English

Fujitsu Software

BS2000 OS DX
Introduction to System Administration

User Guide

Valid for:
BS2000 V21.0B

Edition June 2024

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: bs2000.info@fujitsu.com.

Certified documentation according to DIN EN ISO 9001:2015

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 9001:2015.

Copyright and Trademarks

Copyright © 2025 Fujitsu

All rights reserved.
Delivery subject to availability; right of technical modifications reserved.

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

Table of Contents

Introduction to System Administration 12
1 Preface 13
1.1 Objectives and target groups of this manual 14
1.2 Summary of contents 15
1.3 Changes since the last edition of the manual 17
1.4 Notational conventions 18
1.5 Functions of system administration 19
1.6 Functions of the operating 20
1.7 Automation 21
2 System initialization and termination 22
2.1 Overview of the stages of system initialization 23
2.1.1 Determining the time on system initialization 31
2.1.2 Format of messages during system initialization 33
2.1.3 General notes on initial program loading 34
2.1.4 System initialization on SUs x86 39
2.2 Types of system initialization 40
2.2.1 FAST startup 41
2.2.2 AUTOMATIC startup 43
2.2.3 DIALOG startup 44
2.2.4 Switching modes during system initialization 49
2.2.5 Selecting the startup type and catalog reconfiguration 50
2.2.6 Example of DIALOG startup 53
2.3 System corrections 58
2.3.1 Function and structure of a REP file 59
2.3.2 REP records 62
2.3.3 Temporary backup of REPs 67
2.3.4 RMS: REP delivery and installation 69
2.4 System termination 70
2.4.1 Scheduled termination 71
2.4.2 Unscheduled termination 73
2.4.3 Automatic restart 74
3 Parameter service 75
3.1 Selecting the parameter file 76
3.2 Structure and contents of a parameter file 78
3.3 Starting the accounting system (ACCOUNT) 81
3.4 Startup of dynamic subsystem management (DSSM) 83
3.5 System time control (GTIME) 84

3.6 IOCONF parameter record 89
3.6.1 MOD-IO-UNIT statement 90
3.6.2 MOD-IO-PATH statement 92
3.7 Preset values for NK-ISAM (ISAM) 93
3.8 Defining TSN mode (JMS) 96
3.9 Memory management (MEMORY) 97
3.10 Configuration and suppressing the output of messages at consoles (OPR) . 101
3.10.1 ADD-CMD-ENTRY statement 103
3.10.2 DEFINE-CONSOLE statement 105
3.10.3 SET-CMD-CODE statement 106
3.10.4 SET-CODE statement 107
3.10.5 SET-FILTER statement 110
3.10.6 SET-MSG-SUPPRESSION statement 111
3.11 Selection of the paging area at startup (PAGING) 112
3.12 Snapshot initialization (SNAP) 114
3.13 Modifying system parameters (SYSOPT-CLASS2) 115
3.14 Modifying the IPL options (SYSOPT-IPL) 119
4 Memory management 122
4.1 Virtual address space and virtual memory classes 123
4.2 Main memory 126
4.2.1 Page management algorithm 127
4.2.2 Big pages for CISCFW compiled codes (SUs x86) 129
4.3 Paging area 132
4.3.1 Creating, assigning, releasing and deleting paging files 134
4.3.2 Selecting the paging area at startup 137
4.4 Measures to prevent saturation states 138
4.4.1 Main memory 140
4.4.2 System address space 141
4.4.3 Paging memory 142
5 Device management 144
5.1 Configuration components 145
5.1.1 Hardware units 148
5.1.2 Virtual, reconfigurable connections 151
5.2 Reconfiguration 152
5.2.1 Reconfiguration for multiprocessors 153
5.2.2 Detaching and attaching the components CPU, CHN, CTL and DVC 154
5.2.3 Effect of reconfiguration commands 156
5.2.4 Special information on magnetic tape and disk devices 158
5.3 Dynamic I/O configuration changes (SUs /390) 160
5.4 Dynamic I/O configuration changes (SUs x86) 164

5.5 Reconfiguration of extra and spare CPUs 165
5.5.1 Functionality in BS2000 native operation 166
5.5.2 Functionality as a VM2000 guest system 168
5.6 NDM - resource allocation and reservation 169
5.6.1 Task allocation of volumes 171
5.6.2 System allocation of disks 173
5.6.3 Default values for private disk allocation 174
5.6.4 Controlling resource allocation 176
5.6.5 Supporting NDM handling 178
5.7 Volume monitoring 185
5.7.1 Device selection mechanisms for tape devices 186
5.8 Parallel Access Volumes (PAV, SUs /390) 188
5.9 Administration of private volumes 192
5.9.1 Tape management with MAREN 193
5.9.2 Applications of private disks 195
5.10 IORM: Control of I/O resources 196
5.11 Displaying and checking the SAN configuration 197
6 BS2000 user management 198
6.1 Structure of a user catalog 199
6.2 Maintenance of the user catalog 200
6.3 Security concept for user catalogs 202
6.3.1 Saving the current user catalog of a pubset 203
6.3.2 Restoring a saved user catalog 204
6.3.3 Reconstruction of the SYSSRPM file 205
6.3.3.1 Performing reconstruction 206
6.3.3.2 Logging file 209
6.4 User catalog and the SMS concept 212
6.5 Bulletin file (logon information file) 213
7 POSIX user administration 216
8 File management 219
8.1 File catalog 220
8.1.1 Structure of a file catalog 222
8.1.2 Extending catalog files 226
8.1.3 Enhancing performance for catalog accesses 227
8.1.4 Removing access locks 228
8.2 ACS: Alias catalog system 229
8.3 ADM-PFA: Administrator performant file access 230
8.4 Sending BS2000 file by email 231
8.5 Unicode in BS2000 232
9 Pubset management 233

9.1 Pubset concept 236
9.1.1 Pubset types 237
9.1.1.1 Home pubset 238
9.1.1.2 User pubset (data pubset) 239
9.1.1.3 Paging pubset 240
9.1.1.4 Pubsets with large objects 241
9.1.2 Pubset names (VSN) 242
9.1.2.1 VSN in PUB notation 243
9.1.2.2 VSN in point notation 244
9.1.2.3 Double-point notation for mirror disks 245
9.1.2.4 Special cases 246
9.1.2.5 Pubset addressing 247
9.1.3 SF and SM pubsets 248
9.1.3.1 Structure of an SF pubset 249
9.1.3.2 Structure of an SM pubset 250
9.1.3.3 Pubset metadata 253
9.1.3.4 User quota concept 255
9.2 Pubset properties 258
9.2.1 Physical properties 259
9.2.1.1 Disk formats 260
9.2.1.2 Physical properties of volume sets 261
9.2.2 Global properties 262
9.2.3 User-specific properties 263
9.2.4 Volume set and volume-specific properties 264
9.3 Setting up pubsets 265
9.3.1 Setting up SF pubsets 266
9.3.2 Setting up and replacing SM pubsets 267
9.3.2.1 Setting up and extending volume sets and SM pubsets 268
9.3.2.2 Converting SF pubsets to SM pubsets 269
9.3.3 Generating pubsets with PVSREN 270
9.4 Configuring pubsets 271
9.4.1 Managing MRSCAT entries 272
9.4.2 Defining global pubset properties 273
9.4.3 Defining user-specific settings 274
9.4.3.1 Assigning the default catalog ID (default pubset) 275
9.4.3.2 Defining limit values and access rights 276
9.4.4 Defining properties of a volume set 277
9.4.4.1 Defining availability 278
9.4.4.2 Defining the performance profile 279
9.4.4.3 Defining the usage type 281
9.4.4.4 Usage restrictions 282

9.4.4.5 Defining saturation threshold values for volume sets and SF pubsets ... 283
9.4.4.6 Defining file services ... 284
9.4.4.7 Defining user quotas ... 286
9.5 Activating/deactivating pubsets ... 287
9.5.1 Activating a pubset ... 288
9.5.2 Special startup factors ... 289
9.5.3 Deactivating a pubset ... 291
9.6 Administering pubsets ... 292
9.6.1 Reconfiguring pubsets ... 293
9.6.1.1 Modifying static configuration data of SM pubsets ... 294
9.6.1.2 Modifying dynamic configuration data of SF and SM pubsets ... 295
9.6.1.3 Replacing and removing SF pubsets with the same names ... 298
9.6.1.4 Converting existing pubsets to large pubsets ... 299
9.6.2 Managing an access authorization and user catalog ... 300
9.6.3 Monitoring storage space saturation ... 301
9.6.4 Reorganizing disk storage space ... 303
9.7 Shared pubsets ... 304
9.7.1 Creating a shared pubset network ... 305
9.7.2 Monitoring a shared pubset and correcting errors ... 307
9.7.2.1 Canceling pubset locks manually ... 308
9.7.2.2 Behavior in the event of a system crash ... 310
9.7.3 Shared pubsets in the XCS network ... 311
9.8 Special information on standby pubsets ... 312
9.8.1 Creating standby pubsets ... 313
9.8.1.1 Setting up an individual standby pubset ... 314
9.9 Managing SYSEAM storage space on pubsets ... 319
9.10 Using Speed Catalog Access (SCA) ... 321
9.11 Detecting and rectifying hardware faults on pubsets ... 325
10 Net-Storage management ... 327
10.1 Overview ... 328
10.2 Terms ... 329
10.3 Interoperability on Net-Storage ... 331
10.4 Connecting BS2000 to Net-Storage ... 333
10.5 Access to Net-Storage ... 334
10.5.1 Access from BS2000 to Net-Storage ... 335
10.5.2 Access of open systems to Net-Storage ... 337
10.6 Preparations on the net server and net client ... 338
10.7 Connecting Net-Storage to BS2000 ... 339
10.8 Managing Net-Storage in BS2000 ... 341
10.9 Disconnecting Net-Storage from BS2000 ... 343

10.10 General conditions 344
10.10.1 BS2000 in general 345
10.10.2 Shared pubsets 346
10.10.3 Disk mirroring 347
10.10.4 Interoperability 349
10.10.5 Net-Storage-Volumes from BS2000 V21 350
11 Job and task management 352
11.1 Job management 353
11.1.1 Concept of job classes 360
11.1.2 Job streams, job and class scheduler 364
11.1.2.1 System job scheduler 366
11.1.2.2 Standard job scheduler 367
11.1.2.3 Class scheduler 375
11.1.3 Job management activities during system initialization 376
11.1.4 Data center job management 377
11.1.5 Interrupt-free clock resetting 378
11.1.6 JMP: reconstruction of batch jobs 379
11.2 Task management 380
11.2.1 Priority concept and queues 390
11.2.2 Allocation of resources 394
11.2.3 Management of affinity task groups (TANGRAM) 398
11.3 Time limits in BS2000 399
11.4 PCS: Performance Control System 402
12 Security 403
12.1 System access control 404
12.2 Data access control 405
12.3 Data access control in BS2000 410
12.3.1 Passwords and retention periods 411
12.3.2 File encryption 412
12.3.3 Standard access control (ACCESS/USER-ACCESS) 415
12.3.4 Basic Access Control List (BACL) 416
12.4 Privileges 418
12.4.1 Privileges of the user ID TSOS 420
12.4.2 Privileges for operating 422
12.4.3 Description of the privileges 423
12.4.4 Allocation of privileges 429
12.5 Limiting resources for users 431
12.6 Meeting security requirements using SECOS 433
13 Data backup 435
13.1 Criteria for selecting backup concepts 436
13.2 Data saving using Snapsets 440

13.2.1 Preparing Snapset mode 443
13.2.2 Saving using Snapsets 447
13.2.3 Restoration of files and job variables 451
13.2.4 Restoration of pubsets 453
13.3 Reconstruction of files and volumes 455
13.4 Archiving systems 458
14 Accounting 459
14.1 Control of the accounting systems 460
14.2 Accounting file 462
14.2.1 Characteristics and contents of the accounting file 464
14.2.2 Types of accounting data 466
14.3 Overviews of accounting records 468
15 System messages 478
15.1 BS2000 message system 479
15.2 Message files 480
15.2.1 System message files 481
15.2.2 User message files 482
15.3 Guaranteed messages 483
15.4 Message search 488
15.5 Structure of a message unit 490
15.6 Message code 491
15.6.1 Language identifier 492
15.6.2 Message unit attributes 493
15.6.3 Message text 496
15.6.4 Meaning/response text 499
15.7 Message output 500
15.7.1 Message output forms 501
15.7.2 HELP-MSG-INFORMATION command 503
15.7.3 Messages that require a response 504
15.7.4 Output of messages to user programs 505
15.7.5 Output of messages in S variables 506
16 Operator functions 509
16.1 User task with the OPERATING privilege 511
16.1.1 Providing an ID for the operating function 513
16.1.2 Using the event stream service for operating from user tasks 514
16.1.3 Using the event stream service for any user tasks 516
16.2 Command input via console 517
16.3 Messages 518
16.3.1 Emergency messages 519
16.3.2 Communication between system and operator 520

16.3.3 Message formats 524
16.3.4 Control of message delivery 528
16.3.5 Communication between operators 531
16.4 Use of multiple consoles 534
16.4.1 Main consoles and subsidiary consoles 536
16.4.2 Standby consoles 538
16.4.3 Functional areas and their allocation to consoles 539
16.5 Return messages 550
16.5.1 Command termination messages 551
16.5.2 Return messages on physical consoles 553
16.5.3 Return messages on physical and virtual consoles 554
16.6 Replacement of the STATUS MSG and ASR commands 556
17 Automation of operator functions 558
17.1 Authorized user programs with operator functions 559
17.1.1 Connections with generated authorization names 563
17.1.2 Connections with dynamic authorization names 564
17.1.3 Exchange of messages 568
17.1.4 Message formats 571
17.2 Special operator commands in authorized user programs 576
17.2.1 Command definition 577
17.2.2 Message formats 579
17.3 Software products OMNIS and PROP-XT 582
17.4 Command files for the operator 583
17.4.1 Executing and aborting a command file 585
17.4.2 Structure of command files 588
17.5 System administration functions performed by the operator 591
18 System time administration 593
18.1 System time 594
18.1.1 TODR as hardware clock 595
18.1.2 TODR epoch 596
18.1.3 TODR correction values 599
18.1.4 Synchronization of the system time with external timers or in a network . . . 600
18.1.5 GET-TIME subsystem 602
18.2 SVP time 603
18.3 Initializing the system time 604
18.4 Synchronizing the system time 605
18.4.1 Synchronization with the SVP clock on SU /390 606
18.4.2 Synchronization with the carrier system time on SU x86 607
18.4.3 synchronization in an NTP network 608
18.4.4 Synchronization in an XCS network 609
18.4.5 General recommendation for the XCS network 611

18.4.6 Synchronization outputs in /SHOW-SYSTEM-INFORMATION 612
18.5 Interrupt-free summer time/winter time changeover 613
18.6 System start with special system time 615
18.7 TASKDATE: testing in simulated time 616
19 Appendix 617
19.1 Character set for I/O operations via the console 618
19.2 Overview of test privileges 619
19.2.1 Test privileges for AID 620
19.2.2 Test privileges for other software diagnostic products 623
20 Related publications 624

Introduction to System Administration

1 Preface

- Objectives and target groups of this manual
- Summary of contents
- Changes since the last edition of the manual
- Notational conventions
- Functions of system administration
- Functions of the operating
- Automation

1.1 Objectives and target groups of this manual

The “Introduction to System Administration” is intended for system administration and operating staff of the BS2000 operating system.
Its purpose is to provide assistance in administering, controlling and monitoring the operating system.

1.2 Summary of contents

The “Introduction to System Administration” provides information on topics relating to administering and monitoring BS2000.

The following topics are not described in this manual, but in separate manuals:

- “SDF (System Dialog Facility)”,
  see the “SDF Dialog Interface” [43] and “SDF-A” [44] manuals.
- “Subsystem management”,
  see the “Subsystem Management in BS2000/OSD” manual [18].
- “Product delivery and product installation”, see the “IMON” manual [25].
- “Creating and analyzing diagnostic documents”,
  “error files and logging files (HEL, SERSLOG, CONSLOG, RESLOG)”,
  see the “Diagnostics Handbook” [14].

The commands and macros mentioned in this manual are described in the “Commands” [27], “Executive Macros” [30] or “DMS Macros” [20] manual unless reference is made to another manual.

Architecture of the SE servers

A FUJITSU Server BS2000 SE Series (SE server for short) consists of the following components in the maximum configuration:

- Server Units (SUs /390 and SUs x86)
- Application Units (AUs)
- Peripherals (storage)
- Management Unit (MU) with SE Manager
- Net Unit, for SU /390 with HNC

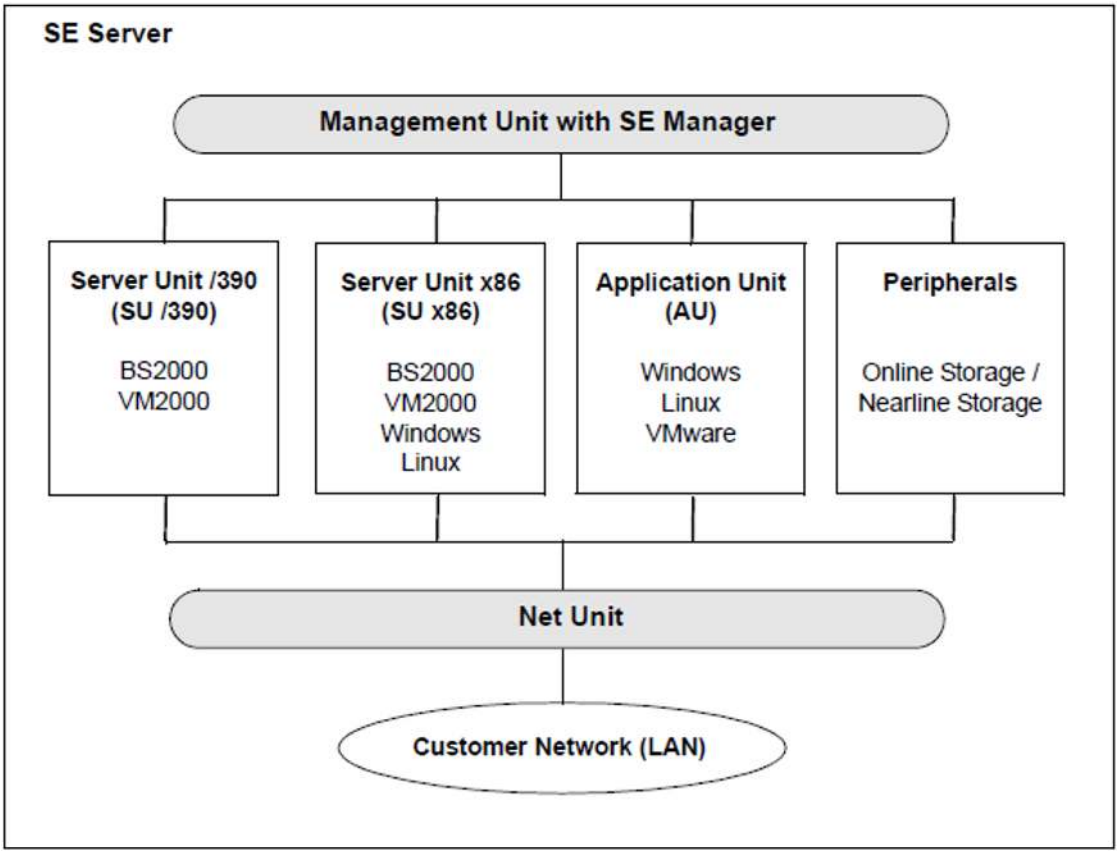


Figure 1: Architecture of the SE servers

With the SE Manager you can operate and manage all components of the SE server centrally from the Management Unit. The SE Manager offers a user-friendly, web-based user interface for this purpose.

In addition to the maximum configuration, variants are offered which do not contain all units.

Operation of the SE Manager is described in the online help for the SE Manager and in the “Operation and Administration” manual [57].

The Net Unit, for SUs /390 with High-Speed Network Connect (HNC), offers highest performance and security for internal communication in an SE server and for the connection to customer networks (LANs).

The released versions of the BS2000 operating system support the Server Unit /390 (/390 architecture) and the Server Unit x86 (x86 architecture).

> ¡ Further configuration levels of the Server Units can be released at various times. For information, please see the product announcements and release information.

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://bs2manuals.ts.fujitsu.com*.

1.3 Changes since the last edition of the manual

The following major changes have been made since the last edition of the manual.

- BS2000 V21.0A supports a new Volume Type NETVOL for NetStorage.
- The support of USER-PFA is discontinued

The new features in BS2000 V21.0A are described in the sales documents and Release Notice , which are available at *http://bs2manuals.ts.fujitsu.com*.

1.4 Notational conventions

For the sake of simplicity and clarity, frequently used names are abbreviated as follows:

- **System parameters** for the class 2 system parameters of BS2000
- **BS2000 operating system** when BS2000 need not be distinguished according to version and configuration.

With standard file names, `<ver>` stands for the internal name of the BS2000 version, e.g. `210` for BS2000 V21.0A.

In the examples the strings `<date>`, `<time>` and `<version>` specify the current outputs for date, time and version of a software product when the examples are otherwise independent of date, time and version.

The following typographical elements are used in this manual:

`Input`	Inputs in examples are shown in bold typewriter font
`Output`	DSECTS, compiler lists or outputs in examples are shown in typewriter font
[ ]	Optional specifications in syntax definitions. The brackets are metacharacters and must not be entered in the statement.
{ | | }	Alternative specifications in syntax definitions. The available operands are separated by a horizontal bar.
	For notes on particularly important information
	This symbol designates special information that points out the possibility that data can be lost or that other serious damage may occur.
[ ]	References to other publications within the text are given in abbreviated form. The full title of each publication referenced by a number is provided in full after the corresponding number in the “Related publications” section.

1.5 Functions of system administration

Administration of BS2000 comprises those tasks that have to be performed so that the system can be used efficiently, securely and with the greatest possible benefit.

System administration functions can be summarized as follows:

- assembling and introducing the appropriate operating system components
- managing and updating the organization program used in the data center and the program libraries
- securing the functionality of the system software and system-related software
- managing access authorization for the system
- managing and distributing the resources
- user and file management
- evaluating error statistics
- backing up system and user data
- installing software products
- taking action to rationalize and automate data center operation
- creating and allocating operator IDs and operator roles (in connection with the administration of privileges)
- creating diagnostic documents in the case of operating system, user program and operating errors

Suitable tools for performing these system administration tasks are available with privileged user IDs at the command and utility level.

Commands

In addition to special system administration commands which are reserved for privileged callers, all the user commands and a subset of the operator commands may be used.The subset of operator commands enables the system administration to exercise a controlling and monitoring influence at the operator interface over domains of the system operator, such as load distribution and task control.

With regard to the user commands, the privileges of the TSOS user ID, for example, provide an extended range of functions available to the system administration. These functions essentially provide a means of obtaining more precise and more extensive information and of enjoying the benefits typical of a “super user” of an operating system which come from the lifting of the resource, process and catalog limits that are set for non-privileged users.

Utility Routines

All the utility routines available to users are also available to the system administration, in some cases with extended functionality (e.g. DPAGE, HSMS/ARCHIVE, JMU, SPCCNTRL).

In addition, the system administration may use special utility routines which run only under a privileged user ID and which are used for controlling and monitoring the operating system.

1.6 Functions of the operating

The operating is responsible for controlling system startup, intervening to control the system, communicating with the user and operating the peripheral devices. System initialization (startup) and system dumps (SLED) can be used exclusively by the operating. There are further SVP commands available for diagnostics and reconfiguration functions which are reserved for the operating at the console.

For the other tasks, the operating uses commands (special operator commands, user or user/system administration commands) whose use are also described in detail in this manual. The following are the principal tasks performed by the operating:

- starting and monitoring the data communication system
- reconfiguring the hardware components and the connections to the peripherals
- providing and allocating external volumes
- monitoring and controlling device allocation
- controlling resource allocation
- monitoring operations
- communicating with the users

1.7 Automation

The main area of use of BS2000 systems in the high-performance and uninterrupted execution of customers' applications.
For many BS2000 customers operation is completely automated, i.e. the following phases are executed fully automatically and reliably:

- Startup (from system startup to availability of the applications)
- Operation (system maintenance and execution of the applications)
- System termination

By using modern hardware, such as robot-controlled magnetic tape cartridge systems, manual operation of devices is largely unnecessary. From the point of view of software, it is necessary to adapt the execution of these phases to the customer's hardware and software ("customizing").

The SDF-P command language and the Job Variable communication tool (both software products, apart from SDF-P-BASYS, the basic component of SDF-P) enables the aforementioned routines to be programmed and automated at command level.

In the startup and system termination phases, in which these tools are not yet or no longer available, the file-oriented or data-oriented interfaces of the SVP (e.g. activation/deactivation sequences), of system initiation (startup parameter service) and of DSSM (subsystem catalog) are offered.

For all further automation tasks involved in systems support, such as automatic response to events, the PROP-XT software product offers suitable user commands and supplies the relevant data in S variables.

Thus, thanks to the supplementary function of PROP-XT, SDF-P and job variables provide a user-friendly, uniform tool for automating all systems support tasks.

2 System initialization and termination

This chapter describes the processes and procedures relating to system initialization and system termination.

2.1 Overview of the stages of system initialization

The BS2000 system initialization procedure consists of “bootstrapping”, i.e. more and more powerful functional units are loaded and started step by step until BS2000 is operational.

Execution of the various routines is initiated on a hardware-dependent basis via the service processor (SVP) on SU /390 or X2000 on SU x86, or via restart processing in the case of automatic restart. This initial program loading (IPL) starts system initialization. Here both the IPL disk and the type of system initialization are defined. The setting for the load options for BS2000 defines whether the system startup is to take place in a convenient or flexible way.

In FAST and AUTOMATIC mode, system initialization is convenient and to a large extent automatic. In DIALOG mode, system initialization is flexible and interactive (see section "Types of system initialization").

For a VM2000 guest system, system initialization is started by means of the SE Manager (see the “Operation and Administration” manual [57]) or using the VM2000 command START-VM (see the “VM2000” manual [60]). On SUs x86 the VM2000 guest system can also be started using the SVP functions on the KVP console which is assigned to the virtual machine.

The most important steps in the system initialization procedure, from the point at which the hardware is made available through to the end of the startup procedure, “System ready”, are:

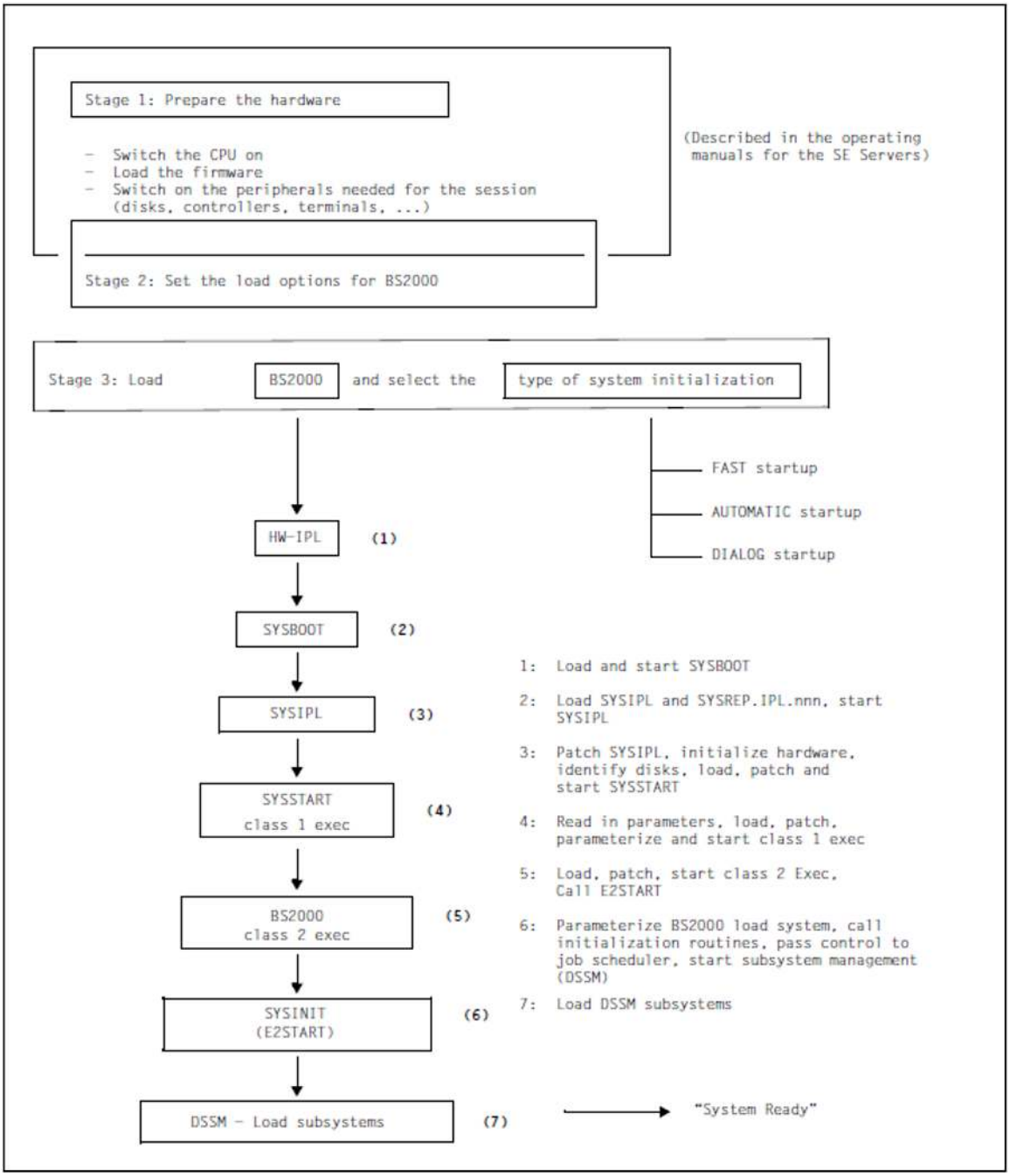


Figure 2: Flow chart of functions for BS2000 system initialization

System initialization for BS2000 can begin when the required hardware units (MU, Server Units, peripheral devices) have been activated and are operational. The relevant operation guides for the SE servers involved describe in detail how to perform these steps, i.e. how to activate the power supply, load the firmware, etc.

Internally, system initialization for BS2000 begins with the loading of the initial program loader (SYSBOOT). This is actually performed by the HW-IPL routine (step (1) in figure 2).

SYSBOOT is the first program in system initialization. It performs elementary checks and initiates other load routines (step (2) in figure 2).

The routine loaded and started by SYSBOOT is SYSIPL, which queries the options in DIALOG mode and determines the current disk and CPU configuration (step (3) in figure 2). The time base for the system time is defined. The disk configuration is checked to ensure that it is complete and unique. With DRV disks in the home pubset, the associated disk pairs are established. In addition, this routine loads and corrects SYSSTART or SLED.

The two programs SYSBOOT and SYSIPL and the IPL REP file reside in fixed locations on a specific disk known as the IPL disk. The IPL disk may be a public disk (a disk belonging to a pubset) or a private disk. For FAST and AUTOMATIC startup, this must be one of the disks of the home pubset.

> To enable automatic restart in the event of a system crash, the disk with the lower subchannel number should always be specified as the IPL disk in the case of DRV pubsets.

Private disks may only be used for initial program loading in a DIALOG startup. At a later stage in the system initialization procedure, the operator must specify which pubset is to be the home pubset for the session.

When initial program loading takes place from a public disk, the pubset to which the disk belongs is automatically selected as the home pubset. Only in the case of dialog startup with the ALLDISK option can the operator still change the pubset.

If the IPL disk does not belong to the later home pubset, particular care must be taken to ensure that the versions and correction statuses on the two are the same.

> A disk configuration may include more than one IPL disk. Setting up IPL disks is a systems support task performed using the SIR utility routine, described in detail in the “Utility Routines” manual [5].
> An IPL disk can be used either for SUs /390 or for SUs x86.

When loading SYSBOOT and SYSIPL/SLED the BS2000 file management functions are not yet available. The necessary files can therefore only be found if they are “anchored” beforehand on the IPL disk. This is done with the CREATE-IPL-VOLUME statement of the SIR utility and consists of the following steps:

- The files needed to load SYSBOOT and SYSIPL/SLED are copied to the IPL disk.
- The backup files used by SYSBOOT and SLED are created on the IPL disk.
- A direct reference to each of these files is entered in the SVL of the IPL disk:

Original file	**File created by SIR**
---	SYSDAT.IPL-CONF.DSKnnn
---	SYSPRG.BOOT.DSKnnn.SAVE
SYSPRG.IPL.<ver>	SYSPRG.IPL.DSKnnn
---	SYSPRG.SLED.DSKnnn.SAVE
SYSREP.IPL.<ver>	SYSREP.IPL.DSKnnn
SYSREP.SLED.<ver>	SYSREP.SLED.DSKnnn

“nnn” stands for the number of the IPL disk within the pubset. If the IPL disk is a private disk, then the VSN of the private disk is used instead of the DSKnnn name section.

- If not yet present on a different disk belonging to the pubset, the backup file for system patches, SYS.NSI.SAVEREP, is created (but not for private disks).

The original files are not needed anymore to initialize the system and can safely be changed while the system is running to, for example, to accept a new correction status. However, all changes only affect the initialization of the system after new copies have been created and anchored with SIR.

The anchored files may not be changed or deleted during operation because that will generally destroy the disk's IPL capability. They are protected by SIR with BACKUP-CLASS=E and MIGRATE=INHIBITED so that they cannot be moved or preempted.

All the other routines required for system initialization reside as “normal files” on either a pubset or a private disk. The configuration tables are generated dynamically.

SYSSTART (step (4) in figure 2) is a program that prepares and carries out system initialization proper for BS2000. The preparation stage mainly involves reading in the parameters for BS2000, determining the object code corrections for the class 1 Executive and checking the SYSSTART and BS2000 versions for consistency. During execution, the individual BS2000 initialization functions are called via tables. These initialization functions also include the virtual memory management data structures and the initialization of the paging areas used by SYSSTART to prepare the transition to BS2000 virtual addressing mode.

Finally, SYSSTART calls the BS2000 load system (step (5) in figure 2), which consists of the two parts “class 1 Executive” (resident) and “class 2 Executive” (pageable).
At this stage of system initialization, a device management application is made for access rights to system initialization devices (disks of the home pubset and paging pubsets). Once this phase of system initialization is complete, the BS2000 load system has been **loaded**, **patched** and **parameterized**.

BS2000 is said to be **loaded** once this part of it is contained in its entirety in main memory. The portions of BS2000 /OSD which are resident in main memory (class 1 Executive) are loaded first. The remaining portions are pageable. SYSINIT copies these routines to the paging memory via main memory.

Patched means that modules of these portions have been modified by SYSSTART during system initialization by means of REP records. REP records can be read in from a maximum of four cataloged disk files in any order. The console can also be defined as the input device, but not for REPs for SYSIPL. During system initialization, all REP records processed from disk and console are written to backup file SYS.NSI.SAVEREP and later logged in the file $SYSAUDIT.SYS.REPLOG.<date>.<sessno>.01.

Parameterized means that a set of parameter records containing statements for the BS2000 initialization routines has been read in. The entire parameter input consists of a series of sections - identified by specific keywords - that affect specific functional units and are evaluated by these functional units (see chapter chapter "Parameter service").

By presetting a value on system start, the operator can specify that loading, patching and parameterization are to be almost fully automatic and noninteractive (in this case, files with default names are used) or that these procedures are to be more flexible, controlled by means of a dialog with the operator.

The final phase of BS2000 system initialization is started using E2START (step (6) in figure 2). This routine is already executing under BS2000 and first determines the name of the command file (CMDFILE) to be started automatically after “System ready”.

To make BS2000 operational, this phase includes the execution of initialization routines for:

- activating the task scheduler
- opening the system files (user catalog, SYSEAM, etc.)

- making file catalog management available
- activating the dynamic binder loader (DBL)
- activating the PLAM library access method
- starting the functions for monitoring disk and tape devices
- activating the SERSLOG function
- starting DSSM (Dynamic SubSystem Management) [1)]

Once the memory occupied by system initialization has been released and the job scheduler started, “System ready” has been reached and processing of the commands stored in the command file CMDFILE is initiated. Although it is not mandatory to use a command file - the name of which may be freely selected - it is highly advisable because of the demand for automation of the operations.

Command files can be used for the automatic activation of the system components and settings that make a specific system operable:

- starting up the optional subsystems
- starting the BS2000 data communication system [2)]
- loading the SPOOL system [3)]
- specific load regulation
- activating special programs by means of ENTER files

Notes

1) DSSM:
During system initialization, control passes to DSSM (Dynamic SubSystem Management). DSSM initializes itself with the specified subsystem catalog and activates subsystems or initiates their activation. The time at which a subsystem is started up (before or after “System ready”) is determined by systems support upon declaration. This makes it possible for subsystems to be activated automatically.

When a subsystem is started, DSSM uses IMON-GPN to determine the path names of all the files of the subsystem from the current SCI. If IMON-GPN is not available (already loaded when DSSM is started) or if there is no file under the specified path name, DSSM uses the standard names entered in the subsystem catalog. If a standard name is used, message `ESM0665` is output.

2) Data communication system (DCM):
To start the data communication system even before “System Ready”, it is possible to specify the DCSTART command as a BCAM parameter in the startup parameter file (see the “BCAM” manual [4], section BCAM BS2000 parameter file).

If that is not the case, the data communication system has to be activated separately after each system initialization. This is done by means of the DCSTART command, which is stored in the CMDFILE for reasons of convenience.

The DCSTART command automatically initiates opening of the following internal privileged applications of the system:

- $DIALOG (application for interactive processing (TIAM))
- $CONSOLE (application for logical consoles, see section "Virtual consoles/$CONSOLE applications" in chapter "Operator functions")

- $BCAM (application for the DCM information service)

If the first DCSTART command is issued later than 10 min. after the “System Ready” or if the DCM is terminated while BS2000 is running (BCEND command) and restarted, the $DIALOG application must be started manually by the operator using the START-DIALOG-APPLICATION command. Another option would be to include /START-DIALOG-APPLICATION in BCAM’s SOF (Start Option File). A prerequisite for this is that a console access has been configured for BCAM with the authorization for START-DIALOG-APPLICATION (see the “BCAM” manual [4]).

In the operating mode with operator logon, the operator must first enter SET-LOGON-PARAMETERS and REQUEST-OPERATOR-ROLE commands after “System ready” before being able to enter further commands. Consequently, it is advisable to also remove the prerequisites for the first two commands from the CMDFILE. The prerequisites are as follows:

- the operator ID must be unlocked
- the operator role must be set up
- the operator role must be assigned to the operator ID

Since the operator ID has to be unlocked under the TSOS ID but an operator role can only be set up and assigned to an operator ID under SYSPRIV, it is advisable to invoke from the CMDFILE an ENTER job which issues the UNLOCK-USER command and calls a further procedure for setting up and assigning the operator roles under the SYSPRIV ID.

The whole process could look something like this:

Call from the CMDFILE:

```
/ENTER-JOB E.OPR-LOGON.TSOS
```

Contents of the E.OPR-LOGON.TSOS file

```
/SET-LOGON-PARAMETERS
/ UNLOCK-USER SYSPRIV
/ SET-JOB-STEP
/ UNLOCK-USER SYSOPR
/ SET-JOB-STEP
/ ENTER-JOB FROM-FILE=$TSOS.E.OPR-LOGON.SYSPRIV,-
/           PROC-ADMISS=*PAR(USER-ID=SYSPRIV,-
/                            ACC=SYSACC,-
/                            PASS=*NONE)
/EXIT-JOB
```

Contents of the E.OPR-LOGON.SYSPRIV file

```
/SET-LOGON-PARAMETERS
/ CREATE-OPERATOR-ROLE OPERATOR-ROLE=SYSADM,ROUT-CODES=*ALL
/ SET-JOB-STEP
/ MODIFY-OPERATOR-ATTR USER-ID=SYSOPR,ADD-OP-ROLE=SYSADM
/ SET-JOB-STEP
/ INFORM-OPERATOR,-
/          MSG='*** OPERATOR-ROLE SYSADM CREATED AND ADDED ***'
/ INFORM-OPERATOR,-
/ MSG='+-----------------------------------------------------+'
/ INFORM-OPERATOR,-
/ MSG='!    THE FIRST OPERATOR COMMANDS AFTER SYSTEM READY    !'
/ INFORM-OPERATOR,-
/ MSG='!    (BEFORE /DCSTART ... )  MUST BE:                  !'
/ INFORM-OPERATOR,-
/ MSG='!    /SET-LOGON-PARAMETERS SYSOPR,SYSACC               !'
/ INFORM-OPERATOR,-
/ MSG='!    /REQUEST-OPERATOR-ROLE OP-ROLE=SYSADM             !'
/ INFORM-OPERATOR,-
/ MSG='+-----------------------------------------------------+'
/EXIT-JOB
```

Only then is BS2000 capable of communication.

3) SPOOL:
After each system start SPOOL has to be loaded and initialized separately. SPOOL startup is initiated by the START-SUBSYSTEM command. The SUBSYSTEM-PARAMETER operand can be used to specify whether a warm or cold start is to be carried out for SPOOL, and whether additionally the software product RSO is to be loaded. This command should either be issued immediately after “System ready” or be included in the command file CMDFILE. If SPOOL is not loaded, no SPOOLIN or SPOOLOUT jobs can be executed. SPOOL requests from the operator (e.g. the commands START-PRINTER-OUTPUT, MODIFY-PRINTER-OUTPUT-STATUS,...) are rejected, ignored or suspended.

To summarize, system initialization consists of the following **internal** steps:

HW-IPL:	- Loading the 1st block of SYSBOOT
SYSBOOT:	- Loading the 2nd block of SYSBOOT - Loading SYSREP.IPL.<version> - Loading and starting SYSIPL
SYSIPL:	- Self correction - Loading, correcting and starting SYSSTART
SYSSTART:	- Reading in the parameters - Loading and correcting the memory-resident portions of BS2000 (class 1 exec) - Initialization of the paging memory

Class 1 Executive:	- Initialization of the resident portion of BS2000Automatic attachment of the disk devices which were generated asDETACHED and on which are mounted the necessary public disks [1] - Loading the pageable portion of BS2000 (class 2 exec) - Correction of the pageable portions
Class 2 Executive:	- Initialization of the pageable portions
SYSINIT (E2START):	- Determining the command file and calling initialization functions for BS2000 functional units (DSSM, PLAM, etc.) - Release of occupied memory and start of the job scheduler
„System Ready“ command file CMDFILE, for example, is now activated)	

[1] During the startup phase, all public disks of the home pubset and all pubsets which contain paging disks and which were specified in the parameter service are automatically attached to the system, even if they were explicitly generated as DETACHED. The devices of the home pubset remain ATTACHED during the entire session.

The SHOW-SYSTEM-INFORMATION command can be used to obtain information on the system configuration, the VM2000 version used, the monitor system and the time setting parameters.

2.1.1 Determining the time on system initialization

See also the chapter “System time administration”.

There are three sources for determining the date and time during system initialization:

- SVP clock: returns local time
  - On SUs /390 the SVP clock is synchronized by the Management Unit.
  - On SUs x86 the SVP clock is emulated by X2000. The SVP time corresponds to the time of the carrier system.
- CPU clock (TODR): provides the local time with a correction value stored on the home pubset. This clock continues to run even if the CPU stops, but not if the power supply is interrupted.
  - On SUs /390 the TODR is an autonomous clock.
  - On SUs x86 the TODR is emulated by X2000.
- Operator
  The currently valid time is displayed for the operator at the console. In the event of an error or in DIALOG mode (using the UNLOCK option), the operator receives the appropriate messages at the console and is requested to confirm or to make corrections.

The local time (LT) is the statutory time at the location at which BS2000 is installed. Because of statutory changes (summer/winter time) it does not run continuously once set; instead, it requires a positive or negative correction at the changeover times. This correction is made automatically with the aid of the GTIME parameter file in which the correction value and the changeover times are stored (see "System time control (GTIME)").

The system time is based on the local time. BS2000 cannot run without the system time.

Systems support uses the parameters for time zone, time increments, summer and winter time and the changeover data to define the basis for calculating the universal world time UTC from the local time LTT and for summer/winter time changes of local time during system execution.
In this way, the system and the users can use the GTIME function to access both a local time reference system and a time reference system valid for the entire system.

Determining the time automatically for manual startup

The time is determined automatically when the following conditions are met:

- The IPL option UNLOCK is not set, i.e. system initialization was carried out in FAST or AUTOMATIC mode or without the IPL option UNLOCK in DIALOG mode.
- The time of the SVP clock is valid.
- The time of the SVP clock has the correct format (e.g. not 31.2.14).
- The time of the SVP clock is greater than the time of the last session and the positive increment compared with the last session is not greater than 6 days (exception: in AUTOMATIC mode, the positive increment may be any size).

If a shutdown is performed immediately before switching from daylight savings time to normal time and the startup immediately thereafter, system initialization starts with a time less than the last session. This is only possible if the UNLOCK option was set in DIALOG mode (see "DIALOG startup").

Negative time increments outside these changeover periods are permissible only by agreement with systems support. There is a risk of introducing inconsistencies into the data resources.

Influencing the time definition in DIALOG startup

If DIALOG mode is used and the UNLOCK option set, the operator is prompted for confirmation even if the value of the SVP clock is valid. If the time (and the date) of the SVP clock is not confirmed, the operator must enter the date and time himself. It is possible to specify large positive and negative time increments in relation to the last session. Once the time has been specified in this way and confirmed by logging, it is used as the valid system time for the next session.

2.1.2 Format of messages during system initialization

Every system initialization message has one of the following formats:

```
? P.msgtext
```

```
? tsn-makz.hhmmss % msgtext
```

where:

?	Message requiring a response (message not requiring a response: %)
P	Message from startup
msgtext	Message text
tsn	Message from BS2000; tsn is the task sequence number of a system task or a job name
-	Hyphen (omitted in messages from the resident portion of system initialization
mjid	Message job ID (3 numeric characters or letters; leading zeros may be omitted). mjid is omitted in messages from the resident portion of system initialization
. / #	Separator; depends on the system parameter SECSTART
hhmmss	Current time of day (omitted in messages from the resident portion of system initialization

The message text always starts with a code number (3-digit message class and 4-digit sequential number, where x is a hexadecimal digit).

Messages from the resident portion of system initialization have the code number NSIxxxx (Nucleus System Initialization).

The messages of the pageable portion of system initialization have the code number EXCxxxx.

Example of a message from the resident portion of system initialization:

```
? P.NSI0050SPECIFY PARAMFILE ...
```

Example of a message after the resident portion of system initialization:

```
? TSC-000.112133%NSI0077ENTER AUTOMATIC COMMAND FILE
```

The response proper begins with the character string (preceded by a question mark) belonging to the message in question (P or tsn), including a hyphen, if appropriate, and the message job ID, followed by a period. The response text must follow without a break.

Example of a response during the resident portion of system initialization:

```
P.FN=PARAMFILE.OPR
```

Example of a response after the resident portion of system initialization:

```
TSC.CMDFILEX
```

System installation messages are always output in English.

2.1.3 General notes on initial program loading

Regardless of the type of server to be controlled by the operator, it is important to answer the following questions in advance in order to make sure that the loading procedure is executed correctly:

- How are the required files and devices identified?
- What points must be observed regarding unique disk configuration?
- How is the home pubset determined and checked for completeness and consistency?
- How does the system react to assigned pubsets or detached devices?
- What can cause devices to be unavailable during system initialization?
- What points must be observed regarding multiprocessors?
- What is the function of the parameter file in system initialization?
- How is the loading procedure logged?
- What must be taken into account with regard to Unicode capability?

Identification of the required files and devices

The volumes required for HW-IPL must be addressed as follows:

For SUs /390, the IPL device is identified by the device number. For devices with a four-digit MN, the device number is the same as the MN. For devices with a two-digit MN, there is a unique assignment between device number and MN, which is described in the "System Installation" manual [55].

For SUs x86, the IPL device is identified by the mnemonic device name (MN).

Only disks of the home pubset should be used as IPL disks. Disks with a physical block size of 4KB are not supported as IPL disks.

All other devices subsequently required for the loading procedure are identified by the mnemonic device name MN.

If files which are stored on private disks are used for system initialization, not only the file name but also the corresponding VSN must be specified. Private disks may be used only if they are included in the disk configuration for system initialization.

Unique disk configuration

After SYSBOOT, SYSIPL determines the available disk configuration (via an online scan). At this point, the BS2000 device tables, i.e. the generated hardware configuration, are not yet available.

Multiple online scan sessions are prevented by creating a separate table for all disks identified that are not added to the system initialization device table. If one of these disks is used later on, then it is added to the system initialization device table without performing another online scan.

Private disks are only accepted into the disk configuration for system initialization if the IPL disk itself is a private disk, if private disks are required for a SLED dump, or if the operator explicitly requests this (by selecting the ALLDISK option in the DIALOG initialization mode).

System initialization is problem-free only if each accessible disk has a unique VSN. If more disks with the same VSN can be accessed, this will normally be recognized.
Exception: the current IOCF of the SU /390 is not correct.

If disks having the same VSN are detected and at least one of these is not a DRV disk, then the system proceeds as follows: When the disks belong to the same pubset as the IPL disk, then the disk with the same time stamp as the IPL disk is selected. If none of the disks has the same time stamp as the IPL disk or the disks belong to a different pubset than the IPL disk, then a check is made to see if one of the disks was already selected by responding to message `NSI2208`. If this is true, then this disk is selected again. This applies not only to pubset disks but also to private disks.

> **CAUTION!**
> In the exceptional case stated above, any discrepancy between the real and generated configurations goes undetected. If system initialization is then continued, this may result in unpredictable I/O errors and the destruction of disk contents!

During system initialization, all attached devices are normally checked to determine the disk configuration. This process can be time-consuming and prone to error, especially if a large number of peripherals are being used (e.g. devices that are not required and therefore not operational are addressed). To circumvent this process, the following option is offered: The disk configuration actually required for system initialization (i.e. all disks of the home and paging pubsets and any private disks used) can be saved in SYSDAT.IPL-CONF.DSKnnn, a file reserved for this purpose on the IPL disk (nnn = number of the IPL disk in the pubset).

