2000 or
- use an interface known to it to start jobs and have them monitored on the server system and to return the result to AVAS.

The user need only ensure that

- the job variable is not deleted and
- the normal or abnormal termination of the run is signaled in accordance with the AVAS statement #AVJ#.

If the end is not signaled, the task and therefore the net remain in the RUNNING status and can only be changed to the ERROR or ABENDED status via CANCEL-NET.

11.1 Installation of AVAS-SV-BS2 on a BS2000 system

After an AVAS server has been installed on a BS2000 system using IMON, the following AVAS libraries must exist under the user ID under which AVSRECV is to be started:

- SYSPRC.AVAS.085
- SYSPRG.AVAS.085.SYSTEM
- SYSPRG.AVAS-SV-BS2.085

The AVS.RECV procedure from the SYSPRC.AVAS.085 library is called to start the AVAS server. The following parameters must be assigned beforehand in this procedure on a user-specific basis:

CONTROL-JV	Name of the job variable used to control the AVAS server Default: SYSJV.AVAS.RECV
COMM-PW	Communication password which clients use to log on to the AVAS-SERVER
PORT	Port number under which the AVAS server can be reached
WORKDIR	Suffix name under which the AVAS server saves its work files
USERID	User ID under which the AVAS libraries are saved Default: AVAS (parameter specification for the procedure)

The AVAS server on the BS2000 system is shut down by entering the value 'SHUTDOWN' in the job variable which is entered under CONTROL-JV.

11.2 Deinstallation of AVAS-SV-BS2 on a BS2000 system

When an AVAS server is deinstalled on a BS2000 system the following AVAS libraries are removed under the user ID under which AVSRECV was started:

- SYSPRC.AVAS.085
- SYSPRG.AVAS.085.SYSTEM
- SYSPRG.AVAS-SV-BS2.085

11.3 Run control and monitoring for server jobs

For communication with the server system, the run control system starts the AVAS agent AVSSINCM. This agent handles the transfer of the server job (if necessary), its start in the server system, the monitoring of its run, the transfer of results to the run control system and the passing on of log files. In the server system, the AVAS AVSRECV server must be started beforehand.

Once it has started, AVSSINCM sets up a connection to AVSRECV, transfers the server job and issues the start instruction. It writes the status of the incoming acknowledgments in its monitoring task job variable so that the run control system and the SHOW-NET-STATUS command can recognize the status of the server job.

AVSSINCM runs independently of the run control system and does not terminate until it has saved the final message of the server job in the task job variable or until the connection to the AVAS server is interrupted. An server job is therefore correctly logged as long as the connection to the AVAS server exists, even if the run control system was terminated in the meantime.

AVSRECV sets up a file in which the status of the server job is saved. This enables the result of the server job to be determined even after the connection has been cleared. This file is called a server job file in the following.

The user gives the desired AVAS server in a BS2000 configuration file a symbolic name, the <server-name> (see [page 475](#)), and ensures that this name is specified correctly by setting the password <server-pw> (see [page 475](#)). AVSSINCM checks these entries before the start of the job and determines from the configuration file and using <server-name> the host name of the server system, the port number of the AVAS server and a communication password <comm-pw> (see [page 476](#)), with which the agent (AVSSINCM) identifies itself to the server as an AVAS partner.

This offers two advantages:

- The description of a job in the net structure remains independent of specifications required by a non-proprietary operating system (server system) to start the job.
- The nets and jobs are not affected if the host configuration, the server user ID, etc. change.

This means that it is not necessary to change the nets if the server system tasks are to run on a different host immediately. Only the host name in the configuration file must be modified.

Alternatively, the configuration data can also be specified as default parameters in the start file for AVSSINCM.

Communication with the server system is presented in the [section “Preparing for operation” on page 514](#).

Run control

In an active net, the run control system recognizes a server job by the specification of a server name (bs2000-servername) in the operand NET-CAT or JOB-CAT in the net/job description. If this server job is managed in the AVAS libraries, the run control system writes it to a BS2000 file. It then starts AVSSINCM in order to transfer the BS2000 file with the server job to the server system before starting the job there.

The run control system starts AVSSINCM via an S procedure with the help of the ENTER-PROC command. The statements necessary for AVSSINCM in the procedure are described on [page 478 ff.](#)

The name and password for the S procedure are passed to the run control system with the parameters SERVER-PROCEDURE-NAME and SERVER-PROCEDURE-PASSWORD (see [section “Starting and terminating the run control system” on page 115](#)).

Before starting a server job, the availability of the server is checked by the run control system.

A job variable named AVS.SERVER.STATE.<server-jv-link-name>.<server-name> is to be created under the BS2000 user ID of the run control system for every server described in the configuration file. This server job variable is used to check the operability of a server. It is created and managed like a job variable used to monitor the computer network status of an MSCF network user for the MSCF network under BS2000.

The assignment and content of the server job variable is described in the [section “Server monitor process” on page 488](#).

The following status values for a server job are set by the run control system depending on the first two positions of the AVS-SERVER job variable:

HOSTWAIT The server is not available

ERROR When server job variable (AVS.SERVER.STATE.<server-jv-link-name>.<server-name>) cannot be read (in this case the server job cannot be started)

With the task job variable for AVSSINCM, the run control system can only monitor AVSSINCM directly. An area of the task job variable is therefore reserved for the status of the server job (SV status), which is updated by AVSSINCM.

The run control system sets the status of a server job after it has evaluated the SV status:

STARTING	The AVSSINCM job is running; it establishes a connection to the server and prepares the job start
RUNNING	The AVSSINCM job and server job are running
FINISHING	The AVSSINCM job is running and is performing housekeeping (transfer of the log files); the server job is terminated
ENDED	if SV status='T' and MONJV for the job AVSSINCM='\$T'
ERROR	all other settings (see AVSSINCM)

The SV status can be output via the SHOW-NET-STATUS statement (see the “AVAS Statements” manual [2] for the parameter description).

AVSSINCM

Once the input parameters have been read, AVSSINCM checks the SV status in its run-monitoring task job variable. The settings tell AVSSINCM whether the server job

- is to be started for the first time (SV status is empty). The job is transferred to the server. AVSSINCM runs in START mode.
- has already run but announces a job error via the value 'RV=n' in the monitor job variable, and a restart was requested. The job is transferred to the server. AVSSINCM works in the START mode.
- has already been started but the result has not yet been returned to the run control system. This might be the case, e.g. if the connection to the server was lost in the course of the job run and the SV status was set to 'E' (byte 184 in the MONJV of the job AVSSINCM). Consequently the job is assigned the status ERROR. The result can be retrieved from the server using RESTART-NET. AVSSINCM then runs in QUERY mode.

AVSSINCM sets up a connection to the AVAS server which is addressed in the configuration data via <host-name> and <port-no>. If the server job is stored in AVAS it is transferred. The request to start the job is then sent.

The <server-name> and <server-pw> are specified

- in the net description (masks AVN001/AVN002 for BS2000 jobs, see “AVAS Statements” manual [2]) or – if *STD is specified there –
- in the definition of the user group (BK in the AVAS.USER.GENPAR) generation file) or – if *STD is specified there –
- in the start procedure AVS.SINCM of AVSSINCM.

Notes

- The version of the AVAS server must be entered in the relevant server job variable so that AVSSINCM can select the appropriate protocol for this server. The entry is made via the AVAS server monitor process. If operation takes place without server monitor, the user must enter the version in the server job variable.
- If AVSSINCM does not address the AVAS server using the correct protocol, the server jobs are assigned the status ERROR-COM.
- The assignment and content of the server job variable are described in the [section “Server monitor process” on page 488](#).

AVSSINCM transfers the status of the server job to the server areas of its run-monitoring task job variable.

When the server job is terminated, the server transfers the log files to AVSSINCM if the server job has signaled for them to be transferred. The logs are saved under AVSSVLOG.<<avas-jobid>.<tsn>.<suffix> in the user ID of the AVSSINCM. They can be signaled using the SIGNAL program which is integrated into AVSSINCM and transferred to the LOGSYS using TRANSFER. Then AVSSINCM takes over the AVAS area, writes it into its run-monitoring task job variable, and sets the SV status according to how the server job was terminated, i.e. to “T” if it was executed without errors, or “A” if errors occurred during the run. AVSSINCM then sends an END message to AVSRECV and terminates normally (T).

If AVSSINCM loses the connection to the AVAS server agent AVSRECV while in operation, it sets the SV status to “E”, leaves the additional status unchanged so that it is evident when the connection was lost, and terminates with an exit code.

The progress of the operation and the results of the server job are shown by the status and the additional status.

The following statuses are used:

Status/add. status	Meaning
SF	Server job in transfer
SS	Server job in start phase
RR	Server job running
TT	Server job terminated without error
AA	Server job terminated with error
EF	Connection to server was lost during transfer of server job
ES	Connection to server was severed during start phase
ER	End status of server job unknown because communication was interrupted after start but before end of server job
PP	AVSSINCM terminated with error because start parameter for AVSSINCM or authentication was incorrect
C	AVSSINCM terminated with error because no connection could be established to the server.
F[x]	These statuses briefly show the error status of a job which has terminated with error for which a restart will take place (between WAITING and STARTING). The following statuses can occur (the corresponding status at job termination is shown in parentheses): F (≙ C), FF (≙ EF), FS (≙ ES), FR (≙ ER) and FP (≙ PP). The statuses are displayed in the dialog masks using STARTING.

11.3.1 Structure of the configuration file

The server configuration file is used for the assignment of real values (server system, server user ID of the server job) to the symbolic specifications for the server system in the net description. The configuration file is created and updated as a BS2000 ISAM file via an editor (EDT).

AVSSINCM, the AVAS agent in BS2000, must have access to the configuration file. If AVSSINCM is to be started under different IDs, either each ID must have its own configuration file or it must be possible for AVSSINCM to access a common file.

There must be an entry in the configuration file for every AVAS server that AVSSINCM is to establish a connection to. This configuration entry contains the specifications needed in the server system for storing and starting a file. The configuration file is best stored under the AVAS ID under which the run control system is running, where it should be protected with READ-PASS against unauthorized access.

Structure of an entry in the configuration file:

```
<server-name> <server-pw> <hostname/port> <userid> <comm-pw> <dir> <path>
<transfer-admission>
```

server-name	Symbolic name of the user ID. 8-byte area (possibly to be filled with blanks). The entry must be uppercase and must begin in column 1. The symbolic name of a server user ID is entered in the CAT-ID field of masks AVN001/AVN002 in BS2000 systems.
server-pw	*NONE must be entered for a BS2000 server. The entry must begin in column 10.
hostname/port	Host name or IP or IPv6 address of the server system and the socket port number (up to 26 bytes) under which the desired AVAS server can be reached.
userid	The user ID (up to 8 bytes) under which the job is to be started in the server system. *NONE must be entered for a BS2000 server. The user ID for the job run is contained in the net description.

comm-pw	Communication password via which the AVAS agent AVSSINCM proves its communication authorization to AVSRECV; i.e. it must match the password also designated as <comm-pw> used to start the server. The agent password is not the logon password for the user ID <userid> stated above that is used to start a server job.
dir	*STD must be entered for a BS2000 server. External jobs are stored under the user ID.
path	Work directory *STD must be entered for a BS2000 server. The work directory is defined by the AVAS server.
transfer-admission	FTAC access authorization on the server (corresponds to the TRANSFER-ADMISSION operand in the REMOTE-PARAMETER entry of the TRANSFER-FILE-SYNCHRONOUS command, see the manual "openFT for BS2000" [12])

The individual areas must be separated by at least one blank. The <server-name> specification must begin at position 1, <server-pw> must begin at position 10.

None of the entries may contain any blanks, as these would be interpreted as separators.

The <comm-pw> is passed to the server system, being converted from EBCDIC to ASCII in the process. It may therefore contain only characters which are defined in the conversion table used. By default, code conversion EBCDIC-ASCII is used.

The file must be created as an ISAM file with KEY-LENGTH=9, KEY-POS=5 and RECORD-FORMAT=VARIABLE. For the ISAM key, the server-name (8 bytes) followed by a blank is used.

It might happen that an entry exceeds the permitted maximum length that EDT can process. In cases like this, continuation lines can be specified. The ISAM key for these lines is formed from the server-name repeated in the first eight columns, followed by a continuation flag specified instead of the blank. This character defines the order of the continuation lines. Once the 9-byte key has been removed, they are added directly after the preceding lines. This mechanism can also be employed for writing every area as of <hostname/port> in a separate line.

If the <server-name> entry of a job is changed between the start and restart of the job, it could be that AVSSINCM can no longer determine the status of the server job at the restart (e.g. if the host name of the server system or the user ID has been modified). In this case, AVSSINCM rejects the restart with TYPE=RESTART, but executes it with TYPE=NORMAL if the server job was already started.

Example of an IP entry in a configuration file

```
BS2000SV *NONE 123.45.67.89/7777 *NONE COMMPW *STD *STD
```

Parameter assignment:

server-name	BS2000SV
server-pw	*NONE
hostname/port	123.45.67.89/7777
userid	*NONE
comm-pw	COMMPW
dir	*STD
path	*STD

Editing the AVAS configuration file

The AVS.EDITCONFIG procedure in the SYSPRC.AVAS.085 library is supplied for editing the AVAS configuration file.

11.3.2 Starting the AVAS agent AVSSINCM

To start the AVAS agent, the run control system issues an ENTER-PROC call for the file specified in SERVER-PROCEDURE-NAME. Inputs for AVSSINCM are predefined via procedure parameters.