Dynamic partitioning of the IPL-CONF file

The IPL-CONF file (SYSDAT.IPL-CONF.DSKnnn) is partitioned dynamically. This enables multiple startup configurations to be stored in this file. Each configuration in the configuration file is identified by the IPL-CPU-ID, home PVS-ID and IORSF date. During startup, if a matching configuration is found, it will be used and no online scan will be performed unless explicitly requested. When no matching configuration exists, online scan will be initiated and a new configuration will be written to the file.

When a pubset is used as the home pubset on various servers, the entries for the startup configuration in the IPL-CONF file are not lost, but each is stored in a separate partition. When the server is switched, the corresponding startup configuration is used without an online scan necessarily being required.

A new startup configuration does not overwrite an existing one, but is entered as a new startup configuration after the last existing one. When an existing startup configuration is modified, the original one is “discarded” and the modified startup configuration is entered like a new one after the last existing one.

However, the IPL-CONF file is not enlarged: When there is a lack of space, the startup configurations declared invalid are first of all deleted one after the other. If the free space in the file is still not sufficient after this, the startup configuration which has remained unused for the longest time is removed. This action is performed without consulting the operator and is repeated until enough space is available for the new or modified startup configuration.

> The IPL-CONF file can contain, for example, 12 startup configurations each with 255 disks. When there are fewer disks, the number of configurations increases: Consequently, 30 configurations each with 100 disks or 57 configurations each with 50 disks are possible, for instance.

Saving a startup configuration in the IPL-CONF file is initiated as follows:

- in DIALOG mode:
  If the operator responds to the message `NSI1110` with the IPL-CONF option SAVE, the startup configuration is saved as a new partition.

- In all startup modes:
  The startup configuration is saved when:
  - the IPL disk is one of the disks of the home pubset
  - and the IPL-CONF option IGNORE has not been explicitly specified
  - and online scan has been performed.

The online scan is always performed automatically when the home pubset or one or more paging pubsets has/have been extended or a new paging pubset has been specified using BS2000 parameters (in the SYSPAR.BS2.nnn file or via the console).

The file is then used automatically for all subsequent system initializations, provided that the VSN of the saved pubsets, the mnemonic device name MN, the disk type and the number of disks belonging to the pubsets remain unchanged.

To support modifications to existing, startup configurations in the SYSDAT.IPL-CONF.DSKnnn file (e.g. if an extended or different pubset is used), in DIALOG mode the message `NSI1110` offers the operator the following functions if the IPL-CONF option is specified:

- IGNORE function:
  The existing valid startup configuration in the SYSDAT.IPL-CONF.DSKnnn file is ignored. An online scan is executed.
- RESET function:
  The current valid startup configuration in the SYSDAT.IPL-CONF.DSKnnn file is still used for the current system initialization and is then declared invalid.
- SAVE function:
  The disk configuration required for startup is saved in the SYSDAT.IPL-CONF.DSKnnn file as a new partition.

All combinations except SAVE and RESET are valid.

Determining the home pubset and checking for completeness and consistency

If the IPL-CONF function is not used for system initialization, the home pubset can be determined as follows according to the type of system initialization, and once the available disk configurations have been determined:

- in FAST or AUTOMATIC mode:
  The home pubset selected is the one to which the IPL disk belongs.
- in DIALOG mode:
  If the IPL disk belongs to a pubset, the home pubset selected is the one to which the IPL disk belongs.
  If the IPL disk is a private disk or the IPL option ALLDISK was specified and the disk configuration consists of more than one pubset, the ID (PVSID) of the home pubset is queried via message `NSI1135`.

Disks with a physical block size of 4KB cannot belong to a home pubset. SM pubsets and shared pubsets are likewise not supported as home pubsets.

The home pubset should have a high degree of availability, i.e. through the hardware, the data for a virtual disk is recorded on several physical disks. This redundancy ensures that there is no loss of data if a physical disk crashes.

All the disks available in the startup configuration and disks belonging to the home pubset according to VSN syntax are compared with the volume list on Pubres. In DIALOG mode or in error situations the disks belonging to the home pubset are logged with the last time stamp and their mnemonic device name MN (with message `NSI1143` or `NSI1145`).

The following error situations are displayed:

- System ID (SYSID) of the home pubset not defined (message NSI1280)
  System initialization proceeds with the default value.
- Invalid SYSID of the home pubset (message NSI1285)
  System initialization proceeds with the default value.
- Missing disk in the home pubset (message NSI3215)
  System initialization cannot proceed.
- Time stamp of a disk not the same as the time stamp of Pubres (message NSI1148)
  Whether or not system initialization can proceed depends on the response to message NSI1148.
- DMS attributes of a disk different to the DMS attribute of the home pubset (NSI3220)
  The relevant DMS attribute s are indicated by message NSI3221.
  System initialization is aborted since such an inconsistent pubset cannot be imported by the IMCAT command.
- Unknown disk in the home pubset (message NSI1145)
  Disks that do not belong to the home pubset are indicated by the words IS NEW in message NSI1145.

Support for DRV disks

DRV pubsets are also permitted as home pubsets. DRV private disks are still not supported, and neither are DRV disks outside of the home pubset at startup paging initialization.

Because the DRV function is not yet available in the early phase of system initialization, the accesses to the DRV disks must be implemented by the IPL EXEC itself.

The following restrictions apply during system initialization:

1. Data is only ever read from or written to one disk of a DRV disk pair. The disk to be used is selected either automatically or in DIALOG startup by the operator specifying the DRV-SELECT option.

   A disk can be selected from a disk pair with the same VSN only if both disks are in the DRV-DUAL state and the time stamp is identical on both of them. If this is not the case, selection of the disk is subject to restrictions (see also the “DRV” manual [17]).

   If the IPL disk belongs to a DRV pair, the system initialization can continue with a different disk after the online scan or after the IPL-CONF file has been evaluated. This means that changes to the data are not written to the disk selected at the start of IPL, but are written instead to a different disk of the DRV pair.
2. Input/output errors on one DRV disk during system initialization do not cause the second disk of the pair to be accessed. As with individual disks, the discovery of irrecoverable I/O errors leads to abortion.
3. The home pubset is started up in the DRV-MONO state; reconstruction must be initialized with the START-DRV command.
4. If the home pubset was in the DRV-DUAL state on shutdown, reconstruction is initiated by DRV. This process reconstructs the files which have been modified since system initialization.

START/STOP mode for multiprocessors

In multiprocessor systems, the START/STOP mode controls selection of the CPUs when the SVP-START/STOP function is used. The SVP-START/STOP function is not needed in BS2000 operation. Nevertheless the START /STOP mode must be set correctly so that when the SVP-START/STOP or address stop function is used there are no undesirable side effects.

The following conditions apply for the settings of the START/STOP mode:

- If all CPUs are to be used, the START/STOP mode must be set to ALL CPU so that in the case of STOP all CPUs are stopped and in the case of START all CPUs are started. This setting should be used as the presetting.
- If only one CPU is to be used or if the SVP-START/STOP or address stop function is required for system initialization or for the dump function, the START/STOP mode must be set to TARGET CPU (CPU currently in use). This prevents the SVP-START function from starting CPUs that can no longer or not yet be controlled by the software.
- If more than one CPU but not all the CPUs are to be used, correct functioning of the SVP-START/STOP function cannot be guaranteed.

Releasing assigned pubsets

If the home and/or paging pubsets required for system initialization were not exported correctly during the last system session (by means of SHUTDOWN), or if they are being used by another system, the operator will be requested to release them (message `NSI424A`).
The operator is informed of the precise nature of this use in a message preceding the request.

It is absolutely essential to assign unique SYSIDs if multiple home pubsets are operated in parallel.

Release is only possible if these pubsets are not currently being used by another system. If pubsets are released without a careful preliminary check, then the pubset data may be destroyed as a result of unrestricted access by two systems.

Parameter service

A startup parameter file is essential if system initialization is to be performed correctly. The parameter service supplies the software components with data.

For a detailed description of the parameter service including the structure and contents of the parameter files, see chapter "Parameter service".

Logging the startup dialog on the IPL console

All messages, including those not output at the console in FAST and AUTOMATIC mode, and all responses are logged in main memory and transferred to the logging file ($SYSAUDIT.SYS.CONSLOG.date.session-no.ser-no) as soon as the CONSLOG function is available.

For information on the CONSLOG logging file structure and contents, please refer to the “Diagnostics Handbook” [14].

If system initialization is concluded before the CONSLOG function is made available, the console dialog can be found in a subsequent dump.

2.1.4 System initialization on SUs x86

On SUs x86 (X86-64 architecture) the bus and Fibre Channel peripherals are configured by the carrier system X2000, and the configuration is determined dynamically at startup. The SU x86 does not require hardware generation with IOGEN.

Filename	Use
instead of SYSPRG.IPL.<ver>: SKMPRG.IPL.<ver>	Load objects SYSBOOT, SYSIPL + SYSIPLEX and SLED, default file
instead of SYSPRG.STRT. <ver>: SKMPRG.STRT.<ver>	Load object SYSSTART + SYSIPLEX
instead of SYSPRG.BS2.<ver>: SKMPRG.BS2.<ver>	Load object BS2000-CL1/2 exec

Table 1: Alternate default file names on SUs x86

Automatic IPL and time-controlled startup/shutdown

On SUs x86 systems support (using the SE Manager) can configure an automatic IPL, a time-controlled startup or shutdown under X2000. In the event of a system shutdown via X2000 the system parameter SHUTPROC determines whether the shutdown should take place immediately or using an enter job.
If automatic IPL was set, then an IPL is automatically initiated from the default IPL disk after the X2000 system has started up.

2.2 Types of system initialization

System initialization can take place in a convenient or flexible way. The FAST and AUTOMATIC modes are available for a convenient system start, and the DIALOG mode for a flexible system start.

The save file has the version-independent name SYS.NSI.SAVEREP. Except in the case of private disks, it is created by SIR when an IPL disk is set up.

For information on the file names for the parameter file see chapter “Parameter service”.

2.2.1 FAST startup

FAST startup permits system initialization to be performed without dialog. To this end, the object code corrections (REPs), parameter and programs to be loaded must be stored in files with version-specific standard names. Alternatively, the file names can be stored in the parameter file. The parameter file itself must have the version-specific standard name, with or without a suffix.

The most important events during system initialization are logged on the IPL console. The detailed log - as in DIALOG mode - is written to the CONSLOG logging file.

If the files SYSREP.BS2.<ver>, SYSREP.STRT.<ver>, SYSPAR.BS2.<ver>[.<name>], SYSPRG.STRT.<ver> or SYSPRG.BS2.<ver> are missing and if the parameter file does not specify a REP file or BS2000, the correction data and parameters may optionally be submitted from disk files with any name or via the console in an operator dialog (see chapter "Parameter service" and section "System corrections").

The choice of startup type and the decision as to whether the user catalog is to be reconstructed depends on the values that have already been set for the system parameters STUPTYPE and RECONUC.
For information on selecting the startup type, see also section "Selecting the startup type and catalog reconfiguration".

The following standard file names are valid for dialog-free system initialization and must be cataloged under user ID TSOS on the home pubset:

Filename	Use
SYSPRG.BOOT.DSKnnn.SAVE	Save area [1]
SYSPRG.IPL.DSKnnn	Load objects SYSBOOT, SYSIPL and SLED, use at startup [2]
SYSREP.IPL.DSKnnn	Reps for SYSIPL, use at startup [2]
SYSDAT.IPL-CONF.DSKnnn	Save area for the function IPL-CONF [1]
SYSPRG.STRT.<ver>	Load object SYSSTART
SYSREP.STRT.<ver>	REPs for SYSSTART
SYSREP.BS2.<ver>	Object code patches (REPs) [3]
SYS.NSI.SAVEREP	Buffer for REP logging [4]
SYSPAR.BS2.<ver>[.<name>]	Parameter
SYSPRG.BS2.<ver>	Load object BS2000-CL1/2 exec [3]

[1] The file is created by SIR as an empty backup file and is anchored in the SVL. “nnn” in the name part “DSKnnn” is the number of the disk on which the file was created by SIR. These file names are not checked during startup. The files with a disk-specific name are not taken into account in a logical save.

[2] The file was copied by SIR from a version-dependent original file into this version-independent file (suffix“DSKnnn”), which was then incorporated in the SVL. “nnn” in the name part “DSKnnn” is the number of the disk on which the file was created by SIR. These file names are not checked during startup.
The files with a disk-specific name are not taken into account in a logical save.

[3] The file name can be specified via the startup parameter service.

[4] The file is created by SIR as an empty backup file. If this file is missing or if it is too small, this has repercussions on the “secure system startup”.

If one of the files is missing or cannot be used, the file name is requested via the console (not in the case of anchored files).

The disk-specific naming of the anchored files, i.e. those stored in the SVL, means that several IPL disks of a BS2000 version can be created in parallel in a pubset.

2.2.2 AUTOMATIC startup

AUTOMATIC mode supports unmanned operation. AUTOMATIC startup is used in conjunction with automatic system power-up following abnormal system termination when the automatic restart function is switched on. Its characteristics and logging correspond to the FAST mode.

AUTOMATIC mode may only be used if the SVP clock is functioning (omitted for restart) and the home pubset disks and paging disks are used only by this system.

In AUTOMATIC mode there are standard responses to certain exception conditions. This means that the operator is not required to make a decision. An example of a standard response is the automatic release of disks which are required but which are still locked as a result of an abnormal system termination.

Automatic restart is described in section “Automatic restart”.

The choice of startup type and the decision as to whether the user catalog is to be reconstructed depends on the values that have already been set for the system parameters STUPTYPE and RECONUC.
For information on selecting the startup type, see also section “Selecting the startup type and catalog reconfiguration”.

2.2.3 DIALOG startup

DIALOG startup enables the operator to control system initialization and select special functions. A detailed dialog takes place, in which messages indicating the permissible operator inputs are displayed on the screen. The operator specifies the desired function by selecting the appropriate entry from those displayed. These functions include:

- free choice of IPL disk, input media for the object code corrections (REPs), parameters and programs to be loaded
- activation of specific debugging functions or functions for dealing with exceptional situations

Special protection is offered for DIALOG startup because DIALOG startup allows the operator to change the settings (parameters, REPs) made by systems support for system initialization:
As an option, system initialization of BS2000 in DIALOG mode can be restricted to a specific console. This restriction can be imposed and canceled only in DIALOG mode; it always relates to the console currently in use. The mnemonic device name of the console is stored in the home pubset as the authorization (routing) code.

Execution of a DIALOG startup

Execution of DIALOG startup is logged in detail on the IPL console.

In DIALOG mode, message NSI1110 shown below prompts the operator to enter options:

```
NSI1110 ENTER OPTIONS OR EOT.
        REPLY (UNLOCK,TEST,ALLDISK,DRV-SELECT,CREATE-DRV,IPL-CONF)
```

Any combinations are possible. The meaning of each of the options is given below:

UNLOCK:	Indicates that, in the event of conflicting information regarding the system time, system initialization can be continued or the operator can determine the system time.

TEST: Message NSI1113 offers the operator certain test options.

```
NSI1113 ENTER TEST-OPTIONS OR EOT.
REPLY (DUMPTEST, STRTNAME, REPERRIGN, WATCH, ASYNSKPIN)
```

Meaning of the test options

DUMPTEST:
Test option, not essential for system initialization.

STRTNAME:
Causes the file name of the load module and of the SYSSTART REP file to be requested before SYSSTART is loaded.

REPERRIGN:
Test option, not essential for system initialization.

WATCH:
Test option, not essential for system initialization. It supports error diagnosis during online scanning. A monitoring function is activated for a given device and system initialization is aborted if problems are detected in connection with this device. In this way, diagnostic documents (SLED) can be generated at the moment the error occurs.

ASYNSKPIN:
Test option, not essential for system initialization.

ALLDISK: In the disk configuration used by startup, private disks are also automatically scanned (via an online scan) so that load objects and REP or parameter files can also be read from private disk. If the IPL disk is a private disk, the ALLDISK option is set automatically. Startup inquires the required home pubset with the following message:

```
NSI1135 ENTER ACTUAL HOME-PVS IDENTIFIER. REPLY (PVSI)
```

The IPL-CONF option IGNORE is set implicitly. This option can only be used if a maximum of 1290 disks are online. In the event of an error, the following message is output:

```
NSI2335 OVERFLOW OF STARTUP DEV.TABLE; SWITCH UNNEEDED DISKS OFF
REPLY (R(RETRY); S(STOP))
```

The online scan is repeated or the system initialization is aborted, depending on your response.

DRV-SELECT: This option means that selection of the disk for startup from a DRV disk pair is left to the operator.

This is meaningful if, due to a disk error on one DRV disk during startup, the other disk of the DRV disk pair is to be used instead.

The IPL-CONF option IGNORE is set implicitly.

CREATE-DRV: This option means that, through implicit starting of the DRV subsystem after system initialization a DRV pubset in the DRV-MONO state is created from an SRV home pubset (Single Recording by Volume).

IPL-CONF: This option controls the processing of the server-specific partition for the SYSDAT.IPL-CONF.DSKnnn file which is always created on the IPL disk (the meaning of this parameter is also described on "Dynamic partitioning of the IPL-CONF file" in section "General notes on initial program loading"). If this option is not specified, an attempt is made to evaluate the server-specific partition of the file, as in all other startup modes (FAST, AUTOMATIC).

When the server-specific partition is processed successfully, the online scan is suppressed.

When this option is specified, message NSI1116 prompts the operator to specify how the server-specific partition in the SYSDAT.IPL-CONF.vvv file is to be processed:

```
NSI1116ENTER IPL-CONF OPTION OR EOT.
REPLY (IGNORE; RESET; SAVE. EOT=IGNORE)
```

Meaning of the possible replies

IGNORE:
The server-specific partition in the SYSDAT.IPL-CONF.DSKnnn file is ignored. An online scan is executed. IGNORE is implicitly set if the ALLDISK option was specified.

RESET:
The partition in the SYSDAT.IPL-CONF.DSKnnn file which is currently valid for this server is still used for the current system initialization and then declared invalid.

SAVE:
The disk configuration required for startup is saved in the SYSDAT.IPL-CONF.DSKnnn file as a new partition.

All combinations except SAVE and RESET are valid.

Once the options entered have been processed, the startup routine checks the available disk configuration. To this end, either a partition which is present in the SYSDAT.IPL-CONF.DSKnnn file on the IPL disk and is valid is evaluated (unless excluded explicitly using the IGNORE option) or the online scan is executed. If disks with the same VSN are encountered, the disk configuration must be limited to disks with unique VSNs (see also the section "Unique disk configuration" in chapter "General notes on initial program loading").

If several pubsets are available on the system, the operator is requested to define the home pubset (NSI1135). In addition to the home pubset and the CPUs, the date and time of day are also logged. If necessary, the operator is requested to enter a confirmation or to make corrections.

The default file name is always used to load SYSSTART. The operator receives the following prompt to enter the file name only if the default file name is not found in the catalog or if STRTNAME was explicitly specified when the test options were entered:

```
NSI1190 ENTER STARTUP-FILENAME.
        REPLY (FILENAME(,VOL=VSN); EOT (USE STANDARD FILE))
```

The default file name is used for the SYSSTART REP file. If this file does not exist or the STRTNAME option was selected, the name of the file or input medium is requested by message NSI0050 (see "Notes on the format of REP records" in section "REP records").

The file name and the input medium for the parameter are then requested (see also the chapter "Parameter service"). The following message is then issued:

```
NSI1190 ENTER BS2000-FILENAME.
        REPLY (FILENAME(,VOL=VSN); EOT (USE STANDARD FILE))
```

The operator can enter the file name of the BS2000 system to be loaded, together with the archive number if the file is on a private disk. If no file name is specified, the parameter file is evaluated. If no file name is entered there either, BS2000 is loaded from the default file SYSPRG.BS2.<ver>.

REP processing then takes place (see "Notes on the format of REP records" in section "REP records"). Then the selected paging disks are logged.

The following message is displayed, requesting the operator to specify the name of the command file. Following “System ready”, the commands contained in the command file are executed automatically.

```
NSI0077 ENTER AUTOMATIC COMMAND FILE NAME.
        REPLY (FILE NAME; N(DO NOT USE); EOT(USE STANDARD))
```

The following message is displayed, requesting the operator to specify the type of system startup (see "Selecting the startup type and catalog reconfiguration"):
(see also the description of the system parameter STUPTYPE)

```
NSI6005 SYSTEM PARAMETER STUPTYPE = (&00). SHALL VALUE BE CHANGED?
        REPLY ( U(NCHANGED), W(ARM), C(OLD), S(ELECTIVE), Z(IP),
        T(SN FILE RESET ONLY), J(OIN AND TSN FILE RESET), EOT=UNCHANGED)
```

By replying the following message the operator can determine whether and how the user catalog is to be reconstructed (see the description of the system parameter RECONUC).

```
NSI6010 SYSTEM PARAMETER RECONUC = (&00). SHALL VALUE BE CHANGED?
        REPLY ( U(NCHANGED), N(O), B(ACKUP), T(SOSCAT), A(LL), R(ESET),
        EOT=UNCHANGED )
```

The further execution of system initialization is controlled by DSSM (see also the note on "Notes" in section "Overview of the stages of system initialization"). System initialization is complete when the “System Ready” message is displayed.

2.2.4 Switching modes during system initialization

In some special cases, too - e.g. when testing new software, parameters or corrections - it may be necessary for system initialization to take place as far as possible without dialog. To this end, the parameter file may contain the deviations from the standard case and also the option of switching to a different mode once these presettings have been processed. The operator selects DIALOG mode from the SVP menu and, while in DIALOG mode, specifies the appropriate parameter file. By entering NEW-IPL-MODE=FAST under the keyword SYSOPT-IPL in the parameter file, systems support specifies that, once the parameter file has been evaluated, system initialization is to continue in FAST mode, i.e. unattended.

For information on the parameter record SYSOPT-IPL, see also section “Modifying the IPL options (SYSOPT-IPL)”.

2.2.5 Selecting the startup type and catalog reconfiguration

In a DIALOG startup, the operator is prompted to select the type of startup by responding to the message:

```
NSI6005 SYSTEM PARAMETER STUPTYPE = (&00). SHALL VALUE BE CHANGED?
        REPLY ( U(NCHANGED), W(ARM), C(OLD), S(ELECTIVE), Z(IP),
        T(SN FILE RESET ONLY), J(OIN AND TSN FILE RESET), EOT=UNCHANGED)
```

In FAST and AUTOMATIC modes, this decision can only be made using the system parameter STUPTYPE.

In DIALOG startup, the following types can be specified:

- U=Unchanged

  The startup is the type defined by systems support with the system parameter STUPTYPE.

- W=Warm start

  This is the default setting for the STUPTYPE system parameter.

  All jobs that were in the JMS job queue (listed according to user ID and task sequence number for type 1 and type 2) and which were not processed during the previous session are now scheduled for processing by job control.

- C=Cold start

  This type of startup ignores any jobs in the job queue. They are deleted without being processed. If, for example, a disk error in the job queue files has caused a system breakdown, the next session must always commence with a cold start. Because after a disk error, the queue must be completely recreated. A cold start should be avoided whenever possible, since it causes loss of one-off and repeat jobs. In the case of one-off and repeat jobs, the JMP utility routine can be used to reconstruct the necessary ENTER-JOB commands prior to the cold start (see section "JMP: reconstruction of batch jobs").

- T=First start with resetting of the TSN file

  This type of startup must be used the first time a pubset is used.

  The resetting of the TSN file ($TSOS.TSOSJOIN.TSNN) causes the JMS job queue and the TSN file to be reset. This means that existing jobs are deleted, the session number is reset to 1 and the next job number (tsn) the system assigns is 0001 or 0AAA, depending on the TSN mode.

- J=First start with resetting of the user catalog

  The user catalog, JMS job queue and TSN file are all reset.

  The resetting of the user catalog causes an existing user catalog to be deleted and a new one to be generated. This new user catalog contains entries for the system user IDs only (SERVICE, SYSAUDIT, SYSDUMP, SYSGEN, SYSHSMS, SYSMAREN, SYSNAC, SYSPRIV, SYSSNAP, SYSSNS, SYSSOPT, SYSSPOOL, SYSUSER, SYSROOT, SYSOPR, SYSWSA and TSOS).

  > **CAUTION!**
  > All files except for those under the TSOS ID are deleted. This means that access (via the catalog) to all user files except the system files is no longer possible. The storage space previously occupied by these deleted user files is also released.

  The user IDs of the system (with the exception of TSOS and SERVICE) are assigned the account number SYSACC and are locked. Systems support can release these user IDs by means of the UNLOCK-USER command. If that is the case, they should be assigned system access control attributes.

  The resetting of the TSN file ($TSOS.TSOSJOIN.TSNN) causes the JMS job queue and the TSN file to be reset. This means that existing jobs are deleted, the session number is reset to 1 and the next job number (tsn) the system assigns is 0001 or 0AAA, depending on the TSN mode.

- S=Selective start

  Selective startup enables the operator to specify that certain queues are to be retained; the remainder are handled by the system as for a a cold start.

  After entering S in response to message `NSI6005` and responding to the message `NSI6010`, the following message appears:

```
? JOBP... JMS0510 SPECIFY QUEUE(S) OF BATCH JOBS TO BE SAVED WITH
          SELECTIVE START. REPLY: (1: AWAITING PROCESSING; 2: BEING
          PROCESSED)
```

  By responding to the `JMS0510` message the operator can specify the JMS jobs (queues Q1 and Q2) which are to be retained.

  Responses have the following format: `JOBP.`n

  where
  n=1 is the job queue for batch jobs to be processed and
  n=2 indicates the batch jobs that were being processed.

  If `JOBP.` is specified, no queues are selected.

  Example

```
JMS0510 SPECIFY QUEUE(S) OF BATCH JOBS TO BE SAVED WITH SELECTIVE
        START. REPLY: (1: AWAITING PROCESSING; 2: BEING PROCESSED)
JOBP.1
```

  The old queue files (JOBPOOL file for the JMS jobs in queues Q1 and Q2) are then checked to see whether they contain any jobs which should be retained. If this is the case, all entries made for jobs to be processed are transferred to the new queue file and the associated counters set accordingly. Once the files have been read they are closed and deleted from the catalog. The new files will be processed in the new session.

- Z=Zip start

  This startup type is used when there is not enough disk space available for a normal start. When “Z” is entered in response to message NSI6005, no attempt is made to allocate disk space for queue files. This mode is only intended for buffering or deleting files after system initialization in order to obtain free storage space on the disks. No other functions should be performed in this session.

By replying the following message in DIALOG startup, the operator can similarly determine whether and how the user catalog is to be reconstructed:
(see the description of the system parameter RECONUC)

```
NSI6010 SYSTEM PARAMETER RECONUC = (&00). SHALL VALUE BE CHANGED?
        REPLY ( U(NCHANGED), N(O), B(ACKUP), T(SOSCAT), A(LL), R(ESET),
        EOT=UNCHANGED )
```

2.2.6 Example of DIALOG startup

The following example illustrates the stages in a DIALOG startup (in extracts) on an SU /390. It corresponds to the logging in the CONSLOG file (partially abbreviated); the message sequence output at the console may differ from the sequence given below.

```
TCLOG    .125439  ***2021-01-25*** 000001 **** UTC+01:00
************************************************************
(C0) %   P-000.125439 NSI00E3 IPL-REPS READ: 1; EXECUTED: 1
(C0) %   P-000.125439 NSI1100 IPL DEVICE = MH20.0; IPL PATH = BE18 (MN=BE18)
(C0) %   P-000.125439 NSI1106 *** BS2000 DIALOGUE STARTUP ***
(C0) ?   P-000.125439 NSI1110 ENTER OPTIONS OR EOT. REPLY (UNLOCK,TEST,ALLDISK,DRV-SELECT,
CREATE-DRV,IPL-CONF)
   P R(C0)-000.125439.
-------------------------------------------------------------------------------  1.
(C0) %   P-000.125439 NSI3135 IPL DISK-SETUP READ FROM IPL-CONF PREPARED 2021-01-12 13:29:55
(C0) %   P-000.125439 NSI1143 002 PUBVOLS OF HOME-PVS MH20 WITH SYSID 117 ONLINE
-------------------------------------------------------------------------------  2.
(C0) %   P-000.125439 NSI1145 MH20.0 2021-01-25 12:53:37 ON BE18
(C0) %   P-000.125439 NSI1145 MH20.1 2021-01-25 12:53:37 ON BE19
(C0) %   P-000.125439 NSI1153 STATE OF PROCESSORS ONLINE:
(C0) %   P-000.125439 NSI1155 CPU 00 ONLINE, ATTACHED (IPL CPU)
(C0) %   P-000.125439 NSI1158 CPU 01 ONLINE, DETACHED
(C0) %   P-000.125439 NSI1163 LOCAL DATE = 2021-01-25, TIME = 12:53:52 FROM SVP
-------------------------------------------------------------------------------  3.
(C0) %   P-000.125439 NSI1180 LOAD ADDRESS OF SYSPRG.STRT.210 = 00777000
(C0) %   P-000.125439 NSI00E6 SYSREP.STRT.210 IS USED AS REP FILENAME
-------------------------------------------------------------------------------  4.
(C0) ?   P-000.125439 NSI0050 SPECIFY PARAM FILE OR DEVICE.
                      REPLY (EOT (USE STANDARD FILE);FN=FILENAME(,VOL=VSN);CONS;END)
   P R(C0)-000.125439.
-------------------------------------------------------------------------------  5.
(C0) %   P-000.125439 NSI00E6 SYSPAR.BS2.210.VM07S700 IS USED AS PARAM FILENAME
(C0) %   P-000.125439 NSI006B FILE SYSPAR.BS2.210.VM07S700 NOT FOUND
(C0) %   P-000.125439 NSI00BC LAST PARAM FILE/DEVICE IGNORED
(C0) %   P-000.125439 NSI00E6 SYSPAR.BS2.210 IS USED AS PARAM FILENAME
(C0) ?   P-000.125439 NSI0050 SPECIFY PARAM FILE OR DEVICE.
                      REPLY (EOT (USE STANDARD FILE);FN=FILENAME(,VOL=VSN);CONS;END)
   P R(C0)-000.125439.END
-------------------------------------------------------------------------------  6.
(C0) ?   P-000.125439 NSI1190 ENTER BS2000-FILENAME. REPLY (FILENAME(,VOL=VSN); EOT (USE
STANDARD FILE))
   P R(C0)-000.125439.
-------------------------------------------------------------------------------  7.
(C0) ?   P-000.125439 NSI0050 SPECIFY BS2000 REP FILE OR DEVICE.
                      REPLY (EOT (USE STANDARD FILE);FN=FILENAME(,VOL=VSN);CONS;END)
   P R(C0)-000.125439.
-------------------------------------------------------------------------------  8.
(C0) %   P-000.125439 NSI00E6 SYSREP.BS2.210 IS USED AS REP FILENAME
(C0) ?   P-000.125439 NSI0050 SPECIFY BS2000 REP FILE OR DEVICE.
                      REPLY (EOT (END); FN=FILENAME(,VOL=VSN);CONS;END)
   P R(C0)-000.125439.END
-------------------------------------------------------------------------------  9.
(C0) %   P-000.125439 NSI0028 AVAILABLE MEMORY = 4.0 GB. CPU = 60029001 26000000
------------------------------------------------------------------------------ 10.
(C0) %   P-000.125439 NSI5103 BS2000 : NAME = G10BXS, VERSION = 21.0A00, GEN-TIME = 2020-10-
06 17:34:20
(C0) %   P-000.125439 NSI5102 COPYRIGHT (C) 2021 FUJITSU TECHNOLOGY SOLUTIONS GMBH. ALL
```

```
RIGHTS RESERVED
(C0) %   P-000.125439 EMM2960 EFFECTIVE SYSSIZE VALUE IS 240 MB
(C0) %   P-000.125439 NSI5150 CPU'S ONLINE: 2  CPU'S INSTALLED: 2
(C0) %   P-000.125439 ETMGT10 GTIME-VALUES FOR SESSION: ZONE=+01:00, DIFF=1:00
(C0) %   P-000.125439 ETMGT01 TODR-EPOCH FOR SESSION: EPOCH=00
(C0) %   P-000.125439 NSI3130 IPL-CONF SAVED ONTO DISK MN=BE18 WITHOUT ERRORS
(C0) %   P-000.125439 NSI5104 IOCF: NAME = SU700001EM-2 29001 26 , 10.10.2020 GEN-TIME =
2021-01-10 11:04:17
(C0) %   E-000.125439 EMM2301 PAGING AREA ON VOLUME MH20.0 SIZE = 02560.00 MB.
(C0) %   E-000.125439 EMM2301 PAGING AREA ON VOLUME MH20.1 SIZE = 02560.00 MB.
(C0) %   E-000.125439 EMM2850 THE SIZE OF THE PAGING-AREA IS 0005120.00 MB;
                     THE RESERVED SIZE OF THE PAGING-AREA IS 0000011.48 MB.
(C0) %   P-000.125439 NSI6102 BS2000 EXECUTIVE: CLASS2 ACTIVE
------------------------------------------------------------------------------- 11.
(C0) %   P-000.125439 NSI4110 /INPUT DISK F=SYSPAR.BS2.210
------------------------------------------------------------------------------- 12.
(C0) %   P-000.125439 NSI4110 /BEGIN GTIME
(C0) %   P-000.125439 NSI4110 ZONE=+00:00
(C0) %   P-000.125439 NSI4110 DIFF=1:00
(C0) %   P-000.125439 NSI4110 SEASON=S
(C0) %   P-000.125439 NSI4110 EPOCH=00
(C0) %   P-000.125439 NSI4110 CHDATE=1900-01-01/00:00
(C0) %   P-000.125439 NSI4110 CHDATE=1980-03-16/02:00
(C0) %   P-000.125439 NSI4110 CHDATE=1980-10-26/03:00
...
(C0) %   P-000.125439 NSI4110 CHDATE=2024-03-31/01:00
(C0) %   P-000.125439 NSI4110 CHDATE=2024-10-27/02:00
(C0) %   P-000.125439 NSI4110 NEXTZONE
(C0) %   P-000.125439 NSI4110 ZONE=+01:00
(C0) %   P-000.125439 NSI4110 DIFF=1:00
(C0) %   P-000.125439 NSI4110 SEASON=S
(C0) %   P-000.125439 NSI4110 EPOCH=00
(C0) %   P-000.125439 NSI4110 CHDATE=1900-01-01/00:00
(C0) %   P-000.125439 NSI4110 CHDATE=1980-04-06/02:00
(C0) %   P-000.125439 NSI4110 CHDATE=1980-09-28/03:00
...
(C0) %   P-000.125439 NSI4110 CHDATE=2024-03-31/02:00
(C0) %   P-000.125439 NSI4110 CHDATE=2024-10-27/03:00
(C0) %   P-000.125439 NSI4110 NEXTZONE
(C0) %   P-000.125439 NSI4110 /EOF
(C0) %   P-000.125439 NSI4110 /BEGIN OPR
(C0) %   P-000.125439 NSI4110 DEFINE-CONSOLE CONSOLE=C0
(C0) %   P-000.125439 NSI4110 SET-CODE CODE=*ALL,CONSOLE=*IPL
(C0) %   P-000.125439 NSI4110 SET-CODE CODE=*ALL,CONSOLE=(CON1,CON2,CON3,CON4,CON5,CON6,CON7)
(C0) %   P-000.125439 NSI4110 SET-CODE CODE=T,CONSOLE=MARE
(C0) %   P-000.125439 NSI4110 SET-CODE CODE=E,CONSOLE=MARE
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE E,ADD-FILE-LINK
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE $,MODIFY-SPACE-SATURATION-LEVELS
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE $,MODIFY-USER-PUBSET-ATTRIBUTES
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE $,PRINT-DOCUMENT
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,FILE
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,SEVER
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,CATJV
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,FSTATUS
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,SHOW-SPOOL-CHARACTER-SETS
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,CATM
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,SHOW-SPOOL-DEVICES
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,COPY
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,IMPORT
```

```
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,SHOW-SPOOL-FORMS
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,DCLJV
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,JOIN
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,SHOW-SPOOL-PARAMETERS
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,ERAJV
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,SHOW-USER-ATTRIBUTES
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,ERAM
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,PASSWORD
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,STAJV
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,ERASE
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,PRINT
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,ADD-MASTER-CATALOG-ENTRY
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,IMPORT-FILE
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,ADD-PASSWORD
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,LOCK-USER
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,ADD-USER
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,MODIFY-FILE-ATTRIBUTES(C0) %   P-000.125439
NSI4110 SET-CMD-CODE @,COPY-FILE
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,MODIFY-FILE-GENERATION-SUPPORT
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,CREATE-FILE
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,MODIFY-FILE-GROUP-ATTRIBUTES
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,CREATE-FILE-GENERATION
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,MODIFY-JV-ATTRIBUTES
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,CREATE-FILE-GROUP
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,MODIFY-MASTER-CATALOG-ENTRY
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,CREATE-JV
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,MODIFY-USER-ATTRIBUTES
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,CREATE-TAPE-SET
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,PRINT-FILE
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,DELETE-FILE
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,REMOVE-MASTER-CATALOG-ENTRY
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,DELETE-FILE-GENERATION
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,DELETE-FILE-GROUP
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,REMOVE-USER
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,DELETE-JV
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,SHOW-FILE-ATTRIBUTES
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,DELETE-TAPE-SET
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,SHOW-JV-ATTRIBUTES
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,DELETE-SYSTEM-FILE
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,SET-FILE-LINK
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,EXPORT-FILE
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,SET-JV-LINK
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,REMOVE-JV-LINK
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,SHOW-JV-LINK
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,EXTEND-TAPE-SET
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,UNLOCK-USER
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,WRITE-SPOOL-TAPE
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,REMOVE-PASSWORD
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,TYPE
(C0) %   P-000.125439 NSI4110 SET-CMD-CODE @,SHOW-LIVE-MIGRATION-HISTORY
(C0) %   P-000.125439 NSI4110 ADD-CMD-ENTRY A-C=R,C-N=CMD1,A-N=CON1
(C0) %   P-000.125439 NSI4110 ADD-CMD-ENTRY A-C=A,C-N=CMD2,A-N=CON2
(C0) %   P-000.125439 NSI4110 ADD-CMD-ENTRY A-C=U,C-N=CMD3,A-N=CON3
(C0) %   P-000.125439 NSI4110 ADD-CMD-ENTRY A-C=A,C-N=CMD4,A-N=CON4
(C0) %   P-000.125439 NSI4110 ADD-CMD-ENTRY A-C=U,C-N=CMD5,A-N=CON5
(C0) %   P-000.125439 NSI4110 ADD-CMD-ENTRY A-C=R,C-N=CMD6,A-N=CON6
(C0) %   P-000.125439 NSI4110 /EOF
(C0) %   E-000.125439 EMM2310 THE SIZE OF MAIN MEMORY IS 00004096 MB; THE MINIMAL SIZE OF
MAIN MEMORY IS 00004096
```

```
MB
(ZA) % UCO-000.125441  %  NBR0706 TEST OUTPUT FOR CONSOLE 'ZA'
...
(C0) % UCO-000.125441  %  NBR0706 TEST OUTPUT FOR CONSOLE 'C0'
  <* % UCO-000.125441  %  EXC0903 CONSOLE 'ZA' INOPERABLE
(C0) % UCO-000.125441  %  EXC0652 CONSOLE 'C0' TAKES OVER ALL FUNCTIONS AND AUTHORIZATIONS
OF CONSOLE 'ZA'
...
  <* ? TSC-000.125442 NSI0077 ENTER AUTOMATIC COMMAND FILE NAME.
                      REPLY (FILE NAME; N(DO NOT USE); EOT(USE STANDARD))
   TSC R(C0)-000.125457.
------------------------------------------------------------------------------------- 13.
  <* % UCO-000.125457  %  NBR0792 QUESTION '000' FROM ' TSC' ANSWERED BY '(C0)'. REPLY:
  <* ?   Q-000.125457  %  NSI6005 SYSTEM PARAMETER STUPTYPE = W. SHALL VALUE BE CHANGED?
REPLY ( U(NCHANGED),
W(ARM),
                         C(OLD), S(ELECTIVE), Z(IP), T(TSN FILE RESET ONLY), J(JOIN AND TSN
FILE
RESET),EOT=UNCHANGED)
   Q R(C0)-000.125500.C
------------------------------------------------------------------------------------- 14.
  <* % UCO-000.125500  %  NBR0792 QUESTION '000' FROM '   Q' ANSWERED BY '(C0)'. REPLY: C
  <* ?   Q-000.125500  %  NSI6010 SYSTEM PARAMETER RECONUC = N. SHALL VALUE BE CHANGED?
                         REPLY ( U(NCHANGED), N(O), B(ACKUP), T(SOSCAT), A(LL), R(ESET),
EOT=UNCHANGED
   Q R(C0)-000.125504. N
------------------------------------------------------------------------------------- 15.
  <* % UCO-000.125504  %  NBR0792 QUESTION '000' FROM '   Q' ANSWERED BY '(C0)'. REPLY:
  <* %   Q-000.125504  %  DMS035B IMCAT TASK 'XABW' FOR PUBSET WITH PUBSET ID 'MH20' CREATED
AND STARTED
...
  <R %DSSM-000.125504  %  ESM0220 FUNCTION 'CREATE' FOR SUBSYSTEM 'CALENDAR/V21.0'
COMPLETELY PROCESSED
...
  <* %   M-000.125505  %  NMH1102 MESSAGE OUTPUT FILE ':MH20:$TSOS.SYSMES.BS2CP.210',
                         ACCESS=DLAM + ISAM, ACTION=STARTUP
  <* %   M-000.125505  %  NMH1102 MESSAGE OUTPUT FILE ':MH20:$TSOS.SYSMES.EKP.01',
ACCESS=ISAM, ACTION=STARTUP
  <* %   M-000.125505  %  NMH1112 MESSAGE PROCESSING READY
...
  <* %SERS-000.125515  %  NER1500 SERSLOG LOGGING FILE ':MH20:$TSOS.SYS.SERSLOG.2021-01-
25.025.01' OPENED
  <H % TSC-000.125515  %  HEL0001 HW ERROR LOGGING FILE ':MH20:$TSOS.SYS.HEL.2020-11-
28.154947'
                         OPENED WITH 'SPACE=  402'
  <R % TSC-000.125515  %  EXC0040 LOGGING FILE ':MH20:$SYSAUDIT.SYS.REPLOG.2021-01-
25.025.01' OPENED
...
  <@ %HT1C-000.125516  %  CMD0695 SYSTEM SYNTAX FILE ':MH20:$TSOS.SYSSDF.SDF.048' ACTIVATED
  <@ %HT1C-000.125516  %  CMD0695 SYSTEM SYNTAX FILE ':MH20:$TSOS.SYSSDF.ACS.210' ACTIVATED
...,
  <A % TSC-000.125522  %  NJA0001 D.E.R INITIALIZED SUCCESSFULLY
  <* % MSG-000.125522  %  ETMRK18 CPU 01 ATTACHED
  <* %ACCT-000.125522  %  NAM0001 NEW ACCOUNTING FILE ':MH20:$TSOS.SYS.ACCOUNT.2021-01-
25.025.01'
                         OPENED WITH 'SPACE=(48,48)'
  <* % DRV-000.125522  %  DRV1010 FILE ':MH20:$TSOS.SYSPAR.DRV.032' IS USED AS DRV
CONFIGURATION FILE.
  <* % TSC-000.125522  %  NSI0000  ***  S Y S T E M   R E A D Y  ***
```

```
------------------------------------------------------------------------------------- 16.
  <J % TSC-000.125522  %  JMS0066 JOB 'JSTREAM' ACCEPTED ON 17-01-25 AT 12:55, TSN = 59Y2
  <R % TSC-000.125522  %  JMS0300 JOB STREAM '$SYSJS' 'ATTACHED'
  <J %59Y2-000.125522  %  JMS0154 'TSOS' LOGGED ON FOR 'NTL'. JOB NAME 'JSTREAM'. CALLER
'TSN  TSC'. TID 00020050
  <R %59Y2-000.125522  %  JMS0300 JOB STREAM 'JSTREAM' 'ATTACHED'
...
  <* %SEST-000.125522  %  NBR3000 SYSTEM EVENT STREAM FILE ':MH20:$SYSAUDIT.SYSLOG.ESS.
SYSTEM' OPENED
OPRR /(C0)-000.125522 RUN CMDFILE
------------------------------------------------------------------------------------- 17.
  <* % UCO-000.125522  %  NBR0972 OPERATOR TASK WITH TSN 'XABY' CREATED FOR PROCESSING OF
/RUN COMMAND SEQUENCES
(C0) +RUNT-000.125522  %  NBR1000 COMMAND '/RUN' RECEIVED. RUN ID '0001' ASSIGNED
(C0) +RUNT-000.125522  %  NBR1001 RUN ID '0001'. READING OF /RUN COMMAND FILE ':MH20:$TSOS.
CMDFILE' STARTED
OPRR /(C0)-000.125522 DCSTART DCSOF=SOF.ABGSE211.SE
OPRR /(C0)-000.125522 START-SUBS CRTEBASY
OPRR /(C0)-000.125522 START-SUBS SPOOL
...
```

1. DIALOG startup is to be performed without a special option (response: `EOT`).
2. The home pubset is online.
3. Date and time as per SVP are displayed
4. The SYSSTART REP file is processed.
5. The default startup parameter file should be processed (response: `EOT`).
6. No other parameter files are to be processed (response: `END`).
7. The file name for BS2000 should be entered. If the response is `EOT`, BS2000 is loaded from the default file.
8. The response `EOT` causes the default REP file to be processed.
9. No other REP files should be processed (response: `END`).
10. Various items of system information are output: name and version of the operating system, time of generation, available memory space, time zone information, TODR epoch, configuration data, size of the paging areas.
11. The paging section of the BS2000 execution section was successfully loaded and corrected (REP processing). It is thus ready for execution.
12. The parameter records selected in (5) are logged in CONSLOG. Processing takes place at different times depending on the parameter keyword.
13. The command file with the name specified by means of the system parameter CMDFILE is to be automatically processed after “System Ready” (response: `EOT`). If the response is `N` (`DO NOT USE`), a CMDFILE can subsequently be started with the RUN CMDFILE command.
14. A cold start is to be performed (response: `C`).
15. Catalog reconfiguration should be performed as defined in the parameter service (response: `N`). In this example, therefore, with RECONUC=N.
16. “System Ready”, the BS2000 system is ready to operate.
17. Automatic start of the command file `CMDFILE` (see (13)).

2.3 System corrections

- Function and structure of a REP file
- REP records
- Temporary backup of REPs
- RMS: REP delivery and installation

2.3.1 Function and structure of a REP file

At system startup, REP files, consisting of REP records, can be used to correct the SYSIPL, SYSSTART and SLED load objects and the Control System.

REP records permit byte-by-byte correction of the load modules itemized above. “Selectable units” that are not linked to Exec can also be corrected by means of REP records. The corresponding REP file is named SYSREP.<selectable unit>.<ver>. REP records cannot be used to exchange entire modules. Instead, these must be imported into the associated library (OML) with LMS.

In general, REP processing is applied to the last object loaded. Loading of the BS2000 operating system takes place in two steps; each of these is corrected individually.
Class 1 REP records are processed immediately after the resident part of the Control System is loaded. They should be created only for those class 1 modules (resident part) of the Control System that are needed for loading and initializing class 2 modules (non-resident, pageable part) of the Control System. The relocation of correction data in class 1 REP records is limited to class 1 modules and system startup modules; entries can only be relativized at the beginning of the module.
Class 2 REP records are processed immediately after the non-resident part of the Control System is loaded. These records can be used to correct all the remaining modules of the Control System. Relocation of correction data is possible for all modules and entries.

REP processing during system initialization

Except when an error arises, REP processing is executed automatically during FAST and AUTOMATIC startup, i.e. without interaction with the operator. In the case of DIALOG startup, REP processing can be influenced by the operator, except for SYSIPL and SLED.

During FAST and AUTOMATIC startup, the REPs to be processed are expected in the $TSOS.SYSREP.BS2.<ver> and $TSOS.SYSREP.STRT.<ver> files on the home pubset, unless the startup parameter file specifies other REP files for the Control System (REPFILEx parameter). A dialog with the operator will only be initiated if errors arise.

The default names of the REP files are:

- for BS2000: SYSREP.BS2.<ver>
- for SYSIPL: SYSREP.IPL.<ver> [1] copied to file SYSREP.IPL.DSKnnn
- for SYSSTART: SYSREP.STRT.<ver>
- for SLED: SYSREP.SLED.<ver> [1] copied to file SYSREP.SLED.DSKnnn

With DIALOG startup, REP files may be on a public disk or private disk or they may be input via the console.

The processing sequence is determined by the operator, who specifies an input device in response to each `NSI0050` message. The REPs are processed immediately. Thereafter, the `NSI0050` message appears again. This procedure is repeated until the operator enters `P.END` (or `P.`, if a REP file with a standard name has already been processed) as the response (see next page).

A disk file can be specified four times and the console twice as the input device. A check is made to see whether the specified file has already been processed. These restrictions apply only to BS2000 REP files. They do not apply to any other objects.

Dialog at the console is opened separately for class 1 REP records and class 2 REP records. The entered data is treated exactly as if it had been input via a REP file. It is also written to the save file SYS.NSI.SAVEREP and later to the REPLOG ($SYSAUDIT.SYS.REPLOG.<date>.<sessno>.01) file.

> *Processing REP files for SYSIPL and SLED*
>
> REPs for SYSIPL and SLED are each held in a REP file which is stored in the SVL by SIR. During system initialization or the generation of diagnostic information, the operator has no opportunity to select another REP file.
> For the same reason, the type of system initialization has no influence on this part of the REP processing.
> If these REP files are modified, they must again be incorporated in the SVL with SIR (CREATE-IPL or MODIFY-IPL function).

[1]These files are copied by SIR and are then anchored in SVL. They cannot be addressed using file names (see below).

Structure of a REP file

A REP file for system initialization has the following structure:

BS2000_LOADER [_comment]	1st record (mandatory)
Class 1 REP records	REP records for selected modules of the resident part of the Control System and system initialization (optional)
_END[_comment]	End statement for class 1 REP records (mandatory)
Class 2 REP records	REP records for the entire Control System (optional)
_END[_comment]	End statement for class 2 REP records
/[_comment]	End-of-file identifier
*%comment or *%%comment	The comment record is output to the console (does not apply to comments in SYSREP.IPL.vvv)

This structure applies equally to SYSIPL, SLED, SYSSTART and BS2000. The distinction between class 1 REP processing and class 2 REP processing is only relevant for BS2000.

Either a second END statement or an end-of-file identifier must exist as the end criterion for class 2 REP records and the REP file.