The following start procedure is supplied with the rest of the AVAS procedures and must be provided at the start of AVSSINCM.

```

/SET-PROCEDURE-OPTIONS LOGGING-ALLOWED=YES, DATA-ESCAPE-CHAR=*STD
/BEGIN-PARAMETER-DECLARATION
/ DECL-PARAM USERID          (INIT=' ')
/ DECL-PARAM CNTAPP          (INIT=' ')
/ DECL-PARAM CNTAPC          (INIT=' ')
/ DECL-PARAM SVD0G-JV (INIT=' '*NONE')
/ DECL-PARAM AVS-AVAK-FILE
/ DECL-PARAM AVS-AVAK-NAME
/ DECL-PARAM AVS-CONF-FILE
/ DECL-PARAM AVS-CONF-PASSW
/ DECL-PARAM AVS-JOB-ID
/ DECL-PARAM AVS-JOB-INDEX
/ DECL-PARAM AVS-JOB-NAME
/ DECL-PARAM AVS-JOB-TYPE
/ DECL-PARAM AVS-MONJV-PASSW
/ DECL-PARAM AVS-NET-NAME
/ DECL-PARAM AVS-SERVER-NAME
/ DECL-PARAM AVS-SERVER-PASSW
/ DECL-PARAM AVS-SERVER-PARAM
/ DECL-PARAM AVS-SERVER-VERSION
/ DECL-PARAM AVS-SYSTEM-ID
/ DECL-PARAM AVS-AVAS-VERSION
/ DECL-PARAM AVS-AVAK-FILE-PASSW
/ DECL-PARAM AVS-JOB-CLASS
/ DECL-PARAM AVS-JOB-LOG
/ DECL-PARAM AVS-JOB-PARAMETER
/ DECL-PARAM AVS-USER-ACCOUNT
/ DECL-PARAM AVS-USER-ID
/ DECL-PARAM AVS-USER-PASSW
/ DECL-PARAM AVS-DEF-COMM-PASSW (INIT=' '*NONE')
/ DECL-PARAM AVS-DEF-DIRECTORY (INIT=' '*NONE')
/ DECL-PARAM AVS-DEF-HOST-NAME (INIT=' '*NONE')
/ DECL-PARAM AVS-DEF-SERVER-PASSW (INIT=' '*NONE')
/ DECL-PARAM AVS-DEF-SERVER-VERS (INIT=' '*NONE')
/ DECL-PARAM AVS-DEF-PORT-NUMBER (INIT=' '*NONE')
/ DECL-PARAM AVS-DEF-USER-ID (INIT=' '*NONE')
/END-PARAMETER-DECLARATION
/ADD-PASSWORD PASSWORD=&(AVS-CONF-PASSW)

```

```

/SET-FILE-LINK LINK-NAME=AVSXID -
/ , FILE-NAME=&(AVS-CONF-FILE) -
/ , ACCESS-METHOD=ISAM, KEY-LEN=9, KEY-POS=5
/ASSIGN-SYSDTA *SYSCMD
/START-PROG FROM-FILE=*MODULE(ELEMENT=AVAS.SYS.LOAD.SINCM -
/ , LIBRARY=$&(USERID).SYSPRG.AVAS.085.SVCOMM -
/ , RUN-MODE=*ADVANCED, PROGRAM-MODE=*ANY)
AVAK-NAME = &(AVS-AVAK-NAME)
FILE-NAME = &(AVS-AVAK-FILE)
FILE-PASSW = &(AVS-AVAK-FILE-PASSW)
JOB-ID = &(AVS-JOB-ID)
JOB-INDEX = &(AVS-JOB-INDEX)
JOB-NAME = &(AVS-JOB-NAME)
JOB-TYPE = &(AVS-JOB-TYPE)
JV-PASSW = &(AVS-MONJV-PASSW)
NET-NAME = &(AVS-NET-NAME)
SERVER-NAME = &(AVS-SERVER-NAME)
SERVER-PASSW = &(AVS-SERVER-PASSW)
SERVER-PARAM = &(AVS-SERVER-PARAM)
SERVER-VERSION = &(AVS-SERVER-VERSION)
SYSTEM-ID = &(AVS-SYSTEM-ID)
AVAS-VERSION = &(AVS-AVAS-VERSION)
JOB-CLASS = &(AVS-JOB-CLASS)
JOB-LOG = &(AVS-JOB-LOG)
JOB-PARAMETER = &(AVS-JOB-PARAMETER)
USER-ACCOUNT = &(AVS-USER-ACCOUNT)
USER-IDENTIFICATION = &(AVS-USER-ID)
USER-PASSW = &(AVS-USER-PASSW)
DEFAULT-COMM-PASSW = &(AVS-DEF-COMM-PASSW)
DEFAULT-DIRECTORY = &(AVS-DEF-DIRECTORY)
DEFAULT-HOST-NAME = &(AVS-DEF-HOST-NAME)
DEFAULT-SERVER-PASSW = &(AVS-DEF-SERVER-PASSW)
DEFAULT-SERVER-VERSION = &(AVS-DEF-SERVER-VERS)
DEFAULT-PORT-NR = &(AVS-DEF-PORT-NUMBER)
DEFAULT-USER-ID = &(AVS-DEF-USER-ID)
SVD0G-JV = &(AVS-SVD0G-JV)
JOBLOG-APPLICATION = &(CNTAPC), &(CNTAPP)
END
/ASSIGN-SYSDTA *PRIMARY
/REMOVE-FILE-LINK LINK-NAME=AVSXID
/REMOVE-PASSWORD PASSWORD=&(AVS-CONF-PASSW)
/SKIP-COMMANDS TO-LABEL=#SW29SET, IF=JOB-SWITCHES(ON=29)

```

```

/CALL-PROCEDURE *LIBRARY-ELEMENT (ELEMENT=AVS.TRANSFER -
/ ,LIBRARY=$&(USERID).SYSPRC.AVAS.085) -
/ ,PROCEDURE-PARAMETERS=(USERID=&(USERID) -
/ ,CNTAPP=&(CNTAPP) -
/ ,CNTAPC=&(CNTAPC))
/ .#SW29SET REMARK
/END-PROCEDURE

```

Note

Parameters which are not specified with an INIT clause are set by the run control system. They must be passed unchanged to AVSSINCM if the procedure is adjusted to the user's requirements or is embedded in another procedure. AVSSINCM expects all parameters other than those for defining default values (AVS-DEF-) to be specified.

USERID	This parameter must be set to the catalog ID and/or user ID if the load library is not present at the point where AVSSINCM executes. The entry must end with a period.
CNTAPP	Name of the CENTRAL application.
CNTAPC	Name of the processor with the CENTRAL application.
AVS-AVAK-FILE	Name of the file to which the run control system has written the script or file name on the server in the case of external jobs.
AVS-AVAK-NAME	Name of the run control system.
AVS-CONF-FILE	Name of the configuration file.
AVS-CONF-PASSW	Password for the configuration file. If no password is used, this parameter is set to X'00000000'.
AVS-JOB-ID	Internal AVAS job ID.
AVS-JOB-INDEX	Job index from the net description.
AVS-JOB-NAME	Job name from the net description.
AVS-JOB-TYPE	Specifies whether the job script is managed in AVAS or is located in the server system.
AVS-MONJV-PASSW	Password for the monitor job variable in hexadecimal form. If no password is used, the parameter is given the value NONE. If fewer than eight characters are transferred, AVSSINCM supplements these with leading zeros.
AVS-NET-NAME	Net name from the net description.
AVS-SERVER-NAME	Server name from the net description.

AVS-SERVER-PASSW	Password with which the user proves that he is allowed to use the specified AVS-SERVER-NAME.
AVS-SERVER-PARAM	String which separates script and parameters in the AVAS-AVAK-FILE file.
AVS-SERVER-VERSION	Version of the AVAS server AVSRECV.
AVS-SYSTEM-ID	ID of the AVAS system.
AVS-AVAS-VERSION	AVAS version.
AVS-AVAK-FILE-PASSW	Password of an external job or *NONE.
AVS-JOB-CLASS	For a BS2000 server job: Name of the job class from the net description Otherwise:*NONE
AVS-JOB-LOG	For a BS2000 server job: Value of the LOGGING parameter (YES / NO) from the net description (LOG parameter) Otherwise:*NONE
AVS-JOB-PARAMETER	For a BS2000 server job: Value of the job parameter from the net description Otherwise: *NONE
AVS-USER-ACCOUNT	For a BS2000 server job: Value of the account from the net description Otherwise: *NONE
AVS-USER-ID	For a BS2000 server job: Value of the user ID from the net description Otherwise: *NONE
AVS-USER-PASSW	For a BS2000 server job: Value of the password of for the user ID from the net description Otherwise: *NONE

AVS-DEF-COMM-PASSW(INIT=""*NONE"")

Default communication password.

Via this password, AVSRECV recognizes that the message comes from an authorized sender. It must match the password that is given to AVSRECV at startup.

AVS-DEF-DIRECTORY(INIT=""*NONE"")

Directory in which AVSRECV is to search for shell scripts on the server system (can be extended via specifications in the file name).

AVS-DEF-HOST-NAME(INIT=""*NONE"")

If SERVER-NAME=*STD:

Host name or Internet address of the server on which jobs are to run.

AVS-DEF-SERVER-PASSW(INIT=""*NONE"")

If SERVER-NAME=*STD:

Password which the user must specify to be able to start server jobs.

AVS-DEF-SERVER-VERS(INIT=""*NONE"")

If SERVER-NAME=*STD:

Version (Vnn.nann) of the AVAS server.

AVS-DEF-PORT-NUMBER(INIT=""*NONE"")

If SERVER-NAME=*STD:

Standard port number under which the AVSRECV can be reached.

AVS-DEF-USER-ID(INIT=""*NONE"")

User ID on the server system.

AVS-SVDOG-JV

Name of the job variable which monitors the SVDOG.

Notes

- Note that under no circumstances must
 - the remaining statements for AVSSINCM be modified,
 - other statements for AVSSINCM be entered or existing ones deleted.
- A #AVJ# statement after the END statement in the sequence of statements for AVSSINCM overwrites entries that the server job has made.
- The start parameters for AVSSINCM must always be written in full.
- When SERVER-NAME=*STD is specified, the parameters for AVSSINCM must be contained in its start procedure.
When SERVER-NAME=*STD is not specified, AVSSINCM must have access to the configuration file. Note here that the configuration file itself should be protected on account of the passwords it contains.
- If a parameter is specified in quotes so that it can be defined with lowercase letters in a procedure, (e.g.: DECL-PARAM AVS-DEF-USER-ID(INIT= '''user'''), AVSSINCM removes the delimiting quotes. Quotes within a text are not changed.
- The SKIP-COMMANDS TO-LABEL=#SW29SET option is set by AVSSINCM if a BS2000 server job is concerned. This option means that the TRANSFER program is not called because signaling and transfer take place directly in the jobs.
- The CENTRAL application is addressed using the procedure parameters CNTAPC and CNTAPP. Alternatively, when the program starts, the JOBLOG-APPLICATION parameter can be supplied with the name of a job variable which contains the application and process names (see the parameter description in section [“Starting the SIGNAL program” on page 146](#)).

11.3.3 Structure of the task job variable and the server job file

The contents of the task job variable and the server job file are predefined by BS2000 in the first 128 bytes. The remaining bytes are used AVAS-specifically.

The task job variable or the server job file is structured as follows (the last fields are distinguished according to the Internet Protocol Version using the hostname entry in the configuration file):

Byte	Length	Info	Set by
1	2	Status of AVSSINCM	BS2000
129	55	AVAS user area	Server job
184	1	Status of server job	AVSSINCM
185	1	Additional status of server job	AVSSINCM
186	10	reserved	
196	1	Job type	AVSSINCM
197	4	reserved	
201	4	Error message number	AVSSINCM
205	2	Error code in message	AVSSINCM
207	1	Mode of AVSSINCM	AVSSINCM
		If Internet Protocol Version 4	
208	16	Host name of server system	AVSSINCM
224	8	Server name	AVSSINCM
		If Internet Protocol Version 6	
208	1	IP version identifier x'FF'	AVSSINCM
209	39	Host name or IPv6 address	AVSSINCM
248	8	Server name	AVSSINCM

Byte 1-2 Status of the AVSSINCM process (after BS2000)

\$S: Task has started

\$R: Task running

\$T: Task has terminated normally

\$A: Task has terminated abnormally

Byte 184 SV status (status of the server job)

S: Server job is ready to start (after file transfer)

R: Server job has started

	T:	Server job has terminated normally
	A:	Server job has terminated abnormally
	E:	AVSSINCM lost the connection to the AVAS server before the end of the server job.
	P:	Server job was not started due to a parameter error in the AVSSINCM procedure or user authentication failed.
	C:	AVSSINCM could not set up a connection to the AVAS server.
Byte	185	Additional status of the server job
	E:	Server job has no connection to AVSSINCM
	F:	File transfer running
	S:	Server job is ready to start
	R:	Server job has started
	T:	Server job has terminated normally
	A:	Server job has terminated abnormally
	P:	Server job was not started due to a parameter error in the AVSSINCM procedure or user authentication failed
Byte	196	Job type I: if script is managed internally (in AVAS). E: if script exists externally (in the server system).
Byte	201	Error message number (4 characters) Message number of the error that caused the processing of the server job to be aborted (by AVSSINCM).
Byte	205	Reserved for error code (2 characters)
Byte	207	Mode in which AVSSINCM is working (1 character). T: TRANSFER mode: A TRANSFER follow-up job is started on the remote BS2000 system for a BS2000 server job. C: CANCEL mode: A started server job is to be aborted.
		If Internet Protocol Version 4:
Byte	208	Host name of the server system (16 characters) The first 16 characters of the host name or its IPv4 address
Byte	224	server-name (8 characters)
		If Internet Protocol Version 6:

- Byte 208 Identifier IPv6 (1 character)
x'FF'
- Byte 209 Host name of the server system (39 characters)
The first 39 characters of the host name or its IPv6 address
- Byte 248 server-name (8 characters)

11.3.4 Job control and monitoring for remote BS2000 systems

Jobs on remote BS2000 systems which are not connected to the local system via MSCF can also be started and monitored via an AVAS server .

11.3.4.1 Starting the AVAS server under BS2000

The AVAS server is started by calling the AVS.RECV procedure from the SYSPRC.AVAS.085 library. The AVSRECV program requires the following input parameters via SYSDTA:

CONTROL-JV	<filename 1..54> Name of the job variable with which the AVAS server is controlled Default: SYSJV.AVAS.RECV
COMM-PW	<name 1..8> Communication password with which clients must log on to the AVAS server
PORT	<integer 1025..65535> Port number under which the AVAS server can be reached
WORKDIR	<name 1..32> Suffix name which the AVAS server uses to store its work files
USERID	<name 1..8> User ID under which the AVAS libraries are stored Default: AVAS (parameter specification for the procedure)

11.3.4.2 Terminating the AVAS server under BS2000

The SHUTDOWN operation terminates the AVAS server AVSRECV.

This operation is input via the monitoring job variable. The operation to be executed must be the first entry and consist of job variable values:

```
/MODIFY-JV JV=jvname,SET-VALUE=C'SHUTDOWN'
```

11.3.5 Server monitor process

In order for server jobs to only be started on active servers, AVAS offers a monitoring function for the servers entered in the configuration file. This function is implemented by the server monitor process.

Server jobs that are to be started on a server currently not available are placed in the HOSTWAIT state.

A job variable named AVS.SERVER.STATE.<server-jv-link-name>.<server-name> is to be created under the BS2000 user ID of the run control system for every server entered in the configuration file. This AVS-SERVER job variable is used to check the operability of a server. The server control job variable is updated via the AVAS server monitor process.

The server job variable contains the following:

Bytes	1..2	Status of the AVAS server:
	\$U	Undefined: The server status is unknown. Jobs for this server are started unconditionally.
	\$R	Running: The server is operational and can accept jobs. Jobs for this server are started and are placed in the RUNNING status.
	\$T	Terminated: The server monitor or the AVSSINCM program could not set up a connection to the server. The server is not operational or cannot be reached via the network. No jobs can be accepted. Jobs for this server are placed in the HOSTWAIT status.
	\$D	Deleted: The server is not contained in the configuration file. The server has attempted to delete the server job variable. Jobs for this server are started and are placed in the ERROR status.
	\$I	Ignore: The server is currently not being monitored. The status must be modified manually. Jobs for this server are placed in the HOSTWAIT status.
Byte	3	Free

Byte	4	Diagnostic information of AVSSVDOG Entry for the socket function which led to the \$T status:
	C	connect
	G	gethostbyaddr
	I	inetaddr
	R	recv
	S	send
	R	select (read)
	w	select (write)
Byte	5	Free
Bytes	6..10	Associated error number
Byte	11	Free
Bytes	12..19	Version of the AVAS server AVSRECV in the form Vnn.nann, e.g. V8.5A00
Byte	20	Free
Bytes	21..39	Start time of the server, format: yyyy-mm-dd hh:mm:ss
Byte	40	Free
Bytes	41..59	Time at which the server first failed to reply, format yyyy-mm-dd hh:mm:ss

Notes

- The AVAS agent AVSSINCM requires the version of the AVAS server (bytes 12..19) to select the correct protocol for communicating with the server.
- The version designation is entered by the server monitor the first time it addresses an AVAS server.
- AVAS servers which cannot respond to the query are assigned *Unknown.
- If operation takes place without the AVAS server monitor, once AVAS-SV-BS2 V08.5 has been installed the value "V08.5A00" must be entered in its job variable. The relevant version or "*Unknown" must be entered for older AVAS servers.