On disk, the REP records may be 1-256 bytes long, although the characters after the 80th byte are not processed. The REP file is a SAM file with variable records and standard blocking BUFFER-LENGTH=STD(SIZE=1) or (SIZE=2). The file name can be freely selected. Changes to the REP file on disk should only be made with RMS (see section "RMS: REP delivery and installation").

The REP files are read and processed in the order specified by systems support.

The console can be specified twice as the REP input device. If any faulty disk REPs were read in, these can be corrected again at the end, from the console.
Comment records (with * in column 1) can be inserted anywhere in the REP file after the record “BS2000 LOADER”. They are ignored by startup. Comment records with the % character in column 2 are logged via the console (does not apply to comments in SYSREP.IPL.vvv).

REP records for modules that are not linked into the Control System but whose names are known to the Control System are skipped without an error message. It is thus possible to integrate REP records for all modules of a given BS2000 version in a REP file.

REP records containing invalid module names are logged as faulty. However, if a REP record contains an “S” or “U” in column 69, the error message is suppressed. In this way REP records for modules that are (still) unknown to the Control System (e.g. selectable units, "Function and structure of a REP file") can be included in the REP file.

2.3.2 REP records

The address of the data to be corrected is always specified with respect to the start of the module. Each REP record is checked for correct format before it is processed. The check data, parity digit and module version are verified only if specified, i.e. they may also be omitted for test purposes. Faulty records are logged together with an error message at the console; the records themselves are not corrected.

Correction data depending on the address of another module can be specified in the form “base + offset”. This type of REP is called a “relative REP”. The offset is specified relative to the start of the module or to the entry or to the ISL entry, and the base in the form of the module/entry/ISL entry name. Thus REP records are not affected by changes in other Control System modules or by regeneration of the Control System.

In the following table:

- **a** stands for an alphanumeric character (0-9, A-Z)
- **x** stands for a he xi decimal number (0-9, A-F)

Column	Contents	Meaning
1	_	Blank
2 - 4	REP	
5	_	Blank
6 -10	xxxxx	REP address relative to start of module
11	_	Blank
12 -14	3-digit number	Sequence number of the object corrections version
15	_	Blank
16	X, I, O, S, P, T	X: Standard or relative Rep I: Relative REP (ISL entry) O, S, P, T: Relative REP for x86-64 code
17 - max. 50	' xxx...x' or ' xx...x' + name	Up to 32 correction items enclosed in single quotes orup to 22 correction items enclosed in single quotes and followed by a plus sign and the name of a Control System module /entry/ISL entry. The start address of this module is added by the system startup component to the last 8 digits of the correction data. The name must be 8 characters long (as in columns 73-80). x86-64 REPs have a special REP format (see "Notes on the format of REP records")
51	_	Blank
52 - 55	xx__ or xxxx	2 or 4 check items; the first (two) byte(s) to be overwritten by the correction data must be specified.
56	_	Blank
57	x	Parity digit for REP address, correction data and check data

58 - 65	aaaaaaaa	Number of problem message

66 - 68	aaa	Module version
69	a	REP identifier
70	1 or 2	1 for class 1 REP records, 2 for class 2 REP records
71	a or blank	Loader version (A-Z)
72	a	Identifier for the selection of REP records for different code variants. 'P' identifies x86-64 code corrections. A blank identifies /390 code corrections.
73 - 80	aaaaaaaa	Module name, 8 characters long. The start address of this module is added to the REP address

Table 2: Format of REP records

Notes on the format of REP records

- Class 1 REP records and class 2 REP records both have the same format and permit the same correction functions. They differ only in the class label in column 70 and in their use, and in the option of specifying entries /ISL entries for “relative REPs”.
- The address of the data to be corrected is always derived by adding the module address (name in column 73ff) and the REP address.
- To allow a distinction to be made in relative REPs, the indicators `I, O, S, P,` and `T` have been introduced to complement the previous format with the indicator `X`. The REP data follows the indicator and is enclosed by quote marks.
  - Indicator `X` designates real modules/entries. Indicators `P` and `T` designate special relative REPs for x86-64 code. There is a special format for this:

    Format: X'distance'+base address
    The address of the real entry is noted for the base address.

  - Indicator `I` indicates that the specified entry name designates the ISL entry.

    Example X'distance'+<name-of-isl-entry>

  - Indicators `O` and `S` designate special relative REPs for x86-64 code. There is a special format for this:

    Example `O'<code>','<distance>'+<entry/module-name>`
    The correction information is assembled from the <code> and the address calculated. The <code> and <distance> must each be exactly 8 characters long.
- Since the correction data is given in hexadecimal notation, its number must always be even.
- In the case of correction data to be relocated, at least 8 correction items must be present.
- If the data to be corrected represents an address, the old value is generation-dependent, i.e. no check data should be specified.
- The parity digit (column 57) serves to safeguard the REP record contents. It is derived from the sum of all the digits of the REP address, correction data and check data.
- The number of correction items is added to the sum. The result mod(16) yields a value between 0 and F for the parity digit.

- mod(16) means that the sum is divided by 16. The remainder is the parity digit. If the parity digit is not yet present, it is automatically generated by means of the tool RMS (REP management system).
- The contents of columns 58-65 are for organizational purposes and are ignored by system startup.
- The check data, parity digit and module version may also be omitted. The corresponding tests during startup are then skipped.
- The following REP identifiers have been defined for column 69:

  `'D'` = diagnostic/interception REP
  `'O'` = optional REP
  `'Q'` = selectable unit, diagnostic/interception REP
  `'S'` = selectable unit
  `'T'` = trace (activate)
  `'U'` = selectable unit, optional
  `'V'` = temporary REP
  `' '` = normal REP
- The identifier in column 72 controls the selection of REP records for different code variants (/390 or x86-64 code). In the event of the identifier 'K', the correction is taken into account for x86-64 code only. If column 72 contains a blank, then the correction applies to /390 code. All other contents are ignored. In accordance with the procedure on the Server Units, the REPs for the other HSI are ignored.
- Instead of a module name, columns 18-49 may contain an ENTRY name in the case of class 2 REP records.
- Instead of a module name, columns 73-80 may contain an ENTRY or CSECT name. In such cases, no module version may be specified. (for class 2 REP records only)

Input of REP records at the console

During the load procedure the following message appears:

```
?P.NSI0050 SPECIFY BS2000 REP FILE OR DEVICE. REPLY (EOT (USE STANDARD
           FILE); FN=FILENAME(VOL=VSN); CONS; END)
```

If the operator’s response is `P.CONS`, the dialog for the class 1 REP records starts:

Message:

```
?P.NSI0070 ENTER CLASS 1 MODULE NAME. REPLY (NAME; EOT; (NO MORE CLASS 1
           CONSOLE REPS))
```

Response: `P.<modulename>`
(Name des Moduls, das geändert werden soll)

Message:

```
?P.NSI0071 ENTER RELATIVE ADDRESS IN MODULE. REPLY (5 CHAR)
```

Response: `P.<patch-address>`
(the relative address of a REP within the module; 5 hexadecimal digits)

Message:

```
?P.NSI0073 ENTER CORRECTION DATA. REPLY (MAX 32 CHARACTERS)
```

Response: `P.<correction-data>`
(data for patching the module object code; 2-32 hexadecimal digits)

Input of relative REPs is extended:
In addition to distance+base_address, `X'<distance>'+<base_address>`, `I'<distance>'+<isl -entry-address>` and the corresponding relative REP formats for x86-64 code may also be input.
`<base_address>` is the address of a real entry or module.

Message:

```
?P.NSI0074 ENTER CHECK INFORMATION. REPLY (CCCC,P,MMM (OLD DATA, PARITY,
           VERSION); EOT(SKIP CHECK))
```

Possible responses:

`CCCC`	1 or 2 check bytes
`P`	parity digit or empty
`MMM`	module version number or empty

A check is made on all the data entered; in the absence of data, the corresponding check is not performed.

or Response: `P.` (Press ENTER key)
When this response is given, no check is made.

Message `NSI0070` is then displayed again, requesting input of the next REP record. The same dialog is repeated until no further class 1 REP records are to be input. If this is the case, `P.` must be entered in response to message `NSI0070`, followed by pressing the ENTER key.

After further information messages, the dialog for class 2 REP records commences. It is initiated with the message

```
?P.NSI0075 CONSOLE ASSIGNED AS REP-LOADER FOR CLASS2 MODULES.
           REPLY (EOT(CONSOLE IS USED); N(NO)).
```

The response is either `P.` or `P.N` (no class 2 REP records) and press ENTER key.

2.3.3 Temporary backup of REPs

All REP records are temporarily backed up in the SYS.NSI.SAVEREP file and are logged in REPLOG. If a current REPLOG file (see "REPLOG") is not available, the SAVEREP file is saved by SLED and may be converted with DAMP.

SAVEREP (system initialization for BS2000)

Each REP record that has actually modified the system is marked with a number in column 72; the other REP records are assigned letters. where:

1 or A: REP from disk

4 or D: REP from console

The loader code is written to the BS2000 LOADER record. The loader code is generated as a function of all REP records (not the SYSIPL REPs) and serves to identify the REP file. It is derived from the sum of the REP addresses and the parity digits of all existing REP records.

For each REP medium, when a save is made, two START and two END records are entered, containing the date, time, component and the full file name to give an unambiguous identification.
REPs for any other HSI than the current one are not logged in SAVEREP (and therefore also not in REPLOG) and are also not counted in the message referring to the processing of IPL REPs.

These records are entered in the following format:

```
START <object> yyyy-mm-tt,hh:mm:ss
START <file or input device>
:
<REP and comment records>
:
END <object> yyyy-mm-tt,hh:mm:ss
END <file or input device>
```

The following values can be used for <object>:

SYSIPL: REPs for SYSIPL
SYSSTART: REPs for SYSSTART
BS2000-CL1: REPs for class 1 exec
BS2000-CL2: REPs for class 2 exec (no date or time)
VM2000: REPs for VM2000 hypervisor (SUs /390)

<file> contains the complete file name of the REP file, made up of the catalog ID, user ID and file name sections.

The following values can be used for <input device>:

IPL path: for SYSIPL REPs

*CONSOLE: for REPs which are input from the console

An additional record is inserted after the last END record for BS2000 class 2 REPs (END BS2000 record); this contains information about the number of REPs which have been processed and the number replaced:

```
END BS2000: LOADER VERSION Z, mmmmm PROCESSED REPS, nnnnn REPLACED REPS
```

During the course of system initialization the SYS.NSI.SAVEREP file is copied to the logging file $SYSAUDIT.SYS.REPLOG.<date>.<session-no>.01. The correction data for BS2000 and all dynamically loaded subsystems is logged there.

REPLOG

The correction data for SYSIPL, SYSSTART, BS2000 and all dynamically loaded subsystems and all corrections made with ROSI in the active system are logged in logging file $SYSAUDIT.SYS.REPLOG.<date>.<sessno.>.01.

> In addition to the corrections made in BS2000 and in the dynamically loaded system parts or subsystems, the old information replaced by the correction (REP before-image) is also logged in memory. This makes it possible to reliably and consistently remove corrections from the system while it is in session.

The system parameter SECSTART can be used by systems support to specify whether the actual correction status of the loaded BS2000 should be logged over several sessions. If SECSTART=N applies, any existing REPLOG files are deleted.
If SECSTART=Y applies, the REPLOG files are retained, and corrections can be traced over several sessions.

The SET-REPLOG-READ-MARK command can be used by systems support to briefly close the REPLOG file of the current session. In this way it is possible to copy and analyze all the data so far logged in a session.

Command	Meaning
SET-REPLOG-READ-MARK	Permits read access to the REPLOG file

2.3.4 RMS: REP delivery and installation

The RMS tool (REP Mounting System) is an interactive program for the installation and supply of REP packages.

The program manages all the corrections and descriptions, their origin and product affiliation as well as the scope and time of all actions in compressed form in a central file. In this way all of the information relevant for the delivery and use of the corrections is available, and every action can be reproduced at will for purposes of error diagnosis.

RMS is described in detail in the “Utility Routines” manual [15].

2.4 System termination

There are two types of system termination:

- Scheduled or “normal” terminations are terminations which are initialized by the SHUTDOWN command.
- Unscheduled or “abnormal” terminations can occur as a result of software or hardware errors.

Command	Meaning
BCEND	Terminate DCM in the server
INFORM-ALL-JOBS	Send message to all user tasks
SET-RESTART-OPTIONS	Control automatic restart
SHOW-RESTART-OPTIONS	Request information on automatic restart
SHUTDOWN	Terminate session
UNLOCK-DISK	Clean up system allocation protocol

Table 3: Overview of commands for system termination

2.4.1 Scheduled termination

Before definitively terminating the session, the operator should send a message to all interactive users with the command SHUTDOWN MODE= QUIET to warn them and give them a chance to terminate their tasks in an orderly manner.

The effect of this is that an INFORM-ALL-JOBS command is simulated to inform timesharing users, and a BCEND W=Y,TERM=N command is issued to warn inquiry and transaction applications of the imminent BCEND. After this time no other users are admitted to the system and no new jobs will be started (e.g. with ENTER-JOB).

The operator initiates definitive system termination with the command SHUTDOWN MODE= END. All tasks still in processing are terminated and a command BCEND W=N is simulated. SPOOL and job entries are saved, however, and can be processed in the next session, depending on the type of system start selected. All private disk allocations are canceled.

The command SHUTDOWN MODE=END(RESTART=*YES) can be used to initiate an automatic restart after system termination.
SHUTDOWN MODE=END(RESTART=*YES(DELAY=...)) can be used to initiate this restart after a certain amount of time has elapsed. Both are relevant to the remote control of the system.

The command SHUTDOWN MODE=END(RESTART=*YES(IPL-DEVICE=...)) can be used to perform an automatic restart with a change of IPL disk. With the exchange of the IPL disk, the restart of another system is also possible (in AUTOMATIC mode).

Guest system shutdown in VM2000 mode

In VM2000 mode a guest system can also be shut down by means of the SE Manager with the action "BS2000 Shutdown" or using the VM2000 command SHUTDOWN-VM.

The monitor system, too, can be shut down by means of the SE Manager with the action "BS2000 Shutdown for Monitor-VM" or with the VM2000 command SHUTDOWN-VM.

For detailed information, see the “VM2000” manual [60].

System shutdown by means of the SE Manager (Systems main menu)

The shutdown of BS2000 can also be initiated by means of the SE Manager in the *Systems* main menu (Server Unit selected) in the *Operation* tab with the action "BS2000 Shutdown". This shutdown is always initiated without a time limit. The shutdown request is executed in a dynamically created system task and logged on the console with the message `NRT1201`.

System shutdown by means of the SE Manager (Hardware main menu)

The shutdown of BS2000 can also be initiated by means of the SE Manager *Hardware* main menu (*Server* selected) in the *Units* tab with the action icon "Shutdown". This shutdown is always initiated on SU /390 without a time limit. Any remaining runtime which may have been set in the SE Manager is available to a BS2000 system on SU x86 to implement a proper shutdown.

The shutdown request is executed in a dynamically created system task. When the remaining runtime is set, it is logged on the console using `NRT1200 SHUTDOWN WITH RUNTIME LIMIT REQUESTED BY 'X2000'. REMAINING RUNTIME: '<HH>:<MM>'`. When no remaining runtime is set, message `NRT1201` is output, as above.

The remaining runtime is stored in the special job variable `$SYSJV.REMAINING-BS2000-RUNTIME` and is decremented as the time elapses. The job variable consists of five characters with the following possible values:

- `UNDEF` No shutdown request by X2000

- UMLIM No time limit
- <hh>:<mm>
  Remaining time up to the abrupt shutdown of BS2000 by X2000

The SHUTPROC system parameter can also be used to control whether BS2000 is shut down immediately after a request by the SE manager or whether an enter job is to be started to shut the system down.

Immediate shutdown takes place in the same way as with the command SHUTDOWN MODE=*END (RESTART=*NO).

The enter job will possibly be expected in the $TSOS.SYSENT.SHUTDOWN file. It is started under the user ID which is specified in the enter job's SET-LOGON-PARAMETERS command. The last action in the customer-specific preparations must be the command SHUTDOWN MODE=*NO/*END(RESTART=*NO). The OPERATING privilege is required to execute the SHUTDOWN command.

A sample command file for the enter job is supplied as the release item SYSENT.SHUTDOWN.TEMPLATE of the BS2CP release unit and installed using IMON. Systems support must adapt this file to the system environment and the customer-specific termination steps and make it available under the name $TSOS.SYSENT.SHUTDOWN.

For further information, please refer to the “Operation and Administration” manual [57].

2.4.2 Unscheduled termination

The following factors may cause “abnormal” session termination:

- hardware errors in peripheral devices or in the server
- software errors caused by saturation or deadlock situations
- conditions in which the system recognizes that continuation of processing is not possible.

In these cases the system cannot be terminated normally, i.e. all jobs active at this time usually have to be repeated after system recovery.

In most cases it is necessary to store the system status at the time of error by means of the SLED utility routine, so as to enable subsequent diagnosis.

The output files for SLED (see the “Diagnostics Handbook” [14]) may also be located outside the home pubset, but only on disks or pubsets that would be suitable as an IPL disk or home pubset - so not, for example, on SM pubsets or DRV private disks.

An error may have occurred which cannot be eliminated by immediately reloading the system, e.g. if the catalog or other important disk contents have been destroyed. In this case the disks have to be restored (from their last save status, see chapter "Data saving").

Private disk allocations cannot be returned, i.e. the system remains in the SVL of the disk and must, if necessary, be removed by means of the UNLOCK DISK command.

Similarly, it is no longer possible to cancel pubset allocations. The next time the system is initialized, message `NSI424A` is displayed to remind the user of this. Pubsets can be released by entering `ACCEPT` in response to this message.
During the BS2000 session the operator can reset any file locks still present for individual disks or entire pubsets using the UNLOCK-DISK command.

2.4.3 Automatic restart

“Automatic restart” is a system function which supports unmanned operation. It enables automatic reloading of the operating system after a system crash. If necessary, a system memory dump can be taken by means of SLED or SNAP without operator intervention.

Automatic restart can be activated or deactivated by the operator (see the commands SET-RESTART-OPTIONS MODE=*ON/*OFF, DUMP=*SLED/*SNAP and SHOW-RESTART-OPTIONS). By default, the function is set to off.

Preconditions for automatic restart with SLED

The preconditions for a FAST startup must have been met and no incorrect REPs or parameters may be present.

If SLED is used to take a system dump, one of the following two files must be available

- $TSOS.SLEDFILE (for SLED without parameter file)
- $TSOS.SYSPAR.SLED.<ver> (for SLED with parameter file)

If, in the case of SLED without a parameter file, the $TSOS.SLEDFILE already contains a dump (i.e. it is not logically empty), a new dump cannot be taken and the system is immediately reloaded in accordance with the specifications in the SET-RESTART-OPTIONS command.

The dump file $TSOS.SLEDFILE can also be a “large” file (more than 32 GB) and be located on the home pubset.

Preconditions for automatic restart with SNAP

The SNAP function must be enabled. This is done using the SNAP-ACTIVE-SWITCH parameter (SNAP parameter record) in the startup parameter service (see "Snapshot initialization (SNAP)") or, when the system is running, using the ACTIVATE-SNAPSHOT command.

If the system file $TSOS.SNAPFILE does not exist, it is created at startup (SNAP-ACTIVE-SWITCH=ON parameter) in the standard size or in the specified size when the ACTIVATE-SNAPSHOT command is executed.

Information on the status of SNAP is provided by the SHOW-SNAPSHOT-STATUS command.

3 Parameter service

With the aid of one or more startup parameter files, various software components are supplied with data during startup.

The following parameters are read in and passed on to the appropriate routines. Shown in parentheses are the “keywords” used in the parameter statements for identification of the software components.

- for the accounting system (ACCOUNT, "Starting the accounting system (ACCOUNT)")
- for commissioning dynamic subsystem management (DSSM, "Startup of dynamic subsystem management (DSSM)")
- for determining the system time and the universal world time (GTIME, "System time control (GTIME)"). GTIME parameters must be specified. If the GTIME parameters are not preset, the operator will have to respond to a question each time the system is initialized.
- for modifying the configuration state (IOCONF, "IOCONF parameter record")
- for presetting NK-ISAM (ISAM, "Preset values for NK-ISAM (ISAM)")
- for determining the TSN mode (JMS, "Defining TSN mode (JMS)")
- for administering the memory space (MEMORY, "Memory management (MEMORY)")
- for defining the console configuration, amending console attributes, assigning routing codes and filter levels, and suppressing particular messages at consoles (OPR, "Configuration and suppressing the output of messages at consoles (OPR)")
- for assigning the paging area on startup (PAGING, "Selection of the paging area at startup (PAGING)")
- for snapshot initialization (SNAP, "Snapshot initialization (SNAP)")
- for modifying system parameters (SYSOPT-CLASS2, "Modifying system parameters (SYSOPT-CLASS2)")
- for modifying the system installation preset values (SYSOPT-IPL, "Modifying the IPL options (SYSOPT-IPL)")
- for defining the parameters for the data communication system (BCAM)
- for defining the security officer when SECOS is used (SRPM)
- configuration data for VM2000 (VM2000)

The parameter records are described below in alphabetical order.
Exceptions: BCAM and VM2000 are described in the related product manuals and SRPM in the “SECOS” manual “Access Control” [46].

3.1 Selecting the parameter file

Automatic selection of the parameter file

If a system is to be operated on changing Server Units, a different parameter file can be used according to the Server Unit.

This parameter file is selected automatically by BS2000 in **FAST** and **AUTOMATIC** startup and in **DIALOG** startup when selecting the standard parameter file (entry of `P.`) in the following order:

- In native mode and in the VM2000 monitor system:
  1. The `$TSOS.SYSPAR.BS2.<ver>.<name>` file (for SUs /390).
     `<name>` here is the value of the `NAME` operand from the GEN statement of IOGEN which is entered in the first eight characters of the IOCF comment field of the active IOCF.
  2. The `$TSOS.SYSPAR.BS2.<ver>.<system-name>` file (for SUs x86).
     `<system-name>` is the system name that was assigned in the BS2000 configuration (option auto IPL) and can still be changed in the IPL menu (dialog screen of the SE manager or SVP menu of the allocated KVP console).
- In a VM2000 guest system:
  1. The file `$TSOS.SYSPAR.BS2.<ver>.<vm-name>` (for SU /390 and for SU x86, if the guest system is started with `/START-VM`).
     `<vm-name>` here is the VM name of the guest system.
  2. The file `$TSOS.SYSPAR.BS2.<ver>.<system-name>` (for SU x86, if the guest system is not started with `/START-VM`).
     `<system-name>` is the system name that was either assigned for the persistent VM (option auto IPL) or entered/changed in the IPL menu (dialog screen of the SE manager or SVP menu of the allocated KVP console).

     > i If the guest system is started on SU x86 with `/START-VM`, the system name will be matched to the VM name.

- If no specific parameter file is found, the file with the default name `$TSOS.SYSPAR.BS2.<ver>` is searched for.
- If none of the above-mentioned parameter files is found, any FAST or AUTOMATIC startup is interrupted and DIALOG startup is switched to for the time it takes to read in the parameter.

Dialog-controlled selection of the parameter file

In the case of a **DIALOG** startup, the operator may allocate either a disk or the console as the input source. Allocation takes places when a response is made to message `NSI0050` (see section "Input of REP records at the console" in chapter "REP records").

- Parameter records can be read in from more than one file or entered directly at the console.
- If systems support has stored the parameter file on disk, allocation is effected via the relevant file name.

- A response of P.CONS to message NSI0050 switches input to the console. The following message invites the operator to enter the keyword for the subsequent parameter records.

  ```
  NSI4120 ENTER PARAM KEYWORD: (<KEYWORD>; ? (DISPLAY LIST OF POSSIBLE
          KEYWORDS); EOT (END))
  ```

  The operator has the option of entering “?” in order to view all the possible keywords. The keyword must be entered directly. (BEGIN is generated internally for logging in the CONSLOG file).

  The following message asks for the individual parameter records to be entered:

  ```
  NSI4125 ENTER PARAM RECORD FOR KEYWORD (&00); (<PARAM>; /- (IGNORE LAST
          RECORD); EOT (END))
  ```

  This message is repeated until EOT is entered. In this case, the EOF statement is generated internally. Message NSI4120 then follows, inviting the operator to enter the next keyword.

  Entering EOT terminates input at the console; an “/END-PARAMS” record is also generated internally

The data records are read in immediately before loading the class 1 exec and prior to reading the class 1 REP records. The parameter and control records are stored in compressed form in a buffer of the startup load module.

All the records processed via the parameter service are written as messages in the CONSLOG logging file.

If readin of the parameter file is abnormally terminated, an error message followed by message NSI008F is output to the console. The operator can then specify whether the records read are to be retained or ignored. Message NSI0050 then appears. In each instance an input device may be specified. This process is repeated until the operator terminates the parameter service with END. If the parameter file contains invalid parameter service statements, these statements are output on the console so that they can be corrected or skipped by the operator immediately.

3.2 Structure and contents of a parameter file

A parameter file consists of parameter records, comment records and control records.

Parameter records are data records that are interpreted by the relevant software component only. The readin routine accepts parameter records without checking them. Comment records always start with an asterisk (*), are ignored by the parameter service and are therefore not logged.
Control records are records that contain statements for the parameter service. They always start with a slash.

BS2000 PARAMS statement

This statement is always the first statement in the parameter file. It is omitted when input is from the console.

BEGIN <keyword> statement

The specified keyword determines to which software component the subsequent parameter records belong. Possible keywords are:

- ACCOUNT
- DSSM
- GTIME
- IOCONF
- ISAM
- JMS
- MEMORY
- OPR
- PAGING
- SNAP
- SYSOPT-CLASS2
- SYSOPT-IPL

The keyword SRPM is processed by the software product SECOS to determine the user ID of the security officer. The corresponding parameter record is described in the “SECOS” manual “Access Control” [46]. The keywords BCAM and VM2000 are explained in the appropriate product manuals.

EOF statement

This statement completes the parameter section for the software component addressed by <keyword>.

ADD statement

A parameter file may also contain the names of additional parameter files to be used. The files specified in the ADD statements are read in as soon as readin of the parameter file containing the ADD statements has been completed.

A maximum of 16 ADD statements may be specified.
An ADD statement must not be followed by any statement apart from another ADD statement or the END-PARAMS statement. No further ADD statements can be specified in a parameter file defined using ADD.

END-PARAMS statement

This statement is the last statement in the parameter file.

The parameter records belonging to a software component need not necessarily be specified in succession. It is quite possible for several BEGIN statements with the same keyword and subsequent parameter records and EOF statements to be contained in the parameter file.

The file SYSPAR.BS2.<ver>.TEMPLATE is supplied as a specimen of the system parameter file. It contains

- the statements BEGIN <keyword> and EOF
- Specimen defaults for parameters, if no default values are valid without defaults (only affects the parameter record for GTIME)

  Explicit statements are only needed for GTIME. The specimen file records the legal time specifications for three time zones known at the time the version is released (Time Zone 0, 1 and 2, including details of the changeover between summer time and winter time). When operating BS2000 in one of these time zones, the specifications for the other time zones can be deleted in each case (when operating in Time Zone 3 the comment characters must be removed).
  When operating in a time zone other than the one specified, the specifications must be generated in the same form in accordance with the applicable changeover times by the customer himself.

The default values for the OPR parameter record mean that all console messages that require an explicit routing default are not output at the main console.
The specimen file must be renamed SYSPAR.BS2.<ver> for operation as a system parameter file. It can be expanded with customized defaults.

Example of a parameter file

```
/BS2000 PARAMS

*Accouning system
/BEGIN ACCOUNT
/START-ACC NAME=$RZ.
/EOF

*Paging
/BEGIN PAGING
PAGING VOLUME=(PUBA04,PUBA05)
/EOF

*Memory management
/BEGIN MEMORY
SHRSIZE 6,UNIT=1MB
ASAMCTRL 4
/EOF

*System time and season
/BEGIN GTIME
ZONE=+01:00
DIFF=1:00
SEASON=S
CHDATE=1900-01-01/00:00
CHDATE=2011-03-27/02:00
CHDATE=2011-10-30/03:00
CHDATE=2012-03-25/02:00
CHDATE=2012-10-28/03:00
/EOF

*TSN mode
/BEGIN JMS
TSN-MODE=A
/EOF

*DSSM
/ADD PAR.FILE.DSSM

/END-PARAMS
```

Every parameter file on disk must be a SAM file with variable records and standard blocking. The record length must be 1-72 bytes exclusive of the record length field or 5-76 bytes inclusive of the record length field.
For BLKCTRL=PAMKEY the hexadecimal block length is expected in the first two bytes of a block and the hexadecimal record length (i.e. record format V) in the first two bytes of a record.
In NONKEY mode the block length is taken from the simulated key within the data field (length 4 bytes).

It is possible, for instance, to use EDT to create, update and merge parameter input files.

3.3 Starting the accounting system (ACCOUNT)

The accounting system can be activated during system initialization or later by means of the START-ACCOUNTING command if the necessary statements are issued via the parameter service. For this purpose, systems support must provide the corresponding data in the parameter file with the default name SYSPAR.BS2.<ver>[.<name>].

If the accounting system is activated via the parameter service and the accounting file resides on a pubset other than the home pubset, the following problem may occur: the file cannot be opened because importation of this pubset has not been completed. For this reason the accounting file should be on the home pubset.

The keyword for the BEGIN record is **ACCOUNT**.
The maximum number of permissible parameter records is 16.

The parameter record begins with the START-ACCOUNTING or STOP-ACCOUNTING statement. The statements and their associated operands can be abbreviated, provided they remain unique.

START-ACCOUNTING statement

The /START-ACCOUNTING statement activates the accounting procedure; here, the same operands can be specified as for the command START-ACCOUNTING.

Format of the parameter record for starting the accounting system

Instruction	Operands
/START- ACC[OUNTING]	[NAME = *STD / file] [,SPACE = STD / (primary,secondary)] [,BLKSIZE = STD / (*STD,n)] [,VOLUME = *STD / vsn] [,ALT[ERNATIVE-FILES] = *NONE / file / (file,...)] [,SET[-RECORD-TYPE] = *STD / *ALL / record / (record,...)] [,ADD[-RECORD-TYPE] = *NONE / record / (record,...)] [,REMOVE[-RECORD-TYPE] = *NONE / record / (record,...)] [,ACCOUNTING-PERIOD = *STD / period] [,JOB-CLASS = *NONE / *ALL / job-class / (job-class,...)]

The START-ACCOUNTING statement may be distributed over several lines by using the continuation character (hyphen).

All continuation lines must be terminated in a syntactically correct manner, since each one is checked by the system separately. For this reason, the same operands may be specified more than once for the START-ACCOUNTING statement.

Since it might not be possible to fit all the values of a list in one line, the operands ADD-RECORD-TYPE, REMOVE-RECORD-TYPE and ALTERNATE-FILES have an additive effect. The list can then be continued in the next line using the same operands (the SET-RECORD-TYPE operand is continued using the operand ADD-RECORD-TYPE).

STOP-ACCOUNTING statement

The /STOP-ACCOUNTING statement prevents accounting records from being collected during the session.

Extract from the parameter file

```
/BS2000 PARAMS
:
/BEGIN ACCOUNT
/START-ACCOUNTING NAME=$RZ.,-                                    1.
SPACE=(99,99),-                                                  2.
SET=*ALL,JOB-CLASS=*ALL                                          3.
/EOF
:
/END-PARAMS
```

1. The accounting file is cataloged under user ID RZ with the default name SYS.ACCOUNT.<date>.xxx.nn, where <date> may be specified in the form yy.mm.dd or yyyy-mm-dd, depending on the setting of the system parameter FMTYFNLG.
2. The primary and secondary memory allocations are 99 PAM blocks each.
3. All accounting records are written and the data of all job classes is to be recorded periodically in the accounting file.

See also chapter "Accounting" for more information on the BS2000 accounting system.

3.4 Startup of dynamic subsystem management (DSSM)

Dynamic subsystem management is started during BS2000 system initialization.
All the information necessary for DSSM initialization is entered via the parameter service. This information includes the name of the static subsystem catalog and the DSSM version number. If absolutely necessary, logging of DSSM-specific data for error diagnosis may be activated at this point.

The keyword for starting subsystem management is **DSSM**.
The maximum number of permissible parameter records is 16.

Format of the parameter record for starting dynamic subsystem management

Format	Meaning
SSMCAT = name	Name of the static subsystem catalog
VERSION = version	Version number of DSSM
LOGGING = ON / OFF	Controls DSSM-specific logging for error diagnosis

Subsystems which are not automatically set up during system initialization can be started in the BS2000 session with the START-SUBSYSTEM command.

Extract from the parameter file

```
/BS2000 PARAMS
:
/BEGIN DSSM
SSMCAT=<name>                                                    1.
VERSION=<version>                                                2.
LOGGING=OFF                                                      3.
/EOF
:
/END-PARAMS
```

1. Any DSSM version can use subsystem catalogs which were generated with particular SSCM versions (currently DSSM V4.3 and SSCM V2.3).
   The control and parameter records need only be contained in the parameter file if systems support wishes to deviate from the following default values:
   SSMCAT=$TSOS.SYS.SSD.CAT.X, VERSION=043 and LOGGING=OFF.
2. The version number refers to all DSSM-specific file names (e.g. SYSLNK.DSSM.043, SYSREP.DSSM.043) and is specified with three digits (e.g. 043).
3. The statement LOGGING=OFF (default value) deactivates logging; (LOGGING=ON would generate a log of diagnostic data during startup of DSSM).

3.5 System time control (GTIME)

The parameter file contains the initialization data for the subsystem GET-TIME, which provides the user with information on the standardized world time and time offsets via the system function GTIME. The operating system also requires this information.

For information on initializing and administering the system time, see chapter "System time administration".

Systems support specifies the relationship between the system time (local time) and the universal world time UTC by means of different parameters. These give both the system and the user of the GET-TIME subsystem access to a local time and a clear time reference system (UTC) which is available across system boundaries.
Without this data (from the parameter file or interactively via the console), system initialization **cannot** be performed.

Except for an automatic restart or guest system operation, the SVP clock must contain the correct local time (system time) for system initialization.

The keyword for setting the relationship between system time and universal world time in the parameter file is **GTIME**.
The maximum number of permissible parameter records is 256.

Format of the parameter record for system time control

Format	Meaning
[NEXTZONE]	Start of a new GTIME parameter block.
ZONE = +hh:mm / -hh:mm	Time zone
DIFF = h:mm	Magnitude of time jump (difference)
SEASON = S / W	Summer/winter time prior to the first time change
EPOCH = 00 / nn	Epoch for the TODR (2 hexadecimal characters). EPOCH=00 specifies the default epoch from 1900 to 2042.
CHDATE = yyyy-mm-dd/hh:mm : CHDATE = yyyy-mm-dd/hh:mm	Time reset point 1 : Time reset point n (max. 125)
SINGLE	No future changeover dates (single season)

> Time reset points can also be managed during ongoing operation using the ADD-/ MODIFY-/REMOVE-/SHOW-CHANGE-DATE commands.
> However, changes to time reset points by means of commands must be entered manually in the parameter file if they are to apply in the next session.

NEXTZONE

This separates the GTIME parameters of different time zones from one another. This means that the data for more than one time zone may be included in the GTIME parameter file. This operand may be omitted if the parameter file contains data for only one time zone.

ZONE = -/+hh:mm
Time zone in hours and minutes.
This value describes the local, legal time zone in comparison to Greenwich Mean Time or UTC (Universal Time Coordinate).
Range: -12:00 <= hh:mm <= +11:59
For example, central European time is one hour ahead of UTC; the value to be specified is thus +01:00.
The value of ZONE **must** be specified in the parameter file.

DIFF = h:mm
Magnitude of the time jump between summer and winter time.
Range: 0:00 <= h:mm <= 9:59
The value of DIFF **must** be specified in the parameter file.
If DIFF is not 0:00, the SEASON operand and at least one CHDATE must be specified.

SEASON = S / W
Specifies whether **S**ummer or **W**inter time was applied before the **first** reset. (“winter time” is taken to be the actual standard time: “daylight saving time” which deviates from the standard time is taken to be “summer time”.)
This value must be specified for internal time calculations if a time reset is declared with the CHDATE operands.
The system function GTIME must determine the valid time from this value, even after several resets.

This value is not evaluated by the system function CTIME. Since other system functions (e.g. JMS, DMS) use CTIME internally, the notes on CTIME and SEASON which can be found in the description of the CHDATE operand should be taken into account.

EPOCH = 00 / xx
Specifies the epoch for the TODR (2 hexadecimal characters).
EPOCH=00 specifies the default epoch from 1900-01-01 to 2042-09-17.

> **!** A prerequisite for the use of a changed TODR epoch is that time stamps are no longer used which were previously obtained with a different epoch designation and stored by an application and which lie outside the range that can be displayed with the new TODR epoch. Such stored timestamps would be misinterpreted with the changed TODR epoch. This must be checked by system or application support before changing the TODR epoch.
>
> "Old" timestamps can be compared with "new" timestamps after conversion to the TODX format, see section "Conversion of TODR timestamps to TODX timestamps" in "TODR epoch". Timestamps in TODX format can always be compared because TODX values are monotonically ascending in the period from 1900 to 4317. However, such a conversion requires that it is known with which epoch designation the "old" timestamps have been created.

CHDATE = yyyy-mm-dd/hh:mm

Declaration of time reset points (1..125). The first data must begin with 1900 and the following dates must be without gaps and in ascending chronological order (see the example on the next page).
Format and range of the date:

yyyy: year (1900 <= yyyy < 2042)

mm: month (1 <= mm <= 12)

dd: day of month (1 <= dd <= 31)

hh: hour (0 <= hh <= 23)

mm: minute (0 <= mm <= 59)

The date details are used by the system at startup time to determine whether summer or winter time currently applies, and hence to determine the difference between local time and universal time, UTC. In this way the parameter file can be used over a number of time resets.

The clock resetting points are also required for the CTIME function; among other things, this converts time specifications from local time to UTC. In interpreting local time stamps, CTIME always assumes that winter time is in effect from 1900-01-01/00:00 up to the first CHDATE, even if, for example, the first CHDATE is 1994-09-25/03:00 and SEASON=S has been specified. From the point of view of the user, time stamps prior to the first CHDATE would then be wrongly interpreted as winter time stamps.
In order to minimize the impact of this problem, it is advisable to ensure that the list is complete with respect to past reset time points. To ensure the CTIME interface functions correctly, it is best to enter 1900-01-01/00:00 as the earliest date, with SEASON=S. The second CHDATE must then interpret a changeover from winter time to summer time. The system can then determine, for any earlier time which is specified, whether it is to be interpreted as summer or winter time.

The difference between two time reset points must be in the 4 to 8 month range (exception: the difference between CHDATE 1900-01-01/00:00 and the second CHDATE may be any size).

SINGLE

Specifies that there will be no summer/winter time changeover in future. This keyword should be specified immediately after the last and final changeover date (CHDATE) for the region. This feature has been introduced to future-proof BS2000 from a possible single season time adoption in certain regions. Omitting this keyword will lead to warning a message (ETMGT18) in the console when future changeover dates are missing.

Incorrect GTIME parameters will invalidate the relationship between system time and universal time (UTC), and therefore have fatal consequences such as an incorrect setting of the SVP clock!

Time changes can be made without interruption,i.e. the system is operated continuously throughout a change in time now that the TODR no longer has to contain the exact local time. The local time is determined from the contents of the TODR and a correction value (see chapter "System time administration").

The centrally supplied parameter file contains GTIME parameters for several zones. In order to select the parameters for the correct zone, the difference from the UTC for the current time zone should be set on the SVP on the SUs /390. On SUs x86 the time zone is set in the carrier system. On all servers the time zone is determined from the SVP time (STORE REAL CLOCK) (see chapter "System time administration").

If the time zone cannot be determined from the SVP time (error), the entry in the startup disk's SVL is used to select the correct parameters. If there is no valid entry there either and more than one time zone is contained in the parameter records, the operator is requested to specify a zone with the messages `ETMGT30` and `ETMGT31`. Alternatively the operator can terminate system initialization.
If the time zone cannot be determined using either the SVP time or the startup disk's SVL but the parameter file contains precisely one time zone, this is taken as the time zone which is to be set.

When a time zone is determined by the SVP time or the SVL but there is no suitable parameter record for it, messages `ETMGT35` and `ETMGT36` ask the operator about the time zone which is to be set.
Alternatively the operator can terminate system initialization.

In all cases in which system initialization is continued, the time zone which is detected or accepted is stored in the startup disk's SVL.

Extract from the parameter file

```
/BS2000 PARAMS
:
/BEGIN GTIME
ZONE=+01:00                                 1.
DIFF=1:00                                   2.
SEASON=S                                    3.
EPOCH=00                                    4.
CHDATE=1900-01-01/00:00                     5.
CHDATE=1980-04-06/02:00
CHDATE=1980-09-28/03:00
CHDATE=1981-03-29/02:00
:
CHDATE=2013-03-21/02:00                     6.
CHDATE=2013-10-27/03:00
CHDATE=2014-03-30/02:00
CHDATE=2014-10-26/03:00
CHDATE=2015-03-29/02:00
CHDATE=2015-10-25/03:00
CHDATE=2016-03-27/02:00
CHDATE=2016-10-30/03:00
:
/EOF
:
/END-PARAMS
```

1. Central Europe is specified as the time zone.
2. The difference of one hour denotes the magnitude of the jump to be made on a reset between summer and winter time.
3. It is essential to set summer time prior to the pseudo-CHDATE.
4. The default epoch from 1900-01-01 to 2042-09-17 applies to the TODR.
5. Pseudo CHDATE: as a result, winter time prevails up to the first actual CHDATE. This complies with the CTIME philosophy which assumes that winter time prevailed from 1900 to the first CHDATE entered.
6. Future reset dates are entered.

3.6 IOCONF parameter record

The BS2000 I/O tables are dynamically constructed on startup.
The channel peripherals (SUs /390) are determined from the active IOCF.
The bus and Fibre Channel peripherals (SUs x86) are determined via X2000.

The configuration state is now determined in the following sequence on startup:

1. From the presettings for the bus periphery (SUs x86):

   ATTACHED: Consoles, network/LAN devices, devices from the '50' family and all disk devices; an optional REP can be used to generate disks as DETACHED by default.

   DETACHED: All other devices.

2. The configuration states of the devices connected to SUs /390 are taken over from the IOGEN data in the IOCF. The default settings for the channels controllers and all connections are:

   ATTACHED: Channels and cluster controllers

   INCLUDED: All connections between the hardware units

3. From the startup parameter service (IOCONF) by means of the statements:

   MOD-IO-UNIT Define configuration state for a hardware unit

   MOD-IO-PATH Define configuration state for a connection

4. By means of automatic reconfiguration during startup:

   ATTACHED: All home pubset disks, paging disks, IPL console

   DETACHED: All tape devices

   The implicit states of all the hardware units and connections are determined.

The keyword for modifications to configuration states is **IOCONF**.
The maximum number of parameter records permitted for this is 256.

Incorrect parameter records are ignored. When startup has terminated, the message `NDI0550` is output. This reports the number of received parameter records and the number of incorrect records. The message `NDI0551` outputs up to 10 incorrect records together with information about the type of error in question.

The values set by means of the parameter service apply to the current session.

instruction	Meaning
MOD-IO-UNIT	Modify configuration state of a hardware unit
MOD-IO-PATH	Modify configuration state of a connection between hardware units

Table 4: Overview of the statements for the IOCONF parameter record

3.6.1 MOD-IO-UNIT statement

Format for the modification of the configuration state of a hardware unit

Instruction	Operands	Meaning
MOD-IO-UNIT	*class(unit)	Class and mnemonic device name of the hardware unit
	,STATE = ATT / DET / INV	New configuration state of the hardware unit
MOD-IO-UNIT	*DEV(*RA[NGE](unit,range))	Range of hardware units
	,STATE = ATT / DET / INV	New configuration state of the hardware units in the range

i With the exception of the explicitly specified abbreviations, all other specifications must be written out in full. It is also not permitted to leave out operand names.

***class(unit)**
Class and mnemonic device name of the hardware unit for which the configuration state is specified.

The following values can be specified for the class *class:

- `*CHA` or `*CHN` (channel)
- `*CON` or `*CTL` (cluster controller, controller)
- `*DEV` or `*DVC` (terminal unit, device)

unit (2 or 4 characters) is the mnemonic device name of a hardware unit of the specified class.

***DEV(*RA[NGE](unit,range))**
Mnemonic device name of a given hardware unit and the number of subsequent devices for which the configuration state is specified.

unit (2 or 4 characters) is the mnemonic device name of a hardware unit.

range is a decimal value between 2 and 999 which specifies the number of devices, starting at unit, for which the configuration state is to be modified.

The range specification is only permitted for terminal units (*class = *DEV/*DVC).

In the case of 4-byte mnemonics, counting continues hexadecimally from unit, while in the case of 2-byte mnemonics counting is alphabetical from unit and concludes with the digits 0 .. 9.

range = 1 is ignored and range = 0 is rejected with a syntax error.

A precondition here is that the first mnemonic must exist in the system (specified in unit).

All the following mnemonics are ignored if they are not present in the system.

Examples

*RANGE(A100,32)	A100, A101, ..., A11F
*RA(A1FF,5)	A1FF, A200, A201, A202, A203
*RA(FA,40)	FA, FB, ..., FZ, F0, F1, F2, ..., F9, GA, GB, GC, GD

STATE =
Defines the new configuration state of the hardware unit.

STATE = ATT / DET / INV
The hardware unit has the status ATTACHED, DETACHED or INVALID.

3.6.2 MOD-IO-PATH statement

Format for the modification of the configuration state of connections

Instruction	Operands	Meaning
MOD-IO-PATH	FROM = *class(unit)	Start point of the connection
	,TO = *class(unit)	End point of the connection
	,STATE = INC / REM / INV	New configuration state of the connection

> With the exception of the explicitly specified abbreviations, all other specifications must be written out in full. It is also not permitted to leave out operand names.

FROM = *class(unit)

Class and mnemonic device name of the start point of the connection for which the configuration state is specified. The following values can be specified for the class *class:

- `*CHA` or `*CHN` (channel)
- `*CON` or `*CTL` (cluster controller, controller)
- `*DEV` or `*DVC` (terminal unit, device)

unit (2 or 4 characters) is the mnemonic device name of a hardware unit of the specified class.

TO = *class(unit)
Class and mnemonic device name of the end point of the connection for which the configuration state is specified. For *class(unit) see FROM.

STATE =
This defines the new configuration state of the connection.

STATE = INC / REM / INV
The connection has the status INCLUDED, REMOVED or INVALID.

3.7 Preset values for NK-ISAM (ISAM)

The indexed sequential access method NK-ISAM of DMS supports the use of disks without PAM keys. NK-ISAM also offers possibilities for performance optimization through a reduction of disk I/Os. This requires the buffer areas of NK-ISAM, the ISAM pools, to be large enough.

These **ISAM pools** are stored in the privileged address space or in data spaces and serve as buffers for the processing of one or more NK-ISAM files. ISAM pools can be created and administered either explicitly by the user by means of special macros and commands or implicitly by the system. Both the user and the system can create task-local or task-independent ISAM pools.
Task-independent ISAM pools are automatically created on a file-specific basis in one data space.

Task-local ISAM pools can be used only by the calling task. Task-independent ISAM pools define a buffer area that is accessible to other tasks as well by specifying the pool name.

If an ISAM file is opened without having been assigned to a specific ISAM pool by the user in a command or macro, DMS uses a standard ISAM pool of the system for processing: If the file is opened with SHARUPD=NO, a task-local standard ISAM pool will be assigned, and in the case of SHARUPD=YES a task-independent standard ISAM pool.

The size of a standard ISAM pool is defined in the parameter file by systems support. The size of a user-defined ISAM pool is also specified in this file if no size was specified when the ISAM pool was created.

The keyword for the BEGIN record is **ISAM**.
The maximum number of permissible parameter records is 16.

The parameters for NK-ISAM may be specified more than once. However, only the last value entered is valid.

Format of the parameter record for preset values for NK-ISAM

LCLDFPS = number	
GLBDFPS = number	*No longer evaluated*
GLBPS = number	
LCLPS = number	
DFPPROC = *STD / *ADV[ANCED]	*No longer evaluated*
GLBDFPN =number	*No longer evaluated*
MAXDSBN = number	

LCLDFPS = number

Defines the size of the task-local standard ISAM pools in PAM pages.
Possible values: 96 £ number £ 8192
Default value: 512

GLBPS = number

Defines the minimum size of an ISAM pool in a data space in PAM pages. Each “number” is, if necessary, rounded up to the next multiple of 512 and, if necessary, once more reduced to the maximum value of 32766.
Possible values: 512 <= number <= 32766.
Default value: 512

LCLPS = number

Defines the default value in PAM pages for the pool size when setting up task-local ISAM pools using a command or macro.
Possible values: 32 <= number <= 8192
Default value: 512

MAXDSBN = number

Defines the maximum number of data spaces to be provided for task-independent ISAM pools.
Possible values: 1 <= number <= 127
Default value: 2

> The SHOW-ISAM-CACHING command informs systems support about the data spaces which are currently used as ISAM caches and which are created to accommodate cross-task ISAM pools and are administered. The number of data spaces can be changed for ongoing operation using the MODIFY-ISAM-CACHING command.

Note on the setting of MAXDSBN and GLBPS

The preset values MAXDSBN=2 and GLBPS=512 enable a maximum of approx. 4000 NK-ISAM files which are opened simultaneously in SHARUPD=YES mode to be buffered in separate ISAM pools. Any further NK-ISAM files opened in SHARUPD=YES mode are buffered in existing ISAM pools.

If you wish to ensure that (almost) every NK-ISAM file opened in SHARUPD=YES mode is buffered separately, you must initially determine how many NK-ISAM files are opened simultaneously with SHARUPD=YES. If the number of files opened simultaneously (=#FILES) exceeds the limit value of 4000, the number of data spaces required must be adjusted (with rounding to the next highest multiple of 2 GB):

```
MAXDSBN = (#FILES x GLBPS) / 1.000.000
```

Comment: As the pool size is specified in units of 2 KB, the size of a data space must be expressed in 1,000,000 x 2 KB (= 2 GB).

The paging area must be expanded if required; the data space requirement for task-independent ISAM pools must be specified as `(MAXDSBN + 1) x 2 GB`.