When starting server jobs, the run control system also sets their status depending on the first two positions of the SV-JV:

RUNNING	If the value in the first two positions of the SV-JV is '\$R'
HOSTWAIT	As long as the value in the first two positions of the SV-JV is not '\$R'
ERROR	If there is no corresponding SV-JV. In this case the server job cannot be started.

11.3.5.1 Functions of the AVAS server monitor process

Determining the server from the configuration file

When the server monitor is started, the configuration files are read and the SV-JVs are set accordingly. The server monitor creates an SV-JV if there isn't one already for a server to be monitored.

After that, the state of the AVAS server is determined. The server monitor sends messages to the server to be monitored and waits for its replies.

If the corresponding server sends a reply, the value '\$R' is entered in the associated SV-JV instead of '\$U'. If no reply is received, then the SV-JV is set to the value '\$T'.

Cyclical server check

The servers are checked when server monitoring is started. This check is performed cyclically in an interval set with the SV-CONTROL-TIME operand.

When new servers are entered in the configuration file or old ones removed, the server environment to be monitored by the server monitor can be updated with the CONFIG operation (see [“Operation for updating the server environment” on page 494](#)).

The run control system must be notified of the modification via the call

```
/INFORM-PROGRAM MSG='USERVER',JOB-ID=*TSN(<tsn>)
```

or by supplying the run control system's job variable with 'USERVER':

```
/MODIFY-JV JV=<jv-name>,SET-VALUE=C'USERVER'
```

Receiving messages and jobs

The server monitor process can receive jobs and messages from other programs if the PORT parameter is assigned a value not equal to *NONE. Which programs are concerned here is regulated by the PORT-USAGE parameter:

- *AVSSURF The AVSSURF parameter can call information about the status of the servers and initiate a check of the server status and of the configuration file.

If the SYSTEM-ID parameter is supplied with a value not equal to *NONE, AVSSURF can perform user authentication vis-à-vis an AVAS system and, if necessary, make changes to the service configuration (editing the configuration file, setting the SV-JV).
- *AVSRECV Servers on which AVSRECV 8.0A or higher runs are no longer checked regularly (SV-CONTROL-TIME). Instead, they themselves are assigned the job of regularly sending in-service messages to the UDP port addressed by means of the PORT parameter. AVSSVDOG only contacts such servers if the in-service messages fail to arrive.

This procedure reduces the load on the net and the CPU requirements of AVSSVDOG.

Note

This functionality can be used only if the socket host name of the host on which the server monitor process is running is known in the net (entry in DNS) or locally on the servers (e.g. entry in /etc/hosts).

11.3.5.2 Starting the server monitor

The server monitor requires the following input parameters via SYSDTA:

JVSVD OG	<filename 1..54> Name of the controlling job variable (mandatory parameter)
SV-CONTROL-TIME	<integer 1..100> Interval in minutes in which the availability of the server is checked Default value: 5 minutes
CONFIG-CONTROL-TIME	<integer 1..1440> Interval in which the configuration file is checked for changes Default value: 60 minutes
CONFIG-FILE-NAME	<filename 1..54> Name of the configuration file (mandatory parameter)
CONFIG-FILE-PASSW	<c-string 1..4> / <x-string 1..8> Password for configuration file Default value: *NONE
SERVER-JV-LINK	list-poss(16):<name 1..8> Link name in the name of the server job variable in which the current server status is entered by the server monitor. (AVS.SERVER.STATE.<server-jv-link-name>.<server>) If the run control system is started without the start parameter SERVER-JV-LINK, the name of the run control system must be specified in the server monitor (see also page 121). <i>Note</i> Before a server job is started the run control system uses the server job variables to check whether the server is active. It must therefore be informed of the link name. This is done using the start parameter SERVER-JV-LINK of the run control system. The names must be identical for the server monitor and the run control system.
RCS-JV	list-poss(16):<filename 1..54> *NONE Names of the controlling job variables of the run control system Default value: *NONE
PORT	<integer 1025..65535 *NONE> TCP or UDP port via which AVSSVDOG can be contacted by AVSSURF and AVSRECV Default value: *NONE

INITIAL-TIMEOUT	<integer 1..60> Wait time (in seconds) for the response from the server when contact is first made (start of AVSSVDOG). The wait time applies for each individual communication step. Default value: 1 second
TIMEOUT	<integer 1..60> Wait time (in seconds) for the response from the server after contacts to servers has first been made. Default value: 10 seconds
PORT-USAGE	list-poss(2): *AVSSURF *AVSRECV *ALL Programs which may use the TCP or UDP port specified with PORT. This parameter is evaluated only if the value for PORT is not equal to *NONE. Default value: *AVSSURF
SYSTEM-ID	list-poss(10): <name 7..7> *NONE Names of the AVAS system IDs which AVSSVDOG uses for authenticating AVSSURF users. Default value: *NONE

Note

When starting the AVAS system, the run control systems may only be started if the server monitor has updated the SV-JVs to reflect the current state (READY status in the control job variables for the server monitor).

11.3.5.3 Terminating the server monitor

The server monitor process is terminated using the SHUTDOWN operation.

The operation is entered via the monitoring job variable. The operation to be executed must be the first entry and consist of job variable values:

```
/MODIFY-JV JV=jvname,SET-VALUE=C'SHUTDOWN'
```

11.3.5.4 Operation for updating the server environment

The operation is entered via the monitoring job variable. The operation to be executed must be the first entry and consist of job variable values.

Example:

```
/MODIFY-JV JV=jvname,SET-VALUE=C'CONFIG'
```

The following operations are available for updating the server environment:

- | | |
|---------|---|
| CONFIG | Server entries in the configuration file have been changed. The server monitor process reads the configuration file and updates the server environment to be monitored. |
| CONFIGn | n=1..1440
Same effect as CONFIG. n is also provided as the value for CONFIG-CONTROL-TIME.
The modification is logged to SYSOUT. |
| SERVER | The servers are checked again. However, the configuration file is not read in again. |
| SERVERn | n=1..100
Same effect as SERVER. n is also provided as the value for SV-CONTROL-TIME.
The modification is logged to SYSOUT. |

11.3.5.5 How to temporarily lock a server

Job distribution by the run control system can be disabled for a short time for a server in order to configure the server, for example. The corresponding server job variable is set to \$I (Ignore) to accomplish this.

If the lock is to be removed, set the value back to '\$R'.

11.3.6 Server interface process

The server interface process provides administrators of server systems with an overview of all the tasks which AVAS has started on their computers.

The AVSSURF program enables the following actions to be performed with an ordinary Web browser:

- Displaying an overview of the server statuses in a table
- Updating the statuses of the servers
- Reading in the configuration file again
- Displaying an overview of all the AVAS server jobs which are running on a server
- Displaying the data of a server job

In addition, users can authenticate themselves to an AVAS system by means of an AVAS ID and the associated password. Depending on the rights of this AVAS ID, they can also perform the following actions:

- Stopping and restarting job assignment for a server (job assignment is stopped by setting '\$!' in the server JV)
- Editing the configuration file
- Viewing script and log files of server jobs
- Modifying attributes of log files
- Aborting server jobs



Server monitoring with AVSSVDOG is required to access a task which is running in BS2000.

The AVSSURF program works as a specialized server for HTTP requests (“Web server”). To permit communication with AVSSURF the socket host name of the host on which the server monitor process runs must be known either in the net (entry in DNS) or locally on the servers (e.g. entry in /etc/hosts).

Communication between the Web browser and AVSSURF can be encrypted if required. To permit this, AVSSURF contains the cryptographic functions of the open source software OpenSSL. The certificates and keys required can be requested from a certificate authority or can be created by the user (e.g. in accordance with the procedure described in the “InterNet Services User Guide” [11]).

11.3.6.1 Starting the server interface

AVSSURF is started using the AVS.SURF procedure in the SYSPRC.AVAS.085 library. AVSSURF reads the start parameters via SYSDTA:

CERTIFICATE-CHAIN-FILE	<filename 1..54> <u>*NONE</u> Name of a file which contains a chain of valid certificates. The parameter is only evaluated when USE-SSL=1.
CONTROL-JV	<filename 1..54> Name of the monitoring job variable (mandatory parameter)
KEY-FILE	<filename 1..54> <u>*NONE</u> Name of a file which contains the program's secret key. The parameter is only evaluated when USE-SSL=1.
PORT	<integer 1025..65535> TCP port under which AVSSURF can be contacted by the Web browser (mandatory parameter)
SECURE-PORT	<integer 1025..65535> <u>*NONE</u> TCP port which is used for encrypted communication. The parameter is only evaluated when USE-SSL=1.
SESSION-TIMEOUT	<integer 1..600> <u>15</u> Validity period of a sign-on session.
STYLE-SHEET	<filename 1..54> <u>*NONE</u> Name of a file which contains a Cascading Style Sheet (CSS). The CSS is linked into every HTML page generated by AVSSURF and enables the design to be customized according to user requirements.
SVDOG-HOST	<name 1..8> <ip-addr> <ipv6-addr> Name or IP/IPv6 address of the host on which the server monitor process is running (mandatory parameter)
SVDOG-PORT	<integer 1025..65535> TCP port under which the server monitor process can be reached (mandatory parameter)
USE-SSL	<u>0</u> 1 Deactivates or activates encryption of communication with OpenSSL

11.3.6.2 Shutting down the server interface

The server interface is shut down by means of the SHUTDOWN operation. This operation is entered via the monitoring job variable. The operation to be executed must be the first entry and consist of job variable values:

```
/MODIFY-JV JV=jvname,SET-VALUE=C'SHUTDOWN'
```

11.3.6.3 Displays of the web-based server interface

The displays of the web-based server interface, which are based on the FHS interface of AVAS in BS2000, are also referred to as “masks”. The mask name is derived from the corresponding AVAS mask.

The displays in the browser can differ from the figures shown below because the various browsers display the same HTML code differently. Furthermore, the appearance can be influenced considerably by the layout file specified in the start parameter STYLE-SHEET.

The title bar of the browser window contains the following information:

- The program name AVSSURF
- The version number
- The current mask name and a brief description of the mask

Each mask contains a button containing a question mark in the upper right-hand corner. When you click on this question mark, a window opens which contains help texts for the current mask.

Setting up a connection to AVSSURF

The AVSSURF program is addressed in the Web browser using the following URL:

`http://host-name:port`

where:

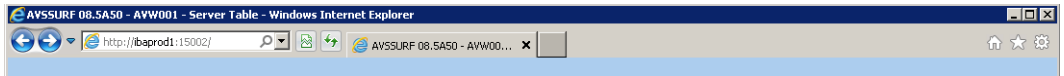
`host-name` Name or IP or IPv6 address of the host on which the server interface process runs

`port` TCP port number which was assigned by means of the start parameter PORT

In the example below the server interface process runs on the host with the name `bs26666n`, and port number 13014 was assigned when it started.

- ▶ Enter the URL in the browser's address bar and load the address.

If the address remains the same, connection setup can also take place by means of a link to the address saved earlier (e.g. as a bookmark or the browser's homepage).



When the connection to AVSSURF has been set up successfully, the browser displays mask AVW001.

Mask AVW001 - Server Table

Mask AVW001 is displayed after the connection to AVSSURF has been set up.

The mask is also displayed when you click on *Back* in mask AVS010, AVW002 or AVW010.

The data displayed is automatically updated. For this purpose the server monitor process which supplies the data notifies the browser when it will next update the server status.

The screenshot shows a web browser window titled "AVSSURF 08.5A50 - AVW001 - Server Table - Windows Internet Explorer". The address bar shows "http://ibaprod1:15002/". The page content is titled "AVAS Server Configuration" and includes the following information:

- received from watchdog on: IBAPROD1, port 15003 (with a "Signon" button)
- Configuration file: SYSDAT.AVAS.SINCONFIG
- Last update of configuration data: 02/14/17 12:38:12 (with an "Update" button)
- Last update of server statuses: 02/14/17 12:38:44 (with an "Update" button)

Below the configuration is a table with the following data:

Server Name	Host	Port	Version	Status	Start date/time	Stop date/time
BS20SD1	172.17.0.13.8	15001	08.5A50	SR	02/10/17 09:39:13	
BS20SD2	172.17.0.13.19	15001	08.5A10	SI		
BS20SD3	172.17.0.13.28	15001	08.5A10	SI	02/10/17 10:20:55	02/14/17 10:54:44

The header contains the following information:

- Name of the host on which the server monitor process (server watchdog) runs and the port number under which it can be reached.

If the user is not signed on (e.g. after a connection to AVSSURF has been set up), the *Signon* button is displayed.

- ▶ Clicking on *Signon* enables the user to authenticate himself/herself: mask AVS010 is displayed for the user so that he/she can enter their signon data (see "[Mask AVS010 - AVAS Signon](#)" on page 502).

If the user is already signed on, the *Signoff* button is displayed instead of *Signon*.

- ▶ The user signs off by clicking on *Signoff* (terminates privileges).

- Name of the configuration file used by the server monitor process
If the user has signed on with an AVAS ID which has the MANAGE-SERVER privilege, the *Edit* button is also displayed.
 - ▶ Click on *Edit* to display the configuration file for editing in mask AVW002.
- Date and time of the last update of the configuration file and the *Update* button
 - ▶ Click on *Update* to request the server monitor process to be read into the configuration file again.
- Date and time of the last check on the server status and the *Update* button
 - ▶ Click on *Update* to request the server interface process to check the server status.

The following information is displayed in the table part:

- Server Name
Name of the server as it is stored in the configuration file.
With active AVAS servers of Version 7.0A or higher, the server name is displayed as a *<server-name>* button.
 - ▶ Click on *<server-name>* to display the information on this server in mask AVW010.
- Host
Host name or IP or IPv6 address of the server as it is stored in the configuration file.
- Port
Port number of the server as it is stored in the configuration file.
- Version
Version number of the AVAS server. If the server monitor process was not able to determine the version number (server does not respond or is too old), “*Unknown“ is displayed.

- Status

Status of the AVAS server. For the meaning of the various statuses, please see the server monitor process.

To make them easier to enter, status \$R (Running) is set against a green background, status \$T (Terminated) against a red background and status \$I (Ignored) against a gray background.

If the user signed on with an AVAS ID which has the MANAGE-SERVER privilege, instead of the current status he/she is offered three options for the statuses \$R, \$T and \$I. The current server status is selected.

The user can change the server status:

- ▶ Clicking on the option of another status causes the status to change:
The status selected is written into the server JV. The server monitor process then updates the server statuses and the content of mask AVW001.

The following status transitions are permitted (when a transition is not permitted, the corresponding status cannot be selected):

Status transition	Meaning
\$R -> \$I	AVAS will start no more jobs for this server for the time being.
\$T -> \$I	AVAS will start no more jobs for this server even if it becomes active in the meantime.
\$I -> \$R	Jobs can be started again on this server. Before jobs are started, the server monitor process checks whether the server is active. Depending on the result of this check, the status which is written to the server JV can be \$R or \$T.

- Start date/time

Start date of the AVAS server.

This field can be empty if the AVAS server has not been active since the start of the server monitor process.