Extract from the parameter file

```
/BS2000 PARAMS
:
/BEGIN ISAM
LCLDFPS=100
GLBPS=512
LCLPS=512
MAXDSBN=4
/EOF
:
/END-PARAMS
```

3.8 Defining TSN mode (JMS)

Systems support can use the JMS statement in the parameter file to specify whether only numeric characters or also alphanumeric characters may be allocated by the system for the 2nd to 4th places of a user TSN during the subsequent session. If the mode is to be switched from alphanumeric to numeric in two consecutive sessions, system initialization can only be performed with a cold start. In this case the operator receives a corresponding message on the console.

The keyword for specifying the TSN mode is **JMS**.
The maximum number of permissible parameter records is 16.

Format of the parameter record for defining the TSN mode

```
TSN-MODE = A[LPHANUMERIC] / N[UMERIC]
```

TSN-MODE = ALPHANUMERIC
This declares that the system may also use alphanumeric characters for the 2nd to 4th places of a TSN.
If no value or an incorrect value is set for TSN allocation in the parameter file, this mode becomes effective for the session by default.

TSN-MODE = NUMERIC
This declares that the system may only allocate numeric characters for the TSN of a user task. If this mode follows a session in which alphanumeric characters were allocated, system initialization can only be performed with a cold start.

Extract from the parameter file

```
/BS2000 PARAMS
:
/BEGIN JMS
TSN-MODE=A                              1.
/EOF
:
/END-PARAMS
```

1. In the subsequent session the system may also allocate alphanumeric characters for the 2nd to 4th places of the TSN of a user task.

3.9 Memory management (MEMORY)

The configuration of the resources managed by the memory management function (virtual address space, main memory and expanded storage) can be defined at system initialization by means of entries in the parameter file.

> The ES-UNIT, MEM-RECONF, HSDABSIZE and PAGING-GSSIZE parameters are no longer evaluated. Their information is ignored.
> The SYSSIZE parameter is not required anymore. See the note on "Notes on the SYSSIZE parameter (size of the system address space on SUs /390)" for more information.

The keyword for the BEGIN record of the parameter service is **MEMORY**.
The maximum number of permissible parameter records is 16.

Format of the parameter record for memory management

SHRSIZE n	
ASAMCTRL n	
ALAMCTRL n	
MEM-TEST n	*Ignored on SUs x86*
SHXSIZE n	*Ignored on SUs /390*
BIG-PAGE-QUOTA n	*Ignored on SUs /390*
BIG-PAGE-SHRSIZE n	*Ignored on SUs /390*
PAGING-SATURATION-WARNING-LIMIT n [,UNIT=4KB / MB]	
PAGING-SATURATION-LIMITS-QUOTA n	

SHRSIZE n
Size of class 4 memory below the 16 MB boundary reserved for virtual address space. This address space is needed for shared code, DSSM subsystems and TU logicals.

n is the size of the reserved space in MB.
The memory areas reserved using SHRSIZE may not exceed 8 MB.
Possible values: 1 <= n <= 8
Default value: n = 2

ASAMCTRL n
This parameter is used to set the Test and Trace Facility (TTF) built into ASAM. It specifies the size of the class 3 memory to be allocated for trace output, in 4Kbyte blocks. The TTF is intended only for error diagnosis and should be activated only if needed.

n is the size of the TTF area in 4Kbyte blocks.
Permissible values: 0 £ n £ 15
Default value: 0

ALAMCTRL n
This parameter serves to set the Test and Trace Facility (TTF) which is built into ALAM. It specifies the size of the class 3 memory to be allocated for trace output, in 4Kbyte blocks. The TTF is intended only for error diagnosis and should be activated only if needed.

n is the size of the TTF area in 4Kbyte blocks.
Permissible values: 0 <= n <= 15
Default value: 0

MEM-TEST n
This parameter specifies if and when a main memory test should be performed.

n designates the option for the main memory test. Possible values for n:

0 No memory test on system start or during system operation.

1 The memory test should be performed on system start. In the case of large main memories, this may slow down system start.

2 Before any main memory page is allocated to a virtual page, the main memory page should be tested. This option may have a negative effect on system performance.

Default value: n = 0

The parameter is ignored on SUs x86.

SHXSIZE n
Size of the class 4 memory created in the area above the 16 MB boundary and below the 2 GB boundary. BLS and DSSM use this address space to load shared programs and non-privileged subsystems which are present in x86-64 code. The area is embedded in all the address spaces and can be read, but not written, in non-privileged mode.

n is the size of the area in MB and must be a multiple of 16.
Possible values for n: 16, 32, 48, ...,128
Default value: n = 64

If other values are entered, system initialization is aborted.

The parameter is ignored on SUs /390.

BIG-PAGE-QUOTA n
This parameter is used to define the target size for the proportion of main memory which is to be reserved for “big pages”. A big page is 4 MB in size and is used for CISCFW compiled code.

n is the size of main memory envisaged for big pages in %.
Possible values: 1 <= n <= 99
Default value: n = 40

The parameter is ignored on SUs /390.

i It is not always possible to create the specified percentage in big pages. This depends on various general conditions (see section "Big pages for CISCFW compiled codes (SUs x86)").

BIG-PAGE-SHRSIZE n
This parameter is used to define the size of the “shared big pages” which are created in the shared memory (class 3) and are used for CISCFW compiled code of shared programs.

n is the size of the shared big pages in MB and must be a multiple of 16.
Possible values: 0 <= n
Default value: n = 64

The parameter is ignored on SUs /390.

PAGING-SATURATION-WARNING-LIMIT n [,UNIT=4KB / MB]
This parameter defines a threshold value for the number of unreserved pages in the paging memory. If the value is below the threshold, the following warning messages are issued:

```
EXC0873 NEW PAGING MEMORY SATURATION LEVEL = 1
```

```
EXC0873 NEW PAGING MEMORY SATURATION LEVEL = 0
```

These messages warn of a saturation state level 2 (threshold value 256), which would be reached otherwise.

You can find further information on the saturation states in the section "Paging memory".

n is the number of unreserved pages in the paging memory.
Possible values: 256 <= n <= 4,194,304 if UNIT=4KB
Possible values: 1 <= n <= 16,384 if UNIT=MB
Default value: n = 256, UNIT = 4KB

PAGING-SATURATION-LIMITS-QUOTA n
With this parameter, a percentage of 100 or higher can be specified. The threshold values for the saturation states level 2 and 3 in the paging memory (in case of memory saturation) and for the saturation states level 1 and 0 (in case of desaturation) are multiplied with this percentage. This ensures that the saturation states level 2 and 3 are reported earlier than with the default settings. Accordingly, the saturation states level 1 and 0 are reported later.All threshold values refer to the number of unreserved pages in the paging memory.

You can find further information on the saturation states in the section "Paging memory".

n is the percentage of the multiplication of the threshold values.
Possible values: 100 <= n <= 1,638,500 (in percent)
Default value: n = 100

Notes on the SYSSIZE parameter (size of the system address space on SUs /390)

This parameter is not needed anymore since the area available for the system address space is calculated from the size of the user address space and the size of the entire address space resulting from this.

However, if a value is specified for SYSSIZE in the MEMORY parameter set in spite of this, then it is compared to the value calculated for SYSSIZE as a consistency check.

Possible values: 240, 256, ..., 512 MB (multiples of 16).
If any other value is entered, the system initialization is aborted.

- If a value is specified for SYSSIZE and it is less than the calculated value, then the specified value is ignored and the calculated value is used.
- If a value is specified for SYSSIZE and the value for the size of the entire address space resulting from the specified value is greater than 2 GB, then the system initialization is aborted with an appropriate message.

- If a value is specified for SYSSIZE and results in the doubling of the size of the entire address space, then the system initialization is continued with the higher value for the entire address space. An appropriate message is output that points out the inconsistency between the generated value and the value in the parameter file.

3.10 Configuration and suppressing the output of messages at consoles (OPR)

The OPR table parameterization function makes it possible for systems support to amend the configuration of the console configuration, which is logically represented by the operating system tables, during system initialization.

This affects:

- the inclusion of operator commands in the operator command table
- Changing the console properties
- changes to the routing code for operator commands
- the assignment of routing codes and filter levels to consoles
- the assignment of filter levels to consoles (does not apply to authorized applications, and, in conjunction with NBCONOPI=Y, applies only until “System ready” and after shutdown)
- the suppression of particular messages at consoles (does not apply to authorized applications, and, in conjunction with NBCONOPI=Y, applies only until “System ready” and after shutdown).

To permit any operator command to be executed, it must be known to the OPR subsystem at the time when it is entered. For this purpose, a command table is built up in memory, containing an entry for each of the commands. This entry consists of:

- the command name
- the alias name (abbreviation) for the command
- the authorization required to execute the command
- the command server identifier
- the password indicator

By means of dynamic changes or additions in the parameter service (ADD-CMD-ENTRY, SET-CMD-CODE), systems support can insert new commands in the operator command table or amend the routing code for existing commands.

Semantic errors within a parameter record, which are not detected by the startup error routine and cannot be corrected interactively, are reported to the console.

The keyword for modifying the aforementioned points in the parameter file is **OPR**. The maximum number of permissible parameter records is 384; there is also a maximum number for each of the parameter records (see following table).

Instruction	Meaning	Max. number
ADD-CMD-ENTRY	Insert operator command into the command table	100
DEFINE-CONSOLE	Define or amend the console, allocate replacement console	24
SET-CMD-CODE	Amend routing code for an operator command	100
SET-CODE	Allocate routing codes	165
SET-FILTER	Specify filter levels	40
SET-MSG-SUPPRESSION	Suppress message output	50

Table 5: Overview of statements for the OPR parameter record

If the operator logon is activated (NBCONOPI=Y), operation at physical consoles and at $CONSOLE applications with dynamic authorization names is performed in a secure mode (in terms of data security and data privacy).

When NBCONOPI=Y - and also when NBCONOPI=N - the assignments to physical consoles in the SET-FILTER and SET-MSG-SUPPRESSION statements are still possible through specification of CONSOLE=*IPL. But assignments via mnemonic device names - including the mnemonic of the IPL console - are ignored.

If specifications in the SET-CODE, SET-FILTER and SET-MSG-SUPPRESSION statements are ignored, the following message is displayed:
`NBR1202 '(&00)' PARAM SETS FOR CONSOLES ONLY ALLOWED FOR *IPL`

The following excerpt from the parameter file shows the relevant statements for modifying the console configuration in the case of NBCONOPI=N:

Extract from the parameter file

```
/BS2000 PARAMS
:
/BEGIN OPR
DEFINE-CONSOLE CONSOLE=C1,REPLACEMENT=C2                          1.
SET-CODE CODE=A,CONSOLE=(C3,C4)                                   2.
SET-FILTER 3,CODE=*ALL,CONSOLE=*IPL                               3.
SET-FILTER *ALL,CODE=*ALL,CONSOLE=C5                              4.
SET-MSG-SUPPRESSION (MSG0001,MSG0002),*ALL                        5.
/EOF
:
/END-PARAMS
```

1. Console C1 is assigned to replacement console C2.
   If C1 has already been assigned during hardware generation, the characteristics of C1 are altered accordingly.
   If C1 has not yet been entered in the console tables, a dummy entry is allocated for C1 with the specified characteristics.
   The REPLACEMENT operand is only significant with NBCONOPI=N.
   It is not evaluated if the operator logon is activated.
2. Authorization (routing) code A is assigned to consoles C3 and C4.
3. Filter level 3 is set for all authorization codes allocated for the IPL operating terminal. This has the effect that any messages distributed by means of the authorization codes that have a message weight in the range 40-59 will not be output at the IPL console.
4. For all the authorization codes allocated to console C5, filter levels 1, 2, 3, 4 and 5 are set. As a result, output of all messages distributed via authorization codes is suppressed at console C5, irrespective of their message weight (this does not apply to queries). This setting corresponds to that which could be made with the ASR NOINF command at console C5.
   The explicit specification of a mnemonic device name is only possible with NBCONOPI=N.
5. Messages with message numbers `MSG0001` and `MSG0002` are suppressed at all the consoles listed in the console table.
   If the operator logon is activated, the message suppression mechanism applies only until “System ready”.

3.10.1 ADD-CMD-ENTRY statement

Format of the statement for the insertion of operator commands in the command table

Instruction	Operands	Meaning
ADD-CMD-ENTRY	AUTHORIZATION-CODE = bs	Authorization code for the command
	,CMD-NAME = cmd	Long form of the command name
	,SAME-NAME = *NONE / cmd1..3	Specifies alias names
	,APPLICATION-NAME = name	Defines the responsible $CONSOLE application
	,PASSWORD-POSSIBLE = NO / YES	Specifies password logging

i The functions of ADD-CMD-ENTRY can also be covered by authorized user programs with the CONNECT-CMD-SERVER command.

AUTHORIZATION-CODE = bs
The authorization code for the command (one character from the set A-Z, 0-9, *, #, @ or $). This authorization code defines the matching key which the issuer of the command must supply (in the form of the character of the same name, also called routing code) before the command can be used. For special functions of authorization codes @, $ and *, see table 45 in section "Functional areas and their allocation to consoles".

CMD-NAME = cmd
The command name (original or long form) for which an entry is to be created in the operator command table. This name may be up to 30 characters long and must conform to the conventions on the naming of commands (in SDF syntax: <structured-name 1..30>).

SAME-NAME =
Specifies whether alias names (i.e. permissible abbreviations) are to be defined for the command, and if so which ones.

SAME-NAME = *NONE
Default setting: no alias names are to be declared.

SAME-NAME = cmd1..3
The set of alias names (max. 3) which are to be treated as equivalent to the command name (original name).

APPLICATION-NAME = name
The name of the $CONSOLE application which executes the command. If it has not already been done, the logical console must be generated internally with this authorization name. This application name must consist of four alphanumeric characters from the set {A-Z, 0-9, @, $, #}, of which the first must not be a digit nor a “#”. In addition, the first character should not be “@”, since this combination of characters is used for dynamic authorization naming.

PASSWORD-POSSIBLE =
Specifies whether the command being defined may contain a password (e.g. if a file password is specified in an operand). If such a password is part of the command string, this operand can be used to prevent the password from being shown in the CONSLOG system logging file if a corresponding SDF entry exists.

PASSWORD-POSSIBLE = NO
Default setting: the complete command string contains no password.
In other words, the command string will not be searched for a password before it is entered in the CONSLOG file.

PASSWORD-POSSIBLE = YES
Passwords may be contained within the complete command string. To prevent this password from being logged in the CONSLOG file, the system task CLOG (which is responsible for the CONSLOG file) is “authorized” to call SDF and to eliminate the password string before it is recorded in the CONSLOG file.

> Together, the ADD-CMD-ENTRY and SET-CODE statements can name a maximum of 384 different logical consoles. If more than 384 logical consoles are specified, only the first 384 are accepted.
>
> The length of a parameter statement is generally limited to 72 characters. You are recommended, as the following examples illustrate, to significantly abbreviate the keywords in statements or to omit them altogether.
> You must also ensure that SAME-NAME is always shorter than CMD-NAME.

Examples

```
---------1---------2---------3---------4---------5---------6---------7--
ADD-CMD-ENTRY A-C=E,CMD-NAME=SPEC-CMD,SAME-NAME=SPEC,APPL-NAM=CON7,P-P=Y
A-C-E F,SPECIAL-COMMAND-FILTER-KDOXYZ,SPECIAL-COMMAND-FILT-KXYZ,CON7,Y
```

3.10.2 DEFINE-CONSOLE statement

Format of the statement for defining or modifying consoles and allocating replacement consoles

Instruction	Operands	Meaning
DEFINE-CONSOLE	CONSOLE = name	Mnemonic device name of the console
	,REPLACEMENT = STD /name	Mnemonic device name of a replacement console
	,TELESERVICE = NO / YES	Teleservice characteristic

CONSOLE = name
Mnemonic device name (2 characters) of the console to which the assignments refer. The name refers to a console which has already been defined.

REPLACEMENT =
Mnemonic device name of a console to be used as a replacement console. If this operand is specified in conjunction with NBCONOPI=Y. It is ignored and the following message is output:
`NBR1203 OPERAND 'REPLACEMENT' IN 'DEFINE-CONSOLE' PARAMETER SET IGNORED`

REPLACEMENT = STD
Default setting: the replacement terminal is assigned by the UCON task.

REPLACEMENT = name
Mnemonic device name of a terminal defined at hardware generation or via a parameter record and which is to serve as a replacement terminal. If the specified terminal does not exist, the replacement terminal is assigned by the UCON task.

TELESERVICE =
Declaration of TELESERVICE characteristics.

TELESERVICE = NO
Default setting: the console is not to have TELESERVICE characteristics.

TELESERVICE = YES
The console is to have TELESERVICE characteristics and can no longer be a replacement console for another console, or a main console.

3.10.3 SET-CMD-CODE statement

Format of the statement for changing the authorization code of an operator command

Instruction	Operands	Meaning
SET-CMD-CODE	AUTHORIZATION-CODE = bs	Authorization code for the command
	,CMD-NAME = cmd	Long form of the command name

AUTHORIZATION-CODE = bs
The new authorization code for the command (a character from the set A-Z, 0-9, *, #, @ or $).
This authorization code defines the matching key which the issuer of the command must supply (in the form of the character of the same name, also called routing code) before the command can be used. For special functions of authorization codes @ and $, see table 45 in section "Functional areas and their allocation to consoles".

CMD-NAME = cmd
The command name (original or long form) for which the entry in the operator command table is to be amended. This name may be up to 30 characters long and must conform to the conventions on the naming of commands (in SDF syntax: <structured-name 1..30>).
The assigned authorization code (AUTHORIZATION-CODE) is also valid for the alias name assigned to the command. This also applies to the so-called special operator commands logged on with the ADD-CMD-ENTRY statement.

3.10.4 SET-CODE statement

Format of the statement for assigning authorization codes

Instruction	Operands	Meaning
SET-CODE	CODE = ac / *ALL	(All) authorization codes
	,CONSOLE = *IPL / name / (name,...)	Mnemonic name of the console

CODE = bs/*ALL
Authorization (routing) code (A..Z, 0..9, or “*, #, @, $”) assigned to a physical console or to a logical console with a generated authorization name.
It is also possible to assign all routing codes (*ALL).
Codes 0, *, $ and @ have special meanings (see "Authorization codes with special meanings" in section "Functional areas and their allocation to consoles").

CONSOLE = name / (name,...,name)/*IPL
Mnemonic name of logical consoles with generated authorization names (4 alphanumeric characters) or of physical consoles (2 alphanumeric characters) for which the assignments are to apply.
The console from which the system is loaded (IPL console) can also be addressed with the value *IPL instead of its name.
If a 4-character name is specified then a logical console with this authorization name is automatically generated if this has not already been done.
The first character of the authorization name must not be a digit, nor the character “#”, nor the character “@”.
However, it is advisable always to include special characters (#, @ or $) among the four alphanumeric characters, to clearly distinguish them from task sequence numbers (TSN).

By assigning an authorization code, a logical console is authorized for entering operator commands and for receiving messages assigned to this code.

For physical consoles, the effect of the assignment depends on the system parameter NBCONOPI.

- NBCONOPI=N

  By assigning authorization codes, physical consoles too are authorized for entering operator commands and for receiving messages assigned to this code. There is no difference to logical consoles with generated authorization names.

- NBCONOPI=Y

  Physical consoles can have explicit or implicit routing codes.

  Only a routing code assigned with the REQUEST-OPERATOR-ROLE command authorizes for both command entry and message receipt (“explicit routing code”).

  Routing codes assigned with SET-CODE are “implicit routing codes” and authorize only for message receipt (and, if required, reply) but not command entry. They also contain the routing code `*` and can only be assigned to the IPL console (value *IPL).
  Assignments to console names will be rejected (message `NBR1202`). If necessary, they are automatically transferred from the IPL console to the active main console during the session.
  The NBIMPRCA system parameter defines how long implicit routing codes remain effective (also see section “Assignment of functional areas to consoles” in chapter "Functional areas and their allocation to consoles").

> Together, the ADD-CMD-ENTRY and SET-CODE statements can name a maximum of 384 different logical consoles. If more than 384 logical consoles are specified, only the first 384 are accepted.

3.10.5 SET-FILTER statement

Format of the statement for defining filter levels

Instruction	Operands	Meaning
SET-FILTER	FILTER = *ALL / no / (no,...)	Filter level
	,CODE = *ALL / ac / (ac,...)	Authorization code
	,CONSOLE = *IPL / name / (name,...)	Mnemonic name of the console

This statement is only permitted for the IPL console in conjunction with NBCONOPI=Y. The filter levels set are then only effective until “System ready” and after shutdown.

FILTER = *ALL
All filter levels are set for the specified authorization codes. This means that messages which are distributed using the authorization codes specified in the CODE operand will be suppressed on the IPL consoles, irrespective of their weight.

FILTER = (number,...,number)
Declaration of a filter level number (digit, range 1...5) to be used for the suppression of messages sent via authorization code and weight.

CODE = *ALL
The filter is to be set for all authorization codes.

CODE = (ac,...,ac)
Authorization (routing) codes (A..Z, 0..9, or *,#,@,$) for which the filter is to be set. Setting filter levels for the authorization code @ does not affect the message output to consoles.

CONSOLE = *IPL
The specified filters are to be set for the console from which the system is loaded. For reasons of compatibility, this value is not rejected in conjunction with NBCONOPI=Y.

CONSOLE = (name,,...name)
Mnemonic device name (2 characters) of consoles at which the filter is to be set. In conjunction with NBCONOPI=Y, this value is ignored.

3.10.6 SET-MSG-SUPPRESSION statement

Format of the parameter record for suppressing message output

Instruction	Operands	Meaning
SET-MSG-SUPPRESSION	MSG-ID = msg-id / (msg-id, ...)	Message number
	,CONSOLE = *ALL / *IPL / name / (name, ...)	Mnemonic device name of the console
	STARTUP-TYPE = ANY / DIALOG / NON-DIALOG	Dependence on startup type

This statement is only permitted for the IPL console in conjunction with NBCONOPI=Y. Message suppression is then only effective until “System ready”.

MSG-ID = (msg-id, ...)
Specifies the seven-digit numbers of messages whose output at the specified console is to be suppressed.

> A maximum of 50 message numbers can be suppressed. Additional message numbers can only be suppressed after SYSTEM READY, by using the command of the same name (e.g. in the CMDFILE).

CONSOLE = *ALL
The declared messages are to be suppressed at all consoles. In conjunction with NBCONOPI=Y, this value is ignored.

CONSOLE = *IPL
The declared messages are to be suppressed at the console from which the system is loaded.
For reasons of compatibility, this value is not rejected in conjunction with NBCONOPI=Y.

CONSOLE = name / (name, ...)
Mnemonic device name (2 characters) of consoles at which the messages are to be suppressed.

STARTUP-TYPE = ANY
The messages are to be suppressed regardless of the startup type.

STARTUP-TYPE = DIALOG
The messages are only suppressed in the case of DIALOG startup.

STARTUP-TYPE = NON-DIALOG
The messages are only suppressed if startup is not a DIALOG startup, i.e. in the case of FAST or AUTOMATIC startup.

3.11 Selection of the paging area at startup (PAGING)

It is possible for systems support to use the parameter service to determine which volumes are to be used for paging. Care is required here to ensure that paging files with the name SYS.PAGING.<vsn> have been created on these volumes and that the maximum permissible number of paging files cannot be used at startup (see below).

The paging area initialized at startup must be at least 50 Mbytes in size, otherwise system initialization will be aborted with message NSI5225. If the paging area is larger than 50 Mbytes but smaller than 200 Mbytes, a warning in the form of message NSI5115 is output on the console, but system initialization continues.

By selecting appropriate paging files to be used for paging, it is possible, as early as startup, to direct paging activity to less heavily utilized volumes.
Not all the paging volumes specified for paging need to be available and usable at startup time since additional paging files can be brought into use during a session.

If systems support did not specify any parameters for paging file selection, all the SYS.PAGING.<vsn> files created on the home pubset are automatically used for paging.

The parameter service allows volumes with paging files both from the home pubset and from other pubsets to be specified.

Restrictions

- At startup time at least one paging disk must be specified or (with no parameters specified) at least one paging file must be set up on the home pubset.
- A maximum of 128 paging disks may be specified at startup time.
- Each paging file SYS.PAGING.<vsn> can consist of a number of extents, but all of these must be on the same disk. However, additional extents reduce the maximum number of usable paging files: For example, if each file has 2 extents, the startup table only accommodates 102 rather than 128 paging files; if there are 64 files, these may have a maximum of 320 extents.
- The paging files specified at startup may be distributed over a maximum of 16 pubsets.
- If a paging file cannot be found for a specified disk, this disk is ignored (message NSI5110).
- Private disks cannot be used for paging.
- Paging pubsets must be available in their entirety and exclusively; i.e. they must not be shared pubsets. They are reserved in their entirety and exclusively for the forthcoming session by startup paging initialization. In the case of an SM pubset, only the volume set which contains the paging disks is checked for external allocations.
- The system initialization device configuration stored in IPL.CONF is automatically extended by any newly specified paging disks.

The keyword for the BEGIN record is **PAGING**.

The maximum number of parameter records allowed for this is 16. Any further specifications are ignored (message NSI0049) and system initialization continues. If a syntax error is diagnosed in the PAGING statement during system initialization, this statement is rejected without specification of a reason.
On the console, the statement can either be repeated with the correct values or ignored.

Format of the parameter record for selecting the paging areas at startup time

Instruction	Operand

PAGING	VOLUME = vsn / (vsn,...,vsn)

VOLUME =
Volumes to be used for paging. The volume serial number VSN is always 6 characters in length.

The following rules apply:

- Two or more identical volume specifications are interpreted as a single entry.
- A maximum of 128 VSN specifications can be processed (16 parameter records with a maximum of 8 VSNs each). Any further specifications are ignored (message `NSI0049`) and system initialization continues.
- If all specified paging statements are ignored due to syntax errors, all paging files `SYS.PAGING.<vsn>` set up on the home pubset are automatically used for paging.
- In a BS2000 session, it is always possible to use combinations of different types of disk simultaneously for paging (with respect to performance, see also the “Performance Handbook” [37]).

Extract from the parameter file

```
/BS2000 PARAMS
:
/BEGIN PAGING
PAGING VOLUME=(PUBA01,PUBA02,PUBB01,PUBB02,PUBB03)                    1.
PAGING VOLUME=ABC.01                                                  2.
/EOF
:
/END-PARAMS
```

1. The paging files (`SYS.PAGING.<vsn>`) on the specified volumes of the pubsets with the catalog IDs A and B become part of the paging area at startup.
2. The paging file on the specified volume of the pubset with the catalog ID ABC is to be counted as part of the paging area at startup.

3.12 Snapshot initialization (SNAP)

SNAP saves BS2000-specific diagnostic information under the control of the independent non-BS2000-specific SNAP-EXEC (see the description of the SNAP function in the “Diagnostics Handbook” [14]). When this is done, BS2000 and its applications are put on hold for a maximum of 24 seconds (see the system parameter SNAPTIME).

The SNAP function is automatically activated when BS2000 starts up if the SNAP-ACTIVE-SWITCH parameter of the SNAP parameter record is not set to “OFF”.

The keyword for the snapshot parameter record is **SNAP**.
There is just one parameter for SNAP.

If an invalid value is specified in the parameter record, SNAP is implicitly deactivated (OFF) and an error message is output during startup of BS2000. If there is no SNAP-ACTIVE-SWITCH parameter, the default (ON) comes into effect.

Format of the parameter record for snapshot initialization

SNAP-ACTIVE-SWITCH = ON / ON*blank* *) / OFF

*) *blank* = 1 blank

SNAP-ACTIVE-SWITCH = ON / ON*blank*
Default: the SNAP function is to be activated for the current session.

If the system file $TSOS.SNAPFILE does not exist, it is automatically created at startup (in the standard size).

If the SNAPFILE still contains a SNAP dump from the previous session (Previous SNAP), this SNAP dump is copied in the the new session, i.e. it is provided with an index structure and written to a new file under the SYSSNAP user id.

SNAP-ACTIVE-SWITCH = OFF
SNAP is not initialized at system initialization time. It is initially not available for this session. SNAP calls are terminated with an appropriate return code. SNAP can be activated dynamically later using the ACTIVATE-SNAPSHOT command. Any Previous SNAP remains unchanged in the new session.

The parameter values ON and OFF can also be specified in inverted commas ('ON' / 'ON ' / 'OFF').

Extract from the parameter file

```
/BS2000 PARAMS
:
/BEGIN SNAP
SNAP-ACTIVE-SWITCH=ON
/EOF
:
/END-PARAMS
```

3.13 Modifying system parameters (SYSOPT-CLASS2)

Class 2 system parameters (in short: system parameters) can be preset via the startup parameter service after loading of and REP processing for the class 2 EXEC. Systems support is therefore able to react flexibly to changing specifications and objectives during system initialization.

The following restrictions must be observed:

- system parameters may not be deleted or added
- modification of particular attributes of the options (e.g. length of the permitted values) is not possible.

The keyword for modifying system parameters is **SYSOPT-CLASS2**.
The maximum number of permissible parameter records is 128.

Format of the parameter record for presetting system parameters

<sysopt> = <value>

sysopt

Denotes the name, not exceeding 8 characters, of a system parameter whose value is to be changed.

value

Denotes the value within the valid range for the specified system parameter. If a system option is assigned an invalid value, the SYSOPT statement is rejected. If more than one valid value is assigned to a system option, the last valid entry will be used.
A statement is valid if the type and the assigned value as well as the length are in keeping with the associated system option.

A brief description of all system parameters with value ranges can be found in the description of the SHOW-SYSTEM-PARAMETERS command in the “Commands” manual [27].

The specific interpretation rules are set forth below:

- Type C system parameters (character strings)

  The assigned value is interpreted as a character string. If the string is to contain blanks, the value must be prefixed with C and enclosed in apostrophes; in such cases, however, the value itself cannot contain any apostrophes. The string is entered left-justified and padded with blanks if necessary. If the specified string is longer than the maximum length of the system parameter, the SYSOPT statement is rejected.
- Type A system parameters (arithmetic values)

  The assigned value is interpreted as a positive decimal value. In cases where a minimum and/or maximum is defined for the specified system parameter, the value is checked accordingly. Otherwise, a minimum value of 0 is assumed and the maximum is calculated from the generated length (see length table).
- Type X system parameters (hexadecimal values)

  The assigned value is interpreted as a hexadecimal value and may therefore contain only hexadecimal digits (prefixed with X and enclosed in apostrophes). A minimum/maximum check is carried out as for type A on the basis of the generated length.

For type A the generated length may be 1, 2 or 4 bytes, for type X 1, 2, 4 or 8 bytes. If no maximum is defined, the following maximum value is accepted:

Length table

Type	Length = 1 bytes	Length = 2 bytes	Length = 4 bytes	Length = 8 bytes
A	255	65535	2147483647	-
H	FF	FFFF	FFFFFFFF	FFFFFFFFFFFFFFFF

Extract from the parameter file

```
/BS2000 PARAMS
:
/BEGIN SYSOPT-CLASS2
BMTNUM=32                              1.
ENCRYPT=Y                              2.
EAMSPVS=X'01'                          3..
TEMPFILE=C'#'                          4.
/EOF
:
/END-PARAMS
```

1. The number of input/output buffers for catalog management is set at 32.
   Type of system parameter: A.
2. It is declared that the system is to operate with encrypted passwords during the next session.
   Type of system parameter: C.
3. This statement controls how the SYSEAM file is handled on a shared pubset.
   Type of system parameter: X.
4. "#" is declared as a special character which must be prefixed to the names of temporary files and job variables throughout the next session.
   Type of system parameter: C.

Dynamically modifiable system parameters

The following system parameters can be modified during operation by the user ID TSOS by means of the MODIFY-SYSTEM-PARAMETERS command:

BLSCOPYN	FILECRYP	NUMBACK	SSMMILOG
BLSCOPYR	FST32GB	PWACTIVE	SSMOUT
BLSLDPXS	ISBLKCTL	PWENTERD	SSMPNOQ
BLSOPENX	ISBLKVAL	PWERRORS	SSMSDEVB
CONSDDE7	JTABSMEM	PWPENTI	SSMSDEVD
DIATTL	JTMAXMEM	RDTESTPR	SYSGJASL
DUMPCL5P	JTSHMEM	SHUTARCH	SYSGJCPU
DUMPCTRL	JTSTDMEM	SHUTCTL	SYSGJPRI
DUMPSD#	MIGHOST	SHUTPROC	TCHOFLO
DUMPSEPA	NBLOGT0	SNAPTIME	TCHREAD
DUMPSREF	NBMSGCSD	SSMAPRI	TCHTACK
EACTETYP	NETCODE	SSMASEC	VMGIORAL
	NRTKILL	SSMCOPT	WRTESTPR

Nonprivileged system parameters

The following system parameters and their current values are also displayed for non privileged users by means of the SHOW-SYSTEM-PARAMETERS command:

ASRSW1	DUMPCL5P	NBACODE	NUMBACK
ASRSW2	DUMPSEPA	NBESSIZE	RDTESTPR
AUDALLOW	ENCRYPT	NBLOGENF	SECSTART
BLKCTRL	FILECRYP	NBMESLG	SECSTENF
BLSCOPYN	FMTYFNLG	NBMSGCSD	SHUTARCH
BLSCOPYR	FREFCRYP	NBOPTINT	SHUTPROC
BLSLDPXS	FST32GB	NBRCILU	SSMLGOF1
CMDFILE	HOSTCODE	NBRCSCK	SSMLGOF2
DEFLUID	ISBLKCTL	NBRCSCKN	SSMOUT
DIATTL	MIGHOST	NBRUNSP	SVC79
DMCMAXP	MSGDEST	NBRUNWT	TEMPFILE
		NETCODE	WRTESTPR

3.14 Modifying the IPL options (SYSOPT-IPL)

The parameter record SYSOPT-IPL provides systems support with an appropriate mechanism for avoiding the rigid choice between flexibility and convenience which has to be made in selecting one of the modes (DIALOG / FAST).

The DIALOG mode offers to systems support staff and the operator an advantageous way of flexibly installing new software or corrections, by means clear dialog with the operating system. However, when this is used allowance must be made for the increased execution time and a higher risk of incorrect entries, made in response to the questions.
On the other hand, the convenient FAST mode, which is non-interactive, does not allow sufficient flexibility in reacting to changed requirements (implementation of new software or corrections), as it is restricted to the use of standard file names.

By using the options of the parameter record SYSOPT-IPL, systems support can store specific instructions for how the parameter file is to be processed at system initialization time, thus providing a compromise between the FAST and DIALOG modes.
Statements of the SYSOPT-IPL type are read in and processed immediately before the BS2000 class 1 exec is loaded (for full details of the execution of system initialization, see the section "FAST startup" and section "DIALOG startup").

The keyword for modifying the IPL options is **SYSOPT-IPL**.
The maximum number of permissible parameter records is 16.

Format of the parameter record for modifying IPL options

Format	Meaning
BS2000 = file	File name for BS2000
IOCF-CHECK = NO / time	Instructions for checking the IOCF identification
REPFILE1 = datei : REPFILE4 = datei	File name for class 1/2 REPs
LINKAGE-AUDIT-SCOPE = NO / INTERRUPT-HANDLING / SYSTEM-LEVEL	Instructions for activating the function LINKAGE-AUDIT
NEW-IPL-MODE = UNCHANGED / FAST	Instructions for changing the system initialization mode

BS2000 = file
This statement specifies the name of the file containing the BS2000 class 1 EXEC and class 2 EXEC to be loaded. If this statement is present in the parameter file, the nominated file will be used for loading the resident (class 1) and pageable (class 2) parts of the control system.

If the specified file cannot be processed, this will be reported by the following message and a dialog request will be made for entry of the name of a replacement file:

```
NSI1192 INVALID FILENAME FOR BS2000 IN PARAMETER-SECTION SYSOPT-IPL
```

IOCF-CHECK =
This statement specifies whether or not a check is to be made on the identification (= generation time) of the current IOCF. This statement enables systems support to ensure that the IOCF (hardware I/O configuration) selected by the operator when starting up the server (IMPL) is the IOCF created at exactly a certain time.

IOCF-CHECK = NO
Default setting: the identification of the current IOCF will not be checked.

IOCF-CHECK = time
A check is to be made on whether the generation time of the current IOCF, logged in the I/O-CONFIGURATION-REPORT for the IOGEN, corresponds with the specified time. The value <time> must be entered in the form `yyyy-mm-dd hh:mm:ss`. If the requested check proves negative, this will be indicated by the following message:

```
NSI5206 IOCF-CHECK FAILED. GEN-TIME REQ.=yyyy-mm-dd hh:mm:ss
        IS=yyyy-mm-dd hh:mm:ss
```

REPFILE[1..4] = file
The REPFILE1, REPFILE2, REPFILE3 and REPFILE4 statements are used to specify the file names of the up to four possible files containing BS2000 REPs (object code corrections) which are to be processed. If this statement appears in the parameter file, the specified files will be used for loading the corrections during system initialization in all modes, in the order REPFILE1 to REPFILE4.
If the specified file(s) cannot be processed, this will be reported by the following message and a dialog request will be made for entry of the name of a replacement file:

```
NSI1192 INVALID FILENAME FOR REPFILEn IN PARAMETER-SECTION SYSOPT-IPL
```

LINKAGE-AUDIT-SCOPE =
This statement can be used to control whether the AUDIT functional unit, linkage AUDIT, which records the destination addresses when the instructions BASR, BALR and BASSM are called, is to be activated immediately after the class 1 exec is loaded, with a specified scope.
The linkage AUDIT function can be used to support diagnosis of problems which may have arisen during an earlier phase of system initialization (local processor linkage AUDIT). If the system parameter AUDALLOW=NO is set, then the AUDIT functions are deactivated for the duration of the session when the class 2 system parameters are set.
If required, the linkage AUDIT function can be deactivated again after “System ready” by a STOP-LINKAGE-AUDIT command.

LINKAGE-AUDIT-SCOPE = NO
Default setting: the logging function is not activated.

LINKAGE-AUDIT-SCOPE = INTERRUPT-HANDLING
Recording of destination addresses will only be activated for the SIH part of BS2000 (functional status SIH).

LINKAGE-AUDIT-SCOPE = SYSTEM-LEVEL
Recording of destination addresses will be activated for the complete privileged-execution part of BS2000 (functional statuses TPR and SIH).

NEW-IPL-MODE =
This statement can be used to specify whether the system initialization mode selected by the operator is to continue being used after the parameter file has been read in, or if the mode is to be changed to FAST.

NEW-IPL-MODE = UNCHANGED
Default setting: there will be no change from the mode established at the start of system initialization.

NEW-IPL-MODE = FAST
If system initialization was started in the DIALOG mode, it will be continued in the non-interactive FAST mode after the parameter file has been read in.

Extract from the parameter file

```
/BS2000 PARAMS
:
/BEGIN SYSOPT-IPL
BS2000=$TSOS.SYSPRG.BS2.<ver>.TEST1                              1.
REPFILE1=$TSOS.SYSREP.BS2.<ver>.REP1                             2.
REPFILE2=$TSOS.SYSREP.BS2.<ver>.REP2
NEW-IPL-MODE=FAST                                                3.
/EOF
:
/END-PARAMS
```

1. The file name for the BS2000 class 1 exec and class 2 exec to be loaded during system initialization is $TSOS.SYSPRG.BS2.<ver>.TEST1.
2. Declares that the system should read in the object code corrections for the following session from the files called $TSOS.SYSREP.BS2.<ver>.REP1 and $TSOS.SYSREP.BS2.<ver>.REP2.
3. If system initialization was started in DIALOG mode, it is to be continued in the non-interactive FAST mode after the parameter file has been read in.

4 Memory management

This chapter begins with a description of the virtual address space and the virtual memory classes. This is followed by a description of big pages. The various storage media are then presented and this is followed by a discussion of the measures for preventing saturation conditions.

Memory management for BS2000 comprises the following tasks:

- Management of system/user virtual address space
- Control of reservation, allocation and release requests for pages in
  - virtual address space
  - main memory
  - paging memory

  as well as creation and updating of tables for managing these storage areas
- Paging control
- Time monitoring of main memory page allocation
- Detection of special situations and initiation of appropriate measures for
  - saturation handling
  - Error handling

Internal mechanisms and functions in BS2000 provide systems support with a number of interfaces and tools. These are described below.

4.1 Virtual address space and virtual memory classes

The virtual address space for the system and for the users, which is implemented via main memory and paging memory, is divided into 6 classes:

- Class 1: resident system modules (e.g. termination analysis, paging routine, task management, physical input /output); resident tables
- Class 2: pageable system modules (e.g. macro and command processing)
- Class 3: resident tables that are dynamically created and cleared (e.g. TCBs, PCBs, address translation tables)
- Class 4: pageable tables that are dynamically created and cleared (e.g. job management tables, DMS tables, tables for teleprocessing); non-resident system modules and shared code that can be loaded and unloaded via DSSM
- Class 5: pageable tables that are required only by the respective tasks (e.g. TFT, I/O areas, tables for managing procedures, password table)
- Class 6: application program, common memory pools

Classes 1 to 4 constitute privileged system address space that is not user-addressable. An exception is shared code in class 4 memory, which can generally also be read and executed by users.

Class 5 is predominantly privileged user address space that serves as a communications area between the user and system address spaces. With the exception of the input/output areas it is not user-addressable.

Division of the virtual address space into system and user address space can be adapted to the installation-specific requirements by modifying the standard EXEC on Server Unit /390 appropriately (see chapter "Customizing the BS2000 Control System" in the manual "BS2000 OS DX System Installation"[55]). Moreover, further data required for memory management in this context is read in via the parameter service (see chapter "Parameter service").

Command	**Meaning**
ADD-USER	Create an entry in the user catalog
MODIFY-USER-ATTRIBUTES	Define the maximum size of the class 6 memory of a user ID (ADDRESS-SPACE-LIMIT)
SHOW-USER-ATTRIBUTES	Output of the maximum permissible size of the class 6 memory of a user ID (ADDRESS-SPACE-LIMIT field)
SHOW-MEMORY-POOL-STATUS	Output of the memory pools currently created in the system and of the tasks which are attached
Macro	**Meaning**
ALESRV	Connect a program to a data space or cancel the connection
ALINF	Output information on the access lists used for managing data spaces and their connections
CSTAT	Change status of the memory pages of a program
CSTMP	Change memory pool read/write status

DISMP	Terminate participation in a memory pool

DSPSRV	Create, expand or delete virtual address space for data addressing, output information on a data space or release this data space
ENAMP	Create memory pool or enable participation in an existing memory pool
MINF	Output information on the allocation and size of class 6 memory or a memory pool
RELM	Release a contiguous memory area of the calling program
RELMP	Release (contiguous) storage space of a memory pool
REQM	Request (additional) storage space for the calling program
REQMP	Request (contiguous) storage space for a memory pool
SHOWMP	Output of the memory pools currently created in the system and of the tasks which are attached

Table 6: Overview of interfaces for managing the virtual address space

4.2 Main memory

Main memory comprises resident pages and pages which are managed in working sets and in the free pool, see section “Page management algorithm” in chapter section "Page management algorithm".
A working set contains virtual pages which can be currently accessed but which can be relocated to paging memory.
The free pool comprises pages whose contents are either no longer needed or - at least temporarily - are to be stored in the paging memory. The free pool is filled from the working sets as required.

Main memory > 2 GB is supported in BS2000 native mode for the VM2000 monitor system and for each VM2000 guest system.

Command	Meaning
ADD-USER	Define the number of resident main memory pages for the user programs of an ID (RESIDENT-PAGES)
MODIFY-MEMORY-PARAMETERS	Define the percentage of main memory which is used for big page memory and whether and when big page memory may be reduced or enhanced
MODIFY-SYSTEM-BIAS	Define the number of resident main memory pages for user programs
MODIFY-USER-ATTRIBUTES	Change the number of resident main memory pages for the user programs of an ID (RESIDENT-PAGES)
SHOW-MEMORY-CONFIGURATION	Display the current configuration of main memory including details of big page memory
SHOW-SYSTEM-STATUS	Display the number of resident main memory pages for user programs (INFORMATION=SYSTEM-PARAM)
SHOW-USER-ATTRIBUTES	Display the number of resident main memory pages for the user programs of an ID (RESIDENT-PAGES)

Table 7: Overview of main memory management commands

4.2.1 Page management algorithm

The page management algorithm (system working set method, SYS-WS) manages the main memory pages globally and the pages are split up into five groups according to their “access age” as follows:

Group 1	Pages that have just been assigned the frame	Active groups: **System working set** Pages in these groups can be accessed without hardware interruption
Group 2	Pages that had a reference, had not run out of CPU time, or deserved a second chance	
Group 3	Pages dedicated to the review of the reference indicator	
Group 4	Pages that have been identified as candidates for displacement from the system working set during the periodic working set review	
Group 5	Read Write Queue: Pages that still need to be saved to the paging area Read Only Queue: Pages that have a copy on the paging area Empty Queue: Free Frames	Inactive group: **Free pool**

The pages in each group - more precisely the hardware frames on which the pages are located - are chained with one another.

Principle of execution

As a rule, as many pages as possible are kept active. As a result the “free pool” is relatively small. However, it must have a certain minimum size in order to satisfy page requests that follow in rapid succession. When the minimum size is reached, free pool regulation is therefore carried out with the aim of always having enough free frames in the Empty Queue.

The system working set algorithm uses the LRU (Least Recently Used) method to select the pages that remain accessible in main memory and the pages that are removed from the system working set to be pushed into the free pool:

- Modified pages are appended to the Read Write Queue because the page content still has to be saved in the paging area. By writing back pages from the Read Write Queue and then reassigning them to the Read Only Queue, the paging routine ensures that the Read Only Queue is normally never empty.
- Unmodified pages are appended to the Read Only Queue, since a copy of the page already exists in the paging area.

When a main memory page is released, the associated frame is placed in the Empty Queue.

If a main memory page is requested, the system takes a frame from group 5, fills it with the current page content from the paging area or - for newly requested pages - with zeros and then puts it in group 1. Then the page is accessible and the reference indicator is set.

Except in the case of a reclaim, in which the owner re-accesses an already deactivated page in the free pool, the system always serves page requests from the Empty Queue, which contains the unused frames. If this makes the Empty Queue too small, it is filled with pages from the Read Only Queue. This contains pages that can be overwritten because their content is already saved on the paging area. With the transition to the Empty Queue, the pages "lose" their owner and then no reclaim is possible.

The Empty Queue and the Read Only Queue form the pool from which the system can immediately satisfy memory requests. If this supply becomes too small, the system writes pages from the Read Write Queue, which contains pages with new / changed content, to the paging area. At the end of the write process, the pages in the memory can be overwritten and are placed in the Read Only Queue. To determine whether a page has been referenced, the hardware (reproduced on x86 servers by X2000) sets the associated reference indicator when the page is accessed.

The following applies to the system working set procedure:

- Referenced pages always remain in the system working set.
- Task local pages remain in the system working set even if not referenced, if the task has not consumed a certain CPU time since the last access.
- Global pages remain in the system working set again if they are not referenced, i.e. they get a second chance and are only displaced in the subsequent check if there is no reference by then.
- Fixed pages always remain in the system working set.

The system working set procedure is carried out both on-demand in the event of bottlenecks in the free pool and periodically at certain intervals or when a logical machine enters the 'IDLE' state.

The simple SYS-WS algorithm results in a reduction in paging management outlay, albeit at the expense of reduced accuracy in determining the working set requirements of the individual tasks. The reduction in outlay is especially noticeable in the case of large main memories.

If a task is deactivated, it retains its working set even if it is in the inactive state. If the task is reactivated, no “page reclaims” are necessary to restore the working set.

While the SYS-WS procedure knows the UPG value (used pages) of every task, it has no means of determining which phase the task is in (even inactive tasks have a UPG value 0.) .

Calculation of the PPC value (planned page count) is therefore approximate: the UPG value is compared with an assumed mean working set value, dependent on the category type.

Since as many pages as possible are kept active, the total UPG value is practically always in the magnitude of the available paging main memory and has only limited relevance. The working set requirements are expressed exclusively by the PPC value. This value represents only a rough estimate. With todays large memory configurations advantages result from the fact that no outlay is necessary for estimating exact memory requirements, as they are superfluous.

4.2.2 Big pages for CISCFW compiled codes (SUs x86)

Generally the main memory is managed in 4K units. These units are called frames and are assigned to the virtual pages.
A (real) “**big page**” is a collection of contiguous frames amounting to several MB, the first frame being aligned to the relevant MB multiple.

BS2000 supports big pages with a size of 4 MB, which corresponds to 1024 frames, on SUs x86. These are used for CISCFW compiled code because this runs quicker on big pages.

The big pages used can be returned by CISCFW when requested (e.g. when the main memory of a VM2000 guest system is reduced).

Parameters in the startup parameter file and the MODIFY-MEMORY-PARAMETERS command are available for defining the required number of big pages. The SHOW-MEMORY-CONFIGURATION command outputs both details of the main memory size and of the big page memory (for an example see "Adjusting the big pages during ongoing operation").

Big pages at system initialization

The startup parameter file can also contain parameters for creating big pages.

Settings in the parameter file

The following is defined in the MEMORY parameter set:

- With the BIG-PAGE-SHRSIZE parameter, the size (in MB) of the big page memory which is to be created in the shared memory (class 3 memory) and used for CISCFW compiled code of shared programs.
- With the BIG-PAGE-QUOTA parameter, the proportion (in %) of main memory which is to be reserved for big pages (required target size).

Quality and validity of the values defined at startup

How much real big page memory (real main memory of the big pages) is to be created in the system is defined in the MEMORY parameter service (BIG-PAGE-QUOTA parameter) when the system is started. The percentage specified there is only a should-be value, however. The actual value may be smaller because the creation of big page memory is linked to the maximum possible number of logical machines (CPUs) in the system. The smaller the memory size, the more CPUs there are, the greater the probability that the actual size of the big page memory will not reach the should-be size.

The values set for working with big pages apply for the current session until they are either modified explicitly using the MODIFY-MEMORY-PARAMETERS command or implicitly by certain system attributes and statuses. These include the increase or reduction of the main memory and the threat of main memory saturation.

Adjusting the big pages during ongoing operation

Big pages can be distributed evenly over the main memory because they may also be above the minimum main memory. As a result it is automatically the case that the size of the big page memory continues to correspond to the BIG-PAGE-QUOTA following main memory configuration (provided an uneven distribution of the big pages had not occurred beforehand, e.g. as a result of the values set being explicitly modified).

Otherwise the actual size of the big page memory can be modified only to a limited extent while the system is running. The MODIFY-MEMORY-PARAMETERS command offers several options for this, see "Main memory": The options are described in detail in the “Commands” manual [27].