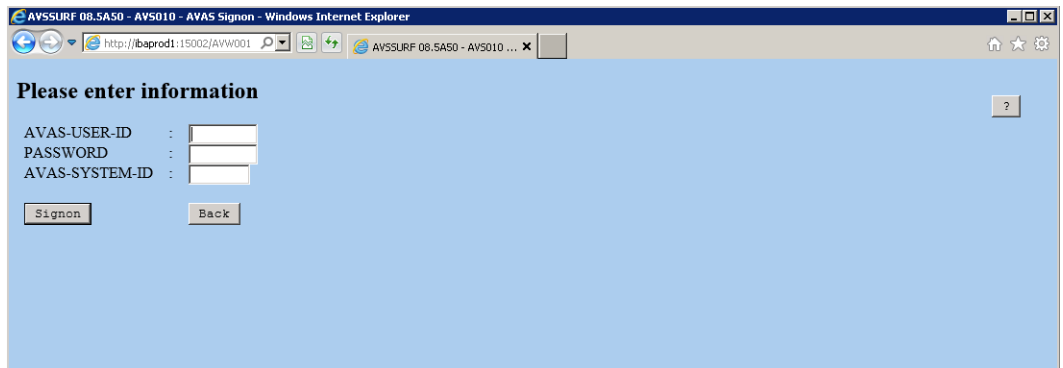
- Stop date/time

Date on which a previously active server first fails to respond.

This field can be empty.

Mask AVS010 - AVAS Signon

Mask AVS010 is displayed when you click on *Signon* in mask AVW010.

The image shows a screenshot of a web browser window titled "AVSSURF 08.5A50 - AVS010 - AVAS Signon - Windows Internet Explorer". The address bar shows "Http://ibaprod1:15002/AVW001". The main content area has a light blue background and the heading "Please enter information". Below the heading are three input fields: "AVAS-USER-ID", "PASSWORD", and "AVAS-SYSTEM-ID". At the bottom of the form are two buttons: "Signon" and "Back". A small question mark icon is visible in the top right corner of the form area.

In its function mask AVS010 corresponds to the AVAS mask of the same name. The input fields and their meaning are also described there.

Successful authentication in this mask is a prerequisite for executing privileged functions in other masks. The privilege expires after 5 minutes of inactivity or after the user signs off using *Signoff* in mask AVW001.

To perform authentication:

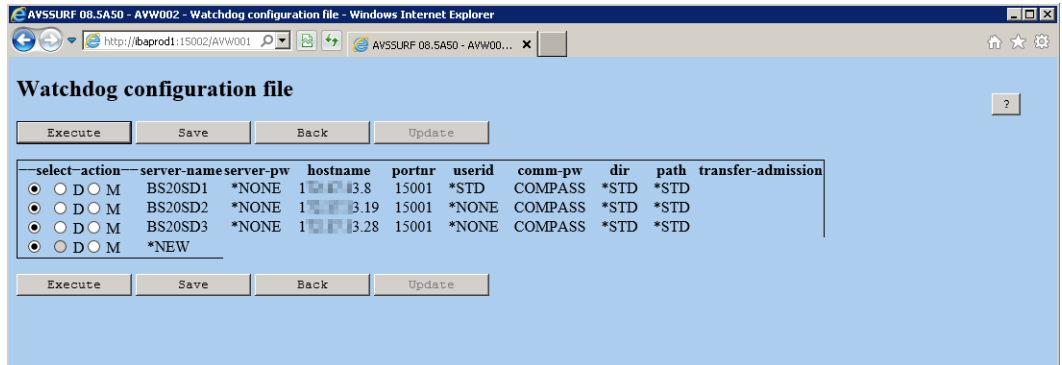
- ▶ Enter signon data in the input fields
- ▶ Click on *Signon*
Only if the signon is successful does the system branch to mask AVW010, otherwise mask AVS010 remains and an AVAS error message is issued.

To abort authentication:

- ▶ Click on *Back* to go to the original mask AVW001.

Mask AVW002 - Watchdog configuration file

Mask AVW002 is displayed when you click on *Edit* for the configuration file in mask AVW001.



A table is displayed containing the entries in the configuration file (refer to this for the meaning of the columns).

To permit the individual entries to be edited, the table begins with a selection column in which one of three options can be selected for each entry:

1. No editing of the entry
The first option (without a label) selects a server entry which is not to be edited. When the mask is displayed or after editing has taken place, this option is enabled for all entries.
2. Select entry to be deleted
Option D (delete) selects a server entry which is to be deleted.
3. Select entry to be modified
Option M (modify) selects a server entry which is to be modified.

Depending on the option selected, the following actions are executed by means of the *Execute*, *Save*, *Back* and *Update* buttons:

- Delete server entry
 - ▶ Select option D for an existing entry (server name not equal to *NEW). This option can also be selected for more than one entry.
 - ▶ Click on *Execute* to delete an entry locally and update the mask.

Further entries can be edited, or editing can be terminated by saving the configuration file.
- Modify server entry
 - ▶ Select option M for an existing entry (server name not equal to *NEW). This option can also be selected for more than one entry.
 - ▶ Click on *Execute* to enable the fields of the entry to be overwritten. Exceptions: the *server-name* and *path* fields cannot be overwritten.
 - ▶ Change the required values in the server entry.
 - ▶ Click on *Execute* to accept the change locally and update the mask.

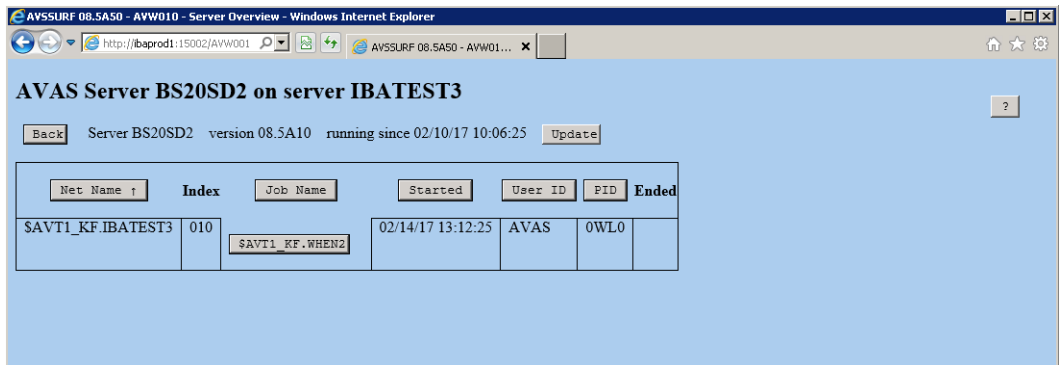
Further entries can be edited, or editing can be terminated by saving the configuration file.
- Create new server entry
 - ▶ Select option M for the last entry (server name not equal to *NEW).
 - ▶ Click on *Execute* to enable the fields of the entry to be overwritten. Exception: the *path* field is preset with “*STD” and cannot be overwritten.
 - ▶ Enter the values for the new server entry.
 - ▶ Click on *Execute* to accept the change locally and update the mask.

Further entries can be edited, or the local editing status can be written back to the configuration file.
- Write changes back to the configuration file
 - ▶ Click on *Save* to write the changes back to the configuration file, terminate editing and go back to mask AVW001.

If the table contains configuration entries with invalid syntax, the faulty entries are offered for correction in mask AVW002. *Save* is only possible once the entries have been corrected.
- Abort editing
 - ▶ Click on *Back* to return to mask AVW001. Changes which have not yet been saved in the configuration file are then discarded.

Mask AVW010 - Server Overview

Mask AVW010 is displayed after you click on `<server-name>` in mask AVW001.



The name of the server, its version and its start date are displayed in the header. The *Back* and *Update* buttons are also contained there.

Information on the following server jobs on this server is displayed in a table:

- all server jobs which are currently active
- all server jobs which have already been terminated whose status could not yet be forwarded to AVAS

Data is displayed in the following information columns for each server job:

- Net Name
Net name of the job
- Index
Index level of the job in the net
- Job Name
Name of the job
- Started
Start date and time
- User ID
User ID under which the job runs

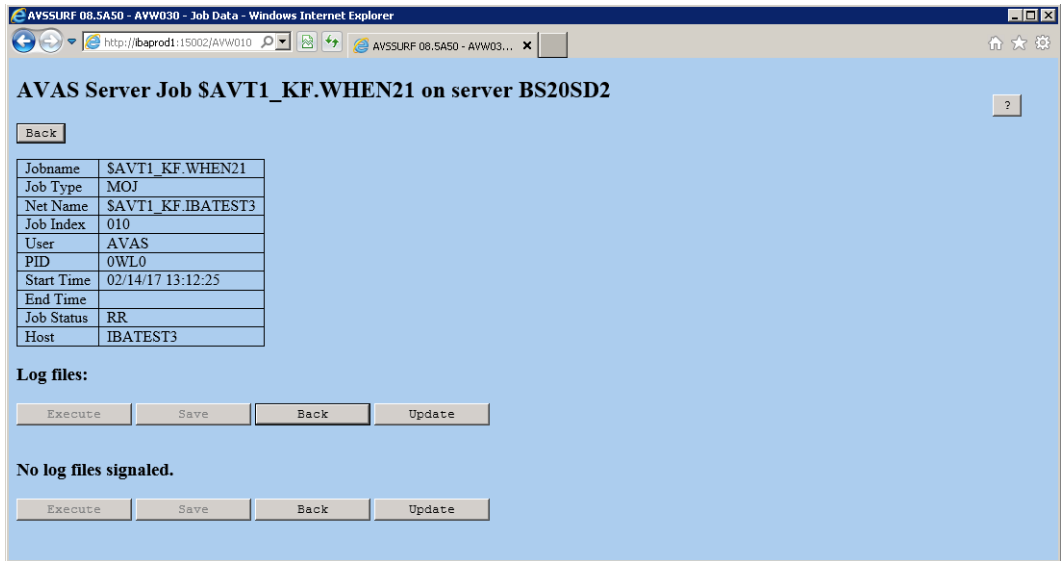
- PID
ID of the job's execution unit (from the viewpoint of the operating system):
TSN (task sequence number)
- Ended
End date and time (only in the case of jobs whose status has not yet been forwarded to AVAS)

The following actions can be performed:

- Change the sort sequence in the table
The server jobs are sorted first according to net name, then according to index level, and finally according to job name. The column headers *Net Name*, *Job Name*, *Started*, *User ID* and *PID* are displayed as buttons.
 - ▶ When you click on one of these buttons, the sort sequence for the current column is reversed (from ascending to descending order or vice versa).
- Request more information on a job
Every job name displayed in the *Job Name* column is displayed as a *<job-name>* button.
 - ▶ Click on *<job-name>* to go to mask AVW030, in which more information on this job is displayed.
- Terminate job
If the user signed on with an AVAS ID which has the functional authorization CANCEL-NET for a job, the *CANCEL* button is also offered for this job.
 - ▶ When you click on *CANCEL*, an attempt is made to terminate the job concerned in the background. The result of the action is consequently not immediately visible in mask AVW010.
- Abort editing
 - ▶ Click on *Back* to return to mask AVW001.
- Update displayed information
 - ▶ Click on *Update* to update the content of the mask.

Mask AVW030 - Job Data

Mask AVW030 is displayed when you click on a job name in mask AVW010.



The job name and the name of the server on which it is running are displayed in the mask header. The *Back* button is also contained there.

The data for this job which was already displayed in mask AVW010 is displayed in a table, as is the following data, too:

- Job Type
Job type from the AVAS viewpoint (MOD, STD, EXT, EXX).
- Job Status
Current status of the server job; for possible values see [page 474](#).
- Host
Host name of the system on which the server job is running.

After the “Log files.” header, the display then shows whether log files are signaled for this job. If no files are signaled, this is displayed with “No log files signaled”.

If files are signaled, a table with the data of the log files is displayed. This contains the following for each signaled log file:

- Zip-FT
Attribute-compressed transfer.
- File Name
Name of the signaled log file.
- Delete
The delete attribute of the log file. When a check mark is set, this indicates that the log file will be deleted after it has been successfully transferred to BS2000.
- Curr. Size
Current size of the log file (in bytes).
- Max. Size
Limit value for transfer to BS2000 in kB. The value “Unlimited” means that the entire file will be transferred to BS2000 (provided the resources to do this are available).
- Head
Specifies what percentage of the log file will be taken from the start of the file when the maximum size is exceeded.

The following actions can be performed:

- Terminate job
If the user signed on with an AVAS ID which has the functional authorization CANCEL-NET for a job, the *CANCEL* button is also offered in the mask header.
 - ▶ When you click on *CANCEL*, an attempt is made to terminate the job concerned in the background. The result of the action is consequently not immediately visible in mask AVW030.
- Display job's script file

If the user signed on with an AVAS ID which has the functional authorization SHOW-PROD-JOB for the job, the job name in the table is displayed in the *<job-name>* button.

▶ Click on *<job-name>* to display the content of this script file in mask AVW030.

- Display log file

If the user signed on with an AVAS ID which has the functional authorization SHOW-PROD-JOB for the job, the file name of a signaled log file is displayed in the *<filename>* button.

▶ Click on *<filename>* to display the content of the log file in mask AVW030.

- Modify attributes of a log file

If the user signed on with an AVAS ID which has the functional authorization MODIFY-SUBMIT-JOB for the job, the attributes *Delete*, *Max. size* and *Header* of a signaled log file are displayed and can be overwritten.

▶ Click on the check box by *Delete* or enter a new value in the *Max. size* or *Header* field to modify the attribute concerned.

▶ Click on *Save* to save the changes on the server.

- Terminate editing

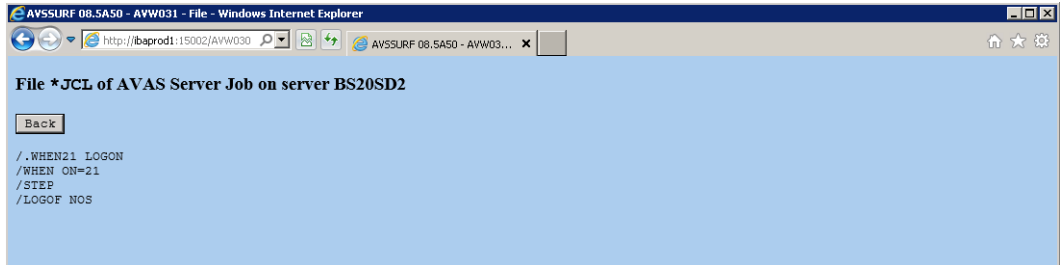
▶ Click on *Back* to return to mask AVW001.

- Update displayed information

▶ Click on *Update* to update the content of the mask.

Mask AVW031 - Displaying a log or script file

Mask AVW031 is displayed after you click on a log file name or job name in mask AVW030. Either the content of this log file or the content of the script file for this job is displayed. You cannot edit the file which is displayed.



- ▶ Click on *Back* to end the display and return to mask AVW030.

11.4 Behavior after system failure

Problems or system failures may occur during productive operation. To ensure that operation continues reliably and smoothly after a system failure, AVAS will recontinue processing after a restart of

- AVAS itself
- the operating system or
- individual system components

at the point where the failure occurred. The results of BS2000 remote jobs that had already been started at the time the failure occurred and that were terminated during the downtime are determined. If jobs have not yet been terminated, AVAS waits for the result.

AVAS carries out the following measures in the course of a system restart:

- AVAS determines the current status of BS2000 jobs started through AVAS. It acquires information on which jobs were in the status RUNNING at the time the failure occurred from the runtime file. AVAS obtains information on the job status from the job's monitoring job variable or through a query sent to the AVAS server (QUERY mode).
- Jobs that have meanwhile been terminated are assigned the status ENDED or ERROR, jobs that are still processing are assigned the status RUNNING.
- Jobs that have failed along with BS2000 or a server system are assigned the status ERROR.

The following failure scenarios may occur from the viewpoint of AVAS:

1. Failure of AVAS in BS2000

AVAS has to be reloaded and determines the current production environment.

- BS2000 jobs that were RUNNING at the time the failure occurred and that have not been terminated when AVAS is restarted are assigned the status RUNNING.
- BS2000 jobs that were RUNNING at the time the failure occurred and that have been terminated by the time AVAS is restarted are assigned a status indicating that they have been terminated normally or abnormally (ENDED/ERROR).
- BS2000 jobs that were to be started while AVAS was unavailable will be handled according to the delay settings in the net description (NET-DELAY-SOLUTION).
- server jobs that have terminated in the meantime and that could not transfer their log files to the CENTRAL task encounter a normal or abnormal end of job (ENDED/ERROR), depending on the settings for the program calls SIGNAL or TRANSFER.

The log files can be transferred to the AVAS library subsequently using the AVAS command ADD-JOB-LOG.

2. Failure of BS2000 and consequently also of AVAS

AVAS is reloaded along with BS2000 and determines the production environment at the time of the failure.

When AVAS is restarted following a BS2000 failure (including AVAS and the server job agent AVSSINCM), AVAS automatically triggers a QUERY for server jobs that were RUNNING at the time the failure occurred. The automatic result query is carried out by the run control system, even if it was terminated using CANCEL.

- BS2000 jobs that were RUNNING when the failure occurred, also failed and are assigned the status ERROR.
- The current status of server jobs that were RUNNING when the failure occurred is determined.
Server jobs that were meanwhile terminated are assigned the status ENDED or ERROR.
Server jobs that have not been terminated retain the status RUNNING.
The log files of server jobs that have been terminated in the meantime are captured.
- Server jobs that were to be started while AVAS was unavailable will be handled according to the delay settings in the net description (NET-DELAY-SOLUTION).

3. Failure of communication between BS2000 and the AVAS server

For productive operation to run correctly, the AVAS agent AVSSINCM must be able to communicate with the AVAS server.

- The server job that can continue in the server system is assigned the status ERROR (as is the job net also). Following a RESTART-NET, when communication is available again, the current status of the server job is determined.
- Server jobs that were RUNNING when communication was lost and that have been terminated in the meantime are assigned the status ENDED or ERROR.
server jobs that have not been terminated yet retain the status RUNNING.
The log files of server jobs that have been terminated in the meantime are captured.
- Server jobs that could not be transferred back to the server due to communication problems are assigned the status ERROR. In the case of a RESTART AVAS will find that the job has not been run yet and will again attempt to start it.

The scenarios described above also apply when the AVAS agent AVSSINCM fails.

4. Failure of the AVAS server

The AVAS server must be restarted.

- Server jobs that were RUNNING at the time the failure occurred continue until the job is ended. Since the AVAS agent AVSSINCM lost the connection to the server, these server jobs are assigned the status ERROR-COM in the runtime file. When these ERROR-COM jobs are restarted the result of the server job is determined (QUERY mode) when the server is available again.
- New server jobs cannot be started until the AVAS server has been restarted. They are assigned the status ERROR-COM. When these ERROR-COM jobs are restarted the server job can also be restarted (START mode) when the AVAS server is available again.

11.5 Preparing for operation

Before server jobs can be started and monitored from AVAS, the preparatory steps listed in this section must be taken.

The following AVAS objects must either be created again or modified:

- net description
- generation parameter
- configuration file
- start procedure for AVAS agent AVSSINCM
- start call for AVAS server AVSRECV
- start procedure for AVAS run control system
- start procedure for AVAS server monitor

The following figure illustrates the relationships between the necessary entries:

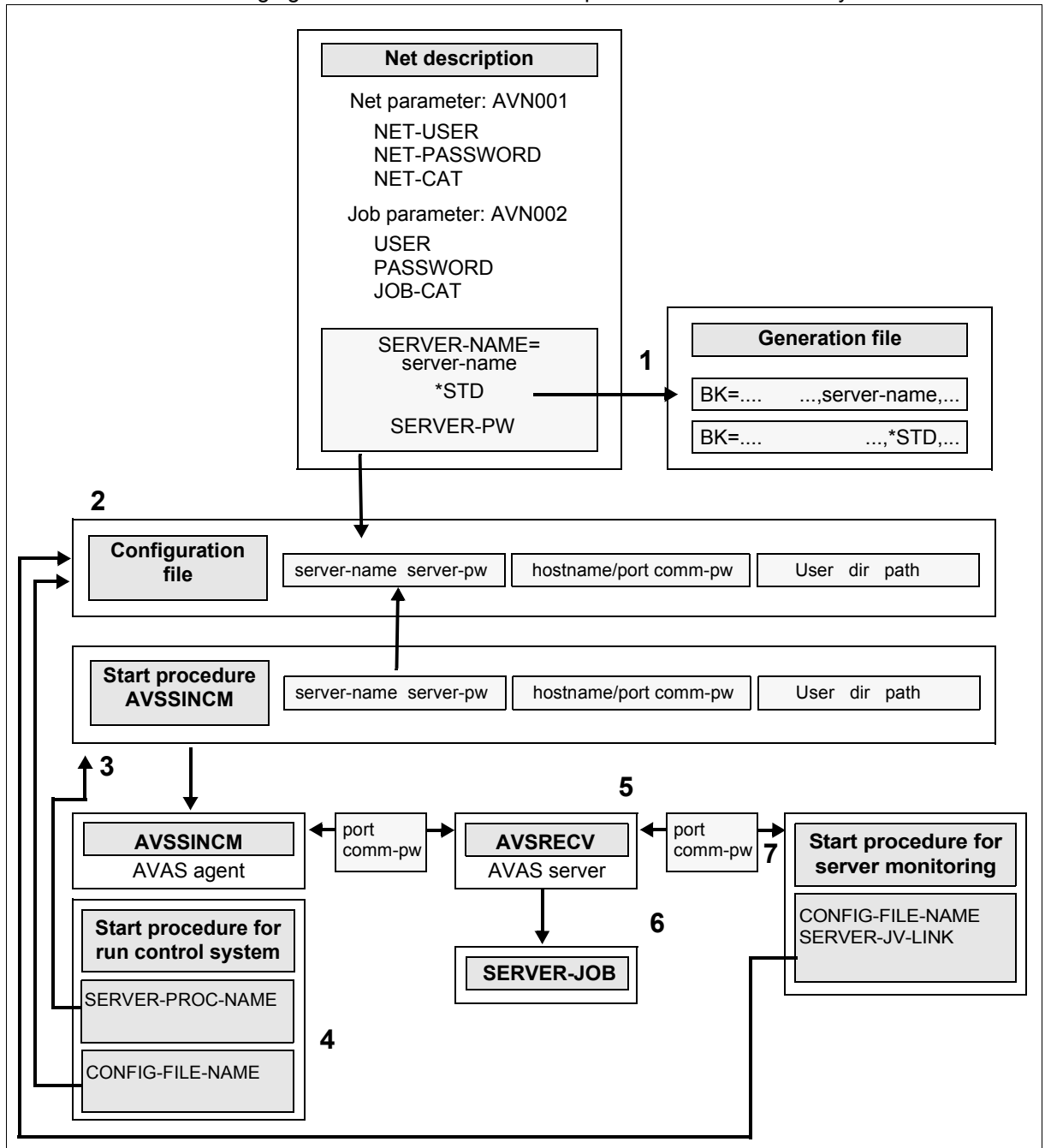


Figure 9: Relationships between the definitions in BS2000 and in the server system

1 Defining the server names for every user group

A server-name is a symbolic name for the processor (host) on which and the user ID (userid) under which an server job is to run.

AVAS takes the server name from the net description:

- Mask AVN001, NET-CAT field, entered value “(bs2000-servername)”
- Mask AVN002, Job-CAT field, entered value “(bs2000-servername)”

If a server-name is set to *NONE for a user group, members of this group cannot execute jobs on server systems.

2 Creating the configuration file

If jobs are to be started on different hosts and/or under different user IDs, a configuration file must be created in BS2000. For every user ID under which AVAS is to start jobs, a server-name and a server-pw must be assigned and then saved as a separate entry in the configuration file.

If jobs under the same user ID are started on different hosts, a separate server-name is to be assigned for each host. The server-names must be unique within the configuration file. They act as identifying keys for further specifications.

The configuration file must be readable from within AVSSINCM.

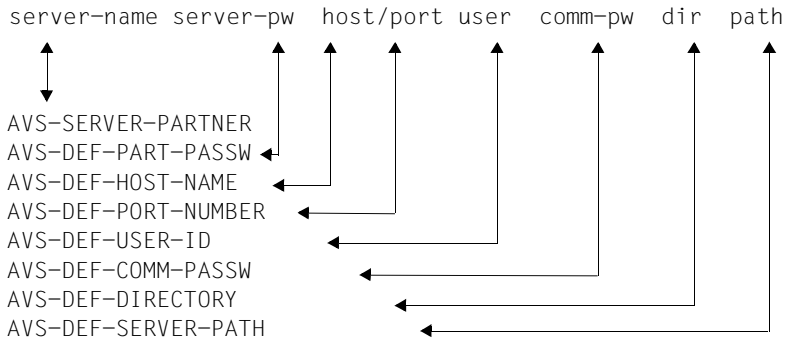
At the port number specified for a server partner, the AVAS server AVSRECV waits for tasks from AVSSINCM. A separate port number is required for every server that is to be started on a host. If AVSRECV is started under *root* (superuser), only one AVSRECV, and therefore only one port number, is required for all users of this host.

The AVAS server AVSRECV knows only one communication password. This means that for all server-names which address the same host and the same port number the same communication password must apply.

3 Adjusting the start procedure for AVSSINCM

The start procedure supplied for AVSSINCM (see example on [page 478](#)) must be adjusted to suit the needs of the application if the server information is not defined by the server name entry in the net description.

The parameters of the start procedure (except for transfer-admission) correspond to the entries in the configuration file:



The name of the start procedure for AVSSINCM and the password for access to it must be entered in the start procedure of the run control system.

4 Adjusting the start procedure for the run control system

The start parameters for server support must be given values in the supplied start procedure of the run control system:

- the name of the start procedure for AVSSINCM (SERVER-PROCEDURE-NAME parameter)
- if necessary, a password authorizing use of the procedure (SERVER-PROCEDURE-PASSWORD parameter)
- the name of the configuration file (CONFIG-FILE-NAME parameter)
- if necessary, a password authorizing reading of the configuration file (CONFIG-FILE-PASSWORD parameter)

The specifications on the configuration file are required whenever SERVER-NAME has been defined in the net description.

5 Starting AVSRECV in the server system

AVSRECV must be started once on each host for each assigned port number. If the assigned numbers were entered in etc/services, AVSRECV can also be started with the service name. The communication password specified at the start of AVSRECV (comm-pw parameter) must match the corresponding entry in the configuration file or in the start procedure (uppercase and lowercase are differentiated).

6 The necessary user specifications for the start of an server job are taken from the configuration file or the start procedure for AVSSINCM.

7 Modifications to the start procedure for the server monitor

In the server monitor start procedure supplied, the following start parameters need to be set:

- Name of the configuration file (CONFIG-FILE-NAME parameter)
- Possibly a password that authorizes read access to the configuration file (CONFIG-FILE-PASSW parameter)
- Names of the run control systems that work with the specified configuration file (SERVER-JV-LINK parameter)

11.6 AVAS interface for external servers

AVAS allows external servers to be used for job processing.

The net description permits a new structure element for starting procedures. FUNCTION=P with TYPE=EXX describes a structure element for starting an external procedure with an externally monitoring job variable.

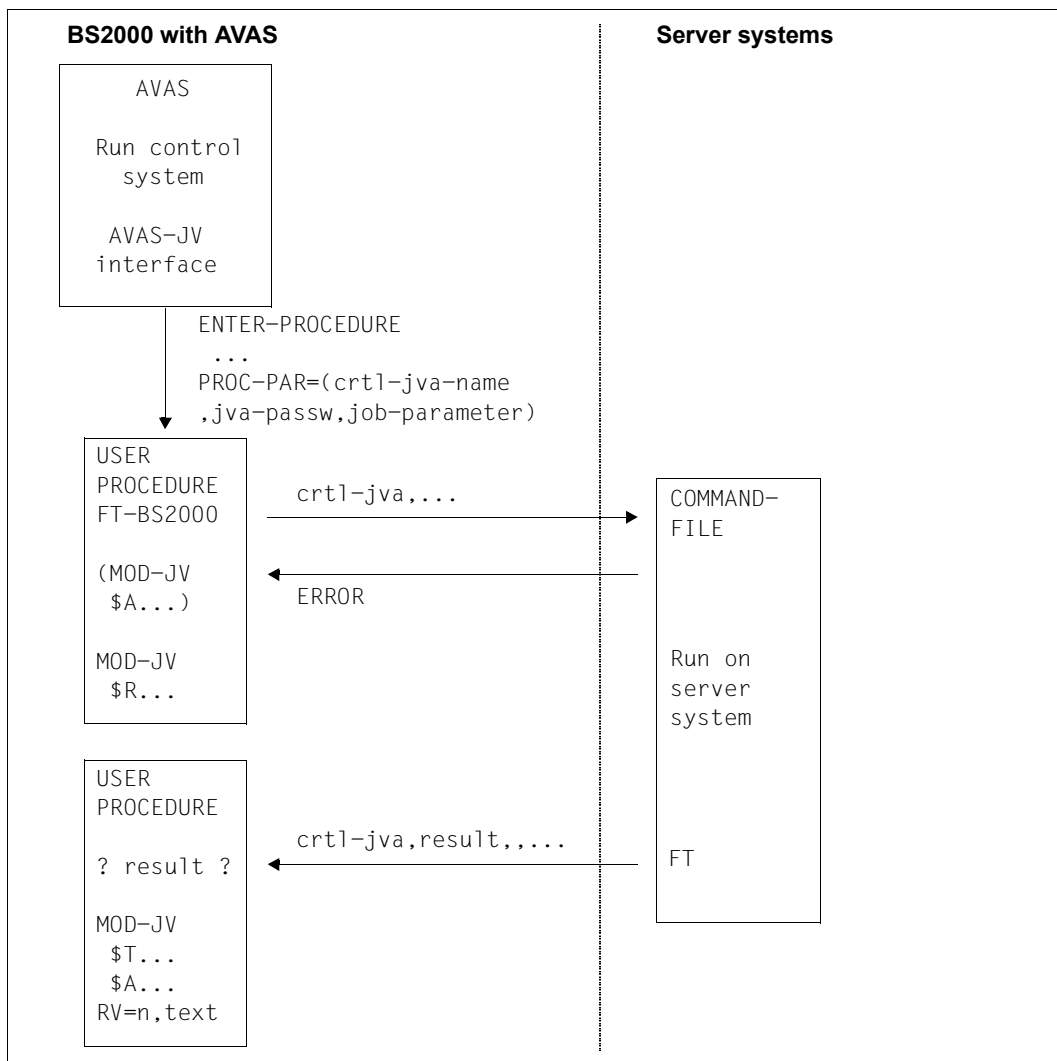
The job variable is set up by the run control system and at the start of the procedure is not assigned as a task job variable but is passed on as a procedure parameter. When the run starts, it must supply the job variable with valid values in accordance with the rules for the BS2000 task job variable and for the AVAS statement #AVJ#.