Outputs of main memory and big page management

The size of the memory area for the big pages is output on the console with the message `EMM2309`. This message is also issued if this size is changed during ongoing operation, e.g.:

```
EMM2309 THE SIZE OF THE BIG PAGE MEMORY IS 168 MB.
```

The SHOW-MEMORY-CONFIGURATION command enables a status check on the assignment and utilization of the entire main memory and the big page areas to be implemented at any time.

Example

```
/SHOW-MEMORY-CONFIGURATION
 REAL MEMORY MANAGEMENT REPORT                                          1.
   MEM-SIZE  MIN-MEM-SIZE  FREE-CORE-SIZE  FREE-MEM-SIZE  PAGE-MEM-SIZE  CSL
    1024 MB        256 MB        32.25 MB       32.25 MB      800.25 MB  0
 BIG PAGE MANAGEMENT REPORT                                             2.
   QUOTA   PLANNED  #LM    ACTUAL     FREE     SHR   FREE-SHR       REDUCE
      40    400 MB    4    168 MB   104 MB   64 MB      16 MB   CORE/QUOTA
```

1. In the example the system has a main memory of MEM-SIZE = 1024 MB, its minimum size being MIN-MEM-SIZE = 256 MB.

   (In a VM2000 guest system MIN-MEM-SIZE can be less than MEM-SIZE. In native mode the values are currently the same.)

   FREE-CORE-SIZE is the proportion of free main memory which is still available for resident class 3 memory requests from the system. This value is also decisive for the degree of main memory saturation.

   FREE-MEM-SIZE is the proportion of free main memory which is available for the remaining resident memory requirements, such as resident memory pools or data spaces. (With a main memory <= 2 GB, FREE-CORE-SIZE and FREE-MEM-SIZE are identical.)

   PAGE-MEM-SIZE presents the main memory size which is available for non-resident, pageable pages.

   (In systems in which MEM-SIZE and MIN-MEM-SIZE are identical, FREE-MEM-SIZE and PAGE-MEM-SIZE are also identical.)

   The last column, CSL, specifies the current degree of main memory saturation, the values 0 (no saturation), 1, 2 or 3 (highest alarm level) being possible (see also "Main memory").
2. QUOTA specifies the percentage of main memory which is to be used as big page memory (and which was set with the BIG-PAGE-QUOTA parameter in the parameter file or with a preceding MODIFY-MEMORY-PARAMETERS command).

   The should-be value which was indicated with PLANNED is calculated from the main memory size (MEM-SIZE) and the percentage (QUOTA).

   #LM is the maximum possible number of logical machines (CPUs) in the system, in other words not the LMs attached at the time the command was issued. Not only the minimum size of the main memory plays a significant role for the current size of the big page memory, but also the number of LMs: the greater the number of LMs, the smaller the big page memory can be.

   The ACTUAL column outputs the current size of the big page memory, i.e. its actual size. This can deviate from the planned should-be value (PLANNED) if, for example, the big page memory is reduced because of the threat of main memory saturation.

FREE outputs the size of the free (currently unused) big page memory.

The SHR column outputs the size of the shared big page memory provided for CISCFW compiled code in class 3 memory, and FREE-SHR shows the size of the free shared big page memory.

Important information is provided in the last column REDUCE: Here you see when big page memory can be reduced. In the example, in the case of the threat of main memory saturation (CORE) and if a new should-be size of the big page memory which is less than the actual size (QUOTA) results from a modification to the main memory size or BIG-PAGE-QUOTA.

General Notes

- The following conditions always apply for main memory management:
  - MEM-SIZE >= MIN-MEM-SIZE > FREE-MEM-SIZE >= FREE-CORE-SIZE
  - MEM-SIZE > PAGE-MEM-SIZE >= FREE-MEM-SIZE >= FREE-CORE-SIZE
- CISCFW storage or DAB buffers are created in the memory designated with FREE-MEM-SIZE. Nevertheless a main memory saturation can be cleared by reducing DAB buffers or terminating programs if FREE-MEM-SIZE minus FREE-CORE-SIZE is less than the size of the DAB buffers and/or of the CISCFW storage. If this is not the case, measures for reducing the class 3 memory or for reducing the address spaces (tasks, data spaces) are required.

Notes for VM2000

- Only in VM2000 guest systems is a so-called minimum value for main memory provided. This minimum value, which falls below the overall memory size, enables the system memory to be reduced.
- In a VM2000 guest system you can get closer to the required should-be size of the big page memory by increasing the overall memory size of the guest system.
- If a memory reduction is envisaged for an active VM2000 guest system, a MIN-MEM-SIZE which differs from the MEMORY-SIZE must be defined for this VM. To permit the memory to be reduced in this way, BS2000 memory management satisfies all requests for resident memory which are below this MIN-MEM-SIZE. The value selected for this parameter must therefore be large enough. For details on this, please refer to the section “Performance aspects in VM2000 mode (Main memory)” in the “Performance Handbook” [37].

4.3 Paging area

Paging files and paging area

Paging memory is required to provide virtual address space, and this is implemented using paging files. Paging files always have the name `SYS.PAGING.<vsn>`.
To permit it to be used for paging, a created paging file must be assigned explicitly to the paging memory.

The sum of the paging files in the system which are assigned to the paging memory is called a paging area.

The paging area can be dynamically extended or reduced while the system is running by assigning paging files to the paging area or releasing them from it. A particular size and structure of the paging area is a prerequisite during system initialization (see "Selecting the paging area at startup").

The virtual address space available for a session is determined by the size of the paging area because one block (two 2K PAM pages) must exist in the paging area for each virtual page. The maximum overall size of the paging area is 4 TB.

During system initialization, the parameter service can be used to select the files that are to be used in the subsequent session from the set of the created and available paging files (see section "Selection of the paging area at startup (PAGING)").
Consequently a distinction must be made between initializing paging files (including reserving memory space) and selecting paging files for the session.

General conditions for a paging file:

- It may not be a file >= 32 GB
- It may not be configured on a private disk
- It may not be configured on a shared pubset

Paging disk and paging pubset

A disk is called a paging disk when a paging file has not only been initialized there, but has also been assigned to the paging memory.

As a general rule, any disk type is permissible. The disks of the paging area need not be of the same type.

A paging file may be located on a disk which was created dynamically as a DRV volume. (Paging files which were specified in the parameter file – PAGING parameter record – during system initialization and are not located in the home pubset are not supported by DRV.) However, only one of the two disks is written to.

A pubset is referred to as a paging pubset when at least one disk of the pubset is used as a paging disk.

If a pubset contains only paging files, it is a pure paging pubset. If a pubset is not a pure paging pubset and consists of multiple disks, no paging file should be created on its pubres.

It is recommended that the paging area be implemented predominantly using single-disk pubsets since this simplifies the reconfiguration of the paging area during operation when a paging file is included.

Command	Meaning
CREATE-PAGING-FILE	Create paging file
DELETE-PAGING-FILE	Delete paging file (overwrite with binary zeros)
EXTEND-PAGING-AREA	Dynamically extend paging area
MODIFY-PAGING-AREA-ATTRIBUTES	Modify attributes of paging files on disk
REDUCE-PAGING-AREA	Reduce paging area
SHOW-PAGING-CONFIGURATION	Display information on used and unused paging files

Table 8: Overview of commands for paging areas

The SHOW-PAGING-CONFIGURATION command outputs the following information:

- Which paging files are used
- Which paging files (on imported pubsets) are unused
- Whether a volume contains a used or unused paging file
- The overall size of the used/unused paging files which were found

Examples of the output for the paging files

```
/SHOW-PAGING-CONFIGURATION VOLUME=*UNUSED,INFORMATION=*ALL
% LIST OF THE FOUND AND UNUSED PAGING-FILES :
%
%  VOLUME  CAT-ID        SIZE                   RESTRICTION
%  1OSH.0   1OSH       256.00 MB                   NONE
%  1OSH.2   1OSH       256.00 MB                   NONE
%  ------------------------------------------------------------------------
%  SUMMARY             512.00 MB
/SHOW-PAGING-CONFIGURATION VOLUME=*USED,INFORMATION=*ALL
% LIST OF THE USED PAGING-FILES :
%
%  VOLUME  CAT-ID        SIZE       FREESIZE     UTILIZATION  REDUCTION  R
%  1OP1.1 * 1OP1      2500.00 MB     1510.99 MB    *LOW       *NOT-REQ   N
%  1OP1.0 * 1OP1      2500.00 MB     2200.00 MB    *LOW       *NOT-REQ   N
%  1OPP.0 * 1OPP      2500.00 MB      395.90 MB    *LOW       *NOT-REQ   N
%  1OSH.1   1OSH       256.00 MB      200.00 MB    *LOW       *NOT-REQ   N
%  ------------------------------------------------------------------------
%  SUMMARY            7756.01 MB
%
%  EMM2317 THE SIZE OF VOLUME(S) - MARKED WITH * - IS INCREASED BY 25%,
            BECAUSE THE PAM-KEYS ARE USED AS PAGING-AREA
```

4.3.1 Creating, assigning, releasing and deleting paging files

Creating paging files

Paging files are cataloged under the system administration user ID with the name SYS.PAGING.<vsn>. A paging file may occupy more than one extent on the volume with the specified volume serial number vsn. In a pubset it is possible for precisely one SYS.PAGING.<vsn> paging file to exist on each volume.

A paging file has a minimum size of 1 MB. For reasons of performance it is recommended that the paging area be distributed over several paging files (i.e. over multiple volumes) and that all these paging files be more or less the same size.

> The higher the paging rate of the disk, the greater the need for the paging area to be distributed among several paging files. The paging inputs/outputs can be accelerated by connecting the disks used to different channels. The paging rate can be measured with openSM2 (see the “openSM2” manual [49]).

Paging files can be created during a session with the CREATE-PAGING-FILE command. This command sets up a paging file SYS.PAGING.<vsn> of the specified size. It is possible to select each disk of an imported pubset via the VSN.

Assigning paging files to the paging area

An existing paging file can be brought into service, i.e. assigned to the paging area, using the EXTEND-PAGING-AREA command. For it to be used right from the start in following sessions it must be entered in the system parameter file (see the PAGING parameter record in section "Selection of the paging area at startup (PAGING)").

Paging files can also be created using the SIR statement CREATE-PAGING-FILE. SIR is described in the “Utility Routines” manual [15].

It makes sense always to create the paging files on the home pubset on system installation since operating system requirements are automatically satisfied in this way.

The required size of the paging area depends on the number and virtual program size of the programs which run simultaneously. To this must be added the system address space currently set.

The following formula is recommended for an initial evaluation of space requirements:

```
required space = 2 * (number of user tasks * virtual program size + system address space)
```

Once operation has begun, this value should be checked (e.g. using the SHOW-PAGING-CONFIGURATION command or openSM2; see the “openSM2” manual [49]) to ascertain the actually required size of the paging area (see also the “Performance Handbook” [37]).

Releasing paging files and partitions from the paging area

The REDUCE-PAGING-AREA command can be used to release a paging file or partition from the paging area during operation.

However, this is only possible if the existing paging area is not yet exhausted and the system will not find itself in a saturation state as a result of the reduction. The REDUCE-PAGING-AREA command executes asynchronously in a server task.

Measures for accelerating the release of paging files are described on "Measures for accelerating the release of paging files".

Deleting paging files

The DELETE-PAGING-FILE command is available for deleting a paging file. For this to be possible, the associated pubset must have been imported.

It takes a relatively long time to execute the DELETE-PAGING-FILE command because the file contents are deleted (overwritten with binary zeros). The command therefore runs asynchronously in a server task.

When deleting a paging file in the home pubset it is necessary to observe the following: Deletion with DELETE-PAGING-FILE is rejected if not enough paging files are left in the home pubset.

Dynamic reconfiguration of the paging area

The paging area can be extended dynamically at any time during the session using the EXTEND-PAGING-AREA command. To permit this, the pubset with the paging file to be included must be imported and the maximum size of the paging area (4 TB) must be observed.

The REDUCE-PAGING-AREA command enables the paging area to be reduced dynamically. A few requirements must be observed here, such as whether the remaining size of the paging area is sufficient for the tasks that are to be performed.

> **CAUTION!**
> You are recommended to make extremely careful use of paging area reconfiguration during ongoing operation and to restrict it to a minimum. Frequent reconfiguration can reduce the overall system performance more and more. This applies particularly for systems with a high paging load.

The following mode of operation enables you to maintain a good overall performance:

- Restrict the procedure to paging files which will presumably only rarely or never be removed from the paging area.
- The use of contiguous paging files is recommended.
- It is better to use a large number of small disks than a few large ones.
- It is recommendable to use these disks exclusively for paging files (pure paging pubsets).

Measures for accelerating the release of paging files

If the paging file to be released is almost fully utilized, relocating the virtual pages to other paging files will result in longer execution times for commands, a higher I/O workload and a higher load on the CPU. To reduce these loads or to simplify the reduction process, the following options are available:

1. The reduction is reported to the operating system some time in advance.

   The MODIFY-PAGING-AREA-ATTRIBUTES command (UTILIZATION=*LOW operand) is issued and, from this point on, the affected paging file is – as far as possible – no longer used for adding virtual pages. Through access to pages within the paging file or through page release, the paging file becomes emptier, and for the actual reduction fewer I/Os are required for transferring these pages.

   This method does not guarantee success: either the system cannot forego reading in pages to this paging file (because the paging rate is too high) or the pages within the paging file are not accessed.

Whether or not this method has been successful can be established using the SHOW-PAGING-CONFIGURATION command: an increase in the displayed free size of the paging file is a measure of success and shows when the reduction is to be started. This can be in a few minutes or a few hours.

2. Already when the EXTEND-PAGING-AREA command (operand UTILIZATION=*LOW) is used, the operating system is informed that this paging file should be used less for adding virtual pages. The likelihood of success is the same here as in the previous point.
3. By specifying the operand LATER-REDUCTION=*REQUESTED in the EXTEND-PAGING-AREA command you can lessen the CPU load during the reduction. However, more resident working memory is required to manage these paging areas.

You are advised to limit all the above measures to no more than two paging files.

4.3.2 Selecting the paging area at startup

At startup time, systems support can use the parameter service to determine which volumes are to be used for paging. Care is required here to ensure that SYS.PAGING.<vsn> paging files have been set up on the volumes concerned and that the maximum number of 128 volumes is not exceeded.

The paging area initialized at startup should be at least 200 Mbytes in size. If the paging area is smaller than 200 Mbytes, a warning in the form of message NSI5115 is output on the console, but system initialization continues.

If systems support did not specify any parameters for paging file selection, all SYS.PAGING.<vsn> paging files created on the home pubset are automatically used for paging. The parameter service allows volumes for paging areas both within and outside the home pubset to be specified. The statements required for this purpose must be stored in the parameter file by systems support (see the PAGING parameter record in chapter “Parameter service”).

4.4 Measures to prevent saturation states

Execution of a storage space request may lead to saturation states in the following areas: main memory, system address space and paging memory.

Memory bottlenecks can also occur in the following situations which are described elsewhere in this manual:

- If the main memory load or migration rate is too high (see "Allocation of resources")
- If storage space is requested on public volumes (see "Monitoring storage space saturation")

The messages output in accordance with the individual saturation states and causes are listed in the following sections. The same also applies to the measures that can be performed either by the operator or by system administration. This section focuses on long-term measures for the prevention of saturation states.

Main memory

- The main memory size should be determined in accordance with the recommendations in the Performance Handbook. If the main memory is too small, this usually has a negative effect on performance since the paging rate and displacement rate are too high.
- In the case of VM2000 guest systems, the recommendations in the Performance Handbook concerning the value of MINIMAL-MEMORY-SIZE must be observed.
- The MODIFY-SYSTEM-BIAS command makes it possible to allow user programs to make a larger number of user programs resident. It is advisable to restrict this authorization to user IDs under which such programs are to run (ADD-USER and MODIFY-USER-ATTRIBUTES commands).

System address space

- The system address space requirement is highly dependent on the system configuration and peripheral devices as well as on the employed subsystems and system load.
- You are strongly advised to be generous in assessing the system address space (the size can only be set on SUs /390, see "System address space").
- An insufficient system address space can result in undesired interruptions to operation (e.g. because of subsystem version changes, increases in load, extension of the system configuration or because of the after-effects of a saturation state).
- You are advised to use the available load limitation possibilities. For example, you can limit the number of users in the job classes (MODIFY-JOB-CLASS command). A user-specific limitation of the program size is also possible (by means of the ADD-USER and MODIFY-USER-ATTRIBUTES commands).

Paging memory

- The size of the paging memory should be chosen on the basis of the recommendations in the Performance Handbook.
- A sufficiently large paging memory must be set up during system initialization (parameter service).
- If the paging memory requirement grows significantly during a session then the extension should be performed as early as possible (by means of the EXTEND-PAGING-AREA command).
- The possibilities for load limitation should be used (see “System address space” above)

To prevent performance degradation, algorithms which are designed to counteract the saturation state come into effect if saturation conditions occur.

In these circumstances the Control System displays the messages contained in the tables below on the console.

4.4.1 Main memory

Saturation state

Message: `EXC0870 CORE SATURATION LEVEL=i`

where `i` is a digit in the range 0 through 3 indicating the saturation level.

Level	Effects on system	Operator action
0	Normal system processing	None
1	No new batch jobs will be started.	None
2	No new batch jobs will be started.	None
3	No new batch jobs will be started. Requests for resident memory pages will be met only for particularly important system jobs.	Jobs can be aborted by the operator

Desaturation

Level	Effects on system	Operator action
2 ... 0	Desaturation Batch jobscan be started again once Level=0 has been reached.	None

Saturation prevention measures

The occurrence of a saturation condition indicates that the system is overloaded. If main memory requirements cannot be permanently reduced, the system needs more main memory. In the case of a VM2000 guest system, the VM2000 administrator will need to increase the MINIMAL-MEMORY-SIZE. You are not advised to set a larger NEW-MEMORY-SIZE while leaving the MINIMAL-MEMORY-SIZE unchanged.

In the case of SUs x86, it is the BS2000 memory that is affected. The proportion of overall memory that this represents can be set via X2000.

Examples for the reduction of system main memory requirements

- Reduce the DAB cache areas in main memory
- Reduce the number of resident catalog buffers for specific pubsets or make the buffers pageable
- Export unused pubsets
- Unload applications which have made user pages resident
- Unload large applications or programs
- Considerably reduce the number of tasks in the system

4.4.2 System address space

Saturation state

Message: `EXC0874 ADDRESS SPACE SATURATION=i`

where `i` is a digit in the range 0 through 2 indicating the saturation level.

Level	Effects on system	Operator action
0	Normal system processing	None
1	No new batch jobs are started	Inform systems administration
2	No new batch jobs are started	

Saturation prevention measures

The occurrence of a saturation condition indicates that the system is overloaded. If the system is affected by recurrent saturation, the system configuration is inadequate.
It may therefore be necessary to decrease the size of the user address space (see also the “System Installation” manual [55], Modifying the BS2000 organization program, procedure SYSPRC.BS2000-EXEC.<ver>).

These configuration options are not available on SUs x86.

Examples for the reduction of system address space requirements

- Reduce the number of resident catalog buffers for specific pubsets.
- Export unused pubsets.
- Unload large applications or programs.
- Considerably reduce the number of tasks in the system.
- You are expressly warned not to unload large subsystems. Even though this normally results in a short-term alleviation of the saturation state it is often not possible to reload the unloaded subsystem later and it is therefore unavailable for the remainder of the session.
- Recurrent saturations of the system address space even though the system is configured correctly may indicate a system error. Please consult Customer Support.

4.4.3 Paging memory

Saturation state

Message: `EXC0873 NEW PAGING MEMORY SATURATION LEVEL=i`
where `i` is a digit in the range 0 through 3 indicating the saturation level.

Level	Effects on system	Operator action
0	Normal system processing	None
1	Warning of a possible saturation state (if the PAGING-SATURATION-WARNING-LIMIT parameter is specified, see "PAGING-SATURATION-WARNING-LIMIT n [, UNIT=4KB / MB]" in section "Memory management (MEMORY)")	See "Saturation prevention measures"
2	Batch jobs requesting class 6 memory are rolled out.[1)] No new jobs will be started.	Operator must try to abort jobs or to dynamically extend the pagingarea with EXTEND-PAGING-AREA
3	Interactive and batch jobs requesting class 6 memory are rolled out. The following message is issued to SYSOUT of dialog jobs. `EXC0844 TASK TEMPORARILY HELD DUE TO PAGING DEVICE SATURATION. TASK WILL BE CONTINUED AUTOMATICALLY` The TCB addresses of the jobs are managed in a queue. There they are arranged as follows: • interactive jobs precede batch jobs • high-priority jobs precede low-priority jobs • jobs of the same priority are arranged according to the number of relocated pages (jobs with fewer relocated pages come first) No new jobs will be started.	Operator must try to abort jobs or to dynamically extend the paging area with EXTEND-PAGING-AREA A deadlock or system crash may result.

[1)] The job's class 6 memory is saved to a temporary disk file (EAM), and then released.
This job must fulfill the following conditions:

- the class 6 memory request does not refer to a common memory pool for several jobs
- the job is not associated with a serialization identifier
- the job has not disabled any file pages

Desaturation

Level	Effects on system	Operator action
1	Level=1 merely indicates that the first job in the queue has been transferred back but there are still further rolled-out jobs.	None
0	All rolled-out jobs are processed. Jobs can be started again.	None

Saturation prevention measures

The occurrence of a saturation condition indicates that the system is overloaded.

If the system is affected by recurrent saturation, then the system configuration is inadequate or the measures taken to reduce the load are insufficient.

- Increase the paging memory.
- Ensure that a sufficiently large paging memory is set up during the system initialization phase of subsequent sessions (parameter service)
- Check the load reduction measures.

Examples for the reduction of a system's paging memory requirements (if an extension is not desired)

- Unload large applications or programs
- Considerably reduce the number of tasks in the system

Settings for threshold values and warnings

The threshold values for the saturation states and the warnings for the saturation levels can be configured in the startup parameter service, parameter set MEMORY, see section "Memory management (MEMORY)".

5 Device management

The device management facility of BS2000 (Nucleus Device Management, NDM) manages the peripheral configuration of a Server Unit. The channel peripherals of the SUs /390 are predefined during hardware generation with IOGEN (see the “System Installation” manual [55]). The bus and Fibre Channel peripherals of the SUs x86 are made known via X2000.

Device management incorporates the following functions:

- controlling the availability of all hardware units and their connections with respect to the system
- making available allocation and release mechanisms for devices and volumes
- handling mount jobs for volumes and protecting these volumes against illegal operator intervention
- enabling the reservation of devices, volumes and files according to job priorities
- providing information on allocation and availability states of the configuration

This chapter looks in detail at configuration, reconfiguration and dynamic I/O reconfiguration for SUs /390. The differences for SUs x86 are pointed out.

The NDM (Nucleus Device Management) component for resource allocation and reservation plus the monitoring, selection and operation of public data volumes are then described.

Next follows detailed information on error handling for selected device controllers and on managing private data volumes Detailed information on the IORM utility routineis closing this chapter.

5.1 Configuration components

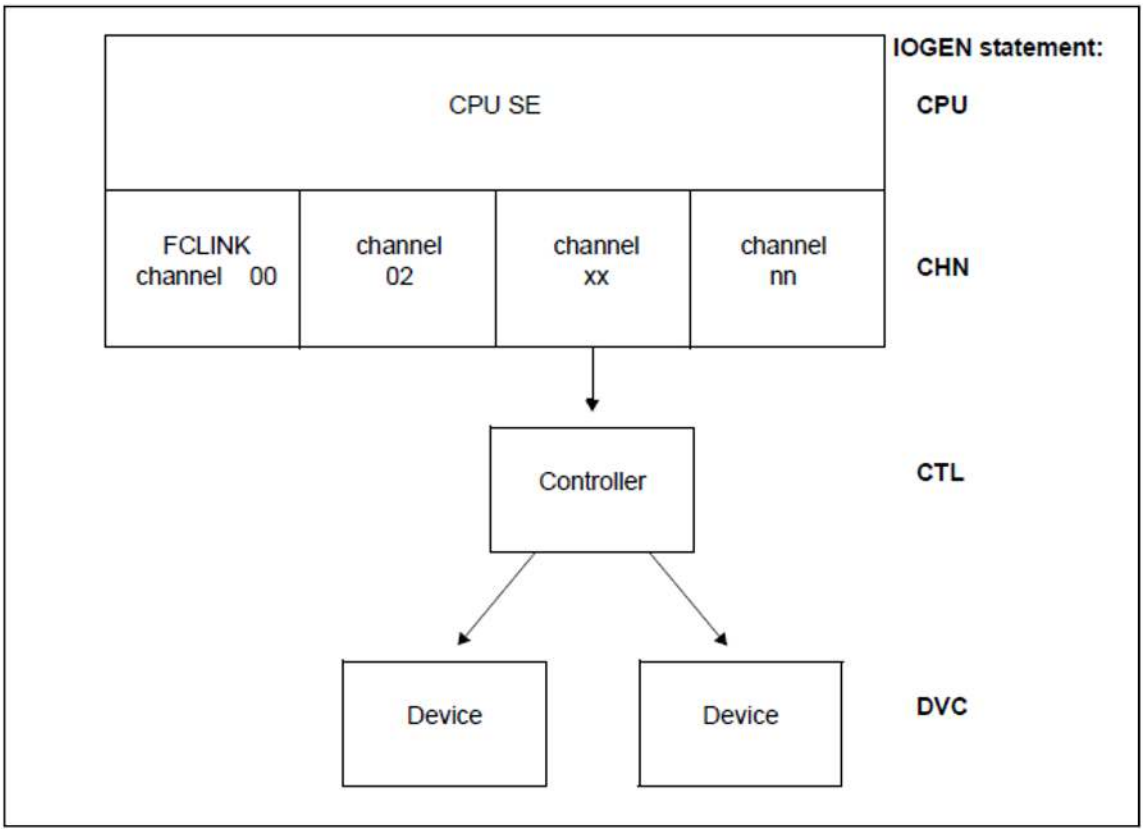


Figure 3: Extract from a configuration (SUs /390)

Interfaces for administering the configuration

Command	Meaning
ADD-IO-UNIT	Dynamically add new I/O unit [1]
ATTACH-DEVICE	Attach a hardware unit locally for channels and controllers or globally in VM2000
DETACH-DEVICE	Detach a hardware unit locally for channels and controllers or globally in VM2000
INCLUDE-DEVICE-CONNECTION	Include a logical connection locally or globally in VM2000
MODIFY-IO-OPTIONS	Modify FastDPAV settings Modify timeout settings [2]
MODIFY-IO-UNIT	Modify a controller's path description [1] Modify a device's preferred path [1] Define the preferred PAV device Modify timeout settings [2]

REMOVE-DEVICE-CONNECTION	Remove a logical connection locally or globally in VM2000
REMOVE-IO-UNIT	Dynamically remove I/O unit [1]

SECURE-RESOURCE-ALLOCATION	Request resources
SHOW-DEVICE-CONFIGURATION	Display the input/output configuration
START-CONFIGURATION-UPDATE	Start dynamic I/O configuration modification
STOP-CONFIGURATION-UPDATE	Stop dynamic I/O configuration modification, Save the current IORSF
UNLOCK-DEVICE	Reset hardware reservation
Macro	**Meaning**
NKDINF	Information on the allocation and availability state of the (peripheral) configuration

Table 9: Overview of interfaces for administering the configuration

[1] These functions are only available if a dynamic I/O configuration change has been initiated on SU /390.

[2] This function is also available outside of a dynamic reconfiguration independently of the architecture of the Server Unit.

5.1.1 Hardware units

Types of hardware unit

The following types of hardware unit are included in a configuration:

- the CPUs (central processing units)
- the channels (CHN), which are defined by means of the IOGEN statement CHN
- the multidevice controllers (CTL), which are defined by means of the IOGEN CTL statement (tape and disk controllers)
- the devices (DVC), which are defined by means of the IOGEN DVC statement

> i On SUs x86 only channels (CHN) and devices (DVC) are known and visible in BS2000, but no controllers (CTL).

With virtual connections a distinction is made between inner and outer units, which are used for input/output.

The channels (CHN) are the inmost hardware units, the devices (DVC) the outmost. The order of the various hardware units, viewed from the inside to the outside, is as follows CHN - CTL - DVC

Thus when, for example, a direct virtual connection exists between the two hardware units CTL and DVC, CTL is the inner and DVC the outer unit.

States of hardware units

These hardware units can assume the following states:

ATTACHED

The hardware unit is attached to the system and can be used for input/output. This state is set:

- after system initialization if the appropriate unit was generated in the hardware generation with IOGEN with the attribute “A” (e.g. DVC A0,CB,**A**,(...))
- after system initialization if the appropriate unit was modified in the startup parameter service with the IOCONF statement MOD-IO-UNIT ..., STATE=ATT
- after successful invocation of the ATTACH-DEVICE (ATT) command.
- after successful invocation of the ADD-IO-UNIT command (with STATE=*ATTACHED)

ATTACH-PENDING

Attachment is performed for a disk or tape device. The hardware unit cannot yet be used for input/output. If the reconfiguration job is executed, the hardware unit assumes the ATTACHED state.
If the reconfiguration job cannot be executed, the hardware unit is placed in the DETACHED state (see section “Special information on tape devices” in chapter "Special information on magnetic tape and disk devices").

This state is set during execution of input/output operations to determine control and device data and to reserve MTC devices.

DETACHED

The hardware unit is detached from the system and cannot be used for input/output. The DETACHED state is further subdivided into:

- DETACHED EXPLICITLY. This state is set:
  - after system initialization if the relevant unit was generated in in the hardware generation with IOGEN with the attribute “D” (e.g. DVC_A0,CB,**D**,(...))
  - after system initialization if the relevant unit was modified in the startup parameter service with the IOCONF statement MOD-IO-UNIT ..., STATE=DET
  - after successful invocation of the DETACH-DEVICE (DET) command (DET)
  - after automatic reconfiguration by the operating system
  - after system initialization if a hardware unit is recognized as “offline”.
- DETACHED IMPLICITLY
  All internal connections of the appropriate unit are in the REMOVED state.

DETACH-PENDING

The unit is still allocated by a task or the system and is no longer available after the end of allocation. Further allocation attempts are not permitted. The DETACH-PENDING state occurs generally only during a specified wait time (operand FORCE=NO...) in the DETACH-DEVICE command. It can also occur if a DETACH-DEVICE command is issued with the FORCE=*YES operand for a magnetic tape cartridge device (see section “Special information on tape devices” in chapter "Special information on magnetic tape and disk devices").
If the reconfiguration job can be carried out during the wait time, the unit is placed in the DETACHED state, otherwise it (re-)enters the ATTACHED state. The DETACH-PENDING state is subdivided into:

- DETACH-PENDING EXPLICITLY
  This state is set if the unit is directly detached.
- DETACH-PENDING IMPLICITLY
  This state is set if no inner connections of this unit are in the INCLUDED state and at least one is in the REMOVE-PENDING state.

INVALID

The device cannot be used nor can it be reconfigured (attached).
Example: The device has been removed with REMOVE-IO-UNIT.

ENABLED

Possible state of a FastDPAV alias device: The FastDPAV alias device can be used for input/output.
The state is set:

- in native mode, if the FastDPAV alias device has not been explicitly detached (DETACH-DEVICE command).
- in VM mode, if the FastDPAV alias device has been assigned to the virtual machine and not been explicitly detached afterwards (DETACH-DEVICE command).

DISABLED

Possible state of a FastDPAV alias device: The FastDPAV alias device cannot be used for input/output.
The state is set:

- in native mode, if the FastDPAV alias device has been explicitly detached (DETACH-DEVICE command).

- in VM mode, if the FastDPAV alias device has not been assigned to the virtual machine or has been explicitly detached afterwards (DETACH-DEVICE command).

5.1.2 Virtual, reconfigurable connections

Types of virtual, reconfigurable connection

The following virtual (reconfigurable) connections exist in a configuration:

- between the channels and the devices generated directly on them
- between the channels and the controllers generated on them
- between the controllers and the devices generated on them

A distinction is made between inner and outer virtual connections:

- An inner connection always goes to an inner unit (e.g. from DVC to CTL or from CTL to CHN).
- An outer connection goes to an outer unit (CHN to CTL or CTL to DVC).

States of virtual, reconfigurable connections

INCLUDED

The virtual connection to the system has been established and can be used for input/output. This state is set:

- after the system startup, unless one of the conditions for the REMOVED state is fulfilled (see below)
- after successful invocation of the INCLUDE-DEVICE-CONNECTION (INC) command

REMOVED

The virtual connection to the system has been cleared and cannot be used for input/output. The REMOVED state is further subdivided into:

- REMOVED EXPLICITLY
  This state is set:
  - after system initialization if the relevant unit was modified in the startup parameter service with the IOCONF statement MOD-IO-UNIT *CON(..., STATE=REM)
  - after automatic reconfiguration by the operating system
  - after successful invocation of the REMOVE-DEVICE-CONNECTION command.
- REMOVED IMPLICITLY
  This state is set if the inner hardware unit is in the DETACHED state.

REMOVE-PENDING

The virtual connection is still in use and thereafter is no longer available to the system. This state occurs only during a wait time in the REMOVE-DEVICE-CONNECTION or DETACH-DEVICE command. If the reconfiguration job can be carried out, the connection is placed in the REMOVED state; otherwise the INCLUDED state is set (reinstated). The REMOVE PENDING state is subdivided into:

- REMOVE-PENDING EXPLICITLY
  This state is set if the connection is cleared directly [with wait time (operand FORCE=NO... in the REMOVE-DEVICE-CONNECTION command)].
- REMOVE-PENDING IMPLICITLY
  This state is set if the inner unit of this connection is in the DETACH-PENDING state.

5.2 Reconfiguration

Reconfiguration comprises detaching and attaching the components of the configuration of a Server Unit. A configuration consists of hardware units and their virtual connections.

For this purpose the operator is provided with the following reconfiguration commands:

Command	Meaning
ATTACH-DEVICE	Attach a hardware unit locally for channels and controllers or globally in VM2000
DETACH-DEVICE	Detach a hardware unit locally for channels and controllers or globally in VM2000
INCLUDE-DEVICE-CONNECTION	Include a logical connection locally or globally in VM2000
REMOVE-DEVICE-CONNECTION	Remove a logical connection locally or globally in VM2000

Table 10: Reconfiguration commands

For other commands and macros relevant to reconfiguration, see table 12 in section "NDM - resource allocation and reservation".

5.2.1 Reconfiguration for multiprocessors

The system reacts automatically to hardware failures. If one of the components (CPU, channel) fails, it is automatically detached from the operating system. Operation continues using the remaining components.

The operator can attach or detach components by means of reconfiguration commands. If a component is to be disabled in order to be serviced or for other reasons, the operator can detach it by means of the DETACH-DEVICE command. Further operations are then performed on the remaining components. If a component which has been detached is to be made available again, the operator can attach it using the ATTACH-DEVICE command.

Reconfiguration fails if:

- after a CPU has been detached, there is no longer at least one CPU still operable
- if inconsistencies arise in the operating system data when a CPU fails and the hardware is unable to make a valid hardware context available
- if public volumes or the last console are affected by the failure.

Certain devices must be switched on or switched over before they can be addressed by the system once more. Switching over in this case means not only the physical switching of the device to an alternative path, although this is a prerequisite for the operation Switching over incorporates all the actions necessary for the system to be able to address the devices again. The process of switching over or switching on devices is the responsibility of the operator.

> **i** If an error occurs in one of the attached CPUs for a Server Unit with more than one CPU, the Server Unit in many cases will be reconfigured automatically. The only time that a reconfiguration cannot be performed is if a valid program context cannot be created for the failed CPU and if the lock mechanism of the operating system is active.

5.2.2 Detaching and attaching the components CPU, CHN, CTL and DVC

Detachment of components by the operating system

Components that fail are automatically detached. The operator then receives the following message at the main console:

NKR0048	'CHANNEL=mn' DETACHED BY SYSTEM **********	(for CHN)
ETMRK48	CPUxy DETACHED BY SYSTEM **********	(for CPU)

mn Mnemonic device name of the component

Message ETMRK48 must be acknowledged by the operator. This prevents components from being detached without the operator's knowledge.

The message is output to the main console connected to the unaffected I/O processor.

Attachment or reattachment is performed by the operator using the ATTACH-DEVICE command, although this is not always possible.

Detachment of components by the operator

The operator can detach components by means of the DETACH-DEVICE command (see the “Commands” manual [27]).

The operating system confirms that each component has been successfully detached by the following message to the console:

NKR0045	'CHANNEL=mn' DETACHED **********	(for CHN)
ETMRK19	CPUxy DETACHED **********	(for CPU)

The DETACH-DEVICE command is rejected with an appropriate message if it cannot be performed at the present time.
For channels and device controllers under VM2000 on SUs /390, detachment can only be performed in the monitor system in the local system or in all BS2000 guest systems (globally in VM2000). The command must then be entered on the monitor system.

Attachment/reattachment is performed by means of the ATTACH-DEVICE command.

Attachment of components by the operator

The operator can attach components by means of the ATTACH-DEVICE command in the following cases:

- if a component becomes available again following a failure or servicing
- if system initialization was performed without a component.

For channels and device controllers under VM2000 attachment can only be performed in the monitor system in the local system or in all BS2000 guest systems (globally in VM2000).

The operating system confirms that each component has been properly attached by means of the following message to the console:

NKR0040	'<unit-class>=mn' ATTACHED	(for CHN, CTL, DVC)
ETMRK18	CPUxy ATTACHED	(for CPU)

Attaching and detaching extra CPUs

Extra CPUs can exist on every Server Unit.

The operator can attach one or more extra CPUs while the system is running with the ATTACH-DEVICE UNIT=*EXTRA-CPU(CPU-ID=...) command.

The extra CPUs are detached with the DETACH-DEVICE command.

See "Attaching and detaching extra CPUs" in section "Functionality in BS2000 native operation" for more detailed information on attaching and detaching extra CPUs.

5.2.3 Effect of reconfiguration commands

A reconfiguration command has no effect if a state is to be selected which has already been set or which cannot be set. The relevant messages are output.

Under VM2000, it is possible to attach or detach channels, controllers and connections in the monitor system only in the local system or in all BS2000 guest systems (globally in VM2000).
The operand SCOPE=OWN-SYSTEM-ONLY/VM2000-GLOBAL has been introduced for this purpose.

ATTACH-DEVICE

If a valid ATTACH-DEVICE command is entered, this has the following effect:

If the specified hardware units were in the DETACHED EXPLICITLY state they enter the ATTACHED state. The hardware units can be used once again.

If the outer connections associated with these hardware units were in the REMOVED IMPLICITLY state, they enter the INCLUDED state. The virtual connections can be used once again.

The outer hardware units which are connected to these units and which were in the DETACHED IMPLICITLY state enter the ATTACHED state if at least one virtual connection to the outer unit is in the state INCLUDED.

DETACH-DEVICE

A valid DETACH-DEVICE command has the following effect:

Each of the hardware units specified assumes the DETACHED EXPLICITLY state. They cannot be used.

All outer connections associated with these hardware units enter the REMOVED IMPLICITLY state. They cannot be used.

All outer hardware units connected to these units assume the DETACHED IMPLICITLY state if the connection to the directly detached unit was the last or the only virtual connection. These external hardware units cannot be used.

Detachment with the parameter SCOPE=*VM2000-GLOBAL is successful only if it was possible to implement detachment successfully in all BS2000 guest systems.

The forced detachment of channels is enabled in the monitor system by the operand FORCE=*UNCONDITIONAL-OFFLINE under VM2000 (SUs /390).

INCLUDE-DEVICE-CONNECTION

The entry of a valid INCLUDE-DEVICE-CONNECTION command has the following effect:

If the virtual connections specified were in the REMOVED EXPLICITLY state, they enter the INCLUDED state. The connections can be used once again.

If the outer units associated with the connections were in the DETACHED IMPLICITLY state, they enter the ATTACHED state. These hardware units can be used once again.

The outer connections extending from these hardware units enter the INCLUDED state if they were previously in the REMOVED IMPLICITLY state. These outer connections can be used again.

REMOVE-DEVICE-CONNECTION

Valid input of the REMOVE-DEVICE-CONNECTION command has the following effect:

The virtual connections specified enter the REMOVED EXPLICITLY state. They can no longer be used.

The outer hardware units attached to these connections enter the DETACHED IMPLICITLY state if the detached connection was the last or only inner connection. These hardware units can no longer be used.

The outer connections extending from these outer hardware units in the DETACHED IMPLICITLY state enter the REMOVED IMPLICITLY state. They can no longer be used.

Interfaces for KVP consoles

The two I/O interfaces (device type SKP2) constitute a single virtual unit for the operating system. With ATTACH /DETACH tasks it ensures that both interfaces assume the same state. A precondition is that both paths from the controller to the interfaces should also be in the same state. If the paths are put in different states using INCLUDE-/REMOVE-DEVICE-CONNECTION, correct error handling of the interfaces by the system is not longer guaranteed.

> **i** Activating hardware units and connections is the responsibility of the operator. Hardware units and connections are deactivated either by the operator or (in serious cases) by the system.
>
> The operator is responsible for coordination when hardware units which are connected to several systems are attached or detached.
>
> Devices in the DETACH-PENDING state can be used by the volume monitoring in the event of remount jobs if no other free devices are available.

FastDPAV alias devices

After successful execution of the ATTACH-DEVICE command for a FastDPAV alias device, the device enters the ENABLED state. This means that the FastDPAV alias device can be used for input/output.

After successful execution of the DETACH-DEVICE command for a FastDPAV alias device, the device enters the DISABLED state. This means that the FastDPAV alias device cannot be used for input/output.

5.2.4 Special information on magnetic tape and disk devices

When the system is initialized, magnetic tape cartridge devices are first explicitly DETACHED from the system.

For ATTACHED generated disk devices, and those belonging to the home pubset, device and controller attributes are determined via input/output commands. A device is explicitly DETACHED from the system if these attributes cannot be determined due to INPUT/OUTPUT errors.

When a magnetic tape or disk device is attached, the first thing that happens is that device and controller attributes are determined. If this cannot be done successfully, the ATTACH procedure is rejected.

Exception

> If the determination of the attributes for magnetic tape devices was rejected as a result of "NO PATH AVAILABLE" (CC=3), the ATTACH procedure is permitted. This means it is possible to overconfigure for the purposes of overallocation.

Special information on tape devices

Tape devices (MTC devices and emulated tape devices) are only ATTACHED if they can be physically reserved for the home system. Once tape devices have been successfully attached, they are assigned exclusively to a system.

ATTACH-DEVICE

When a tape device is being attached, an attempt is made to reserve the device. The message `NKR0042 DEVICE=mn ATTACH ACCEPTED` is issued and the device is placed in the ATTACH-PENDING state.

Once the device has been successfully reserved, the message `NKR0110 DEVICE=mn DEVICE ATTACHED AND ASSIGNED` is issued, and the tape device assumes the ATTACHED state.

If the tape device has been reserved by another system, the messages `NKR0111 DEVICE=mn DEVICE ASSIGNED ELSEWHERE` and `NKR0044 DEVICE=mn ATTACHMENT REJECTED` are issued, and the tape device is placed in the DETACHED-EXPLICITLY state.

If the tape device cannot be assigned for other reasons, the following message is issued:
`NKR0112 DEVICE=mn  COULD NOT BE ASSIGNED.`

A dummy tape device will be attached even if assignment is rejected due to the unavailability of the device or device controller. In this case, the following messages are issued instead of the `NKR0110` message:

```
NKR0112 DEVICE=mn COULD NOT BE ASSIGNED
NKR0040 DEVICE=mn ATTACHED
```

DETACH-DEVICE

In the case of detachment (explicit or implicit), an attempt is made to cancel the hardware reservation:

- During this time, all devices/paths affected by the reconfiguration job are placed in the DETACH-/REMOVE-PENDING state.
- The message: `NKR0114 DEVICE=mn DEVICE UNASSIGNED` indicates successful execution; The message: `NKR0115 DEVICE=mn COULD NOT BE UNASSIGNED`. indicates that the hardware reservation could not be canceled.
- If an ATTACH-DEVICE command is issued during the wait time, a new allocation attempt is made.

Effects of a DETACH-DEVICE/REMOVE-DEVICE-CONNECTION command with the FORCE=*YES operand on the controller:

1. The controller acknowledges write jobs if the data to be transferred is stored in one of the controller's buffers; It then transfers the data asynchronously to the volume.
   The system then waits a maximum of 2 minutes for successful transfer to the tape device or for data that could not be transferred to be read in again. One of the following messages is issued:
   `NKR0020 DETACH DEVICE ACCEPTED`
   `NKR0021 REMOVE DEVICE CONNECTION ACCEPTED`

   During this wait time the tape device is in the DETACH-PENDING state.
2. After expiration of this wait time, at the latest, an attempt is made to cancel the hardware reservation.
3. Issuing an ATTACH-DEVICE command during the wait time results in cancellation of the DETACH job.

When the system is terminated, an attempt is made to cancel all (still) existing hardware reservations for the system. If the attempt fails, the following message is issued:
`NKR0115 DEVICE=mn  COULD NOT BE ASSIGNED`

If an attempt to attach a device is rejected with the message `NKR0111 DEVICE=mn DEVICE ASSIGNED ELSEWHERE`, the tape device is reserved by an external system. The UNLOCK-DEVICE command can be used to cancel the reservation by the external system. After a system crash this command can be used to release the tape device.

5.3 Dynamic I/O configuration changes (SUs /390)

Reconfiguration during system operation means the attachment and detachment of previously defined components of the configuration of a Server Unit (see section "Reconfiguration").

In addition, it is possible to add or remove I/O units to and from the Server Unit configuration during ongoing operation.

It is possible to modify the I/O configuration during system operation, i.e. the existing configuration can be extended or reduced without the need for a new IMPL. The modifications are implemented directly on the Server Unit's active IORSF.

This method of changing the configuration is referred to as "dynamic I/O configuration change" below. Dynamic I/O configuration changes are supported for the following device classes:

- disk and tape devices
- Network/LAN devices

Preparations for dynamic I/O reconfiguration

To make it possible to add new devices and controllers to the configuration during system operation, the system generates additional free table entries for devices and controllers during system initialization. The table entries for the channels and connections between the I/O units are created dynamically and do not need to be considered by systems support during IORSF generation.

The number of free table entries for devices and controllers depends on the size of the reserve area in the IORSF. In the case of an IORSF created with IOGEN this is sufficient for 512 additional entries (channels, controllers, devices and their connections), see the "System Installation" manual [55]. An IORSF that has already been modified can have more or less free entries either through adding or removing units in the active IORSF or through writing back the active IORSF with the STOP-CONFIGURATION-UPDATE ..., DEVICE-RESERVE= command.

At system startup, 64 free entries are provided in the device table for network/LAN devices, and for disk and tape devices the maximum possible number for each resulting from the free entries in the IORSF (but at most 2048 each). The total number of units which can be added dynamically may not exceed the limits set in the IORSF.

Measures to avoid bottlenecks

Every device or controller which is added to the configuration occupies a free table entry. The attempt to add the unit is rejected if no further free table entries are present. In such cases, it is not possible to add a new device or controller until a corresponding I/O unit has been removed.

If at IORSF generation time, it is possible to predict that these free table entries will not be sufficient for the dynamic I/O configuration changes expected in the next session then a sufficiently large number of future I/O units should also be configured in the IORSF. When they are taken into service, these additional I/O units can then be redefined to match the real configuration using the dynamic reconfiguration commands.

Performing dynamic I/O configuration changes

The following commands are available for dynamic I/O configuration changes. The commands are described in detail in the “Commands” manual [27]:

Command	Meaning
ADD-IO-UNIT	Define new I/O units
MODIFY-IO-UNIT	Change path description for a controller, modify the preferred path for a device, modify the timeout settings, specify preferred device for PAV.
REMOVE-IO-UNIT	Remove I/O units
SHOW-DEVICE-CONFIGURATION	Display the input/output configuration
START-CONFIGURATION-UPDATE	Start dynamic I/O configuration modification
STOP-CONFIGURATION-UPDATE	Stop dynamic I/O configuration modification, Save the current IORSF

Table 11: Commands for dynamic I/O configuration changes

Before changes can be made to the configuration, dynamic I/O configuration modification must be started using the START-CONFIGURATION-UPDATE command. The commands for configuration modification are not accepted until dynamic reconfiguration has been started successfully.

Exception

> If you simply wish to modify device timeout settings then the MODIFY-IO-UNIT command (operands: UNIT=*DEVICE(...) and TIMEOUT=...) can be entered outside of dynamic reconfiguration. This is also true when specifying a preferred device for PAV (operand PAV-PREFERRED-DEVICE).
> **These changes can be performed independently of the Server Unit's architecture!**

Once dynamic I/O configuration modification has been started successfully, systems support can use the ADD-IO-UNIT, REMOVE-IO-UNIT and MODIFY-IO-UNIT commands to modify the I/O configuration dynamically. The modifications are implemented in the active IORSF (Input/Output Resource File) and therefore take effect immediately.

General conduct of a dynamic I/O configuration change

Every command-initiated configuration change is processed in a number of stages under the control of the system component IORECONF:

1. Inspection phase:
   IORECONF checks whether the requested configuration change can be performed and rejects requests which cannot be satisfied. Thus, for example, an ADD-IO-UNIT command for a device may be rejected for one of the following reasons:
   - The controller or channel to which the device is to be connected is not defined.

- The device is already defined.
- There is no free table entry available.

After this, all the other system components which are affected by the configuration change are called (e.g. device and volume management components). The request can then be rejected by one of the called components.

2. Modification of IORSF:
   When the check has been terminated, the configuration change is entered in the active IORSF. During the period required to modify IORSF, all the affected devices are locked for I/O operations.
3. Modification of tables:
   Once IORSF has been successfully modified, IORECONF modifies the I/O tables and informs system components affected about the configuration change.

If the request for a configuration change cannot be completed in full, IORECONF attempts to reconstruct the initial status. The completion and rejection of configuration change requests are logged by means of console messages (`NKR....`).

Dynamic extensions to the I/O configuration

Systems support can use the ADD-IO-UNIT command to add new I/O units (device, controllers, channels) to the configuration. Devices can be defined as PAV alias devices (see section "Parallel Access Volumes (PAV, SUs /390)").

New I/O devices must be specified in the sequence: channel controller device. Up to 256 devices can be added to a controller by means of an ADD-IO-UNIT command.