If the name of the job variable is passed on, possibly with a password, in a task chain, the job variable can be given valid status information in the follow-up tasks.

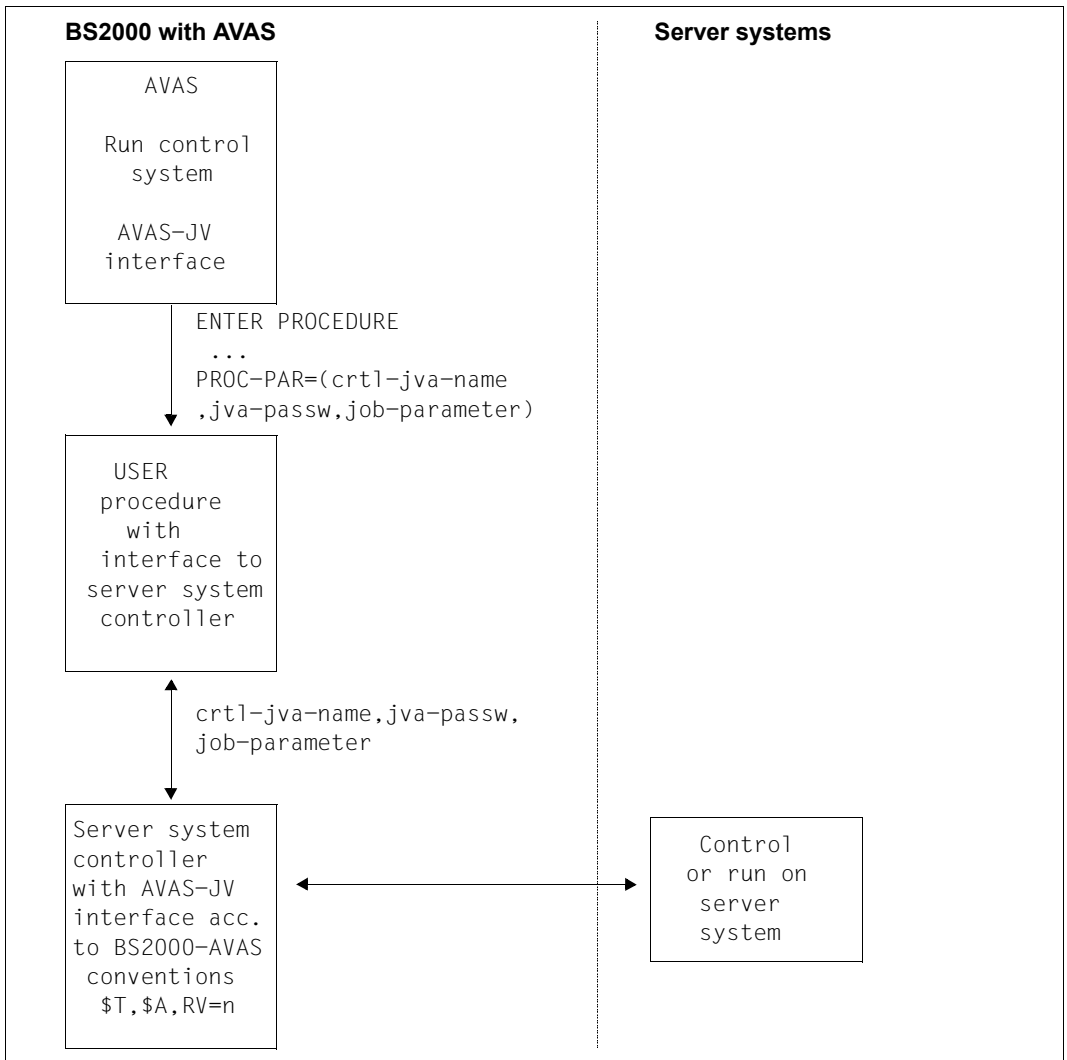
The run control system handles the job variable like the task job variable of a job which was started as a procedure via a structure element with TYPE=EXT. An event is set for the information \$T and \$A. You need only make sure that the last task in the sequence stores valid status information (\$T or \$A) in the job variable and that the job variable is *not* deleted by the user.

Two application examples are illustrated on the following pages (refer to the two diagrams on the following pages).

1. Application:
Starting and monitoring a run on a server system via AVAS



2. Application:
Starting and monitoring a run with monitoring by a BS2000 server system controller



Note

With the start of the procedure, a BS2000 TSN (task sequence number) is assigned, and is stored in the run control file and the journal file.
The TSN is a component of the sort key of the JOBLLOG logging and is displayed there.
The TSN is not entered in the job variable by BS2000 because it is not a task job variable in this case.

The user must take into account that the saved TSN is used in the JOBLOG logging and in the display via SHOW-NET-STATUS and SHOW-JOURNAL. This applies particularly when a run involves several processes.

If AVAS is to signal events to the server system, this can only occur via the start of jobs and procedures (priority is given to FUNCTION=P, TYPE=EXX), which are always triggered by AVAS. The event itself must be set by the user job in the server system.

If the server system is to signal events to AVAS, it must start a job in BS2000 which modifies the value or status of a condition description via the batch statement MODIFY-COND-DESCR, or modifies the contents of a job variable which is queried via COND-TYPE=JVA.

Glossary

ABLDAT

Link name for the run control file.

ABLDUP

Link name for the copy of the run control file.

automatic restart

AVAS automatically restarts an errored net at the relevant restart point.

AVAS report

Evaluation of the AVAS production plan and the AVAS journal file according to predefined criteria.

AVAS-JV interface

Executing jobs are generally monitored using a task job variable.

In the case of the AVAS-JV interface monitoring takes place using the same job variable, but this is not supplied with values as the task job variable by BS2000, but appropriate values must be supplied by the user.

AVAS-SYSTEM-LIBRARY

Name of a central AVAS library.

AVAS-USER-LIBRARY

Name of an AVAS user library.

BATCH statements

BATCH statements are selected statements which can be entered in procedures.

calendar

List of days, delimited by a start date and an end date. Each day is assigned a day of the week. Each day can also be assigned one or more symbolic start dates. Each user group is assigned a standard calendar. Nets can also be assigned to a specific calendar. Calendars are stored and managed under unique names in the CALLIB library.

CALLIB

Link name for the calendar library.

condition

Prerequisite for starting a net or an index level of a net; see also **CONDITION-TYPE**.

condition description

Part of the **ABLDAT** for conditions of type **NET/JOB/RES/VAL**; a record contains all the necessary information for the **CONDITION-TYPE** concerned.

CONDITION-JVA-NAME

Name of the job variable which has to assume a desired value at a certain position in order to satisfy the structure variable's condition.

CONDITION-TEXT

Brief description of the structure element.

CONDITION-TYPE

The type of a structure element which specifies a condition (**NET/JOB/RES/VAL/JVA/TIM/TRA**). Accordingly the following terms are used: Condition **NET**, Condition **JOB** (also includes conditions of the type **TYP=TRA**), Condition **RES(OURCE)**, Condition **VAL(UE)**, Condition **JVA**, Condition **TIM**

CONDITION-VALUE

Value of a condition description of a job variable.

configuration file

The configuration file is used to assign a real connection between the **BS2000** system and a server system to the symbolic name of a connection to a server system (**SERVER-NAME**).

dependency

Situation where a net or an index level of a net is waiting for an event to occur before the start can take place.

DELAY-SOLUTION

Measures to be taken if a net is not started at the appropriate time.

DOCLIB

Link name for the library containing the documentation elements.

DOCSYS

Link name for the central library of documentation elements.

DUE key

Same as ENTER key (qv).

EARLIEST-START

Resolved start time provided for the net. It is formed with CREATE-PLAN-NET and can be modified by means of MODIFY-PLAN-NET and SUBMIT-NET. It is a search criterion when nets are selected via the operand PERIOD-NAME, but it is **not** part of a name.

ENTER-FILE

This file is used to store the JCL of jobs not managed by AVAS (jobs with JOB-TYPE=EXT).

ENTER-PARAMS

Specifies whether values should be assigned for the ENTER parameters from the net description or from the jobs.

ENTER key

Triggers transfer of the data in a mask to AVAS.

EXTERNAL-FILE

Name of an external PLAM library or SAM file as an input or output file for transferring AVAS library elements.

FILE-PASSWORD

Password for the ENTER-FILE.

FORMAT-NAME

Name of a user mask.

FT control record

Part of the net description. It describes the position of an FT request within the net as well as its parameters.

FT request

File transfer which was requested using openFT (TRANSFER-FILE command, see the “openFT User Guide” [12]).

The request is defined fully by the entries in the AVAS structure element and handled using openFT. Runtime monitoring and CONDITION handling takes place in the same way as for jobs.

FT-STATUS

Status indicator of an FT request.

FT-TEXT

Brief description of the FT request.

FUNCTION (also FU or F)

The function which a structure element performs within the net description. FUNCTION can take on the following values:

J (Job)	The specification required to execute a job
F (File Transfer)	The specification required to execute an FT request
P (Procedure)	The specification required to execute an S procedure
S (Start)	Description for starting a subnet
A (Add)	Create a condition description
M (Modify)	Amend a condition description
D (Delete)	Delete a condition description
C (Compare)	Test a condition descriptions
W (Wait)	Wait until a date and time

hypernet

A hypernet is a net with structural elements of type FU=S. Subnets can be run and monitored in it.

index level

Hierarchy level of the net structure. The structure elements of one index level are processed or brought to execution simultaneously. The index levels are processed consecutively in ascending order if the index level was terminated normally. If errors occur, processing is interrupted at the end of the index level involved. The sequence in which an index level is processed (or waited for) can be broken by specifying a synchronization index (SYNC-INDEX).

JCL element

Externally stored JCL of one or more jobs or S procedures. It is reincorporated in the job or S procedure via an AVAS statement within the framework of parameter modification.

JCLLIB

Link name for the library of jobs, S procedures, server jobs and JCL elements.

JCLSYS

Link name for the central library of jobs, S procedures, server jobs and JCL elements.

JMDLIB

Link name for the library of modified jobs, S procedures and server jobs.

JMDSYS

Link name for the central library of modified jobs, S procedures and server jobs.

Job

BS2000 job, FT request (without JCL), S procedure or server job

JOB

BS2000 command sequence beginning with '/SET-LOGON-PARAMETERS' and ending with '/EXIT-JOB' or '/LOGOFF'. It is also possible to incorporate special AVAS statements in the command sequence.

JOB-ACCOUNT

Parameter for the ENTER call of the job, S procedure or server representative.

JOB-CAT

Catalog ID of a SLAVE processor or server name of a remote processor.

JOB-CLASS

Parameter for the ENTER call of the job, S procedure or server representative.

job control record

Part of the net description. It describes the position of the job or S procedure within the net as well as its parameters.

JOB-DOC

Name of the documentation element for a job, an S procedure or a server job.

JOB-INDEX

Index level of a job, an S procedure or a server job in the net.

JOB-LOG

Job execution logs stored under AVAS.

JOBMAP

Link name for the library of user masks related to individual jobs or S procedures.

JOB-PARAMETER

Parameter for the ENTER call of a job, an S procedure or a server representative.

JOB-STATUS

Status indicator of a job, an S procedure or a server job.

JOB-TEXT

Brief description of the job, S procedure or server job.

JOB-TYPE

Indicates how the JCL of a task (job, S procedure) is managed in the AVAS system and how the task is monitored via a job variable (STD/MOD/EXT/ EXX).

journal file

Output medium for logging the actions of the user on the AVAS system as well as the run control system activities.

JRLDAT

Link name for the emergency journal file.

JRNDAT

Link name for the journal file.

JVA-LENGTH

Length of the value of a job variable.

JVA-NAME

Name of a job variable.

JVA-PASSWORD

Password for a job variable.

JVA-POSITION

Start position of the value in the job variable.

LATEST-START

Latest start time for the net or a task in the net.

LIFE-TIME

Time span relative to PLAN-START; indicates how long the event 'end of net' or 'end of job' is to remain valid and recognizable.

LOG

Parameter for the ENTER call of the job, S procedure or AVAS agent AVSSINCM.

LOGSYS

Link name for the central job log library (AVAS pool).

M

Column in the AVAS system masks where marks are entered to select elements.

net

Set of consecutive jobs, S procedures or server jobs whose execution is structured and defined in accordance with their logical and temporal interdependencies.

NET-ACCOUNT

Default value for JOB-ACCOUNT.

NET-CAT

Catalog identifier of a slave processor or server name of a remote processor.

NET-CLASS

Default value for JOB-CLASS.

net control record

Part of the net description. It contains parameters valid throughout the net.

NET-DELAY-SOLUTION

Action for untimely net start.

net description

Structure description of the net and information on the contents and sequence of processing steps within a net. It is created by production planning.

NET-DOC

Name of the documentation element for a net.

NETLIB

Link name for the net description library.

NET-LOG

Default value for LOG.

NETMAP

Link name for the library of user masks related to nets.

NET-NAME

Name of the net.

NET-PARAMETER

Default value for JOB-PARAMETER.

net processing

Processing of the net description (create, modify, copy, delete, display).

NET-STATUS

Status indicator for the net.

NETSYS

Link name for the central net description library.

NET-TEXT

Brief description of the net.

NET-TYPE

Control variable for serializing the processing of two or more like-named nets (but with different start times).

NET-USER

Default value for USER.

NPRLIB

Link name for the library of planned nets.

operation

Short string used to control the dialog in masks. It is entered via the CMD: field in the mask.

OUT-OF-PLAN report

This report lists nets which have exceeded a defined delay and/or which have a selected status.

PERDAT

Link name for the period file.

period

Interval delimited by start and end times. Periods are stored and managed under unique names in a separate collection of data.

PLANNED-NET-MODIFICATION report

This report lists nets which have been modified after production planning.

planning period

Time span for which selected nets are scheduled to run. It is specified via PERIOD-NAME. Those nets are processed whose symbolic start dates are entered in the calendar section corresponding to the planning period.

PLAN-START

Start time envisaged for the net during the planning operation. It is made part of the name of the nets in the NPRLIB during the planning operation and cannot be modified thereafter. The envisaged start time is modified after the planning operation using EARLIEST-START.

production plan

“Directory” for the library of planned nets, i.e. a list of the planned nets with individual resolved start times and production status.

release period

Time span during which two or more nets can be released together. It is set by the PERIOD-NAME operand. Those nets are processed whose resolved start times lie in the release period.

REPORT generator

Process for creating AVAS reports.

REPORT statements

Instructions to the REPORT generator.

resolved dependency

The event on which the start of a net or an index level depends has occurred. The condition of a waiting net has been satisfied.

resolved start time

This always consists of a date and a time of day and means that the symbolic start date of a net has been replaced by a real date. This operation takes place at the “production planning” stage.

RESTART-INDEX

Index level at which restart is to take place.

restart job

Additional job that must be performed following an interrupt before normal processing can resume.

RESTART-NAME

Name of the structure element at which any required restart is to take place.

RESTART-NET

Restart of a previously interrupted net.

restart statement

Facilities incorporated in the JCL for restart following an error.

RESTART-TYPE

This indicates whether restart statements are to be processed in a restart situation.

RESTART-VARIANT

This indicates which of the three possible restart variants is to be processed. Description of the three restart variants, consisting of RESTART-TYPE, RESTART-INDEX, RESTART-NAME.

run control file

File containing all the information needed to control execution of linked run control systems. At the "release for production" stage, the structure description of the planned net and the corresponding JCL are added to the run control file.

run control system (RCS)

This consists of an AVAS run control and monitoring routine with the name defined at generation time (RUN-CONTROL-SYSTEM), as well as all nets assigned via the run control file and the jobs brought to execution within the nets.

RUN-CONTROL-SYSTEM

Name of the run control system.

SELECT-TURNUS

Processing cycle (monthly, weekly, daily, etc.), which is always assigned a numeric value. All jobs and conditions whose SELECT-TURNUS is 0 or equal to the SELECT-TURNUS of the net control record are taken into account for processing. SELECT-TURNUS is also used as a selection criterion when defining net run variants within the framework of net planning.

SERVER-NAME

SERVER-NAME is a symbolic name for the host on which and the user ID under which a server job is to run.

standard net

Net description generated by production planning, including all job descriptions assigned to the net.

start parameter

Start parameter of a net: LATEST-START, DELAY-SOLUTION, LIFE-TIME.
Start parameter of a structure element: LATEST-START, DELAY-SOLUTION

static jobs/server jobs

Jobs/server jobs in the JMDLIB which may be assigned to two or more nets.

structure element

Individual element of a net structure for starting a task, editing a condition task or querying a condition.

subnet

A subnet is a net that is started as a structural element of a hypernet. A subnet cannot start other subnets.

symbolic start dates

Dates for the net start time, assigned when standard nets are generated and processed. They are entered in the net parameter PLAN-START. The AVAS administrator enters the symbolic start dates in the calendar and also takes charge of their further management with regard to the calendar. Symbolic start dates are also selection criteria for defining net run variants during net planning.

SYMDAT-NAME

Name of a symbolic start date.

SYNC-INDEX

Synchronization index in the net description.

task

BS2000 job or SDF-P S procedure

temporary jobs/server jobs

Jobs with the name <netname_jobname> in the JMDLIB which can be assigned uniquely to a net.

USER

Parameter for the ENTER call of the job, the S procedure or the AVAS agent AVSSINCM.

user group

Group of users who access public AVAS libraries.

USER-PARAM-FILE

User file with current values of the net run parameters supplied to the jobs of a net during production for the planned process.

Related publications

You will find the manuals on the internet at <http://manuals.ts.fujitsu.com>. You can order printed versions of manuals which are displayed with the order number.

- [1] **AVAS (BS2000)**
AVAS Functions and Tables
User Guide
- [2] **AVAS (BS2000)**
AVAS Statements
User Guide
- [3] **ASSEMBH (BS2000)**
User Guide
- [4] **BS2000 OSD/BC**
Utility Routines
User Guide
- [5] **BS2000 OSD/BC**
Introductory Guide to DMS
User Guide
- [6] **BS2000 OSD/BC**
Introductory Guide to Systems Support
User Guide
- [7] **BS2000 OSD/BC**
Commands
User Guide
- [8] **MAREN (BS2000)**
MTC Management
User Guide
- [9] **SDF (BS2000)**
SDF Dialog Interface
User Guide

Related publications

- [10] **IMON (BS2000)**
Installation Monitor
User Guide
- [11] **interNet Services**
User Guide
- [12] **openFT for BS2000**
Enterprise File Transfer in the Open World
User Guide

Index

(#SCR#) or (#SCR#,n) for output tapes 285

(#VOL#) or (#VOL#-n) for input tapes 285

* - authorization 40

/INFORM-PROGRAM command

for operating the run control system 127

#AVA# 52

#AVD# 52

#AVJ# 52

#AVM# 52

#AVS# 52

#RA 52

#RI 52

#RU 52

0-authorization 40

1-authorization 40

A

ABLDAT 32, 523

ABLDUP 32, 523

access tasks 61

abnormal termination 114

central 23

normal termination 113

starting 108

terminating 125

accessing

files 61

libraries 61

the emergency journal file 61

the journal file 61

the period file 61

the run control file 61

adjusting start procedure

for AVSSINCM 516

for run control system 517

assignment of CC exits to EXIT module 78

assignment of mask libraries to job libraries

definition 39

assignment of mask libraries to net libraries

definition 39

assignment of net libraries to production plans

definition 39

authorizations of a user, changing 40

automatic restart 523

AV03EXTV module 79

avak 42

avak-use 36

AVAS administrator 105

AVAS agent

AVSSINCM 470, 472

AVAS configuration file

editing 477

AVAS files, creating 61

AVAS libraries 31, 64

AVAS pool 138, 156

AVAS report 303

AVAS reports 523

AVAS run control system, operation 106

AVAS server

changing status 501

status 501

AVAS server AVSRECV 470

AVAS server interfaces 465

AVAS statements, symbolic names 52

AVAS system

generating 23

installing 21

- AVAS system parameters 53, 56
- AVAS users 33
- AVAS variables, symbolic names 52
- AVAS version changes 66
- AVAS, program interface 399
- AVAS-EXIT module 77
- AVASEXKO macro 83
- AVAS-JV interface 468, 519, 523
- AVASNET
 - RECORD=A3 390
 - RECORD=C3 388
 - RECORD=D3 392, 394, 396
 - RECORD=J1 381
 - RECORD=J2 383
 - RECORD=J3 386
 - RECORD=M3 391
 - RECORD=N1 372
 - RECORD=N2 374
 - RECORD=N3 376
 - RECORD=N4 378
 - RECORD=N5 380
 - RECORD=W3 393
- AVASNET macro 369
 - expansion 372
- AVAS-QUER 21, 239
 - CREATE-FORMAT statement 252
 - CSV format 239
 - database 239
 - entering statements 247
 - error handling 279
 - importing data into the database 259
 - installing 21
 - SIGNON statement 251
 - start procedure 242
 - structure of the database 260
 - structure of the output file 248
- AVAS-SV-BS2
 - deinstalling 469
 - installing 469
- AVASSYS (link name for SYSPAR) 108
- AVAS-SYSTEM-ID 28
- AVAS-SYSTEM-LIBRARY 523
- AVAS-USER-LIBRARY 523
- AVEX0001 (CC exit) 71
- AVEX0002 (CC exit) 71
- AVEX0101 (CC exit) 85
- AVEX0102 (CC exit) 72, 86
- AVEX0401 (CC exit) 72
- AVEX0402 (CC exit) 73
- AVEX0403 (CC exit) 73
- AVEX2001 (CC exit) 74
- AVEX6601 (CC exit) 74
- AVEX6602 (CC exit) 74
- AVEX6801 (CC exit) 75
- AVEX6802 (CC exit) 75
- AVEX7101 (CC exit) 75
- AVEX7102 (CC exit) 76
- AVEXSVV (macro) 79
- AVS.GENPERIOD procedure for standard periods 347
- AVS010 (AVSSURF mask) 502
- AVSASSAD (macro) 415
- AVSASSAN (macro) 419
- AVSASSBC (macro) 370, 426
- AVSASSBO (macro) 430
- AVSASSBW (macro) 431
- AVSASSCD (macro) 433
- AVSASSCE (macro) 435
- AVSASSJV (macro) 437, 440
- AVSASSRT (macro) 442
- AVS-AVAK-FILE 480
- AVS-AVAK-FILE-PASSWORD 481
- AVS-AVAK-NAME 480
- AVS-AVAS-VERSION 481
- AVSCMDC key, define 457
- AVSCOBAD (COPY element) 447
- AVSCOBAN (COPY element) 448
- AVSCOBBC (COPY element) 452
- AVSCOBBO (COPY element) 456
- AVSCOBQQ (COPY element) 457
- AVSCOBWW (COPY element) 458
- AVSCOBBD (COPY element) 459
- AVSCOBCE (COPY element) 460
- AVSCOBJD (COPY element) 461
- AVSCOBJN (COPY element) 462
- AVSCOBRT (COPY element) 463
- AVS-CONF-FILE 480
- AVS-CONF-PASSW 480

- AVS-DEF-COMM-PASSW 482
 - AVS-DEF-DIRECTORY 482
 - AVS-DEF-HOST-NAME 482
 - AVS-DEF-PORT-NUMBER 482
 - AVS-DEF-SV-PASSW 482
 - AVS-DEF-USER-ID 482
 - AVS-JOB-CLASS 481
 - AVS-JOB-ID 480
 - AVS-JOB-INDEX 480
 - AVS-JOB-LOG 481
 - AVS-JOB-NAME 480
 - AVS-JOB-PARAMETER 481
 - AVS-MONJV-PASSW 480
 - AVS-NET-NAME 480
 - AVSRECV 491
 - AVAS server 470
 - starting in server system 518
 - AVS-SERVER-NAME 480
 - AVS-SERVER-PASSW 481
 - AVS-SERVER-VERSION 481
 - AVSSINCM
 - adjusting start procedure for 516
 - AVAS agent 470, 472
 - start parameter (see start parameter for AVSSINCM) 480
 - start procedure 478
 - AVSSURF 491, 495
 - authentication 502
 - displaying job data 507
 - displaying log file 510
 - displaying script file 510
 - displaying server overview 505
 - displaying server table 499
 - editing Watchdog configuration file 503
 - encrypted communication 495
 - setting up a connection 498
 - start parameters 496
 - starting 496
 - terminating 497
 - AVS-SVDOG-JV 482
 - AVS-SV-PARAM 481
 - AVS-SYSTEM-ID 481
 - AVS-USER-ACCOUNT 481
 - AVS-USER-ID 481
 - AVS-USER-PASSW 481
 - avuser 37
 - AVW001 (AVSSURF mask) 499
 - AVW002 (AVSSURF mask) 503
 - AVW010 (AVSSURF mask) 505
 - AVW030 (AVSSURF mask) 507
 - AVW031 (AVSSURF mask) 510
- B**
- batch mode, optimizing 164
 - BATCH statements 336, 523
 - BS2000 password, visible 37
- C**
- calendar 35, 523
 - calendar library, CALLIB 61
 - CALLIB 30, 61, 169, 524
 - CC exit 71
 - AVEX0001 71, 84
 - AVEX0002 71, 84
 - AVEX0101 72, 85
 - AVEX0102 72, 86
 - AVEX0401 72, 87
 - AVEX0402 73, 89
 - AVEX0403 73, 90
 - AVEX2001 74, 92
 - AVEX6601 74, 93
 - AVEX6602 74, 95
 - AVEX6801 75, 96
 - AVEX6802 75, 98
 - AVEX7101 75, 100
 - AVEX7102 76, 101, 102
 - CC routines
 - connecting 79
 - incorporating in processing 71
 - information area 81
 - linking in AVAS 77
 - central access tasks (ZDs) 23, 61
 - defining 28
 - starting 108
 - system parameters 28
 - CENTRAL task
 - starting 143, 158
 - terminating 145, 160

- CERTIFICATE-CHAIN-FILE 496
- CNTAPC 480
- CNTAPP 480
- COMAREA (communication area) 406
- command /REMARK #UID# 286
- comm-pw 470, 476
- communication area 83
 - (COMAREA) 406
 - defining 370, 426, 452
 - structure of 406
- communication password comm-pw 470, 476
- condition 524
- condition description 524
- CONDITION-JVA-NAME 524
- CONDITION-TEXT 524
- CONDITION-TYPE 524
- CONDITION-VALUE 524
- CONFIG, operation for updating the server environment 133
- CONFIG-FILE-NAME 122
- CONFIG-FILE-PASSWORD 122
- configuration
 - data as default parameters 470
- configuration file 524
 - areas for entries 475
 - creating 516
 - editing 477
 - example 477
 - of server system 475
- connecting CC routines 79
- contents of job variable for server jobs 484
- control statement NETC 42
- CONTROL-JV 496
- controlling
 - interoperation with server systems 121
 - released nets via /INFORM-PROGRAM 128
- control-time 42
- coupling AVAS with MAREN 281
- create
 - EXIT module 80
 - variable period 344
 - volume lists 289
- CREATE-FORMAT (AVAS-QUER statement) 252
- CREATE-GENPAR 56
- CREATE-PLAN-NET 399
- CREATE-PROD-NET 399
- creating and formatting AVAS files 61
- creating emergency journal file 32
- creating the configuration file 516
- D**
- data area, identification of 195
- database queries for AVAS-QUER 239
- default value
 - DEFAULT-OPERATOR-START 49
 - defining for net description 45
- define
 - AVSERLTC key 457
 - condition description in run control file 435
 - journal directory 437
 - journal record header 440, 462
 - key for AVSCMDC + AVSERLTC 457
 - net contents directory entry for the run control file 415
 - net directory run control file 447
 - net directory specification 461
 - net structure data 419, 448
 - result area header 430, 456
 - results area 442
- defining default values
 - for displaying hypernets 49
 - for displaying net structures 47
 - for job editing 47
 - for net planning 46
 - for net release 49
 - for restart processing 49
 - net description 45
- definition
 - for function control 51
 - of assignment of mask libraries to job libraries 39
 - of assignment of mask libraries to net libraries 39
 - of assignment of net libraries to production plans 39
 - of function authorizations 40
 - of production plans 38

- definition (cont.)
 - of run control system 42
 - of user group 33, 35
 - overview processing 45
 - periods using batch statements 344
 - system variables of the users 41
 - users 37
- deinstalling AVAS-SV-BS2 469
- DELAY-SOLUTION 524
- deleting, nets from run control and journal files 170
- dependency 524
 - resolved 531
- diagnostic documentation, outputting 106
- dialog, break 37
- directory record 140
- displaying server table 499
- DOCLIB 31, 169, 524
- DOCSYS 31, 524
- documentation libraries DOCSYS and DOCLIB 62
- DUE key 525
- E**
- EARLIEST-START 525
- EDT procedure for JOBLOG log 107
- elements
 - displaying from system libraries 35
 - of system user group 34
- emergency journal file
 - creating 32
 - JRLDAT 63
- emergency journals, merging in 125
- encryption 495
- encryption of passwords 24
- END (BATCH statement) 357
- END (reorganization statement) 177
- ENTER file, for access tasks 108
- ENTER key 525
- ENTER-FILE 525
- entering
 - VSNs for input tapes 285
 - VSNs for output tapes 287
 - VSNs in JCL 285
- ENTER-PARAMS 525
- error handling in AVAS-QUER 279
- event signaled by server system to AVAS 522
- execution of the REPORT generator 306
- EXIT module, creating 80
- EXIT modules 77
- external creation, of nets 369
- EXTERNAL-FILE 525
- F**
- F and K keys, defining 44
- F# 52
- file extension for AVAS files 63
- file name
 - of journal file 32
 - of run control file 32
- FILE-PASSWORD 525
- FORMAT-NAME 525
- formatting the run control file and journal file 63
- FT control record 525
- FT request 525
- FT-STATUS 525
- FT-TEXT 526
- function authorizations, definition 40
- function control, definitions for 51
- function keys 44
- FUNCTION of a structure element 526
- function record 141
- further processing, output file 298
- G**
- generate GENPAR file 59
- generation file
 - creating 24
 - GENPAR 61
- generation of the AVAS system 23
- generation parameters, create 56
- H**
- history
 - job description 72
 - net description 72