Restrictions and special aspects

A new channel module can be added by means of the following steps:

1. ADD-IO-UNIT *CHN(...),STATE=DET command for each channel in the module
2. Installation of channel group and activation via CM frame
3. ATTACH-DEVICE *CHN(...) command for each channel in the module

Channels in modules which were already present on IMPL can be dynamically generated using ADD-IO-UNIT but cannot then be switched “online”. For this reason, all channels that are already present should be pre-generated.

To operate a new tape device, the following measures are necessary in addition to the ADD-IO-UNIT definition and when an MTC archiving system is used:

1. The ADD-DEVICE-DEPOT command must be issued to assign a storage location.
2. If you are using MAREN and the new device is to be subject to free tape allocation, the MARENUCP program must be terminated and restarted.
3. If the new device is part of an MTC archiving system controlled by ROBAR and has not yet been defined in the ROBAR configuration file, ROBAR-SV must be terminated and the ROBAR-SV configuration file extended. ROBAR-SV can then be restarted.

In configurations without MTC archiving systems, the tape device need only be ATTACHED (ATTACHDEVICE command) if it has not already been implicitly attached with the ADD-IO-UNIT ...,STATE=*ATTACHED command.

Before or after addition, a new network/LAN device must be declared in DCM by means of the BCIN command.

The IOTRACE subsystem does not take account of newly added I/O units unless it is restarted following the configuration change.

Dynamic reductions to the I/O configuration

Systems support can use the REMOVE-IO-UNIT command to remove IO units (devices, controllers, channels), which currently have the status DETACHED, from the current configuration. When the “innermost” unit is removed (channel or controller), then the

system also implicitly removes the “outer” units (controller or devices) if these are not associated with any other “inner” units. I/O units for removal must be specified in the sequence: device controller channel.

Dynamic modifications to the I/O configuration properties

Systems support can use the MODIFY-IO-UNIT command for the dynamic modification of the configuration attributes of controllers and devices. In the case of controllers, it is possible to redefine the I/O parts and the availability of the connection between the channel and controller (INCLUDED or REMOVED).

In this way, a controller which possesses two or more channel connections can be switched over without interruption. In the case of a device, it is possible to modify the preferred I/O path. You can specify with the STATE=*PAV-PREFERRED-DEVICE(...) operand that the PAV device under VM2000 is preferred for I/O (see section "Parallel Access Volumes (PAV, SUs /390)" for more information on PAV).

The command can also be used to modify device timeout settings (MODIFY-IO-UNIT UNIT=*DEVICE(...), TIMEOUT=<value_in_seconds>).
If only the device timeout settings are modified or a preferred device for PAV is specified, then the command can also be issued outside of dynamic reconfiguration (i.e. independently of the architecture of the Server Unit).
The timeout value should only be changed temporarily for certain actions, e.g. the online update of a disk storage system controller. After that the timeout should be reset to its default value with TIMEOUT=*DEFAULT.

Stop dynamic I/O configuration change

Dynamic I/O configuration modification is terminated with the STOP-CONFIGURATION-UPDATE command. To make the current I/O configuration available for a following session, the current IORSF can be saved using the IORSF-UPDATE operand.
If the system crashes before the active IORSF is written to the hard disk then the configuration changes that have already been performed are not lost on the next IPL since the BS2000 I/O tables are derived from the active IORSF on system initialization.

Following the successful termination of dynamic reconfiguration, the reconfiguration commands ADD-IO-UNIT, REMOVE-IO-UNIT and MODIFY-IO-UNIT are no longer accepted (exception: timeout settings).

Dynamic I/O configuration modification under VM2000

Under VM2000 it is only possible to enter the I/O reconfiguration commands in the monitor system. The necessary changes in the I/O tables are performed in the Hypervisor, in the monitor system and automatically in the active guest systems. When the START-CONFIGURATION-UPDATE command is entered, message `NKR0178` is displayed indicating how many guest systems are active and which of them support dynamic configuration changes.

5.4 Dynamic I/O configuration changes (SUs x86)

On SUs x86 bus peripherals are not configured anymore with IOGEN but are instead configured via X2000. Due to this, the ability to overdefine the I/O configuration, which then allowed predefined devices to be put into operation dynamically later on, is also eliminated.

Disks, tape and network/LAN devices can be added to the configuration during system operations. BS2000 creates additional empty device table entries for this purpose during system initialization:

- for network/LAN devices, 64
- for disk and tape devices, approx. 10% each of the number of devices already defined in the device table (at least 64, at most 2048)

5.5 Reconfiguration of extra and spare CPUs

The Server Units offer improved high availability for CPUs as well as the ability to temporarily attach preinstalled CPUs that expand the computing capacity of the server slightly beyond its nominal value.

When a BS2000 CPU fails, an existing spare CPU is automatically attached. The nominal performance of the Server Unit is thus guaranteed.
On an SU/390 one spare CPU is available for this purpose.
On an SU x86 a CPU not previously used by BS2000 is employed when system startup takes place following a CPU failure.

In addition to the spare CPU, if necessary, additional Server Unit power can be temporarily installed beyond the nominal number of multiprocessors (MP level) on the Server Unit in the form of extra CPUs whose duration of usage is specified in a contract. The extra CPUs are attached and detached by systems support using extended CPU reconfiguration commands. The usage intervals are recorded in BS2000 in a separate log file and also appear as messages sent to remote service.

See the “VM2000” manual [60] for more information on using spare/extra CPUs in VM2000.

5.5.1 Functionality in BS2000 native operation

Installation

In the context of a new installation or the reconfiguration of a Server Unit, the model lock diskette read in by customer support stores the model and hardware characteristics on the Server Unit. These characteristics are: the model ID, the globally unique IDs of the installed CPUs, the Server Unit's nominal MP level, the number of spare CPUs available (only for SUs /390) and the number of extra CPUs as agreed to in the contract.

IPL

During startup the CPUs are put in to operation in accordance with the number of multiprocessors (ATTACHED, ONLINE, NORMAL).

Extra CPUs remain detached in BS2000 (DETACHED, OFFLINE, EXTRA).

spare CPUs also remain detached (DETACHED, OFFLINE, SPARE) if the model is a model with two or more CPUs. The spare CPU is attached (DETACHED, ONLINE, SPARE) for models with only one CPU.

IPL specifies a CPU for normal operations using the firmware, and it is checked by the BS2000 software. The state of the CPU is displayed with the SHOW-DEVICE-CONFIGURATION CLASS=*CPU command.

CPU failure

When a CPU fails (MFA=Malfunction Alert, MCKI=Machine Check Interrupt) on an SU /390, BS2000 detaches the defective CPU (DETACHED, OFFLINE, ERROR) and simultaneously attaches a spare CPU that is ready for operation.

The following messages (which are also sent to the service center as a teleservice call) are output when this is done:

```
ETMRK48 CPU (&01) DETACHED BY SYSTEM
ETMRK20 CPU (&01) ATTACHED BY SYSTEM
```

After a MFA the detached CPU is locked for any following "IPL/Power on Reset" or IMPL. BS2000 does not display it anymore after the next IPL or IMPL. Only after the CPU is repaired or exchanged by the service (the entire server must be offline to do this) can the CPU be declared to be available via an SVP frame.

When a CPU fails for a different reason (e.g. a CPU loop), detaching the CPU normally has no effect on the next startup (DETACHED, ONLINE, NORMAL). A spare CPU is not attached in this case.

Attaching and detaching extra CPUs

In accordance with the contractual duration of usage, extra CPUs can be attached by the operator for a limited time. On BS2000 systems this is done using the expanded operands of the ATTACH-DEVICE and DETACH-DEVICE commands:

```
/ATTACH-DEVICE UNIT=*EXTRA-CPU(CPU-IDENTIFIER=<x-string 2..2>/*ALL/*ANY)
```

*ALL	All detached extra CPUs are attached.
*ANY	Any one of the detached extra CPUs is attached.

Additional messages when the commands are being processed (asynchronous, no command return codes):

```
ETMRK58 EXTRA-CPU (&00) ATTACHED
ETMRK5A ALL EXTRA-CPU'S ALREADY ATTACHED
ETMRK5E OBJECT EXTRA-CPU NOT EXISTING ON THIS HARDWARE
```

The extra CPUs are detached in a similar fashion:
`/DETACH-DEVICE UNIT=*EXTRA-CPU(CPU-IDENTIFIER=<x-string 2..2>/*ALL/*ANY)`

*ALL	All attached extra CPUs are detached
*ANY	Any one of the attached extra CPUs is detached.

Additional messages when the commands are being processed (asynchronous, no command return codes):

```
ETMRK59 EXTRA-CPU (&00) DETACHED
ETMRK5B ALL EXTRA-CPU'S ALREADY DETACHED
ETMRK5E OBJECT EXTRA-CPU NOT EXISTING ON THIS HARDWARE
```

The attaching and detaching is recorded in the special SYS.RESLOG.<server-id> log file (see the “Diagnostics Handbook” [14]).

An operable spare CPU is also automatically attached when an extra CPU fails (for SUs /390 only).

The use of the extra services terminates once the BS2000 session terminates or when it terminates abnormally. The extra CPUs are added again explicitly when a new session is started.

Expanding the output of the information commands

Example: Information on CPUs (SU /390 with 2 extra CPUs and 1 spare CPU

```
/SHOW-DEVICE-CONFIGURATION UNIT=*SELECT(CLASS=*CENTRAL-PROC)
MNEM  UN-CLASS  UN-TYPE  CONF-STATE  POOL/SIDE
00    CPU       SU700-30  ATTACH /ON  NORMAL
01    CPU       SU700-30  ATTACH /ON  NORMAL
02    CPU       SU700-30  ATTACH /ON  NORMAL
03    CPU       SU700-30  ATTACH /ON  EXTRA
04    CPU       SU700-30  DET(EX)/OFF EXTRA
05    CPU       SU700-30  DET(EX)/OFF SPARE
```

5.5.2 Functionality as a VM2000 guest system

The extra CPUs are invisible to BS2000 guest systems. They can only be used via VM2000 commands – see the “VM2000” manual [60].

With VM2000, in addition to the normal virtual CPUs (whose number is specified when configuring the VM), all guest systems are assigned as many virtual spare CPUs as the Server Unit configuration has real spare CPUs. The virtual CPU that fails when the real CPU it is currently running on fails is replaced by the affected guest system by a virtual spare CPU.

This is especially important for the availability of guest systems that only have one CPU assigned since such systems without a spare CPU are terminated immediately when CPU errors occur while it is running. With a (virtual) spare CPU, there is a realistic chance when a failure occurs that the guest system will be able to continue operation without interruption after an internal CPU reconfiguration.

5.6 NDM - resource allocation and reservation

NDM (Nucleus Device Management) is the actual device management tool of BS2000. The basic function of device management is to allocate devices. Since, as a general rule, not every device request can be fulfilled, NDM must also offer a device reservation function, manage queues and process these queues.

NDM records the number and status of the devices at system startup time and changes in these statuses during operation.

If an application program issues a task to a particular device, if this device is available and if it is allocated by NDM, then NDM must also ensure that the device is flagged as occupied and no other application program has access to the device. This device remains occupied until the application program with access authorization has finished working with the device and has notified NDM.

Command	**Meaning**
ADD-DEVICE-DEPOT	Defines assignment of tape device to depot
CHANGE-DISK-MOUNT	Locks private disk for access
CHANGE-TAPE-MOUNT	Changes mount status of tapes
CHECK-DISK-MOUNT	Checks mount status of a disk
CHECK-TAPE-MOUNT	Checks mount status of tape devices and MTCs
MODIFY-MOUNT-PARAMETERS	Changes parameters for mounting and demounting tapes and disks
MODIFY-RESOURCE-COLLECTION	Controls collector selection
REMOVE-DEVICE-DEPOT	Cancels assignment of tape device to depot
SECURE-RESOURCE-ALLOCATION	Requests resources
SET-DISK-DEFAULTS	Defines default values for DISK parameters
SET-DISK-PARAMETER	Sets defaults for monitoring disks
SHOW-DEVICE-CONFIGURATION	Display the input/output configuration
SHOW-DEVICE-DEPOT	Shows the assignment of tape device to depot
SHOW-DEVICE-STATUS	Requests allocation and monitoring information for devices
SHOW-DISK-STATUS	Shows assignment and DISK parameters
SHOW-MOUNT-PARAMETER	Shows monitor defaults for disks and tapes

SHOW-RESOURCE-ALLOCATION	Shows task allocation and open operator actions
SHOW-RESOURCE-REQUESTS	Shows status of device queue and collector task
SHOW-TAPE-STATUS	Shows tape allocation and monitoring
START-RESOURCE-COLLECTION	Starts collector selection
STOP-RESOURCE-COLLECTION	Terminates collector selection
UNLOCK-DISK	Corrects system allocation log
Macro	**Meaning**
NKDINF	Information on the allocation and availability status of the (peripheral) configuration
NKGTYPE	Information on the name, device type code, device attributes etc. of a device or volume type or on the names and device type codes of the device types that belong to a device family or class

Table 12: Overview of interfaces for resource allocation and reservation

The NDM information services provide the operator with certain output fields appropriate to the specified command and the desired scope of information.

A list of the SHOW commands and an explanation of the information which is output are provided in the Appendix to the “Commands” manual [27].

5.6.1 Task allocation of volumes

Device management controls and monitors the allocation of resources requested by the user (devices, volumes, files) depending on the volume usage mode.

For tapes and private disks, the following modes exist:

- "DMS"
  for all DMS accesses (using commands SECURE-RESOURCE-ALLOCATION, CREATE-FILE, ADD-FILE-LINK, COPY-FILE...)
- "SPECIAL"
  special applications (e.g. utility routines INIT, VOLIN, FDDRL ...)

In addition, the following usage mode is provided for tapes:

- "WORK"
  for DMS work tapes, see commands SECURE-RESOURCE-ALLOCATION, ADD-FILE-LINK in the “Commands” manual [27]

Task-exclusive allocation:

A volume cannot be allocated by more than one task at a time. All other task allocation requests for the same volume within the same system or from other systems are rejected.

Exception: exclusive allocation requests (SECURE-RESOURCE command) of batch tasks wait until the volume has been released.

Task-shareable allocation:

A volume can be allocated by more than one task at a time. The only volumes which can be allocated as task-shareable are disks in USE-MODE=DMS (public and private disks).

Allocation types for private volumes and other resources:

- Allocation of tapes

  Tapes are always allocated as task-exclusive; task-shareable allocation is not possible. The system determines whether the tape is only to be allocated or if it is actually to be used. If it is only allocated, i.e. other tasks have no access to the tape but I/O operations do not occur, the operator is requested via a premount message to carry out device allocation if the tape is not recognized as being mounted. If, however, the tape is to be used for I/O operations, the operator is informed of this via a mount request.

- Allocation of private disks

  Private disks are allocated as task-shareable for DMS operation (usage mode DMS) by default. Task-exclusive allocation is only possible via an appropriate reservation (SECURE).

  Disks for the usage mode SPECIAL are allocated as task-exclusive.

  The operator can use the disk parameter USER-ALLOCATION to specify which task allocation is permitted for a disk (usage mode DMS).

  USER=*EXCL enables a disk to be allocated as task-exclusive only (with SECURE-RESOURCE-ALLOCATION DISK= ...,ALLOC=*EXCL). The disk allocated in this way is then only available for this task. Any attempt to allocate this disk as task-shareable is rejected.

  USER=*SHARE enables the disk to be allocated for all DMS and SECURE requests exclusively as task-shareable. All task-exclusive allocation requests for this disk are rejected.

  USER=*ALL enables the disk to be allocated for all task-shareable and task-exclusive DMS and SECURE requests. This is only valid for initial allocation of the disk. If the disk is allocated as task-exclusive, all other allocation requests for this disk from other tasks are rejected.

  USER=*NO enables the disk to be reserved by a user. This setting is made automatically by the system in the event of: `/CHANGE-DISK-MOUNT UNIT=mn/*VOLUME(vsn), ACTION=*CANCEL` or if a remount message is answered with `tsn.N`.

- Allocation of other resources

  All allocation requests for devices (tape/disk devices, etc.) are allocated as task-exclusive without operator intervention.

5.6.2 System allocation of disks

Each disk allocation is stored in its standard volume label (SVL). For this purpose the SVL of the disk contains a system allocation log into which up to 16 catalog IDs of systems reserving the disk can be entered in the case of public disks. The types of allocation for disks are as follows:

System-exclusive allocation

A disk can be reserved by one system only. The system ID of the home pubset of this system is entered in the system allocation log of the SVL to prevent simultaneous use of the disk by other systems.

System-shareable allocation:

A disk of a shared pubset can be reserved by up to 16 systems simultaneously. The system ID of each system reserving the disk is entered in the system allocation log so that a maximum of 15 further systems can simultaneously reserve the disk.

> When the software product MSCF is used, the information in section “Shared pubsets” must be observed.

5.6.3 Default values for private disk allocation

ASSIGN-TIME

The DISK parameter ASSIGN-TIME enables the operator to set default values for the time of allocation or for the release of a private disk and of a disk device with the appropriate updating of the system allocation log on the disk.

- The allocation of the disk and the device on which the disk is mounted begins with ASS=*USER with the reservation of the disk by the user. At the same time the system ID is entered in the system allocation log. The disk and the device are released when the disk is released by the user, and the system allocation log is cleared.
- If ASS=*OPERATOR is used, the disk and device on which it is mounted are allocated independently of the allocation by the user. If the disk is already online, the disk and the device are allocated immediately, otherwise allocation occurs after the activation interrupt but no later than at the time of an allocation request by the user. The disk and the device remain allocated until the operator returns the allocation by entering ASS=*USER and all users have released the disk.

  Disks which are to be accessed by a system over an extended period, should be allocated via ASS=*OPERATOR. This avoids unnecessary allocation and release logs.

  If a disk is allocated with ASS=*OPERATOR then it is important to ensure that the setting ASS=*USER is made before the device is detached using the command DETACH-DEVICE ...,FORCE=*NO since the DETACH-DEVICE command will otherwise be rejected. Only in this way can the allocation log be cleaned up correctly.

Allocation start

- ASS=*USER with the first user request
- ASS=*OPERATOR
  - with the activation interrupt when mounting (ATTACH-DEVICE)
  - immediately if the disk is already online
  - at the first allocation request by the user

Allocation release

- ASS=*USER with last user release
- ASS=*OPERATOR
  - with or after SET-DISK-PARAMETER ASSIGN-TIME=*USER
  - with or after SET-DISK-PARAMETER ASSIGN-TIME=*STD (and default value ASSIGN-TIME=*USER set)
  - if no task is still reserving the disk

OPERATOR-CONTROL

By means of this DISK parameter the operator specifies whether initial reservations of disks by tasks are to be checked. If the operator activates monitoring for a disk, the following message is displayed with each initial allocation of the disk:

```
NKA0004 ALLOCATION OF DISK '(&00)' IN USAGE MODE '(&01)' BY USER TASK
        PERMITTED? REPLY (Y=ALLOCATION ACCEPTED; N=ALLOCATION REJECTED)
```

This message requests the operator to decide whether the specified disk of the requesting task is to be allocated in the allocation mode. Only in the case of a positive response to this message (`TSN.Y`) is the disk reserved by the task, otherwise this and all subsequent allocation attempts are rejected without any additional operator action.

If a disk for which the DISK parameter OPERATOR-CONTROL is set is already reserved by a job at this time, message NKA0004 is output for this disk only when it is released by the allocating job and is to be reallocated.

5.6.4 Controlling resource allocation

If a reservation job is given by a user (with the SECURE-RESOURCE-ALLOCATION) command), the system tries to reserve all the resources requested (devices, volumes or files) for this task.
If the requested resources are available and can be reserved for the task, the reservation job is completed and other user jobs can be processed.

If the reservation job cannot be executed at all or only in part, the job is entered in a queue (secure queue). Waiting jobs receive no resources (except possibly the collector task).

Collector task

The operator has the option of selecting one job in a secure queue for privileged handling. This means that the task stands at the head of the queue and is able to collect the resources requested. This task is known as the collector task. The commands START-RESOURCE-COLLECTION, MODIFY-RESOURCE-COLLECTION and STOP-RESOURCE-COLLECTION are available to the operator for controlling collector selection.

These commands enable the operator to execute the following functions:

- Starting COLLECTOR selection

  The operator activates the selection of a collector task by means of the START-RESOURCE-COLLECTION command. The system calculates a weight for each task in the secure queue in accordance with the following formula:

  `W = T + N * U`

  `W` Weight;

  `T` time; wait time the task has already spent in the secure queue

  `U` urgency; this value is calculated from the task priority, the task with the highest priority having the lowest urgency and vice versa

  `N` factor which is entered by the operator in the TIME-WEIGHT operand of the START-RESOURCE-COLLECTION command (default value is 10). The operator can influence the calculation of the weight by selecting factor N:
  If N is low, the wait time the task has already spent in the secure queue is assigned high significance.
  If N=0 is specified, the wait time is equal to the weight and the urgency has no influence on the calculation.

  If, however, N is high, the significance of the task priority increases for the calculation of the weight. For N=600 the weight is calculated almost exclusively from the urgency; the wait time is hardly significant in the calculation.

  After the system has carried out this calculation for all tasks in the secure queue, the task with the highest weight becomes the collector task. The collector task is placed at the beginning of the secure queue and can collect the resources requested. Once the collector task has collected all the resources it requested, it is removed from the secure queue, and the system recalculates the weights for the tasks remaining in the queue, whereupon a new task becomes the collector task.

- Excluding tasks from selection

  The operator can exclude a task from selection by means of the command MODIFY-RESOURCE-COLLECTION ACTION=*REMOVE-COLLECTOR. If the system determines a new collector task (see 1.), this task is not considered, i.e. it cannot become the collector task. If the task specified is the collector task at the time of exclusion, it loses its collector attribute and thus all resources which have been reserved up to this time.

- Admitting tasks for selection

  The operator can readmit a previously excluded task for selection (see 2.) by means of the operand ACTION=*ADD-COLLECTOR; This task is again considered by the system at the next collector selection.

- Declaring a task as the collector task

  The operator can declare a task as the collector task by means of the command MODIFY-RESOURCE-COLLECTION ACTION=*SET-COLLECTOR. If another task is the collector task, it loses its collector attribute and therefore the resources it has already collected. The task specified in the command becomes the collector task. This function is also possible without a previous START-RESOURCE-COLLECTION command.

- Terminating collector selection

  The STOP-RESOURCE-COLLECTION command enables the operator to terminate the collector selection started with the START-RESOURCE-COLLECTION command (see 1.). If a task is the collector task, it does not lose its collector attribute and can continue to collect the resources it requested. If the operator withdraws the collector attribute from this task (see 2.) or the collector task can collect all resources requested, no new collector task is selected by the system.

5.6.5 Supporting NDM handling

NDM offers the operator a wide range of control capabilities for volumes. Since these parameter settings are not dependent on the online status (and consequently not on dismounting the corresponding volumes), abnormal system behavior can result from not observing the current parameter settings (for example, this may result in the inability to allocate a disk even though it has already been mounted; if allocation of the disk is to be permitted anew, the operator has to modify the appropriate parameter value). All the values can be queried using information functions.

Other unexpected reactions can occur if the state of the hardware does not match the state of the software. (For example if a tape is mounted on a device in the ATTACHED state and the tape controller is not connected to the Server Unit, the tape cannot be allocated.)

The following sections describe the most important aspects of system behavior, including an explanation of the relevant situation in the SHOW output. Information is also given on how these situations can be dealt with.

The following situations are described:

- mount message in spite of volume being mounted
- SVL allocation by other systems
- permanent hardware error for allocated devices
- private disk permanently locked by another system
- checking for free disks
- SECURE deadlock situations
- changes to USER-ALLOCATION
- information on reservations
- proposals for performance-enhancing NDM parameter settings

Mount message in spite of volume being mounted

It may happen that a mount request is output at the console even though the operator has already mounted the volume requested. This may have the following causes:

1. Several volumes with the same VSN are online (SHOW-DISK-STATUS, SHOW-TAPE-STATUS). The operator must decide which volume is to be allocated.
2. The device on which the volume has been mounted is reserved by another task (SHOW-DEVICE-STATUS). The volume must be remounted.
3. The configuration state of the device (it is explicitly or implicitly detached) prevents allocation:
   - Controller or device is detached (SHOW-DEVICE-STATUS),
   - Path is (partially) removed (SHOW-DEVICE-CONFIGURATION).

   Remount volume or attach device/path (ATT/INC).
4. The hardware state of the device prevents allocation:
   - Device is not yet activated (disk),
   - Tape is not yet positioned to BOT,
   - Controller is not attached.
5. The activation interrupt for the mounted volume was not provided by the hardware. Reading the VSN can be enforced by CHECK-DISK-/CHECK-TAPE-MOUNT.

6. The mounted volume has a different VSN from the one specified by the user (SHOW-DEVICE-STATUS, SHOW-DISK-STATUS, SHOW-TAPE-STATUS).
7. Other causes in the case of disk requests:
   - The device type specified by the user does not match the type of private disk mounted (SHOW-DISK-STATUS VOL=vsn,INF=*PAR). Reject the mount message or, if required, assign a second disk with the same VSN but with another device type.
   - VOLIN: UNIT specification requires a device different from the device on which the disk has been mounted. To continue, the volume must be remounted or the mount message must be rejected and the user be made to release the device.
   - The requesting task has reserved the device on which the disk is mounted by means of /SECURE-RESOURCE-ALLOCATION UNIT=mn and wishes to reserve the disk for DMS operation (USE=*DMS). Since disks with UNIT requests are only reserved for the usage mode SPECIAL, the mount message must be rejected and the user must be requested to release the device.
   - A disk cannot be reserved by other systems on the basis of its SVL allocation: For a description of the possible reactions see section "SVL allocation by other systems").
   - A disk is to be reserved which was used during the last session as a DRV disk and the DRV subsystem is not (yet) loaded. As a reaction the DRV subsystem must either be loaded and the mount message rejected to enable the user to make a new allocation attempt or the disk must be forced to operate in SRV mode by means of a positive response to the mount message.
8. Other causes in the case of tape requests:
   - MODIFY-MOUNT-PAR ALLOC=*NO is set (SHOW-MOUNT-PAR): A reply to the mount message is required.
   - The volume type specified by the user is not supported by the device on which the volume is mounted (SHOW-RESOURCE-ALLOCATION, SHOW-DEVICE-CONFIGURATION). The tape must be remounted.
   - A different device from the one on which the tape was mounted was explicitly reserved by the user (SECURE-RESOURCE-ALLOCATION UNIT=mn). The SECURE-UNIT request of the user is mandatory: the tape must be remounted on the specified device.

SVL allocation by other systems

Even if a disk has already been logged as online, a user's allocation request can result in a mount message for this disk. This is the case if an inconsistency is detected between the valid SYSTEM-ALLOCATION of the disk and its actual SVL allocation by other systems. The mount message, which is preceded by a notice indicating this discrepancy, is designed to request the operator to make a decision or a response.

The possible operator responses and actions are explained below. The starting point for the considerations is the output of SHOW-DISK-STATUS VOL=vsn,INF=*ALL

1. All systems stored in the SVL (SYSTEMS) are no longer active:
   The systems entered can be removed with UNLOCK-DISK VOL=vsn,SYS-ID=(...). There is then an automatic reply to the mount message and the allocation is thus accepted.
2. Systems entered are still (to some extent) operating with the disk and the disk is to be allocated as system-exclusive by the home system:
   The mount message must be rejected.
3. Systems entered are still (to some extent) operating with the disk and the disk is to be allocated as system-shareable by the home system:

- SVL-ALLOC=EXCL:
  The mount message must be rejected as the disk cannot be allocated concurrently by another system.
- SVL-ALLOC=SHARE und alle Systeme aktiv:
  16 external systems (in the case of shared disks) are stored in the SVL: Procedure as described in 3a)
- SVL-ALLOC=SHARE and a part of the system entered are no longer active: the inactive systems can be removed as described in paragraph 1 above.
  For other procedures see 3b).

Permanent hardware error for allocated devices

If a tape or disk unit reserved by a volume which is mounted and used by one or more users constantly reports hardware errors (INOP,...), the following responses are possible:

1. The volume is a fixed disk:

   If the hardware error cannot be corrected, the remount message must be rejected if the disk is a private disk (implicit volume cancel); if it is a public disk, BS2000 operation comes to a standstill.

2. The volume is a tape:

   The affected volume is to be mounted on another device (command CHANGE-TAPE VOLUME=vsn, ACTION=*MOVE). If there is still a free device available, the system proposes a standby device to the operator by means of the remount message. Rejection of the command can have the following causes:

   - Volume allocation was preceded by SECURE-RESOURCE-ALLOCATION UNIT=mn. Consequently the volume is forced to remain allocated to the device while it is being used by the allocating task. If the hardware error cannot be corrected, `TSN.N` must be entered in response to the remount message. This results in the volume being cancelled.
   - No further standby device is available.

   In each case, the device must be detached with DETACH-DEVICE UNIT=mn, FORCE=*YES.

   Detachment of the device can have the following effects:

   - The volume is implicitly canceled if it has a non-standard label, or if the allocation is preceded by SECURE UNIT=*mn.
   - The volume is placed in the action state “NO DEVICE”, i.e. as soon as a device becomes free it is allocated to the private volume.

   When detaching devices with replaceable volumes, waiting for a suitable standby device is supported in all phases of use, i.e. both in the IN-USE state and during loading (MOUNT or PREMOUNT state) and for pure device type allocations, e.g. after SECURE-RESOURCE-ALLOCATION DEVICE=(TYPE=TAPE-C4,NUMB=1).

   The “NO DEVICE” action state may occur for volumes in use or during loading immediately following DETACH UNIT=mn,FORCE=*YES if no suitable standby device is available. A “free” device may be available in the sense that although a volume is not currently assigned to it, a type allocation exists for it. This allocation is suppressed.

   The relevant task waits - during the loading phase for the volume linked to this type allocation - for a suitable standby device to become free.

   Volumes in the IN-USE state displace volumes in the PREMOUNT state. Displacement of a volume in the MOUNT state is performed automatically only for manually operated devices. In the case of devices supported by ROBAR, a mount operation cannot be aborted once it has been started, i.e. the remount/recover routine is terminated with NO DEVICE and the mount operation is completed (attributes of ROBAR).

When a device becomes free as a result of ATTACH-DEVICE or the end of allocation of a another volume, the volume to be assigned is selected according to the phase (the sequence being: IN-USE before MOUNT before PREMOUNT) if more than one volume which, on the basis of the type and depot allocation, could be processed on this device is in the NO DEVICE state.

Private disk permanently locked by another system

A disk cannot be allocated if another system is stored in the SVL of the disk as a VTOC lock holder (see the VTOC-SYS field of SHOW-DISK INF=*SYS). The operator is informed of this state in the following cases:

- An allocation request by the user results in message `NDV0002` being output (the disk is permanently locked by another system).
- An implicit UNLOCK for the home system ID when the disk is mounted, or an explicit UNLOCK (UNLOCK-DISK VOL=vsn,SYS-ID=sys-id) also results in message `NDV0002`.
- SET-DISK-PARAMETER VOL=vsn,ASS=*OPER is rejected with a reference to the VTOC lock holder.

It must be ensured that the system entered as the VTOC lock holder is no longer working with the disk and the lock is not retained because of a malfunction (e.g. system crash) or due to the cancellation of the disk. Only in such a case can the lock holder be removed; In all other cases the operator must wait until the lock is released by the allocating system, otherwise side effects may occur.

For a) and b) the lock holder can be removed by entering `tsn.F` in response to message `NDV0002`.

For c) the removal of the allocating system must be initiated explicitly with UNLOCK-DISK VOL=vsn,SYS-ID=sys-id. This request also results in message `NDV0002`, to which the operator must respond with `tsn.F`, as described above.

Checking for free disks

A check is carried out to determine whether a disk is free, for example, when it is to be initialized. A disk is not allocated by users if the SHOW-DISK command returns the following information:

- PHASE=ONLINE
- PHASE=IN-USE and VOL-A=FREE for disks with usage mode USE=DMS; In this case, the disk is allocated only by its ASSIGN-TIME and can be released by SET-DISK-PARAMETER VOL=vsn,ASS=*USER[,USER=*NO].

SECURE deadlock situations

The messages `NKS0022` and `NKS0054` inform the operator of deadlock situations recognized during SECURE processing of the tasks specified. These two messages refer to the following deadlock situations:

1. `NKS0022` - Deadlock due to resources which the tasks can retain after SECURE processing:

   The deadlock can only be resolved by canceling one or more tasks.

Example

TASK 11.	TASK 2
/CREATE-FILE FILE-NAME=DAT1,- SUPPORT=PRIVATE-DISK(VOLUME=- VOL1,DEVICE-TYPE=DEV1) /ASSIGN-SYSLST TO-FILE=DAT1 : /SEC-RES DISK=(VOL=VOL2,- TYPE=DEV2,- ALLOC=EX),- WAIT=.....	/CREATE=FILE FILE-NAME=DAT2,- SUPPORT=PRIVATE-DISK(VOLUME=- VOL2,DEVICE-TYPE=DEV2) /ASSIGN-SYSLST TO-FILE=DAT2 : /SEC-RES DISK=(VOL=VOL1,- TYPE=DEV1,- ALLOC=EX),- WAIT=.....

Both task 1 and task 2 reserve for themselves a private disk in task-shareable mode by opening a private disk file (ASSIGN-SYSLST). The SECURE request of both tasks for task-exclusive allocation of the disk allocated as task-shareable by the other task results in both tasks waiting for the disk to be released.

Solution:
This classic deadlock situation can only be resolved by canceling one of the two tasks.

2. NKS0054 - Deadlock due to collecting

   If a collector task is selected during the current session, the following deadlock situation can occur:

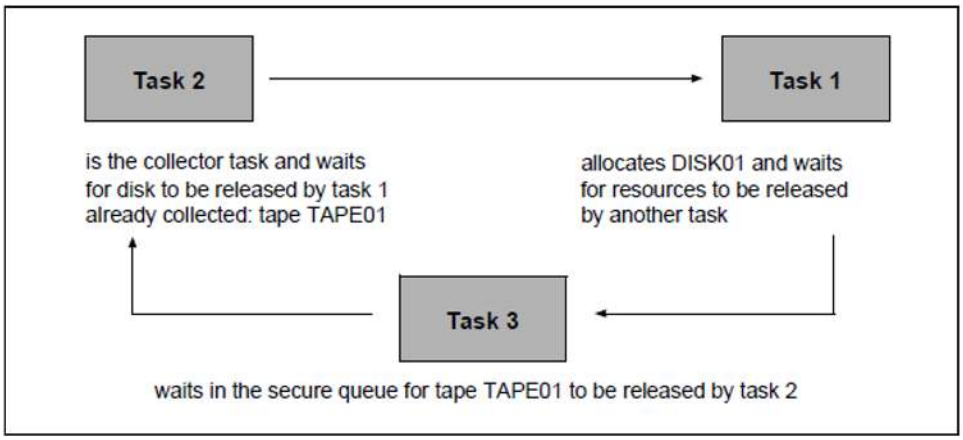


Figure 4: Deadlock situation because of collecting

Task 1 allocates disk DISK01 with /CREATE-FILE FILE-NAME=abc and /ASSIGN-SYSLST TO-FILE=abc and waits for resources to be released by another task with /SECURE-RESOURCE-ALLOCATION FILE=(NAME=FILE1,ALLOC=EX).

Task 2 is the collector task which waits for the disk to be released by task 1 with

/SECURE-RESOURCE-ALLOCATION DISK=*PAR(VOL=DISK=01,TYPE=D3435,ALLOC=EX),

TAPE=(VOL=TAPE01,TYPE=TAPE-C4),WAIT=...

Task 3 waits in the secure queue for the tape to be released by task 2 during the allocation of the file FILE1 /SECURE-RESOURCE-ALLOCATION
TAPE=(VOL=TAPE01,TYPE=TAPE-C4).

Solution:
In this case the establishment of task 3 as collector task would cause tape TAPE01 to be released by task 2 (task 2 loses its collector attribute). Task 3 therefore receives the resources it requested and leaves the secure queue. After task 3 has released the resources, the request of task 1 can be met, and after that task's resources have been released, the request of task 2.

Changes to USER-ALLOCATION

The value of USER-ALLOC for a disk determines whether the disk can be reserved by the user. If "NO" is set, all allocation requests are rejected without any indication for the operator. It is therefore important for the operator to know in which cases (implicit) parameter modification occurs.

1. SET-DISK-PARAMETER VOL=vsn,USER=....
2. If USER-ALLOC=*STD is defined for a disk, the applicable actual value is that set with SET-DISK-DEFAULTS USER=.... . Thus if the default value changes, this private disk is also affected.
3. When a disk is canceled with CHANGE-DISK-MOUNT VOL=vsn,ACTION=CANCEL then USER-ALLOC=*NO is set.
4. Response to a remount message with `tsn.N` (rejection) results in implicit cancellation of the disk (see point 3 above).

Information on reservations

1. Task-specific

   The command SHOW-RESOURCE-ALLOCATION (SH-RES) provides information on the following resource allocations:

   - unit reservations
   - device type reservations
   - tape/disk reservations (explicitly via SECURE-RESOURCE-ALLOCATION or implicitly by opening or reserving files on private volumes)

2. System-global

   SHOW-DEVICE-STATUS UNIT=*SEL-TYPE(ATTR=*FREE[,TYPE=xxx] is used to output a list of all devices (of a specific type) which have not yet been allocated explicitly (by SECURE-RESOURCE-ALLOCATION UNIT=mn) or implicitly (by allocation of a volume mounted on them);; This list, however, does not reflect the actual number of devices that can still be allocated: there are reservations which have not yet been allocated to a device (unit), e.g. type reservations by means of the SECURE-RESOURCE-ALLOCATION command.

   An overview of the actual number of devices reserved, the type of reservation and the number of devices of a certain type which can still be allocated is provided in the output for the command SHOW-DEVICE-STATUS INF=*SUM[,TYPE=xxx].

   SHOW-DEVICE-STATUS UNIT=*SEL-TYPE(TYPE=xxx), INF=*TASK shows the allocating or reserving tasks and the number of devices they occupy/reserve.

Proposals for performance-enhancing NDM parameter settings

1. Private disks used in DMS mode are repeatedly reallocated and released when files are processed by DMS at command level (COPY-FILE, DELETE-FILE, ADD-FILE-LINK, CREATE-FILE). If task allocation and release results in SVL reallocation or release of the private disk, SVL management actions (entry/removal of the system ID in the SVL of the disk) must be effected in addition to DMS input/output. If, however, the disk remains allocated after it has been released by the user (owing to continued valid reservation by another user or ASSIGN-TIME=*OPERATOR), the considerable overhead for SVL management is no longer required. A private disk used for DMS mode should therefore remain allocated through its ASSIGN-TIME until it is no longer needed and is specifically released by the operator. This can be effected by means of
   - SET-DISK-PAR VOL=vsn,ASS=*OPER
   - SET-DISK-DEFAULTS ASS=*OPER (valid for all disks with ASS=*STD)

     ℹ If SPECIAL applications are required, b) is not recommended as an alternative, since disks mounted for SPECIAL usage later on are automatically allocated and must be explicitly released by the operator before being processed by the SPECIAL application (with SET-DISK-PAR VOL=vsn,ASS=*USER).

2. For data center operation where the SPOOLOUT is to be stored externally on tape, MODIFY-MOUNT-PAR UNLOAD=*NO (presetting) should be set, since otherwise the relevant tape is unloaded and made ready again by the operator each time a SPOOL file is rolled out.

5.7 Volume monitoring

Device management supports and monitors disk and tape usage, executes mounting and remounting procedures and protects used tapes and disks from erroneous intervention.

For this purpose device management uses two permanent tasks, the DISK-MONITOR (DM) for disk monitoring and the TAPE-MONITOR (TM) for tape monitoring. Temporary tasks are started for functions which temporarily inhibit permanent tasks (e.g. tape positioning).

Mounting of a volume before it is used or in the case of an allocation request

If the user requests a volume, a mount message is issued at the console if the volume has up to now not been recognized as mounted (`NKVD013/NKVT013`; this depends on whether a disk (D) or a tape (T) is to be mounted).

If volume monitoring recognizes the mounting of the requested volume (by evaluating the activation interrupt from the device), the outstanding message is automatically answered by the system. The operator is therefore not required to respond to the mount message.

Remounting of volumes during use

If a (defective) device is detached by reconfiguration (DETACH command), the operator is requested by the volume monitor to carry out remounting (if a standby device is available).

If no standby device is available, a temporary overload is accepted, i.e. the system holds a renewed mount request until a device is available or can be attached (ATTACH command). However, the operator can terminate this wait state by explicit withdrawal of the volume (CANCEL function of the CHANGE-TAPE-MOUNT command).

Protection of volumes from illegal device intervention and repositioning of tapes

Each operator intervention on a tape or disk unit (unloading a tape, deactivating the device) with a volume that is in use results in a temporary I/O lock for the user.

If during an I/O request it is determined that the volume is no longer available, a request is issued for the operator to make the volume ready again (remount message).

After reassigning the volume, the I/O operations can be continued. With tapes, automatic repositioning takes place. If hardware resets of the devices are reported to the monitors, this also leads to identification of the volumes and, in the case of tapes with PHASE=IN-USE, to repositioning.

(Automatic) removal of a volume overload

If a device is withdrawn from an allocated tape in USE-MODE=DMS (DETACH UNIT=..., FORCE=*Y) without a standby device being available, a temporary overload is created (SH-TAPE: output column ACTION=NO DEVICE), i.e. more volumes of this device type are in use (SHOW-TAPE-STATUS: column PHASE=IN-USE), than devices are available.

The operator can use the command CHANGE-TAPE-MOUNT ...,EXCHANGE=(...) to influence or determine which volumes should temporarily not be available. If a device of an appropriate type is free, the operator is automatically requested to mount the volume (by means of ACTION=NO DEVICE) by the volume monitor.

5.7.1 Device selection mechanisms for tape devices

In the selection process, NDM searches the device table from the beginning, i.e. in the sequence defined at hardware generation time, and selects the first suitable free device (BEST-GENERATED-DEVICE). As a result, the first devices in the table are the ones more often selected and – particularly in connection with robot-operated magnetic tape archive systems such as ROBAR which always follow the mount proposal of NDM – are the ones subject to greater wear than the devices at the end of the table.

The “wrap around” function is provided for users interested in ensuring all their tape devices suffer the same amount of wear. This function can be activated with the MODIFY-MOUNTPARAMETER NEXT-TAPE-MOUNT=LEAST-RECENTLY-USED-DEVICE command. From the suitable free devices in the device table, NDM then selects the one which has remained unused for longest.

The MODIFY-MOUNT-PARAMETER ...,NEXT-TAPE-MOUNT=*BY-CONTROLLER command is offered for users who are interested in a device selection which is optimized for input/output throughput. A device can be selected from the suitable free devices on whose controller the fewest devices are in use. This ensures even distribution of the devices used over the controllers and channel paths available. Device selection in NDM only takes into account the local device assignments. The IORM function DDAL (see "IORM: Control of I/O resources") is used to extend optimization to all the VM2000 guest systems of a system.

The default setting of the device selection mechanism is NEXT-TAPE-MOUNT= *BESTGENERATED-DEVICE, i.e. NDM searches the device table from the beginning in the order specified during hardware generation and selects the first suitable free device.

Device selection in accordance with storage locations

Storage location management has been developed in NDM, in particular to support robot-controlled archives in BS2000 (software product ROBAR, see section "Archiving systems").
Volumes and the devices on which they are to be mounted can be assigned to storage locations in BS2000. Declaring storage locations enables volumes and devices from a precisely defined set to be selected and reserved.

Tape devices are assigned to storage locations by using the ADD-DEVICE-DEPOT command. 1024 tape devices are permitted per storage location. Assignment of a VSN to a storage location is managed by the software product MAREN. The storage locations set down there must match the specifications made by the operator with the ADD-DEVICE-DEPOT command.

Each storage location is identified in the MAREN catalog with its name (max. 8 characters), its type (REMOTE or LOCAL) and its operating mode.

> ¡ Working with storage locations is only possible in conjunction with MAREN (see "Tape management with MAREN").

The effects of the ADD-DEVICE-DEPOT command on device selection in NDM are explained below in the description of the basic principles. This description makes a distinction between two different types of allocation request (tape device request and tape volume request).

Device selection for tape device requests

There are three types of request that can be issued by a user for a tape device:

1. Request via the mnemonic device name (MN)

   The user explicitly requests a certain device via its mnemonic device name; no selection takes place.

2. Request with specification of storage location

   The user requests a tape device by specifying a device or volume type together with a storage location.

   If the request is made using the SECURE-RESOURCE-ALLOCATION command, NDM only takes the specified storage location into account when making the selection. If no device is available there the user's job is rejected.

   If the request is made via an internal interface used by ARCHIVE, the caller must determine the storage location beforehand using MAREN. If this storage location is known to NDM, i.e. was set up by means of the ADD-DEVICE-DEPOT command, NDM selects a tape device from this storage location. Otherwise, NDM accesses the set of devices that supports the specified type but is not assigned to any specific storage location. This set of devices is called the "restpool".

3. Request without specification of storage location

   The user requests a tape device by specifying a device or volume type but without specifying a storage location.

   If the request is made using the SECURE-RESOURCE-ALLOCATION command with the operand TYPE=TAPE-C4,...,LOCATION=*USER-DEF, for example, NDM attempts, via MAREN, to determine a default storage location for the specified volume type or a volume type supported by the specified device type. If this storage location is known to NDM, a tape device is selected from it. Otherwise, NDM accesses the "restpool".

   If the request is made using the SECURE-RESOURCE-ALLOCATION command with the operand LOCATION=*NONE, the MAREN call is omitted and the "restpool" is accessed directly.

Implicit device selection for tape volume requests

There are two types of request that can be issued by a user for a tape volume:

1. Request with specification of VSN

   If the VSN is specified in the request, the storage location defined for this VSN in MAREN is used. If there is no entry for this VSN in MAREN, then MAREN will return either a storage location determined via the MAREN exit routine or a default storage location.

2. Request without specification of VSN

   The tape volume request is made without specifying a VSN.

   In this case, NDM attempts, via MAREN, to determine a default storage location for the specified volume type or a volume type supported by the specified device type. If this storage location is known to NDM, a tape device is selected from it. Otherwise, NDM accesses the "restpool".

5.8 Parallel Access Volumes (PAV, SUs /390)

In BS2000 every disk is represented by a PDT (Physical Device Table) entry. Every I/O request for a disk is passed on by the device driver for execution to the I/O controller IOCNTRL in the form of a channel program.

If the PDT entry shows that the device is not active, the I/O request is started. Every additional I/O request for this device is only placed in a device queue by IOCNTRL at first. Only when the device is not active anymore will the next request in the queue be started. Since only one I/O request is allowed per disk at a time, the total duration of an I/O operation is the sum of the actual duration of the I/O operation plus the wait time.

Long wait times can be caused by:

- several applications with high I/O loads working at the same time on a disk.
- an application issuing its I/O request asynchronously to the IOCNTRL.

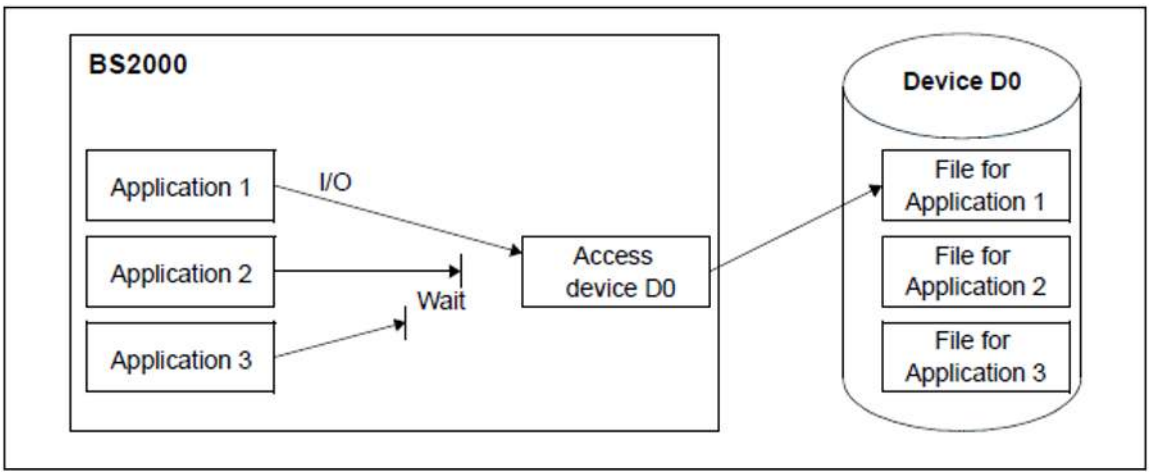


Figure 5: Several applications with I/O accessing a disk (traditional)

> On the SU x86, IOCNTRL can by default start multiple requests for emulated disks in parallel (RSC function). No special settings are required.

To implement PAV functionality, BS2000 utilizes the fact that on a Fibre Channel a device can accept an I/O request while another I/O is still active. However, “Command Queuing” may not be disabled on the RAID system. PAV can be placed in service without any intervention in the RAID system.

A logical Parallel Access Volume is represented by a base device (BASE) and up to 7 alias devices (ALIAS). Base and alias devices are generated during the hardware generation as independent devices of the same type with different unit addresses and mnemonic device names, although they are generated in the same logical controller.

Alias devices are generated using IOGEN (see the “System Installation” manual [55]) or, during ongoing operation, using the ADD-IO-UNIT command with the PAV-ALIAS-DEVICE operand set.

The relationship between the base device and alias devices is determined dynamically by BS2000 when a device is attached with the ATTACH-DEVICE command. When BS2000 detects a base device, the corresponding alias devices are searched for on the same logical controller. These alias devices are implicitly attached. Alias devices can also be explicitly attached with ATTACH-DEVICE. However, an alias device cannot be attached if the associated base device is not attached.

When a base device is detached with the DETACH-DEVICE command, associated alias devices are always detached implicitly. An alias device can also be explicitly detached with DETACH-DEVICE.

When a base device and its alias devices are attached, IOCNTRL can start I/O requests for the PAV volume in parallel via base and alias devices.