- history data records
 - record structure 232
 - structure and contents 226
- history file
 - keys 225
- HISTORY-DELETE-DATE (reorganization statement) 179
- HISTORY-DELETE-OPTION (reorganization statement) 180
- HISTORY-KEEP-RECORDS (reorganization statement) 181
- HISTORY-SAVE-OPTION (reorganization statement) 182
- HLPLIB 38
- hostname/portno 475
- HOSTWAIT 488
- HOSTWAIT (for server jobs) 471
- HSTSYS 169
- hypernet 526

- I**
- identification, of data area 195
- IMON 21
- index level 526
- information area, CC routines 81
- INFORM-PROGRAM command 127
- input file, further processing 297
- input in job variable 124
- input in job variables 130
- in-service message 491
- installation
 - AVAS 21
 - AVAS-QUER 21
 - AVAS-SV-BS2 21, 469
- integration of MARENAV 79
- interface function code 457

- J**
- JCL element 526
- JCLLIB 62, 526
- JCLnnn 31, 35, 169
- JCLSYS 31, 62, 526
- JMDLIB 31, 62, 169, 527
- JMDSYS 31, 62, 527

- job 527
 - control remote BS000 487
 - runtime logs 138
 - terminating 506
- job control record 527
- job data 507
- job description, history 72
- job variable
 - input in 124, 130
 - interface 468
- job variable for server jobs, contents 484
- JOB-ACCOUNT 527
- JOB-CAT 527
- JOB-CLASS 527
- JOB-DOC 527
- JOB-INDEX 527
- JOB-LOG 527
- JOBLOG log, EDT procedure 107
- JOBMAP 62, 527
- JOB-PARAMETER 527
- JOB-STATUS 528
- JOB-TEXT 528
- JOB-TYPE 528
- journal file 528
 - deleting nets 170
 - file name of 32
 - formatting 63
 - JRNDAT 63
 - read access 402
 - reorganization 172
- journal listings, outputting 191
- journal records
 - editing for print file 191
 - of statements 198
 - record structure of fixed portion 204
 - record structure of variable-length portion 207
 - saving 190
 - structure of 194
- JRLDAT 32, 63, 528
- JRNDAT 32, 63, 528
- JV file 470
- JVA-LENGTH 528
- JVA-NAME 528

JVA-PASSWORD 528

JVA-POSITION 528

javak 42

JVCENTRAL 28

JVPLAMZD 28

JVUPAMZD 28

K

K and F keys, defining 44

KEY-FILE 496

keyword

AVAK 42

FUTABnnn 40

JOB-MAP 39

NET-MAP 39

NET-PRO 39

PRODTABnnn 38

USER 37

L

LATEST-START 528

libraries

available throughout the system 64

for jobs and JCL elements, JCLSYS and
JCLLIB 62

for production jobs, JMDSYS and JMDLIB 62

licensing regulations 16

LIFE-TIME 528

link name

AVASSYS 108

for AVAS-QUER 241

for AVAS-QUER tables 22

link name for parameter file SYSPAR
(AVASSYS) 108

link statements 77

for Exit module 80

linking, CC routines in AVAS 77

list

L06001 330

L06002 331

LOG 528

log

collecting 158

opening 157

log data library, LOGSYS 62

log data, save 193

log element 140

log entries, saving 193

log file 510

current size 508

delete attribute 508

displaying 508, 509

displaying threshold values for file
transfer 508

modifying attribute 509

log message record 141

log records 142

length 143

log status (overview) 155

logs

as collection of data 139

as JOBLLOG record 139

collect 143

edit 154

element structure 140

measures for optimizing collection of
store 139

LOGSYS 31, 169, 529

M

M (column) 529

macro

AVASEXKO 81

AVASNET 369

AVEXSVV 79

MANAGE-SERVER (privilege) 501

manage-servers 37

MAPLIB 62

MAPnnn 38, 169

MAREN, coupling with AVAS 281

MARENAV 281

integrating 79

messages 299

mask libraries, MAPLIB 62

MAX-BATCH-PROCESS 29

merging in emergency journals 125

MODIFY-COND-DESCRIPTION 399

module

AV03EXTV 79
MARENAV 281

monitoring

batch tasks via job variables 135
tasks via job variables 135
the run control system job variable 42
via JVs 135

MONJV-PASSWORD 115

MONJV-RDPASS 116

mscf-control-time 43

MSG7-DEST-DIALOG 30

N

N# 52

name

of a system user group 34
user group 35

net 529

net control record 529

net creation in external library 369

net description 11, 529

history 72

net libraries, NETSYS and NETLIB 62

net processing 530

NET-ACCOUNT 529

NETC control statement 42

NET-CAT 529

NET-CLASS 529

NET-DELAY-SOLUTION 529

NET-DOC 529

NETLIB 62, 529

NETLIB record descriptions 369

NET-LOG 529

NETMAP 62, 529

NET-NAME 530

NETnnn 31, 35, 169

NET-PARAMETER 530

NET-STATUS 530

NETSYS 31, 62, 530

NET-TEXT 530

NET-TYPE 530

NET-USER 530

normal termination, of access tasks 113

normal user 40

notational conventions 15

NPRLIB 169, 530

NPRnnn 31

O

OpenSSL 495

operand

MACID (generation of names) 415

MF (code generation) 415

PREFIX (generation of names) 415

operation 530

copying system file 131

for administering signed-on users 124

for terminating access tasks 125

of AVAS run control system 106

of run control system 127

reassigning system files 131

terminating access task 130

terminating access tasks 132

OUTAREA 408

OUT-OF-PLAN report 303, 530

output file, further processing 298

outputting, journal listings 191

overview

of batch statements 336

of journal records 198

of record keys 196

P

P# 52

password encryption 24

password, visible 37

PASSWORD-ENCRYPTION 28

password-visibility 37

PERDAT 30, 61, 169, 530

performing volume checks 281

period 530

creating variable 344

variable (real dates) 347

variable date 346

with wildcards 344

period file, PERDAT 61

PLAM-ZD 61

- PLANNED-NET-MODIFICATION report 303, 531
 planning period 531
 PLAN-START 531
 PORT 496
 prefix (emergency journal file) 32
 privileged user 40
 procedure, AVS.GENPERIOD 347
 processing elements, in system libraries 34
 processor table, updating 128
 production
 job library 38
 plans, defining 38
 production execution 12
 production monitoring 12
 production net libraries NPRLIB 62
 production net library 38
 production plan 11, 531
 production preparation 11
 program interface 399
 calling for Assembler 412
 example in Assembler 413
 example in COBOL 445
 statements 399
- Q**
- queues 164
- R**
- read access, journal file 402
 READ-AVAS-LIBRARY 399
 Readme file 14
 real dates for a variable period 347
 record description NETLIB 369
 record keys, journal 196
 references to other publications 15
 register conventions 78
 regulations, licensing 16
 release
 for production 12
 release period 531
 released nets
 controlling 128
 reorganization procedure 170
 reorganization statement
 AVAS-SYSTEM-ID 176
 AVAS-USER-ID 176
 DELETE-JOB-LOG 177
 DELETE-PLAN-NET 177
 END 177
 FUNKTION 178
 HISTORY-DELETE-DATE 179
 HISTORY-DELETE-OPTION 180
 HISTORY-KEEP-RECORDS 181
 HISTORY-SAVE-OPTION 182
 JOBLOG-DELETE-DATE 182
 JOBLOG-DELETE-STATUS 183
 JOBLOG-SAVE-LINK 184
 JOBLOG-SAVE-OPTION 183
 NET-NAME 184
 OUTPUT-FILE 184
 OUTPUT-LINK 185
 PERIOD-NAME 185
 START 186
 UPDATE-JRIDAT 186
 USER-PASSWORD 186
 ZDD-PASSWORD 187
 ZDL-PASSWORD 188
 reorganization, execution 172, 188
 reorganizing
 the journal file 172, 173
 the log file 173
 the run control file 172
 the user files 169
 REPORT generator 303, 531
 REPORT generator execution 306
 REPORT statements 303, 531
 resolved dependency 531
 resolved start time 531
 restart
 automatic 523
 job 531
 statement 532
 RESTART-INDEX 531
 RESTART-NAME 532
 RESTART-NET 399, 532
 RESTART-TYPE 532
 RESTART-VARIANT 532

- result area (OUTAREA) 408
- routing-code 42
- run
 - on any server systems 468
- run control
 - for server jobs 466, 468, 470
- run control file 532
 - ABLDAT 63
 - and journal file formatting 63
 - copy 32
 - file name 32
 - file name of 32
 - read access to program interface 402
- run control system 532
 - definition 42
 - operating 127
 - password assigned by 115
 - start parameters 115
 - starting 115
 - terminating 122
- run monitoring
 - for server jobs 466, 468, 470
- run monitoring system, for server systems
 - via AVAS 520
- RUN-CONTROL-SYSTEM 532
- runtime logs
 - of jobs 156
 - opening 156
- runtime logs of jobs 138

- S**
- S# 52
- saving, journal records 190
- script file 510
 - displaying 508
- SECURE-PORT 496
- security, in the AVAS system 106
- selection criteria (OOP report) 304
- SELECT-TURNUS 532
- serialization argument to the UPAM-ZD 165
- serialization run 112, 165
- server
 - failure 512
 - temporarily lock 134
- server configuration file 122, 514
 - creating 475
- server entry
 - creating 504
 - deleting 504
 - modifying 504
- server interface (process) see AVSSURF 495
- server job
 - starting and monitoring (preparations) 514
 - static 533
 - status and additional status 473
 - temporary 533
- server job file 484
- server jobs
 - run control 466, 468, 470
 - run monitoring 466, 468, 470
- server monitor 123
 - cyclical check 490
 - functions 490
 - receiving jobs 491
 - receiving messages 491
 - update 133
- server monitor (process) 106, 488
- server name 515
- server overview 505
- server systems
 - start parameters 121
- SERVER-JV-LINK 121
- SERVER-NAME 532
- server-name 36, 470, 475, 515
- SERVER-PROCEDURE-NAME 121
- SERVER-PROCEDURE-PASSWORD 122
- server-pw 470, 475, 515
- shift procedure MARENAV 294
- SIGNAL program 139
 - starting 146
 - terminating 148
- signoff 502
- SIGNON 399
- signon 502
- SIGNON (AVAS-QUER statement) 251
- signon information in job variables 107
- sort criteria (OOP report) 304
- SOUT-APPLICATION-JV 30

- SOUT-APPLICATION-NAME 30
- SOUT-APPLICATTION-PASSWORD 30
- standard net 533
- standard parameter
 - AVS-DEF-SERVER-VERS 482
- standard period 344
 - create using procedure 347
 - deleting 355
- START (reorganization statement) 186
- start dates symbolic 533
- start file for AVSSINCM 470
- start parameter for AVSSINCM
 - AVS-AVAK-FILE 480
 - AVS-AVAK-FILE-PASSWORD 481
 - AVS-AVAK-NAME 480
 - AVS-AVAS-VERSION 481
 - AVS-CONF-FILE 480
 - AVS-CONF-PASSW 480
 - AVS-JOB-CLASS 481
 - AVS-JOB-ID 480
 - AVS-JOB-INDEX 480
 - AVS-JOB-LOG 481
 - AVS-JOB-NAME 480
 - AVS-JOB-PARAMETER 481
 - AVS-JOB-TYPE 480
 - AVS-MONJV-PASSW 480
 - AVS-NET-NAME 480
 - AVS-SERVER-NAME 480
 - AVS-SERVER-PASSW 481
 - AVS-SERVER-VERSION 481
 - AVS-SVDOG-JV 482
 - AVS-SV-PARAM 481
 - AVS-SYSTEM-ID 481
 - AVS-USER-ACCOUNT 481
 - AVS-USER-ID 481
 - AVS-USER-PASSW 481
 - CNTAPC 480
 - CNTAPP 480
 - USERID 480
- start parameter for AVSSURF
 - CERTIFICATE-CHAIN-FILE 496
 - CONTROL-JV 496
 - KEY-FILE 496
 - PORT 496
- SECURE-PORT 496
- STYLE-SHEET 496
- SVDOG-HOST 496
- SVDOG-PORT 496
- USE-SSL 496
- start parameter run control system
 - CONFIG-FILE-NAME 122
 - CONFIG-FILE-PASSWORD 122
 - controlling log processing 120
 - GENERATE-JOB-LOG 120
 - interoperation with server systems 121
 - SERVER-PROCEDURE-NAME 121
 - SERVER-PROCEDURE-PASSWORD 122
 - TRANSFER-PROCEDURE-NAME 120
 - TRANSFER-PROCEDURE-PASSWORD 121
- start procedure
 - AVSSINCM 478
 - AVSSURF 496
- start time, resolved 531
- starting
 - CENTRAL task 158
 - the access tasks 108
 - the run control system 115
- statements, program interface 399
- static jobs 533
- static server jobs 533
- status
 - log entries 142
 - log entries, priority 143
 - of server job (SV status) 471
- structure
 - of communication area 406
 - of journal records 194
 - task job variable 484
- structure element 533
- STYLE-SHEET 496
- SUBMIT-NET 399
- subnet 533
 - serialization 112
- SVDOG-HOST 496
- SVDOG-PORT 496

- symbolic names
 - of AVAS statements 52
 - of AVAS variables 52
- symbolic start dates 533
- Symdat 261
- symdat 260
- SYMDAT-NAME 533
- SYNC-INDEX 533
- system failure, behavior 511
- system files 61
- system library elements, processing 34
- system optimization 161
 - application-specific steps 163
 - organizational measures 161
- system parameter file, managing 137
- system parameters
 - of central access tasks 28
 - of users 33
- system start 106
- system termination 106
- system user group
 - defined by keyword BKSYS 33
 - name 34
- system variables of the users, defining 41

T

- tape mount listing 293
- tape, reserving 287
- task 533
- tasks, monitoring via job variables 135
- temporary job 533
- temporary server job 533
- terminating
 - access tasks, abnormally 114
 - access tasks, delayed 113
 - access tasks, normally 113
 - run control system 122
- time scheduling 11
- TRANSFER program 139, 148
 - starting 148
 - terminating 150
- transport list 291
- type, of authorization 40

U

- ug (user group) 35
- UPAM-ZD 61
- USER 533
- user files 61
 - reorganizing 169
 - symbolic names 61
- user group 534
 - defined by keyword BK 35
 - definition 33, 35
 - name 35
- user ID reserving tape for 287
- user identification 37
- user password 37
- USERID 480
- USER-PARAM-FILE 534
- USER-PASSWORD (reorganization statement) 186
- users
 - defining authorizations 40
 - definition 37
 - general parameters 38
- USE-SSL 496

V

- variable date in periods 346
- variable period
 - real dates 347
 - wildcards 344
- version changes 66
- visibility, password 37
- volume checks, performing 281
- volume lists, creating 289
- VSN
 - enter for input tapes 285
 - enter for output tapes 287
 - enter in JCL 285
 - in a net, further processing 295

W

Watchdog configuration file, editing [503](#)

wildcard (#SCR#) [287](#)

wildcard (#VOL#) [285](#)

wildcards, for variable periods [344](#)

work area program interface

 defining [431](#), [458](#)

 suppressing message output [411](#)

 WRKAREA [411](#)

WRKAREA [411](#)

Z

ZDD-PASSWORD [144](#)

ZDL-PASSWORD [145](#)