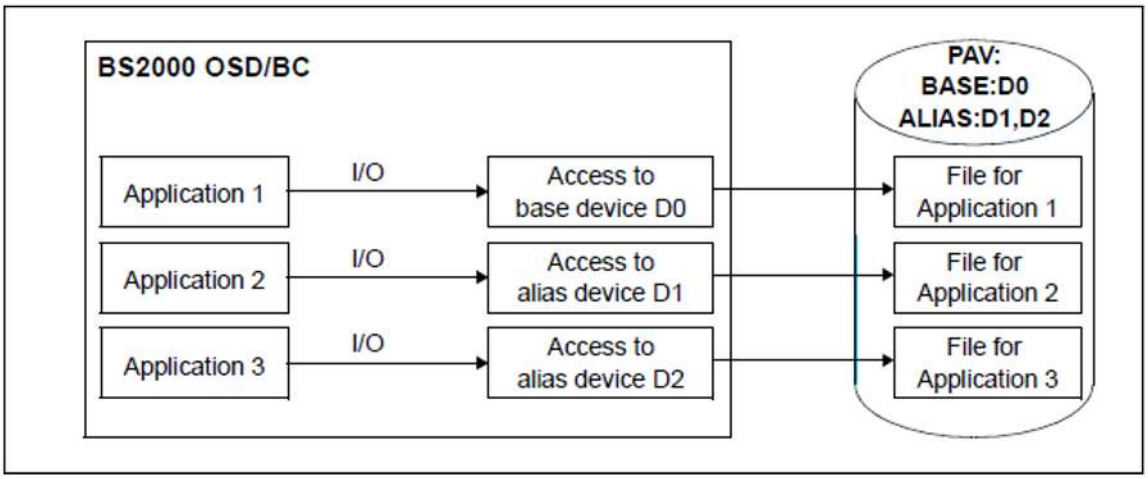


Figure 6: Several applications with I/O accessing a disk (PAV)

A PAV volume is allocated (IMPORT-PUBSET or SET-DISK-PARAMETER command) via the base device. Even the read and write I/O requests are made via the base device in BS2000.
IOCNTRL distributes the I/O depending on the loads on the base and alias devices onto one of these. IOCNTRL in the native mode prefers the base device when all have the same load.

Information on all PAV devices is output with the SHOW-DEVICE-CONFIGURATION ..., INFORMATION=*PAV command.

Notes for use under VM2000

In order to obtain as even a distribution of I/Os for shared disks as possible under VM2000, IOCNTRL in the guest systems prefers the base device or an alias device, depending on the VM index of the virtual machine.
However, base or alias devices can also be explicitly defined for each guest system. The MODIFY-IO-UNIT UNIT-DEVICE=*DEVICE(NAME=...,STATE=*PAV-PREFERRED-DEVICE(...)) command is available for this purpose. One alias device can be specified as the preferred device for performing I/O for each guest system.

You will find detailed information on PAV under VM2000 in the “VM2000” manual [60].

Recommended applications for PAV

The use of PAV is urgently recommended for ETERNUS disk storage systems.

PAV provides enormous improvements in TP and batch mode when a large proportion of the data is located in the disk storage system’s cache. Simultaneous cache hits are supported; in parallel, a cache miss with physical disk access can also be executed. For a heavily utilized disk connected via n paths it can therefore make sense to generate n-1 alias devices. This enables the n-fold throughput of a disk without alias devices to be achieved.

When data backup takes place, it must be assumed that the cache is not sufficient and that the data must therefore be fetched from the disks for saving or written to the disks when it is restored. This is highly optimized in the disk storage system by means of asynchronous “Read Ahead” and “Delayed Fast Write”. The performance of these functions restricts the throughput during data backup.

To permit optimum data backup, the use of “large” Raid systems with high-speed disks is recommended, as is generation of a sufficient number of alias devices.

Extended PAV (XPAV)

The “extended PAV (XPAV)” function removes the restriction that an alias device must be located in the same logical controller as the base device. In addition to an existing logical controller, one or more logical controllers can be configured with 256 alias devices.

XPAV is particularly suitable for configurations in which PAV was not planned from the outset or in which no or only a few free addresses are available for alias devices.

Two XPAV variants are possible:

1. When the existing logical controller and the additional controller with the alias devices are generated on the same controller ports, no manual intervention in the disk controller nor any additional cabling is required.
2. When the existing logical controller and the additional controller with the alias devices are generated on different controller ports, the volumes in the disk controller must be assigned to additional ports. Additional cabling may also be required for the alias devices.

The generation restrictions are described in the “System Installation” manual [55].

Dynamic PAV (DPAV)

Static PAV requires careful planning with regard to future device utilization by generating a suitable number of alias devices. Naturally it is also possible to assign one or more alias devices to all disks in advance.

Dynamic PAV (the IORM function DPAV, see "IORM: Control of I/O resources") requires fewer alias devices. As with static PAV, alias devices must be generated, but it is not necessary to provide the maximum number of alias devices required for each volume up front. DPAV autonomously assigns alias devices to the volumes which profit most from this.

> For Server Units that support FastDPAV, it is no longer recommended to use DPAV.

Fast Dynamic PAV (FastDPAV)

The “FastDPAV” function, an optimized DPAV, is offered for Server Units SU /390 that support a modification of the logical unit number (LUN) for alias devices when starting an I/O.

Advantages of FastDPAV compared to static PAV:

- FastDPAV requires significantly fewer alias devices.
- FastDPAV allows for a higher number of “normal” volumes, clones and snaps on the channel.
- FastDPAV allows for larger volumes despite identical I/O performance.

Advantages of FastDPAV compared to DPAV:

- FastDPAV does not require IORM for monitoring device utilization and switchover.
- FastDPAV requires no coordination with other systems.
- Alias devices are immediately available whenever required.

When using FastDPAV, a pool of FastDPAV alias devices is generated for a set of logical volumes with identical channel paths (connection from the channel to the port on the disk storage system), without a permanent assignment to one of these logical volumes. For a FastDPAV alias device, the LUN is only defined/modified when an I/O is started.

Notes for use

In native mode, BS2000 automatically activates all generated FastDPAV alias devices for I/O.

In VM mode, BS2000 only activates those generated FastDPAV alias devices that have been exclusively assigned to the guest system (or monitor system). The devices can be explicitly (using ADD-VM-DEVICES) or implicitly (using ATTACH-DEVICE) assigned, as long as the system has the required authorization.

When performing I/O, BS2000 only accesses the activated FastDPAV alias devices.

They can be deactivated using DETACH-DEVICE. If, in VM mode, the alias device has only been assigned implicitly, it is also withdrawn from the virtual machine.

FastDPAV can be used with or without static PAV.

5.9 Administration of private volumes

- Tape management with MAREN
- Applications of private disks

5.9.1 Tape management with MAREN

The software product MAREN is provided to assist systems support in the management of tapes in the data center. MAREN stores its information about tapes in an ISAM file, the MAREN catalog.

Without MAREN it is not possible to work with storage locations (see "Device selection mechanisms for tape devices").

MAREN catalog

To keep the information in the MAREN catalog up to date at all times, the MAREN system is linked to BS2000. MAREN registers every DMS access. The MAREN catalog is updated each time a tape is processed.

The MAREN catalog is based on the volume serial numbers (VSN) of the tapes. The volume serial numbers must be unique, but require no special numbering concept.

Tape assignment and automatic initialization

An important component of the MAREN system is the automatic free-tape assignment facility MARENUCP. In the event of an unspecified tape request (“SCRATCH” tapes), for example, free tapes can be automatically assigned from the MAREN tape pool. At the same time, the processed tapes are marked as reserved in the MAREN catalog. MARENUCP checks whether a tape that has been requested and mounted by the operator has to be initialized. If necessary, initialization is first performed before reservation and assignment take place.

Interfaces for BS2000 tape processing

In magnetic tape reservation and processing via the BS2000 Data Management System, the MAREN system is automatically activated for the following tasks:

- Access authorization check
- Availability check
- updating of archive entries
- Output of transport messages
- Insertion of device parameters

Invocation of these test routines may be triggered by the following events:

- reserving a tape
- creating a TFT entry
- opening a file
- changing reels
- closing a file
- releasing a tape

Access authorization check

Before a tape is processed, MAREN checks whether the caller is authorized to access the volume. The following conditions must be met before each access:

- If the tape is assigned to a foreign user ID, it must possess the attribute USERACCESS=*ALL-USERS in the MAREN catalog (exception: callers under TSOS).

- If the tape is protected by a password, this password must have been specified.
- If the tape is associated with an ARCHIVE directory , this tape can only be accessed with ARCHIVE.
- For output to tape, write access must be permitted; the FOREIGN-READ-ONLY attribute must not be present in the MAREN catalog (exception: callers under TSOS).
- For output to tape the file expiration date must be smaller than or equal to the current date.
- If MAREN exit routines are being used, they must permit access to the tape.

If any of these conditions is not met, access to the tape in question is refused and a corresponding message is issued.

Availability check

Before allowing access to a tape, MAREN checks whether the following conditions for the availability are met:

- The VSN of the tape exists in the MAREN catalog.
- The VSN designates a tape according to the DEVICE-TYPE attribute.
- The tape has not been lent out.
- The tape has been reserved for the caller.
- The tape is available in the local data center, i.e. the storage site defined in a special field of the archive entry is not marked “remote” in the storage-site table.
- The tape is not being processed on another system (the check whether processing is taking place on the local system is performed not by MAREN but by BS2000.)
- In the INPUT processing mode (except for access via ARCHIVE), the file names in the MAREN archive entry coincide with those in the TFT. Name portions denoting the catalog ID, user ID and version are ignored.

Detailed descriptions of the installation and use of MAREN in the data center can be found in the “MAREN” manuals [31].

5.9.2 Applications of private disks

In BS2000, disks can always be utilized either as private disks or as public disks.

Prior to their first use the volumes must be initialized with the VOLIN utility routine. There will be written a volume serial number (up to 6 characters) to the standard volume label. This volume serial number (VSN) serves to distinguish private from public disks.
The VSN of a private disk is freely selectable, with the restriction that the name must not be the same as that of a public disk. The rules and possibilities for assigning a VSN for public disks are explained in the section "Pubset properties".

There are two possible applications for private disks:

- DMS applications
  The disk is reserved by one or more DMS applications.
  By default, the allocation mode is defined as task-shareable. Task-exclusive allocation is effected by means of the user command SECURE-RESOURCE-ALLOCATION. The system allocation mode depends on the generation characteristics of the device on which the disk is mounted. With the operator command SET-DISK-PARAMETER, the system allocation mode can be set independent of the generation value and as a function of a specific disk.
- SPECIAL applications
  The disk is reserved for a privileged application, e.g. VOLIN, FDDRL.
  Special applications are always task-exclusive and system-exclusive allocations. The duration of an allocation is determined by the special application itself and cannot be influenced by the operator.

Dual recording by volume (DRV)

With the help of the DRV subsystem, not only public disks can be managed in duplicate but also private disks. Shared pubsets cannot be used for DRV. Each DMS write job is executed in duplicate and each read job is processed on the disk that is less heavily used at that time.

Requirements

In order for two disks to be operated as a single logical disk, the following conditions must be met:

- the same device type
- the same controller type
- the same channel type
- the same logical structure based on the same VOLIN parameters
- the same VSN
- the same time stamp
- homogeneous entry for the DRV recording method

Operation and control

The entire functionality of DRV is implemented in the input/output system, in NDM and in the DRV subsystem and need not be considered either by DMS or by the application programs.

DRV operation is initiated, controlled, monitored and terminated exclusively by systems support.

The DRV product is described in detail in the “DRV” manual [17].

5.10 IORM: Control of I/O resources

The IORM utility routine enhances the I/O properties of BS2000 in native and VM2000 modes.

The following functions are implemented in IORM (and the IORM subsystem) for autonomous, dynamic control of the I/O resources channel, controller, path and device:

- IOPT: I/O Priority Handling for Tasks
- DPAV (SUs /390):
  Dynamic Parallel Access Volume
- DDAL: Optimized Dynamic Device Allocation when using ETERNUS CS
- TCOM: Dynamic Tape Compression
- IOLVM (SUs /390):
  I/O Limit for Virtual Machines (individual VM2000 guest systems)

During ongoing operation, IORM collects data on the utilization of the I/O resources and controls I/O operation in accordance with specified threshold values.

The IORM functions IOPT, DPAV and IOLVM control disk devices.
The IORM functions DDAL and TCOM control tape devices.

When IORM is used in the monitor system and in the BS2000 guest systems involved in VM2000 mode, the IORM subsystems exchange I/O data and control information via an internal interface.

IORM can be used in native mode and under VM2000. IORM does not work across more than one Server Unit, however.

A detailed description of IORM is provided in the “Utility Routines” manual [15].

5.11 Displaying and checking the SAN configuration

The Server Units are connected to state-of-the-art storage systems via Fibre Channel. Here the storage systems are generally not connected directly to a Server Unit’s Fibre Channel Adapter but via a **switch**. Such an FC switch at the same time permits multiple connections between the devices which are connected at its ports. Other switches can, in turn, be connected to an FC switch via special ports. A network formed from one or more FC switches is called a fabric. A network of several storage systems which are connected with FC switches is referred to as a Storage Area Network (SAN).

From the BS2000 viewpoint, the FC switches are transparent. BS2000 uses the controllers and devices which are connected via Fibre Channel without having information on the connections in the fabric.

When problems are encountered in attaching devices or faults occur in ongoing operation, it is often difficult to detect the reason for these. An INOP or NINT message of the device error recovery can be caused by connection problems at any point in the SAN. Possibly a device cannot be attached at all because the paths or WWPNs (World Wide Port Number) generated in BS2000 do not exist physically or because the connections generated between the channel and controller are not permitted in the switches.

FC networks in SE Manager

The FC fabrics and connection paths through the fabrics are also displayed in the SE Manager under "Hardware" > "FC networks". Detailed information on this is provided in the “Operation and Administration” manual [57].

These information provides support for:

- Detection of generation errors (SUs /390)
- Detection of connection errors between Server and Storage
- Location of error states in the SAN

6 BS2000 user management

BS2000 user management of a pubset is implemented using what is known as a user catalog.

This chapter describes

- Structure of a user catalog
- Maintenance of the user catalog
- Security concept for user catalogs
- User catalog and the SMS concept
- Bulletin file (logon information file)

Detailed information on pubsets is provided in chapter "Pubset management".

Command	**Meaning**
ADD-USER	Create an entry in the user catalog
LOCK-USER	Lock access to system
MODIFY-USER-ATTRIBUTES	Modify entry in user catalog
MODIFY-USER-PUBSET-ATTRIBUTES	Modify pubset-specific user attributes of user ID
REMOVE-USER	Delete user entry from user catalog
SHOW-USER-ATTRIBUTES	Request information from user catalog
UNLOCK-USER	Cancel access lock to system
Macro	**Meaning**
RDUID	Query user ID
SRMUINF	Transfer data from user catalog to area

Table 13: Overview of interfaces for user catalog management

6.1 Structure of a user catalog

Every pubset contains the SYSSRPM file under the TSOS ID. This forms the user catalog, in which information on the user IDs of BS2000 is stored.

The SYSSRPM.BACKUP file (also called .BACKUP) also exists under the TSOS ID. The .BACKUP file is a copy of the user catalog. It acts as a scratch file if the user catalog is backed up or has to be reconstructed from a backup.

Both files are opened by a system task when a pubset is imported, and are closed again by the same task when the pubset is exported. Direct access to the files is therefore not possible.

The $TSOS.SYSSRPM file

The system file $TSOS.SYSSRPM is an NK-ISAM file.

The $TSOS.SYSSRPM file is accessed from a server task (the SRPM task TSN RP01 or RP02). With the import, the file is opened by the server task and remains open until export. $TSOS.SYSSRPM cannot be accessed directly.

The file $TSOS.SYSSRPM includes:

- information on access control for user IDs (also for extended access control with SECOS)
- system-global privileges
- BS2000 user groups

The SYSSRPM file forms the user catalog.

User catalog of the home pubset

The user catalog of the home pubset contains **system-specific** data such as BS2000 access rights (user ID, account number, logon password, SPOOLOUT class, mailing address, etc.) and the standard catalog ID.

User catalogs of the imported pubsets

The user catalogs of the imported pubsets contain **pubset-specific** data. Among other details, this includes information about:

- the maximum storage space available to the individual users on each pubset
- if applicable, the permission to exceed this limit
- catalog allocation

The system-specific information of the home pubset may also be contained in the user catalogs of the imported pubsets; However, for checking purposes the system refers only to the user catalog of the home pubset. The system-specific data on the imported pubsets is required only if these pubsets are used as reserve home pubsets for BS2000 system initialization and for the subsequent session.

6.2 Maintenance of the user catalog

When a pubset is imported for the first time (after generation with SIR), by the command IMPORT-PUBSET ACTUAL-JOIN=*FIRST, the user catalog is set up. The user catalog contains the following user IDs by default:

TSOS	User ID of system administration
SYSPRIV	User ID for the assignment of privileges in conjunction with the software product SECOS
SYSDUMP	User ID under which system dumps are stored
SYSOPR	User ID for the operator
SYSSPOOL	User ID for SPOOL management
SERVICE	User ID for service staff; under this ID, special programs for operational security are used by hardware and software maintenance personnel
SYSGEN	User ID for hardware generation
SYSHSMS	User ID for the HSMS data archive
SYSSNAP	User ID under which SNAP dumps are stored
SYSUSER	User ID for User dumps that cannot or should not be stored under the ID of the person responsible for the dump
SYSAUDIT	User ID for REPLOG management, and for SAT analysis and SAT file management in conjunction with SECOS
SYSNAC	User ID for the Network Administration Center
SYSROOT	User ID for POSIX management
SYSSNS	User ID for the SPOOL Notification Service
SYSMAREN	User ID for MAREN management
SYSSOPT	User ID for the SPACEOPT product
SYSWSA	User ID for the Web Service API

CAUTION!
If a pubset which has already been imported once is imported again with ACTUAL-JOIN=*FIRST not only is the user catalog reset to its default settings but also all the data it contains - with the exception of files under $TSOS - is deleted.

Except for TSOS and SERVICE, the system’s user IDs are set to “locked” on initialization, and can be unlocked using the UNLOCK-USER command.

Entering and managing users

For every user who is to have access to the system, systems support must create an entry both in the user catalog of the home pubset and in that of the assigned default pubset. After LOGON validation, the entry in the user catalog of the home pubset has priority. After due consultation, the user is allocated specific resources and privileges in addition to the data for user identification:

Identification: user ID, password, SPOOLOUT class, mailing address

Resources: default pubset, user address space, storage space on public disks, CPU time, message language, assignment of group syntax file

Privileges: right to exceed the allocated storage space, right to use specific task attributes, runtime priority, test privileges, use of hardware and linkage AUDIT, use of Net-Storage

The ADD-USER and MODIFY-USER-ATTRIBUTES commands are used to create or update entries in the user catalog.

Systems support can use the SHOW-USER-ATTRIBUTES command to request display of the user entries and thus obtain an overview of the contents of the user catalog.
Furthermore, with the aid of the LOCK-USER and REMOVE-USER commands, systems support can lock or delete users, and using UNLOCK-USER can unlock them again.

The system user IDs cannot be deleted (exception: the SERVICE user ID if the product SECOS is in use, see the “SECOS” manual “Access Control” [46]).

6.3 Security concept for user catalogs

- Saving the current user catalog of a pubset
- Restoring a saved user catalog
- Reconstruction of the SYSSRPM file
  - Performing reconstruction
  - Logging file

6.3.1 Saving the current user catalog of a pubset

To save the user catalog, the $TSOS.SYSSRPM.BACKUP file must be saved with the ARCHIVE subsystem (SAVE statement) or HSMS subsystem (BACKUP-FILES statement). For the user, the process is basically the same as saving any other file, with the file name functioning simply as a placeholder. During the SAVE run, the current data is transferred from user management to ARCHIVE/HSMS while the actual.BACKUP file continues to act as a placeholder and remains empty.
For the saved data to take effect, the .BACKUP file must be read in from the backup prior to IPL or IMCAT (with RESTORE/RESTORE-FILES) and the IPL or IMCAT must be parameterized accordingly.

If a differential save of the file is required (operation SAVE ...,CHANGED=YES), this will always be performed, because the file will have been reopened by the system task that manages it after it was last processed by ARCHIVE.

An export for the .BACKUP file is rejected.

> ¡ Since it is impossible to say for sure whether or not the contents of the saved user catalog have been manipulated during reconstruction, systems support must protect the file with suitable organizational measures.

6.3.2 Restoring a saved user catalog

To restore a user catalog, you must restore the $TSOS.SYSSRPM.BACKUP file with the help of the ARCHIVE subsystem (RESTORE operation).
If the file was renamed when saving $TSOS.SYSSRPM.BACKUP, this renaming must be reversed during restoration. Because the file is a permanent one, the REPLACE operand must be set to one of the values YES, ALL or ALLP in the RESTORE operation.

> i It is permitted to restore a saved user catalog to a pubset other than the one from which it was saved. You should note that in this case attributes such as the default pubset have to be updated manually.

6.3.3 Reconstruction of the SYSSRPM file

A reconstruction affects not only the user catalog but also the TSOSCAT file catalog because the file catalog keeps a separate list of user IDs which is used for mapping user IDs to PBNs (the primary block number identifies the file owner). The reconstruction must preserve the consistency of the user catalog and file catalog. The user structure of the existing user catalog SYSSRPM is not included in the reconstruction process.

> ¡ Ideally, a pubset should be reconstructed with its own backup. If this is not the case, there can be inconsistencies between the system environments at saving and reconstruction.

The aim is to unite the two user structures of .BACKUP and TSOSCAT, at the same time eliminating any inconsistencies.

6.3.3.1 Performing reconstruction

Reconstruction of the user catalog can be initiated at startup (in the case of the home pubset) or with the IMPORT-PUBSET command. Provided that a user catalog was successfully restored in the .BACKUP file before the last shutdown or export.

If a new pubset is created, the SYSSRPM file can also be reconstructed using SIR.

System parameter RECONUC

The reconstruction is controlled by the system parameter RECONUC. The system parameter RECONUC can be set and modified via the startup parameter service. A DIALOG startup allows an additional option of modifying RECONUC: the default value for RECONUC is output via message `NSI6010` and you are asked if you wish to modify it.

Note that the system parameters RECONUC and STUPTYPE are connected. If STUPTYPE=J or T is selected, first the first startup is executed (with or without resetting of the user catalog), then the value for RECONUC is evaluated. Message `NSI6220` informs you if the value for RECONUC or STUPTYPE is invalid and that it has been set to a default value, which is specified in the message.

The values N, B, T, A and R can be specified for RECONUC. N means no reconstruction.

Example

A: Set of all users existing in the .BACKUP file but not in TSOSCAT

B: Set of all users existing in both the .BACKUP file and TSOSCAT

C: Set of all users existing in TSOSCAT but not in the .BACKUP file

Ideally, sets A and C should be empty, otherwise either user attributes or files could go missing during the reconstruction.

The list below shows which value must be specified for which reconstruction request and how this may possibly affect the sample sets A, B and C. The relevant value is contained in parentheses in the IMPORT-PUBSET command.

- Reconstruction by means of BACKUP: RECONUC=B (SCOPE BACKUP)

  Based on sets A, B and C, this means:

  - A: Recreated with the saved attributes.
  - B: Existing user attributes are updated to the saved ones.
  - C: Their files and job variables are deleted.

  This mode is recommended in the case of regular saving.

- Reconstruction via TSOSCAT: RECONUC=T (SCOPE TSOSCAT)

  Based on sets A, B and C, this means:

  - A: no transfer to the reconstructed user structure. This can disrupt the distribution of privileges or the group structure on the pubset concerned such that, for example, a user who in the .BACKUP file exclusively possessed a particular privilege is not transferred to the new user catalog or no group administrator is transferred.
  - B: Existing user attributes are updated to the saved ones.
  - C: Set up with default attributes and the preservation of the files, job variables and guards.

  This mode is recommended when the preservation of files is paramount and user IDs, if these already exist at the time of the save, are to be reconstructed.

- Restoration via BACKUP and TSOSCAT: RECONUC=A (SCOPE ALL)

  As with SCOPE BACKUP, the user structure at the time of the save is restored.

  As with SCOPE TSOSCAT, files of user IDs which were set up after the save are retained through recreation of the user IDs with default attributes. In the case of two large, disjunctive user structures, this may exceed the capacity of TSOSCAT - a maximum of 8189 user IDs. If so, the import is aborted as soon as this is detected, and can be repeated in one of the modes BACKUP or TSOSCAT.

- Resetting the user catalog to the status of TSOSCAT: RECONUC=R (RESET)

  This function allows systems support to restore the format of the user catalog while preserving the files. The contents (i.e. the user attributes) must be restored in a second stage with the help of the reconstruction function.

  The only alternative in the event of an error is the first startup, in which all IDs, except for those of system administration, and all files which do not belong to the TSOS ID, are lost.
  If the SYSSRPM is destroyed by a system error, you should opt for complete pubset reconstruction, as this system error could also have destroyed or damaged other files.

The operator can follow the reconstruction by means of two messages, where the first documents the reconstruction basis at the beginning via the catalog ID and the time of the save (`SRM2017` for reconstruction with the .BACKUP file (*BY-BACKUP) or `SRM2018` for reconstruction without (*RESET)), and the second specifies at the end the number of reconstructed user IDs (`SRM2019` for reconstruction with the .BACKUP file (*BY-BACKUP) or `SRM2020` for restoration without (*RESET)).

A logging file is also made available to systems support, see "Logging file".

Reconstruction of a defective user catalog

If the purpose of the reconstruction is to restore a defective user catalog but this catalog cannot be imported due to a defect, an initial rudimentary correction must be made via the RESET function, in which a new user catalog is created on the basis of the user IDs contained in TSOSCAT. The individual user IDs are given default attributes and are all locked except for TSOS and SERVICE. With this method, unlike with a first startup, all files can be retained and the complete rebuilding of the pubset can be avoided.
Unless a zip import is requested, the defective user catalog is not deleted but is stored under the name : catid:$TSOS.SYS.SRPM.RECON.DIAG.<date.time> for later diagnosis of the problem that led to the reset.

> **CAUTION!**
> Please bear in mind that when a defective user catalog is passed on to a third party, an unauthorized reconstruction of the user data cannot be ruled out.

As well as the specification of the reconstruction type (see above), the RESET function can also be requested as a response to message SRM2012. This message is only displayed in the event of errors in user administration; errors in other components, such as group administration, lead to abortion of IMPORT-PUBSET processing.

Repercussion on the attributes

- Default Pubset

  The DEFAULT PUBSET attribute of all user IDs is set to the catalog ID of the reconstructed pubset. The DEFAULT-PUBSET attribute of the remaining user IDs is retained. If the backup of an incompatible pubset configuration was reconstructed, the systems support has the responsibility of assigning the locally valid default pubsets.

- Logon passwords

  If the user catalog of a system is saved without password encryption (system parameter ENCRYPT) and is reconstructed on a system with password encryption, all logon passwords are encrypted. The corresponding measure in the reverse case is not possible!

  Through the reconstruction, all user IDs are given back their logon passwords at the time of the save. It is up to each individual user to remember this password. This applies especially for TSOS or any other ID with the USER-ADMINISTRATION privilege.

- SECOS attributes

  If the user catalog of a system which has installed SECOS is saved and is then reconstructed on a system without SECOS, the settings of the logon parameters and the privileges of all user IDs are set to their defaults settings. This prevents settings that were made with SECOS from hindering operation without SECOS, even though it is not possible to change these settings.

  If, in a reconstruction in a system on which the SECOS subsystem is not available, a .BACKUP file is used (i.e. RECONUC=B, T or A) which was saved on a system which had SECOS available, all privileges are set to their default values as in a first startup. If in this case the .BACKUP file contains a privilege allocation which is only known in a higher BS2000 version (this is possible after changing versions), the unknown privileges in the current version are also reset.

  If SECOS is in use on the systems of both the saved user catalog and the reconstructed user catalog, password expiry dates may be reached in the time that elapses between save and reconstruction. The impending locking of the user ID is prevented by reconstruction of the runtime remaining at the time of the save.

- SM pubset attributes

  If the user catalog of an SF or SM pubset is saved and is then reconstructed on an SM or SF pubset respectively, the SM-pubset-specific attributes are set to their default values (during conversion from SF to SM pubset) or deleted (during conversion from SM to SF pubset).

6.3.3.2 Logging file

The logging file enables systems support to check the result of the reconstruction. This file is created under the TSOS user ID and is called SYS.SRPM.RECON.LOG.<date.time>.

It contains the following information:

- Pubsets of the user catalog that has been saved (BACKUP) or is to be reconstructed (SYSSRPM):

PUBSET	Catalog ID
TYPE	SM or SF pubset
DATE	Date of the save or reconstruction
TIME	Time of the save or reconstruction
ENCRYPT	NO/YES: logon-password encryption
SECOS	NO/YES: product SECOS in use

- user ID

USERID IN SYSSRPM	
USERID	All processed user IDs
PRESENT	NO/YES: entry in the reconstructed user catalog
USERID IN BACKUP	
PRESENT	NO/YES: entry in the backup file
ACTION	ADDED/REMOVED: new addition or deletion
USERID IN TSOSCAT	
PRESENT	NO/YES: entry in TSOSCAT
ACTION	ADDED/REMOVED: new addition or deletion
DELETED	
FILES	Number of deleted files with SCOPE=BACKUP
JV	Number of deleted JVs with SCOPE=BACKUP

Depending on the command entered: IMPORT-PUBSET PUBSET=..., RECONSTRUCT-USERCAT=*BY-BACKUP(SCOPE=*ALL/*BACKUP/*TSOSCAT) the user IDs are handled differently during reconstruction.

Example

```
 USERID IN SYSSRPM | USERID IN BACKUP  | USERID IN TSOSCAT |      DELETED      |
 USERID  | PRESENT | PRESENT | ACTION  | PRESENT | ACTION  |  FILES  |   JV    |
---------+---------+---------+---------+---------+---------+---------+---------+
USERID1  |   YES   |   YES   |         |   YES   |         |         |         |
USERID2  |   YES   |   YES   |         |   NO    | ADDED   |         |         |
USERID3  |   YES   |   NO    | ADDED   |   YES   |         |         |         |
USERID4  |   NO    |   YES   | REMOVED |   NO    |         |         |         |
USERID5  |   NO    |   NO    |         |   YES   | REMOVED |       4 |       3 |
---------+---------+---------+---------+---------+---------+---------+---------+
```

- SCOPE=*ALL/*BACKUP/*TSOSCAT

  USERID1 was saved in the .BACKUP file and has an entry in the TSOSCAT file catalog when reconstruction takes place.
  USERID1 is reconstructed in the SYSSRPM user catalog with the saved user attributes and retains its existing files.

- SCOPE=*ALL/*BACKUP

  USERID2 was saved in the .BACKUP file but has no entry in the TSOSCAT file catalog when reconstruction takes place.
  USERID2 is reconstructed in the SYSSRPM user catalog with the saved user attributes and is assigned an empty entry in the file catalog.

- SCOPE=*ALL/*TSOSCAT

  USERID3 was not saved in the .BACKUP file but has an entry in the TSOSCAT file catalog when reconstruction takes place.
  USERID3 is configured in the SYSSRPM user catalog with the default attributes and is assigned its existing files.

- SCOPE=*TSOSCAT

  USERID4 was saved in the .BACKUP file but is assigned no entry in the TSOSCAT file catalog when reconstruction takes place.
  USERID4 is not entered in the SYSSRPM user catalog.

- SCOPE=*BACKUP

  USERID5 was not saved in the .BACKUP file but has an entry in the TSOSCAT file catalog when reconstruction takes place.
  USERID5 is not entered in the SYSSRPM user catalog, its entry in the file catalog is deleted together with its files and JVs.

- Summary:

  LOGON PASSWORDS ENCRYPTED: NO/YES:
  The logon passwords have been encrypted

  EXPIRATION DATES UPDATED: NO/YES:
  The logon password expiry dates have been adjusted

  SECOS ATTRIBUTES RESET: NO/YES:
  The SECOS attributes have been set to their defaults

  USERIDS RECONSTRUCTED:
  Number of user IDs in the reconstructed user catalog

  USERIDS ADDED TO BACKUP:
  Number of user IDs added to the new user catalog compared with the status of the .BACKUP file; SCOPE=*ALL / *TSOSCAT

  USERIDS REMOVED FROM BACKUP:
  Number of user IDs deleted from the new user catalog compared with the status of the .BACKUP file; SCOPE=*TSOSCAT

  USERIDS ADDED TO TSOSCAT:
  Number of new user IDs added to TSOSCAT; SCOPE=*ALL / *BACKUP

  USERIDS REMOVED FROM TSOSCAT:
  Number of user IDs deleted from TSOSCAT; SCOPE=*BACKUP

Responses to errors

In the event of a system error which causes abortion of the reconstruction and therefore of import processing, the future user structure depends on the status of the reconstruction when it was aborted. This status can be determined from console messages:

- before message `SRM2017`
  The reconstruction has not yet begun; a new pubset import with or without reconstruction can be started.
- between messages `SRM2017` and `SRM2019`
  The reconstruction is in full swing; only a new pubset import without reconstruction can be started. The backup file may then have to be read in again. The user structure has the old status.
- after message `SRM2019`
  The reconstruction is so far gone that it can no longer be undone. A new pubset import without reconstruction can be started. The user structure has the new status.

6.4 User catalog and the SMS concept

The user catalogs of SF and SM pubsets are basically the same.

When SF pubsets are converted to SM pubsets, the TEMP-SPACE-LIMIT and PUBLIC-SPACE-LIMIT quotas which are relevant for SF pubsets are interpreted as follows on SM pubsets:

SF pubset attribute		**SM pubset attribute**
TEMP-SPACE-LIMIT	becomes	TEMP-SPACE-LIMITS/TOTAL-SPACE
PUBLIC-SPACE-LIMIT	becomes	PERM-SPACE-LIMITS/S0-LEVEL-SPACE

The additional quotas for SM pubsets can only be set via the MODIFY-USER-PUBSET-ATTRIBUTES command. The new quotas can be queried via the parameter INFORMATION=PUBSET-ATTRIBUTES of the SHOW-USER-ATTRIBUTES command.

6.5 Bulletin file (logon information file)

Systems support can set up a file which is stored under the name $TSOS.BULLETIN and is automatically output to all users on logon. $TSOS.BULLETIN must be shareable and should be protected with a write password or permit read access only:

Structure and content of the bulletin file

It is advisable to structure the bulletin file as follows:

1. General information for all users
2. Special information for certain user groups or individual users
3. Additional information that can be queried by all users

Alongside messages which are addressed to all users, special information can be sent to certain target groups or individual users. This is made available to dialog users by means of the message
`CONTINUE userid (Y,N)?`

If the user responds with `N`, the output of the bulletin file is aborted.

The bulletin file should be structured in such a way that, after the first current message, the dialog user can choose whether or not to output subsequent messages. For this reason the query should be inserted before every subsequent message.
`CONTINUE (Y,N)?` or `CONTINUE userid (Y,N)?`

> *Note on batch operation*
> All user-specific messages are output to SYSLST.
> Only the first message of the bulletin file is output to SYSOUT. The most recent messages should therefore always be located at the start of the bulletin file.

Example

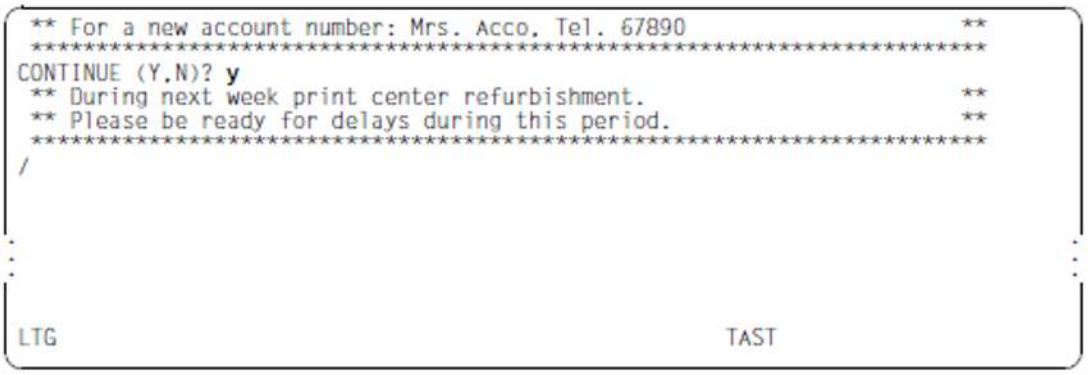
In the bulletin general information for all users should contain additional information which can be queried for the users ALLKIND and general information for all users which can be queried:

```
/START-EDT
  ************************************************************************
  ** Welcome to the server X/YZ !                                       **
  ** Normal OPEN SESSION operation from 07:00 to 21:00.                 **
  **                                                                    **
  ** Service numbers:  Tape announcement Tel. 12345                     **
  **                   List output       Tel. 34512                     **
  **                   Control center    Tel. 51234                     **
  ************************************************************************
  ** IMPORTANT!                                                         **
  ** On Friday the system will be shut down at 17:00                    **
  ** for maintenance work.                                              **
  ************************************************************************
**STOP**$ALLKIND
  ** IMPORTANT!  URGENT !                                               **
  ** Release the volumes MINE1 and MINE4 which you have reserved.       **
  ** ** For questions, telephone the Control Center on 89067           **
  ************************************************************************
**STOP**$ALLKIND
  ** IMPORTANT!                                                         **
  ** Your private disk PRIV03 is not currently available.               **
  ** Please contact us (Tel. 51234).                                    **
  ************************************************************************
**STOP**$ALLKIND
  ** Your account number M0815 will become invalid.                     **
  ** For a new account number: Mrs. Acco, Tel. 67890                    **
  ************************************************************************
**STOP**
  ** During next week  print center refurbishment.                      **
  ** Please be ready for delays during this period.                     **
  ************************************************************************
W'BULLETIN'
HALT
```

Output 1st screen

```
% JMS0160 INSTALLATION ' SU700-60', BS2000 VERSION 'V190', HOST 'DOYZZEOX': PLE
ASE ENTER '/SET-LOGON-PARAMETERS' OR '?'
/.test logon someone,m0815,'any#'
% JMS0066 JOB 'TEST' ACCEPTED ON <date> AT <time>, TSN = 12CF
 ************************************************************************
 ** Welcome to the server X/YZ !                                       **
 ** Normal OPEN SESSION operation from 07:00 to 21:00.                 **
 **                                                                    **
 ** Service numbers:  Tape announcement Tel. 12345                     **
 **                   List output       Tel. 34512                     **
 **                   Control center    Tel. 51234                     **
 ************************************************************************
 ** IMPORTANT!                                                         **
 ** On Friday the system will be shut down at 17:00                    **
 ** for maintenance work.                                              **
 ************************************************************************
CONTINUE SOMEONE (Y,N)? y
 ** Important !  URGENT !                                              **
 ** Release the volumes MINE1 and MINE4 which you have reserved.       **
 ** For questions, telephone the Control Center on 89067              **
 ************************************************************************
CONTINUE SOMEONE (Y,N)? y
 ** Your account number M0815 will become invalid on 03/01/            **
%PLEASE ACKNOWLEDGE
LTG                                                  TAST
```

Output follow-up screen



7 POSIX user administration

This chapter describes the interfaces for managing the POSIX user attributes of a BS2000 user ID. These interfaces are part of the SRPM component which is implemented in the SECOS software product and in BS2000. It is possible to work with POSIX, however, without installing the SECOS software product.

Each BS2000 user is, at the same time, a POSIX user. Apart from having a BS2000 user ID with valid individual POSIX user attributes, there are no other conditions to be met in order to gain access to POSIX and its interfaces.

For further information on SRPM, see section "Description of the privileges" and the “SECOS” manual “Access Control” [46].

The POSIX functionality in BS2000 is described in detail in the “POSIX Commands” [38] and “POSIX Basics” [39] manuals.

What is POSIX?

POSIX (Portable Open System Interface for UNIX) is a range of UNIX-based standards. These standards ensure the compatibility and interoperability of applications in a heterogeneous network. A heterogeneous network consists of servers and products from different manufacturers and of system and user software from different software suppliers.

The POSIX standard was defined as the national American standard by the Institute of Electrical and Electronics Engineers (IEEE) in 1989. It was then adopted by the X/OPEN consortium and in 1990 became the international standard (X/OPEN Portability Guide IV, XPG4).

The library functions of the POSIX standard are available to the user via a C library and a defined set of commands is available via a shell (POSIX shell).

Application programs can be easily ported with POSIX, irrespective of the operating system being used. Programs compliant with XPG4 can therefore also run in BS2000 following recompilation.
POSIX program interfaces are offered together with BS2000 interfaces. It is possible to use a combination of both BS2000 and POSIX program interfaces in the same program.

Some BS2000 software components and software products have been expanded to include functions for processing POSIX files. SPOOL, for example, can also be used to print out POSIX files and HSMS is able to save and reconstruct POSIX files, directories and file systems.

Command	Meaning
ADD-POSIX-USER	Defines the POSIX user attributes
ADD-USER	Creates a user entry in the user catalog [1]
MODIFY-LOGON-PROTECTION	Modifies protection attributes [2]
MODIFY-POSIX-USER-ATTRIBUTES	Modifies POSIX user attributes of a BS2000user ID
MODIFY-POSIX-USER-DEFAULTS	Modifies the default POSIX attributes of a pubset
MODIFY-USER-ATTRIBUTES	Modifies the catalog entry of a user [1]
SET-LOGON-PROTECTION	Defines protection attributes [2]

SHOW-LOGON-PROTECTION	Displays protection attributes [2]
SHOW-POSIX-STATUS	Displays POSIX status
SHOW-POSIX-USER-ATTRIBUTES	Displays the POSIX user attributes of a BS2000 user ID
SHOW-POSIX-USER-DEFAULTS	Displays the default POSIX attributes of a pubset
SHOW-USER-ATTRIBUTES	Outputs information on the entries in the user catalog [1]
START-POSIX-SHELL	Makes the POSIX shell available
Macro	**Meaning**
SRMUINF	Reads data from the user catalog and transfers it to a previously defined area

Table 14: BS2000 interfaces for POSIX user administration

1 Commands for administering accounting numbers for access via a remote computer.

2 Commands for administering access authorization via a remote computer:
If the SECOS software product is used, it is possible for existing BS2000 user IDs to define whether the user of a remote computer may gain access to the system with the UNIX command rlogin. The operand POSIX-RLOGIN-ACCESS=*YES(PASSWORD-CHECK=*YES/*NO) or POSIX-RLOGIN-ACCESS=*NO in the SET-/MODIFY-LOGON-PROTECTION command is available for this purpose.

The commands are described in the “SECOS” manual “Access Control” [46] and “Commands” [27] manuals.

POSIX user attributes

See the chapter “Administering POSIX users” in the “POSIX Basics” [39] manual.

Privileges for administering the POSIX user attributes

The POSIX-ADMINISTRATION privilege exists for POSIX. Owners of this privilege are referred to as POSIX administrators. They have the following tasks and rights:

- administration of the POSIX user attributes of all BS2000 user IDs on all pubsets
- administration of default values for the POSIX user attributes on all pubsets
- calling privileged POSIX functions

The POSIX-ADMINISTRATION privilege is automatically linked to the SYSROOT system user ID. This privilege cannot be withdrawn by SYSROOT. The security administrator (SECURITY-ADMINISTRATION privilege) can also grant the POSIX-ADMINISTRATION privilege to other BS2000 user IDs, and likewise withdraw it.

SYSROOT is the POSIX counterpart to the system administrator ID *root* in UNIX systems. SYSROOT is set up following first startup of the BS2000 system and automatically receives the user number 0. No other user number can be assigned to SYSROOT.

Holders of the USER-ADMINISTRATION privilege also receive authorization to administer the POSIX user attributes and the default values for these. In this instance, they are treated as if they were POSIX administrators.

The authorization of the group administrator of the *UNIVERSAL group is extended to include the POSIX user attributes. When administering the POSIX user attributes on the pubset managed by the user, the user is treated as if he/she has the privilege USER-ADMINISTRATION. In this case, the restrictions for group administrators within the user's hierarchy described below do not apply to the user.

Group administrators may also administer POSIX user attributes. However, the following restrictions apply:

- They cannot administer the default values for the POSIX user attributes.
- The type of POSIX user attributes which they can use depends on their authorization (ADM-AUTHORITY).
- The value range of the POSIX user attributes is restricted for group administrators.
- They can only administer the group and subgroup members for whom they are responsible.

For further information on the subject of privileges see section "Privileges".

Allocating a user number to a BS2000 user ID

See the chapter “Administering POSIX users” in the “POSIX Basics” [39] manual.

Administering BS2000 and POSIX groups

See the chapter “Administering POSIX users” in the “POSIX Basics” [39] manual.

Entering new POSIX users

See the chapter “Administering POSIX users” in the “POSIX Basics” [39] manual.

Mapping POSIX user attributes in the POSIX file system

The POSIX user attribute “user number” is closely linked with the POSIX file system: the user number documents the owner of a file. In contrast to BS2000, it is simple for the root administrator to assign a new owner to a file or directory (POSIX command *chown*).

Reading user information by program

See the chapter “Administering POSIX users” in the “POSIX Basics” [39] manual.

8 File management

Files are managed in file catalogs. The file catalogs themselves are managed by means of the catalog management system CMS.

8.1 File catalog

The file catalog of an SF or SM pubset serves as a container for the catalog entries of the files on public disk storages and Net-Storage, the job variables that belong to the pubset, as well as for the catalog entries of private disk files and tape files. The file catalog is created implicitly when the pubset is set up. It can be extended by systems support.

All user/system files and job variables possess an entry in the TSOSCAT file catalog. The files and job variables are identified by a unique name.

The **name is structured** as follows:

:catid:$userid.name

catid	Each pubset is addressed via the catalog ID (catid). It has a maximum length of four characters. (see also section "Pubset properties").
userid	The user identification (user ID) may be up to 8 characters in length.
name	The name of the file or job variable may be up to 41 characters in length. If a catalog ID comprising more than one character and a maximum length of the user ID are used, this number is reduced accordingly.

catid, *userid* and *name* must not be longer than 54 digits in total.

Command	**Meaning**
CHECK-IMPORT-DISK-FILE	Import files / check file generations
CREATE-FILE	Define name and attributes of a new file
CREATE-FILE-GENERATION	Create file generation of a file generation group
CREATE-FILE-GROUP	Create file generation group
DELETE-FILE	Delete file
DELETE-FILE-GENERATION	Delete file generation of a file generation group
DELETE-FILE-GROUP	Delete file generation group with associated file generations
EXPORT-FILE	Delete catalog entry for files on private volumes or Net-Storage
EXPORT-NODE-FILE	Delete catalog entry for node files (export node files)
IMPORT-FILE	Create catalog entry for files on private volumes or Net-Storage
IMPORT-NODE-FILE	Create catalog entry for node files (import node files)
MODIFY-FILE-ATTRIBUTES	Modify attributes of a file
MODIFY-FILE-GENERATION-SUPPORT	Modify attributes of a file generation

MODIFY-FILE-GROUP-ATTRIBUTES	Modify attributes of a file generation group
REMOVE-CE-LOCK	Unlock catalog entry
REPAIR-FILE-LOCKS	Remove illegal file lock
SHOW-CE-LOCK	Display catalog entry lock
SHOW-FILE-ATTRIBUTES	Display attributes of a file
SHOW-FILE-LOCKS	Display file locks
SHOW-PUBSET-CATALOG-ALLOCATION	Display information on catalogs of a pubset

Table 15: Command overview for file catalog management and catalog entries

8.1.1 Structure of a file catalog

The file catalog contains an entry for every file on the pubset, including itself; This entry includes information on the file attributes, the protection criteria and the location of the file on the volume.

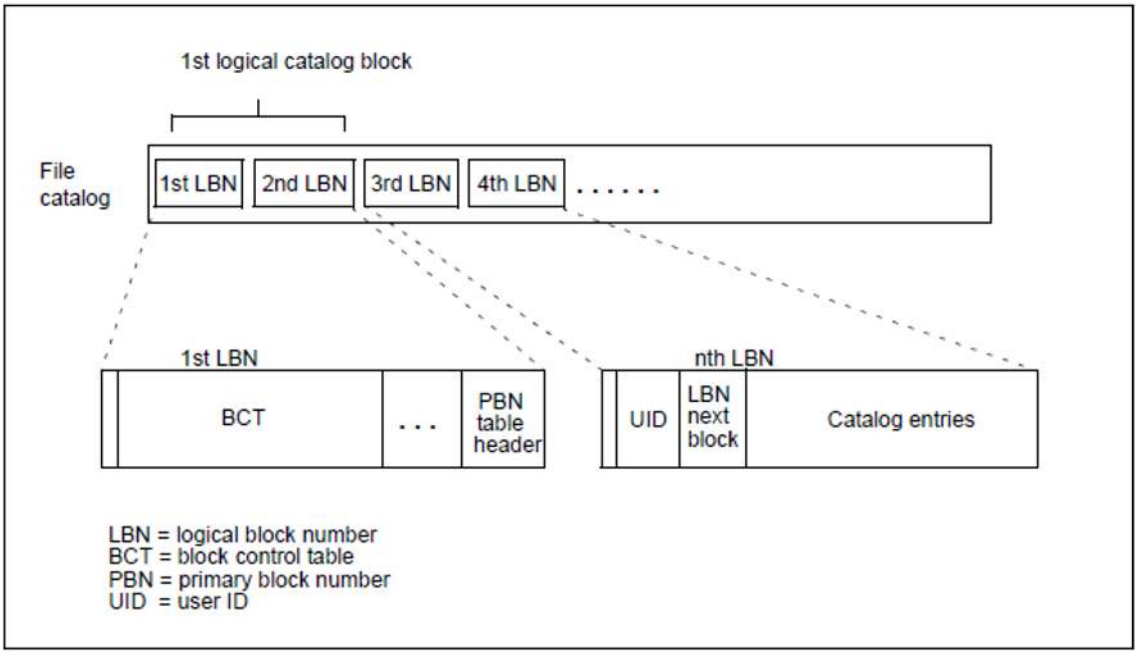


Figure 7: Structure of the file catalog

The first catalog block (4K block) contains data for management of the file catalog. The catalog entries (CEs) are stored in the subsequent catalog blocks. The catalog is structured according to user IDs. Any given block of a catalog contains only CEs of one user ID; the CEs are stored sequentially and unsorted. If there is not enough space in the block to accommodate additional CEs, another block is requested and chained to the last block of this user ID.
All users in the catalog are managed together with the beginning of their user chain PBN, (primary block number) in the PBN table of the catalog.
When a new user ID is added to the user catalog of an SF pubset, a new catalog block, the PBN of this user ID, is reserved automatically. The logical block number (LBN) of this catalog block is entered in the PBN table together with the user ID.
In an SM pubset initially only one entry in the PBN table of the control volume set catalog is generated for the new user. PBN assignment in an SM pubset catalog takes place only when the first CE of the user is stored in this catalog.

The second 4K block of the file catalog always contains the CE of the file catalog itself and is the beginning of the user chain of the TSOS ID.

The logical block size in TSOSCAT is 4-Kbytes; a logical block number (LBN) is maintained as the number of the logically smallest PAM page stored in the 4-Kbyte block.

A catalog block can accommodate from 1 to 13 catalog entries. The maximum length of an entry is 4079 bytes, the minimum length (including a 1-byte file name) is 301 bytes, as long as no space has been assigned. Otherwise this value increases by the length of the extent list. This length is at least 14 bytes for volumes < 32 Gbyte and 16 bytes for volumes >= 32 Gbyte.

Catalog formats

Whenever possible, BS2000 OSD/BC from V11.0 onwards only supports the EXTRA LARGE catalog format (max. Size 64,016 PAM pages). The NORMAL and LARGE catalog formats are now obsolete.

Catalogs of a preceding version in the NORMAL or LARGE format are automatically converted into the EXTRA LARGE format as soon as the pubset is imported exclusively or as master pubset. At the same time, the special catalogs are renamed as well:

- `:<catid>:$TSOS.TSOSCAT.#MIG` to `:<catid>:$TSOS.TSOSCAT.#M00`
- `:<catid>:$TSOS.TSOSCAT.#PVT` to `:<catid>:$TSOS.TSOSCAT.#J00`
- `:<catid>:$TSOS.TSOSCAT.#JVC` to `:<catid>:$TSOS.TSOSCAT.#P00`

These converted catalogs can be used in a preceding version as well.

Only when a shared pubset is imported in slave mode, the catalog format specified by the pubset master is retained. Even in this case, the NORMAL and LARGE catalog formats are only possible if the master runs in an older version.

File catalog on SF pubsets

The file catalog is stored in the file called `:<catid>:$TSOS.TSOSCAT`. It contains all user and system files as well as the job variables.

File catalog on SM pubsets

Unlike with SF pubsets, the catalog of an SM pubset is made up of several catalog files. An example is used to illustrate its structure in the figure below. In the example the SM pubset consists of the three volume sets A, BLA and C7, the catalog ID is X, and the catalog format is EXTRA LARGE.

Subcatalogs with catalog format EXTRA-LARGE

With the catalog format EXTRA-LARGE, up to 100 subcatalogs are created with each necessary catalog enhancement, with the names:

`:<catid>:$TSOS.TSOSCAT.#Mnn` (nn = 00 through 99)
`:<catid>:$TSOS.TSOSCAT.#Pnn` (nn = 00 through 99)
`:<catid>:$TSOS.TSOSCAT.#Jnn` (nn = 00 through 99)

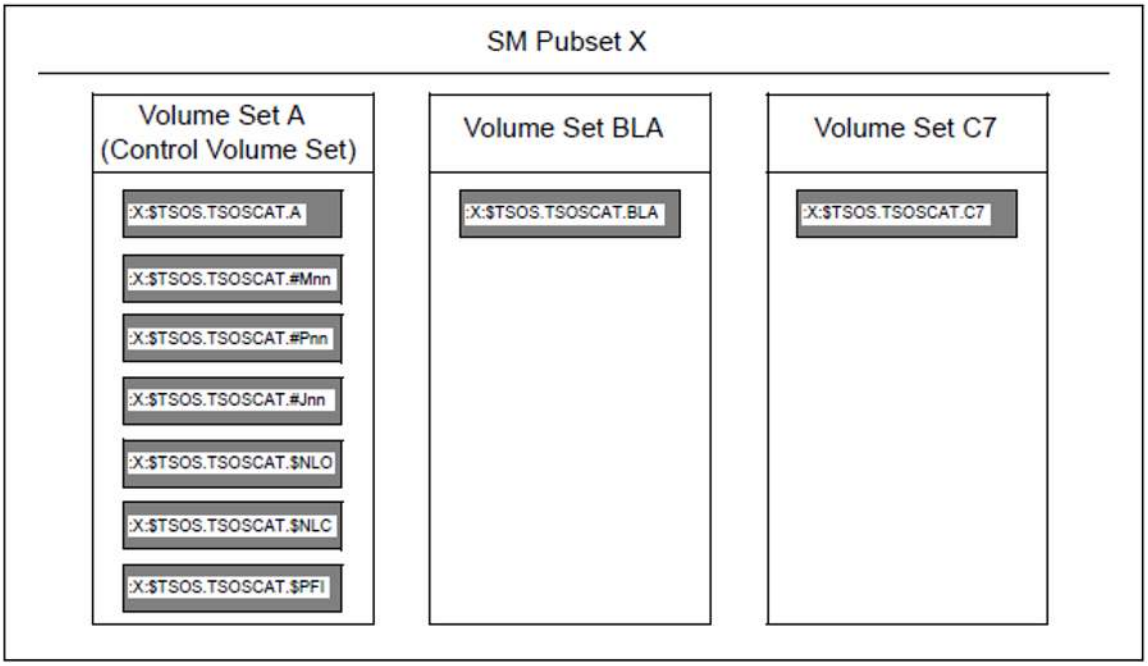


Figure 8: Structure of the file catalog of an SM pubset

Explanation

```
:X:$TSOS.TSOSCAT.A
:X:$TSOS.TSOSCAT.BLA
:X:$TSOS.TSOSCAT.C7
```

Each volume set of an SM pubset contains a volume-set-specific catalog file with the catalog entries for the files stored on it. The path name of the volume-set-specific catalog file is composed of the pubset ID and the volume set ID, as follows: :<catid>:$TSOS.TSOSCAT.<volume-set-id>.

The catalog files below are all located on the control volume set of the SM pubset:

`:X:$TSOS.TSOSCAT.#Mnn`

Catalog entries for files migrated to a background level as well as cataloged files which do not occupy any space.

`:X:$TSOS.TSOSCAT.#Pnn`

Catalog entries of files on Net-Storage, tape files and private disk files.

`:X:$TSOS.TSOSCAT.#Jnn`

Container for job variables.

`:X:$TSOS.TSOSCAT.$PFI`

Catalog index file, used for saving the catalog index when an SM pubset is deactivated. This file is an administrative entity superordinate to SM pubset catalogs and contains information on the location of a file or a job variable in the SM pubset.

`:X:$TSOS.TSOSCAT.$NLO`

Name list, used for data security. If a volume set fails, this file can be used to reconstruct the data lost as a result. The name list contains one entry for each file existing in a volume set of the SM. There are also entries for temporary files. Only job variables and files which are stored in catalogs on the control volume set are not referenced in this list, because if the control pubset fails the name list can no longer be accessed. The entry for a CE in the name list comprises the ID of the catalog which contains the CE, the user ID of the file owner, and the name of the file.

`:X:$TSOS.TSOSCAT.$NLC`

Name list copy, required to reconstruct the name list after a system crash.
The name list and the copy of the name list should be created with an equal size because each time the catalog index is restored the name list becomes the copy of the name list and vice versa. When managing the name list, CMS might then only take the smaller of the two file sizes into account.

File catalog on Net-Storage volumes

Each Net-Storage volume has its own file catalog with the name `.BS2FSCAT`.

For each file which is stored on this Net-Storage volume this file catalog contains a copy of the catalog entry in the TSOSCAT of the pubset to which the Net-Storage volume is assigned.

8.1.2 Extending catalog files

The SHOW-PUBSET-CATALOG-ALLOCATION command displays the catalog type, the occupancy rate per catalog (utilization/file size) and the extendability per catalog. This output can be implemented either for a single pubset or for all pubsets imported locally on a system and in master mode.

Automatic extension of the catalog

CMS recognizes when a catalog is 90% full and automatically extends it provided this is possible without changing the catalog format.

For special catalogs in EXTRA LARGE type a new subcatalog is then created if none of the existing subcatalogs can be extended.

Manual extension of the catalog

The catalog files can be extended during operation, e.g. with the MODIFY-FILE-ATTRIBUTES ...,SPACE=... command. This applies on both SM and SF pubsets. The catalog extension becomes effective immediately instead of after the next import of the pubset.

If the product SCA was activated for an SF pubset, SCA is implicitly terminated by the system and restarted if the TSOSCAT file is extended (to adapt the SCA table structure). This generally has no effect on the system.

Up to 99 further subcatalogs can be created for SM pubsets with catalogs of the EXTRA LARGE type for each of the special catalogs #M00, #J00 and #P00. The ADD-CATALOG-FILE command is used to do this. 8

8.1.3 Enhancing performance for catalog accesses

- It is advisable to split up the system administration ID into TSOS and the default user ID because the number of system files which must be contained under TSOS amounts to several thousand. Without splitting, these catalog entries must also always be searched each time a user accesses $file.
  - System administration ID (TSOS)
    All files required for maintaining system operation are cataloged under this ID. The user catalog, file catalog, logging file and accounting file, for example, are concerned here, but also all files which need to be accessed before SYSTEM READY.
  - Default user ID
    This ID is defined with the system parameter DEFLUID. All utility routines, language processors, libraries, etc. that are frequently needed by the user should be cataloged under this ID.
    This parameter can also be used to specify a pubset. However, it is better to assign the standard pubset with the aid of the ADD-USER command.
    Should the pubset with the user ID specified for DEFLUID become defective, a standby pubset can then be used. In this case only the user catalog entry of the user ID would have to be modified.
- For the START-/LOAD-EXECUTABLE-PROGRAM, CALL-PROCEDURE, ENTER-PROCEDURE and ENTER-JOB commands the user should specify *$filename* instead of *filename*.
  For instance, if START-EXECUTABLE-PROGRAM FROM-FILE=$EDT is entered, the user ID specified with DEFLUID is referenced immediately.
  If START-EXECUTABLE-PROGRAM FROM-FILE=EDT is entered, a file with the name EDT is first searched for under the calling user ID, and then under the user ID specified with DEFLUID.
- The system administration should specify a suitable value for secondary allocation when the system is generated (system parameter DMSCALL), because the secondary allocation of storage space is very time-consuming as the corresponding catalog entry is changed with every user task request.
- To prevent saturation of the public storage space it is advisable to release storage space not used by user files. The system administration can create procedures for this purpose. See section “Monitoring storage space saturation”

8.1.4 Removing access locks

Removing the catalog entry lock (CE lock)

To guarantee the consistency of catalog entries (e.g. for system communication), system components can temporarily lock the catalog entry of a file or JV against modification. If it is not possible to remove a CE lock, for example because of an error during system communication, the catalog entry remains locked even though the lock is no longer required.

This type of “leftover” CE lock hinders the operation of tasks which require this file or JV for processing.

Systems support can use the REMOVE-CE-LOCK command to remove a lock for the catalog entry of a file or JV. The command is executed only when the lock holder’s task no longer exists or is in the PENDING INDEFINITELY status.
If the lock holder task possesses a different status, then the CE lock cannot be released and the command is rejected. In this case, it is necessary to check whether the lock holder task can be terminated (e.g. CANCEL-JOB command). The SHOW-CE-LOCK command can be used to ascertain the TID of this task and the SYSID of the system in which the task is running.

In the case of shared pubsets, the REMOVE- and SHOW-CE-LOCK commands can be entered at any system in the network.

Removing a file lock

The owner of the file or systems support (TSOS user ID) can use the REPAIR-FILE-LOCKS command to remove “illegal” file locks for a file. These are file locks which are no longer required but which could not be automatically reset by the system for one of the following reasons:

- temporary loss of a connection in a network
- system error preventing the file lock reset

The locks present for a file can be set by a job, by a system component or for data transfer.

The SHOW-FILE-LOCKS command informs the file owner and all users with access authorization about the locks which currently apply to a file. File locks which are effective because of the catalog entry are not displayed. These locks can be displayed by means of the SHOW-CE-LOCK command.

8.2 ACS: Alias catalog system

With ACS ((Alias Catalog Service) it is possible to access files and JVs under names which are, within certain limits, freely selectable. Users thus have the option of defining the files/JVs they require and storing them in special catalogs, the alias catalogs, together with the assignment to the real file/JV. Files for which aliases are defined can be subsequently processed by specifying only the alias. Alias catalogs are managed locally for each task.

Detailed information on ACS is provided in the “Introductory Guide to DMS” [19].

Command	Meaning
ADD-ACS-SYSTEM-FILE	Declares a new identifier for an ACS system file, and assigns this identifier to a file name
ADD-ALIAS-CATALOG-ENTRY	Adds an entry to the alias catalog of the current task and defines where the substitution applies.
HOLD-ALIAS-SUBSTITUTION	Interrupts the ACS function for the current task.
LOAD-ALIAS-CATALOG	Transfers entries saved in an AC file to the alias catalog.
MODIFY-ACS-OPTIONS	Modifies options (for local task).
MODIFY-ACS-SYSTEM-FILE	Modifies the definition of an AC system file
MODIFY-ALIAS-CATALOG-ENTRY	Modifies an existing entry in the alias catalog.
PURGE-ALIAS-CATALOG	Deletes the alias catalog of the current task
REMOVE-ACS-SYSTEM-FILE	Deletes the definition of an AC system file
REMOVE-ALIAS-CATALOG-ENTRY	Deletes an existing entry from the alias catalog.
RESUME-ALIAS-SUBSTITUTION	Resumes the alias substitution function
SET-FILE-NAME-PREFIX	Defines a prefix for file/JV names and specifies where the prefix applies.
SHOW-ACS-OPTIONS	Displays options (for local task).
SHOW-ACS-SYSTEM-FILES	Shows the predefined alias catalog system files.
SHOW-ALIAS-CATALOG-ENTRY	Displays selected alias catalog entries (on SYSOUT).
SHOW-FILE-NAME-PREFIX	Shows the current prefix and its scope.
START-ACS	Activates ACS system-wide
STORE-ALIAS-CATALOG	Stores the alias catalog in a file

Table 16: Overview of ACS commands

8.3 ADM-PFA: Administrator performant file access

The file caching formerly known under the name ADM-PFA is described in the “DAB” [10] manual.

8.4 Sending BS2000 file by email

The MAIL-FILE command or the MAILFIL macro enables you to send BS2000 files (including SYSLST and SYSOUT) to the specified user ID. This email is sent to one or all recipient addresses which are entered in the user ID's user entry. The email address must be entered in the user ID, see "Entering email addresses for a user ID".

The mail service of the software product interNet Services must be installed because the email is actually sent with the mail sender (SEND-MAIL interface) of interNet Services.

The character set (file attribute CCS-NAME) is taken into account for the text file which is to be transferred, and when it is transferred, conversion takes place to a character set of the open systems environment. You can add a "subject" to the email and specify whether the file should be deleted after it has been transferred successfully.

You can also send the following files using this email concept:

- MAIL-FILE for system files SYSLST and SYSOUT, see "Output of SYSLST and SYSOUT using MAIL-FILE".
- HSMS reports with HSMS, see the "HSMS" manual [24].
- Output files of MAREN, see the "MAREN" manual [31].

When an email address has been selected successfully via the job name, this email address is used as the sender of the email, otherwise the first email address of the execution ID is used. The first email address of the TSOS ID is also provided in case of delivery errors. This ensures that the administrator receives a so-called "bounce mail" when recipient addresses are incorrect and can then correct the email addresses concerned in the user management.

Entering email addresses for a user ID

Email addresses are entered in the user entry of a BS2000 user ID with the EMAIL-ADDRESS operand of the ADD-USER and MODIFY-USER-ATTRIBUTES commands. The MAIL-ADDRESS attribute remains unchanged.

For the TSOS user ID an email address must be entered.

The email addresses which are entered are output using the SHOW-USER-ATTRIBUTES command or the SRMUINF macro.

After new email addresses have been entered, systems support should check these by sending a test mail to the associated user ID.

Output of SYSLST and SYSOUT using MAIL-FILE

MAIL-FILE can output the current system files SYSLST and SYSOUT. For the system output, the value MAIL can be specified instead of PRINT in the relevant output operands in the EXIT-JOB, CANCEL-JOB and ENTER-PROCEDURE commands. Furthermore, for these commands and for the LOGOFF command the default value for system output can be set to PRINT or MAIL in the system parameter SSMOUT.

If no email can be sent when the job terminates, system files which in accordance with the settings would have to be sent in an email are output instead to SPOOL as was previously the case.

Error analysis in the mail service of the software product interNet Services

The system administrator can specify settings for logging and tracing emails by means of the configuration file for the mail sender backend (file `SYSDAT.MAIL.<version>.SERVICE.OPT`) and by means of the MODIFY-MAIL-SERVICE-PARAMETER command. The logging and trace files contain detailed information on sending emails, in particular in the event of errors.

8.5 Unicode in BS2000

Unicode incorporates almost all the text characters known around the world in a single character set. Unicode is also independent of the various manufacturers, systems and countries.

Unicode support in BS2000 extends the EBCDIC character sets which are available in BS2000 systems by additional characters which are required in the European language area. The programming and runtime environment is made available to users, who require this to expand their existing applications by the addition of Unicode data fields. For this purpose a corresponding software configuration is provided under BS2000. The XHCS subsystem for Unicode support in BS2000 is loaded by default.

For general information on Unicode, please refer to the website of the Unicode Consortium: http://www.unicode.org.

For detailed information on Unicode in BS2000, please refer to the “Unicode in BS2000” manual [58].

9 Pubset management

Pubsets (public volume sets) are sets of shared volumes and are used in BS2000 together with private volumes for storing files. In addition to the files themselves, a pubset also contains all the metadata required for file management (file catalog, user catalog, etc.).

i BS2000 supports Net-Storage in the context of the pubsets. This is described in chapter "Net-Storage management".

Command	Meaning
ADD-MASTER-CATALOG-ENTRY	Create entry in the MRSCAT catalog list
ADD-NET-STORAGE-VOLUME	Create a Net-Storage volume and assign it to a local pubset
ADD-USER	Create entry in the user catalog and define pubset access
CANCEL-PUBSET-EXPORT	Cancel export of a pubset
CANCEL-PUBSET-IMPORT	Cancel import of a pubset
CHECK-PUBSET-MIRRORS	Check the homogeneity of pubset mirroring
EXPORT-PUBSET	Export a previously imported pubset
FORCE-PUBSET-EXPORT	Force a pubset export
IMPORT-PUBSET	Import a pubset and define user catalog handling
MODIFY-MASTER-CATALOG-ENTRY (EDIT-MASTER-CATALOG-ENTRY)	Modify entry in the MRSCAT catalog list
MODIFY-PUBSET-DEFINITION-FILE (EDIT-PUBSET-DEFINITION-FILE)	Modify the definition of an SM pubset
MODIFY-PUBSET-PROCESSING	Modify the configuration of a pubset
MODIFY-PUBSET-RESTRICTION	Modify the usage restrictions for a pubset
MODIFY-PUBSET-SPACE-DEFAULTS (EDIT-PUBSET-SPACE-DEFAULTS)	Modify the storage space management default values

MODIFY-SPACE-SATURATION-LEVELS (EDIT-SPACE-SATURATION-LEVELS)	Modify the storage space saturation levels
MODIFY-USER-ATTRIBUTES	Modify entry in user catalog
MODIFY-USER-PUBSET-ATTRIBUTES	Modify the SM pubset attributes of a user ID
REMOVE-MASTER-CATALOG-ENTRY	Delete entry in the MRSCAT catalog list
REMOVE-NET-STORAGE-VOLUME	Remove a Net-Storage volume from a local pubset
REMOVE-PUBSET-LOCK	Delete pubset lock
RESUME-PUBSET-RECONFIGURATION	Terminate pubset reconfiguration request correctly
SET-PUBSET-ATTRIBUTES	Define the characteristics of a pubset
SET-SPACE-SATURATION-LEVEL	Define the saturation levels for using storage space on an SF pubset (only supported for compatibility reasons)
SHOW-MASTER-CATALOG-ENTRY	Request information on the state, occupation and accessibility of a pubset
SHOW-PUBSET-ATTRIBUTES	Request information on pubset properties
SHOW-PUBSET-DEFINITION-FILE	Display information on the global and the performance attributes of a volume set
SHOW-PUBSET-LOCKS	Display information on pubset locks
SHOW-PUBSET-FILE-SERVICES	Display the file property combinations supported in an SM pubset
SHOW-PUBSET-IMPORT-EXPORT	Output information about the processing status of import/export jobs
SHOW-PUBSET-NET-STORAGE	Display the Net-Storage of a pubset
SHOW-PUBSET-OCCUPATION	Display information on the jobs which occupy pubset space
SHOW-PUBSET-PROCESSING	Display information on the physical pubset configuration
SHOW-PUBSET-RESTRICTION	Display information on the physical pubset configuration
SHOW-PUBSET-SPACE-ALLOCATION	Display the storage space assignment for a pubset

SHOW-PUBSET-SPACE-DEFAULTS	Display information on the pubset-specific default values for storage space allocation
SHOW-SHARED-PUBSET	Request a su058mmary of the participants in a shared pubset network
SHOW-SPACE-SATURATION-LEVELS	Display information on the pubset-specific storage saturation levels
SHOW-XCS-PUBSET	Request a summary of the participants in an XCS pubset network
SHOW-XCS-OCCUPATION	Display the current TU tasks using XCS

Table 19: Overview of pubset management commands

9.1 Pubset concept

A pubset is a group of shared disks which form a unit. Because of their self-contained structure, pubsets are objects which are almost completely independent of each other. If one pubset fails, it is the only one affected.

9.1.1 Pubset types

The pubsets are distinguished according to usage types:

- For information on the home pubset, which must exist in every BS2000 system, see "Pubset names (VSN)".
- For information on data pubsets (which are also called user pubsets), see "User pubset (data pubset)"
- For information on paging pubsets, see "Paging pubset"
- For information on pubsets for large files and volumes, see "Pubsets with large objects"
- For information on standby pubsets for increasing data security, see "Special information on standby pubsets"
- For information on shared pubsets for shared access, see "Shared pubsets"

9.1.1.1 Home pubset

On each BS2000 system, at least one pubset must exist which contains the system-specific data and files and must therefore be available throughout the complete BS2000 session. This special pubset is known as the home pubset. It cannot be connected to Net-Storage.

The system ID (SYSID) of the home pubset is used to identify the active system. It is entered in the disk occupancy information and in the path group ID. Controllers use the path group ID to differentiate between I/Os from different systems. For these reasons, it is essential that unambiguous system IDs are assigned when home pubsets are operated in parallel. When PUB notation is used this is always the case, provided that unique pubset IDs have been chosen. In the case of pubsets with point notation which are to be used as home pubsets, you must explicitly ensure that unique system IDs are used, see "Creating a shared pubset network".

BS2000 imports the home pubset at system initialization and it is exported at system shutdown. The home pubset is exported during system shutdown (when the SHUTDOWN command is processed). User files and job variables may be cataloged on the home pubset. This must, however, be permitted by systems support (ADD-USER or MODIFY-USER command).

9.1.1.2 User pubset (data pubset)

These pubsets are used exclusively by DMS for storing files. They can also be connected to Net-Storage. They only contain user files or job variables. Systems support can control the availability of this pubset at any time during the session with the IMPORT-PUBSET and EXPORT-PUBSET commands. A user may have access rights to several pubsets.

Each user is assigned a default pubset. He/She can create, process and delete files and job variables on this pubset without entering the catalog ID.
The default pubset for the TSOS user ID is always the home pubset, regardless of what is specified in the DEFAULT-PUBSET entry in the ADD- or MODIFY-USER-ATTRIBUTES command.

9.1.1.3 Paging pubset

The pubsets containing paging files to be used in the session are known as paging pubsets. They are required for the complete BS2000 session but may be activated and deactivated as required independently for DMS use but only, however, for the same system.

The paging pubsets are handled by DMS in the same way as pubsets not containing paging files: The import request for a data pubset containing a paging file must be explicit (IMPORT-PUBSET command). Paging pubsets are not imported automatically during system initialization. They should be imported by starting a RUN file to ensure their availability immediately after system initialization.
The paging pubsets are identified as such in the MRSCAT catalog list and cannot be deleted there.
All system paging pubsets can be displayed with the command SHOW-MASTER-CATALOG-ENTRY CATALOG-ID=*ALL,SELECT=*PAGING.

9.1.1.4 Pubsets with large objects

Volumes and files whose size exceeds the 32 GB limit are called “large volumes” and “large files”, or together they are called “large objects”. The maximum possible size is 4 TB (terabytes) There are two pubset types for supporting large objects:

- pubsets with large volumes and without large files
- pubsets with large volumes and large files

This makes it possible to introduce large volumes and large files step-by-step. The introduction of large volumes is transparent for the most part to existing programs. Some changes may need to be made when large files are introduced.

9.1.2 Pubset names (VSN)

All volumes are identified in BS2000 by a name, the VSN (Volume Serial Number).

For pubsets, a distinction must be made between single-character names (VSN in PUB notation) and multiple-character volume set/pubset names (VSN in point notation): This convention must not be used for private volumes.

The pubset names must be unique within a network.

there is a special notation for Net-Storage volumes, see "Notation of Net-Storage volumes" in section "Terms"

9.1.2.1 VSN in PUB notation

A VSN in PUB notation begins with the fixed string PUB (for public). It always consists of six characters, with the format **PUBpxx.**

PUB fixed name part to distinguish between private volumes (3 characters “PUB”) = type identifier

p Catalog ID (catid), (1 character; A..Z, 0..9)

xx sequential number within a pubset/volume set, (2 characters; 00..31) = sequential number

A maximum of 36 pubsets or volume sets, consisting of up to 32 volumes, can be addressed with the single-character catalog ID.

Examples: PUBA00, PUBA25, PUB502

9.1.2.2 VSN in point notation

A VSN in point notation always consists of six characters, with the format **pp[pp].[xy]z.** Where:

pp[pp] catalog ID (catid), (2-4 characters; A..Z, 0..9), the “PUB” prefix is not allowed

. period (1 character) = type identifier

[xy]z sequential number within a pubset/volume set, (1-3 characters) = sequential number

A pubset or volume set in point notation can consist of up to 255 volumes.

Examples: AA.001, AB.309, XYZ.23, OTTO.0, J19P.8

With SF pubsets, the catalog ID corresponds directly to the “catalog ID” part of the name in the VSN. With SM pubsets, the catalog ID differs from all volume set names of the SM pubset concerned and it therefore differs from the “catalog ID” part of the VSN name for all volumes of the pubset.

Maximum number of disks per pubset/volume set for point notation

The maximum number of volumes per pubset/volume set depends on the allocation unit AU (ALLOCATION-UNIT), the volume types and the length of the sequential number.

AU	Sequential number value range x	Sequential number value range y	Sequential number value range z	Maximum number of addressable volumes
6 KB	0	0	A..V, 0..9	32
8 KB	A..Z, 0..9	A..Z, 0..9	A..Z, 0..9	36 / 255 [1]
64 KB	A..Z, 0..9	A..Z, 0..9	A..Z, 0..9	36 / 255 [1]

[1] A maximum of 36 volumes may be addressed per pubset with a single character sequential number

9.1.2.3 Double-point notation for mirror disks

Double-point notation is used to designate mirror disks unambiguously after they have been detached from the original unit (e.g. for performing a backup with HSMS). Mirror disks are created using the replication functions of external disk storage systems.

For this purpose the point created in a VSN with point notation is changed into a double point or, in PUB notation, the character string “PUB” is changed to “P:B”.

Examples

			Double-point notation
Point notation:	ABC.04	becomes	ABC:04
	XY.123	becomes	XY:123
PUB notation:	PUB023	becomes	P:B023
	PUBX88	becomes	P:BX88

When a mirror pubset is created with SHC-OSD, this renaming can take place implicitly. However, only SF pubsets can be renamed implicitly.

9.1.2.4 Special cases

Converting or renaming the VSN format

The utility PVSREN (see the “Utility Routines” manual [15]) can be used to convert pubset notations or rename pubset or volume set names within a notation type.

It is always possible to convert a PUB notation to a point notation. It is only possible to convert a point notation to a PUB notation if the number of volumes in the pubset or volume set concerned does not exceed 32.

Snapset notation

In the case of snap units which were used for generating a Snapset, the Snapset identification in lower case is used to name the VSNs unambiguously in such a manner that on the one hand the associated pubset disk can be derived from the Snapset disk, and on the other hand the VSN of the Snapset disk is outside the name space of the VSNs for private disks and pubsets. (For details see "Snapset identification").

9.1.2.5 Pubset addressing

Objects cataloged on a pubset are addressed via the catalog ID (catid) and complete path name.

The catalog ID is identical to the name of the corresponding pubset. The catalog ID of an SM pubset must be different from all volume set names of all SM pubsets.

The addressing of files stored on a specific volume set on an SM pubset via the name of this volume set is not supported.

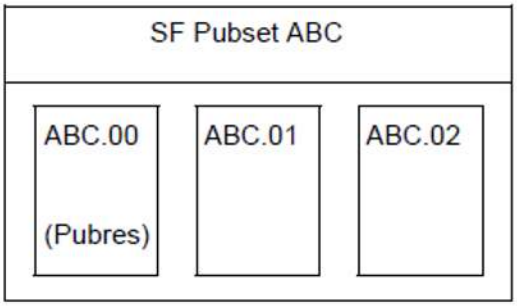
9.1.3 SF and SM pubsets

This section describes the structure of SF and SM pubsets and also the configuration data which is important for these pubsets.

9.1.3.1 Structure of an SF pubset

A single-feature pubset (SF pubset) consists of one or more homogeneous volumes which must have identical properties (disk format, allocation unit). The disk assigned number 0 is known as the **Pubres**. The Pubres is the obligatory system disk of an SF pubset. It contains, among other things, a directory of the disks belonging to the pubset and a reference to the start of the TSOSCAT file catalog.

In the example below the SF pubset with catalog ID ABC consists of three volumes with the VSNs (in point notation) ABC.00, ABC.01 and ABC.02.



Volume configuration of an SF pubset (example)

If a file named “MY.LIST” exists under user ID “ALLKIND” on one of the volumes in SF pubset ABC, the path name is: “:ABC:$ALLKIND.MY.LIST”

Properties of SF pubsets

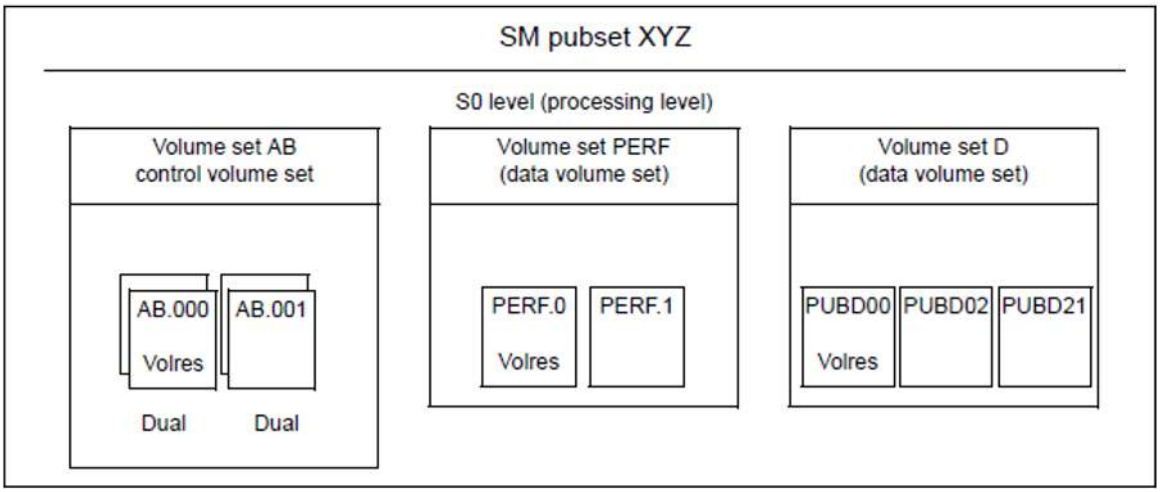
- An SF pubset represents the provision of a limited, homogeneous service. If the service requirements of a file change the user must move the file to another pubset.
- SF pubsets can be enlarged by adding empty volumes. The volume properties must match those of the pubset (see "Setting up and replacing SM pubsets").
- The size of an SF pubset can be can be reduced by removing empty volumes. The system provides the option of setting allocation restrictions as an aid for clearing volumes.
- If a single volume in an SF pubset fails, the complete pubset is taken as failed.
- SF pubsets are suitable for use as home or paging pubsets.

9.1.3.2 Structure of an SM pubset

An SM pubset consists of one or more **volume sets** which, as with an SF pubset, consist of a group of several homogeneous disks. A volume set can only be operated within the SM pubset. An SM pubset can consist of a maximum of 255 volume sets.

The control volume set has a special role among the volume sets. The **control volume set** is used for storing all pubset-global metadata of the SM pubset. The control volume set must therefore always be available and cannot be removed from the pubset.

The example below shows the structure of an SM pubset with the three volume sets AB, PERF and D:



Volume configuration of an SM pubset (example)

If the file “LST.PHONE” exists under the user ID “ALLKIND” on one of the volumes in any of the volume sets of the XYZ pubset, its path name is: “:XYZ:$ALLKIND.LST.PHONE”, i.e. the internal structure of the SM pubset and where the system stores files is generally irrelevant for the user.

Properties of SM pubsets

- The services offered are volume-set-specific, i.e. because of its properties, each volume set represents a special service type within the SM pubset. Volume sets with the same properties are assigned to the same service type. When a file is created, the system decides which volume set is most suitable as a file storage location with respect to the requirements of the user (performance, availability, use of special applications).
- Within the SM pubset files can be relocated to a different volume set without changing the address, i.e. the relocation is not visible for the user.
- An SM pubset can be extended with or reduced by empty volume sets during operation.
- Empty volumes can be added to separate volume sets or removed from them.
- If a single volume in an SM pubset fails, the damage is limited to the volume set concerned. This means that all the files on this volume set may be damaged, but the rest of the pubset is not affected and can be used further without interruption.

  If the control volume set fails, the complete SM pubset is taken to be failed. Systems support should therefore apply special fail-safe measures for the control volume set by using the mirror functions (DRV or RAID1) . However, an SM pubset that contains volume sets operated with DRV cannot be used as a shared pubset.
- SM pubsets cannot be used as home pubsets.

- Paging files can be set up on SM pubsets. If the paging files are used then only the volume sets to which the paging volumes belong are checked for external allocations.

Properties of volume sets

- A volume set is identified by a volume set ID, which forms the main part of the VSN for all volumes in the volume set.
- All volumes in the set have a uniform disk format (K, NK2, NK4) and allocation unit.
- Each volume set possesses a so-called **Volres** which mainly contains the physical configuration of the volume set.
- Only one cache area can be activated for a volume set.
- Each volume set forms a sub-container for complete files, i.e. single files cannot be spread over several volume sets.
- A volume set can adopt the following operating states:

DEFECT	the volume set is defective
DEFINED-ONLY	the volume set is defined but not accessible
IN-HOLD	the volume set is temporarily not functional
NORMAL-USE	the volume set is functional and accessible
CONTROL-VOLSET	the volume set is the control volume set

- The control volume set contains:
  - the pubset configuration file (pubset definition file) (this file contains a list of all volume sets belonging to the SM pubset)
  - special catalogs for job variables, migrated files, no-space files and files on private volumes
  - the user catalog
  - the guards catalog

SM pubsets with background levels

An SM pubset must contain a processing level.

Pubsets can be provided with background levels with the help of the software product HSMS. These form a three-level storage hierarchy together with the processing level with respect to access time, availability and costs:

- S0 level (for the processing level)
- S1 level (for the background level available online)
- S2 level (for the background level formed from volumes which can be accessed offline)

If files are not processed for long periods, it is advisable to migrate them to a more economical background level. In SM pubsets, the S1 level is implemented by a volume set belonging to the pubset, which must be reserved exclusively for this purpose. A tape pool made up of magnetic tapes (or MTCs) is used as the volume for the S2 level. The background levels are assigned exclusively to an SM pubset and cannot be used by more than one pubset. The metadata required for using the background levels, such as the migration archive directory, is on the SM pubset itself on the control volume set.
In the example below, the SM pubset with the catalog ID XYZ consists of three volume sets in the processing level.

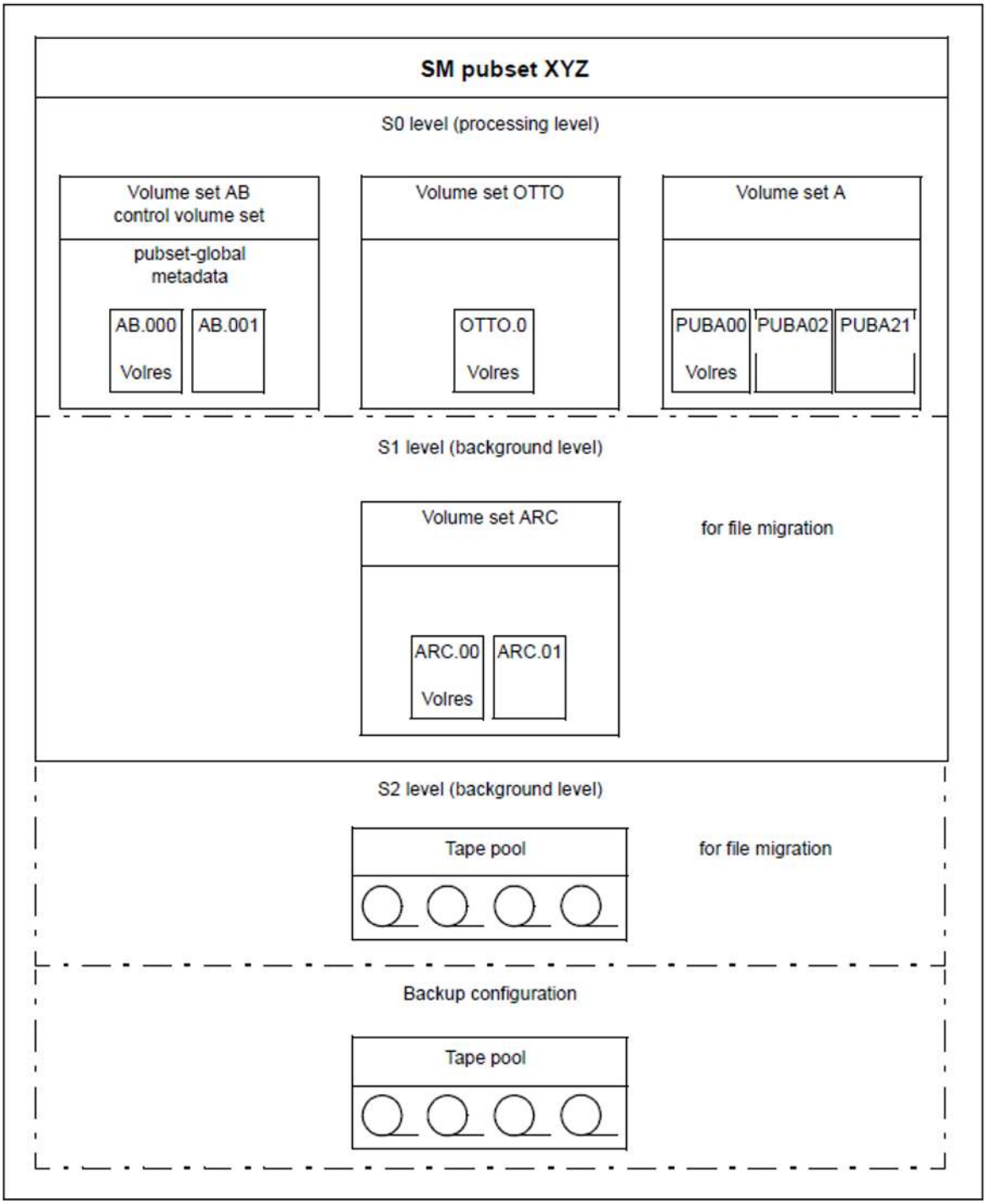


Figure 9: SM pubset structure with HSMS configuration

9.1.3.3 Pubset metadata

The term pubset metadata is taken to mean all data that is ultimately required for pubset operation, e.g. the physical configuration data, the physical and logical properties and the operating parameters (user catalogs, user quotas, etc.).

In addition to the pubset metadata described below, the system files $TSOS.SYSCAT.STORCLS (storage class catalog) and $TSOS.SYSCAT.VSETLST (volume set list catalog) can also be contained on SM pubsets, see the "System Managed Storage" manual [28].

Standard Volume Label (SVL) of Pubres and Volres

All volume set-global information is stored in the SVL (standard volume label) of the Pubres of an SF pubset or the Volres of a volume set.

The SVL is built up of two different record types.

- DMS record
  This contains a reference to the TSOSCAT or TSOSCAT.<volume-set-id> file catalog, the value of the allocation unit, an indicator for normal/abnormal termination of the last pubset session, an operating mode identifier (local or shared) and information on any cache used.
- Pubset record
  This contains the volume configuration (volume catalog) of the SF pubset/volume set. The volume catalog is stored in different locations, depending on the maximum number of volumes:
  - If the maximum number of volumes is 32, it is stored in the pubset record itself. The volume catalog in the pubset record contains the VSN, the volume type of each volume and the time stamp from the last operating phase of the volume.
  - If the maximum number of volumes is 255, it is stored in a special 4KB block on the Pubres/Volres. In this case, the pubset record only contains a reference to this block.

The complete SVL is set up at the time of pubset generation with the SIR and VOLIN utility routines.

Pubset configuration file

SF pubsets have no pubset configuration file. In the case of SM pubsets the pubset configuration file is set up when the SM pubset is generated or when several SF pubsets are grouped together to form an SM pubset.

The pubset configuration file ($TSOS.SYS.PUBSET.CONFIG) has two functions:

- Storing all configuration data of an SM pubset
- Storage medium for the reconfiguration data in the case of a reconfiguration job for the pubset

The pubset configuration file is an ISAM file and consists of three record types:

- Pubset record
  Contains all pubset-global data for the SM pubset, e.g. the number of volume sets and the name of the control volume set.
- Volume set record
  The number of these records corresponds to the number of volume sets. These contain all volume set-specific operating data.

- Reconfiguration record
  Contains just the information that is required to resume an interrupted reconfiguration job.

The pubset configuration file can only be modified while the SM pubset is operational.

MRSCAT list

MRSCAT is a list of all pubsets which are known to a BS2000 system. It contains the static (permanent) and dynamic operating parameters of the pubset. It is also the list of all file catalogs and provides information on their availability. MRSCAT is a central DMS data structure and is always the starting point for file addressing.

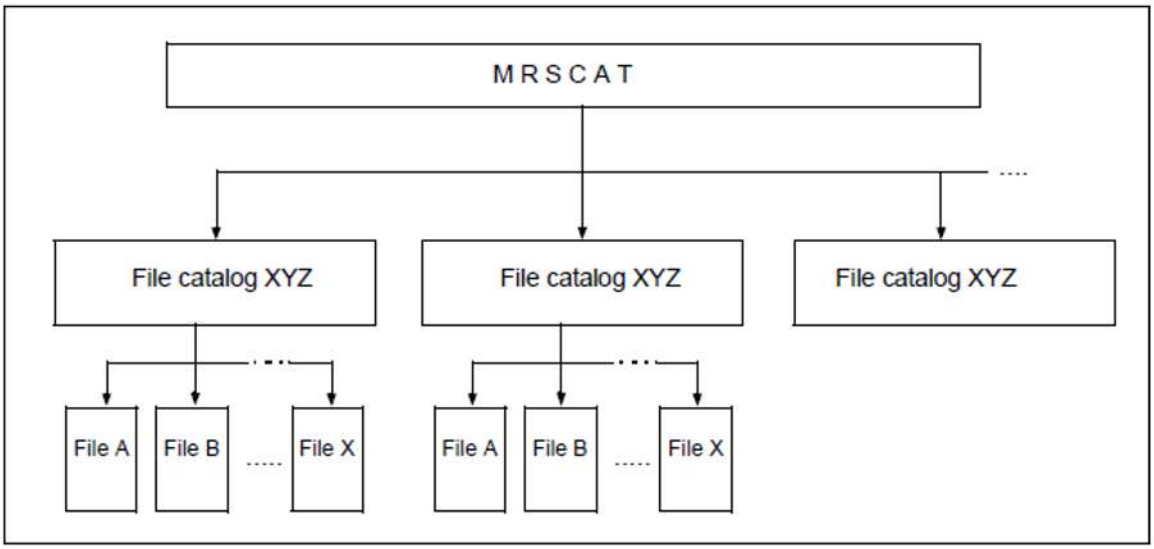


Figure 10: MRSCAT catalog structure - file catalog

Static operating parameters

The two ISAM files $TSOS.SYSTEM.MRSCAT and $TSOS.SYSTEM.MRSCAT.COPY are used for permanent storage of the operating parameters. They are stored on the home pubset and kept consistent to enable access to valid data if one of the two files fails. The files are set up during system initialization when a pubset is used as the home pubset for the first time.

MRSCAT information for the home pubset and for control volume sets of SM pubsets is also stored on the first page of the catalog file TSOSCAT or TSOSCAT.<control-volume-set-id>. This is necessary in order to avoid deadlock situations during startup as the ISAM files accesses are only possible when the pubset concerned is operating. The catalog file, by contrast, can be accessed at a very early stage with CMS functions.

Dynamic operating parameters

The second function of the MRSCAT list is the high-performance storage of all pubset operating parameters in main memory during the BS2000 session. An entry exists in MRSCAT for each volume set, SF pubset or SM pubset. The catalog ID, which corresponds to the current pubset or volume set ID, is used as the identifier for the MRSCAT entry.

9.1.3.4 User quota concept

Systems support can use the user quotas to check and restrict the space taken up on a pubset for each separate user.

The quotas are distinguished primarily by the type of files:

- PERM-SPACE for permanent files
- TEMP-SPACE for temporary files
- WORK-SPACE for work files

For these three types it is also possible to some extent to differentiate according to attributes (availability, performance) and storage levels (processing level, all levels).

The PERM-SPACE and TEMP-SPACE quotas exist for all pubsets; the other quotas are only available for SM pubsets.

Quota structure on SM pubsets for permanent files

The separate quotas have the following meaning:

- TOTAL-SPACE
  This quota is the space for all permanent files (for a user) on the pubset, regardless of whether they are on a background level (S1 or S2) or the S0 processing level.
  Background level is taken to mean file storage locations used by HSMS. The unit of measurement is the space that the file would occupy when in the S0 level and not the actual space occupied.
- S0-LEVEL-SPACE
  Total quota for permanent files on the S0 processing level.
- HIGH-PERF-SPACE
  Quota for files with a *HIGH performance value.
- VERY-HIGH-PERF-SPACE
  Quota for files with a *VERY-HIGH performance value.
- HIGH-AVAILABLE-SPACE
  Quota for files with a *HIGH availability value.

The connection between file attributes and quotas for permanent files is shown in the figure below. It must thereby be noted that loading a quota also implicitly loads all superordinate quotas.

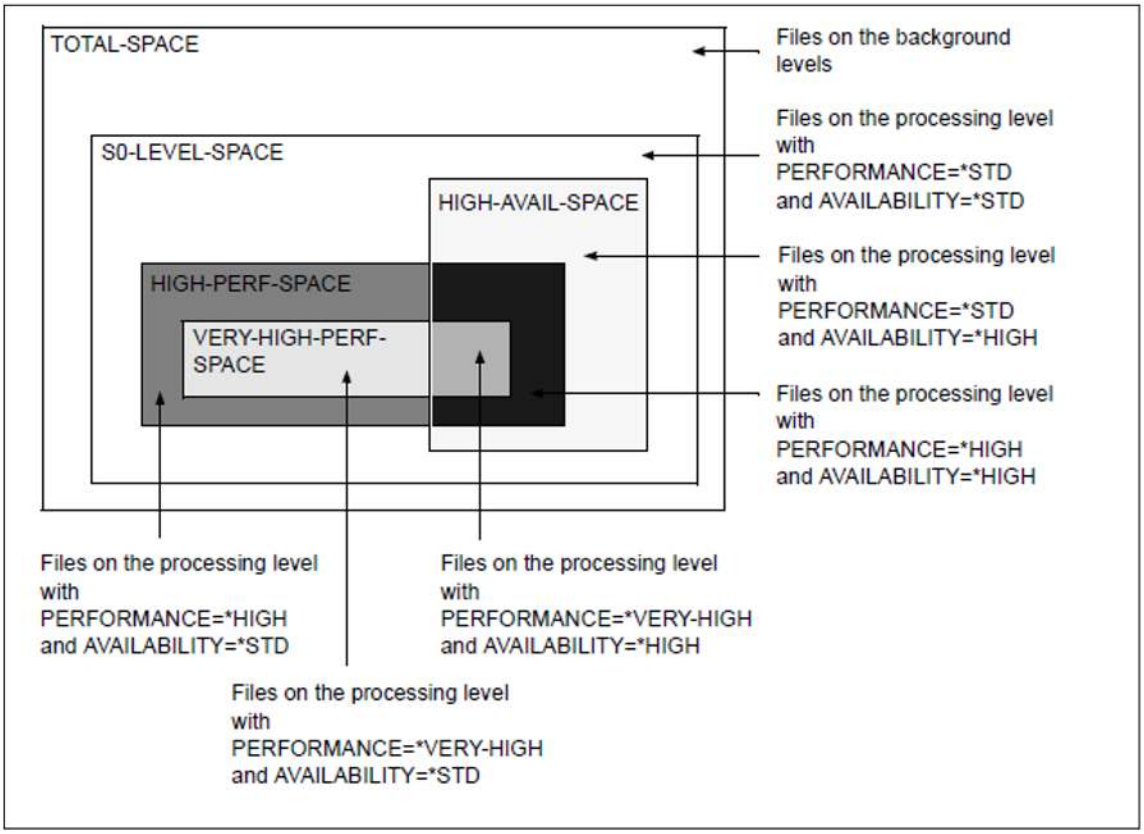


Figure 11: Quota structure for permanent files on an SM pubset and assignment to file attributes

Example for the figure above

Creating a file with the attributes AVAILABILITY=*HIGH and PERFORMANCE=*HIGH, causes the occupied values for the S0-LEVEL-SPACE and TOTAL-SPACE quotas, in addition to those for HIGH-AVAIL-SPACE and HIGH-PERF-SPACE, to be increased by the number of half-pages assigned to the file.

Quota structure on SM pubsets for temporary and work files

Temporary files and work files cannot be migrated to background levels or assigned the file attribute AVAILABILITY=*HIGH. This results in a temporary and work file quota structure which is simpler than that of permanent files.

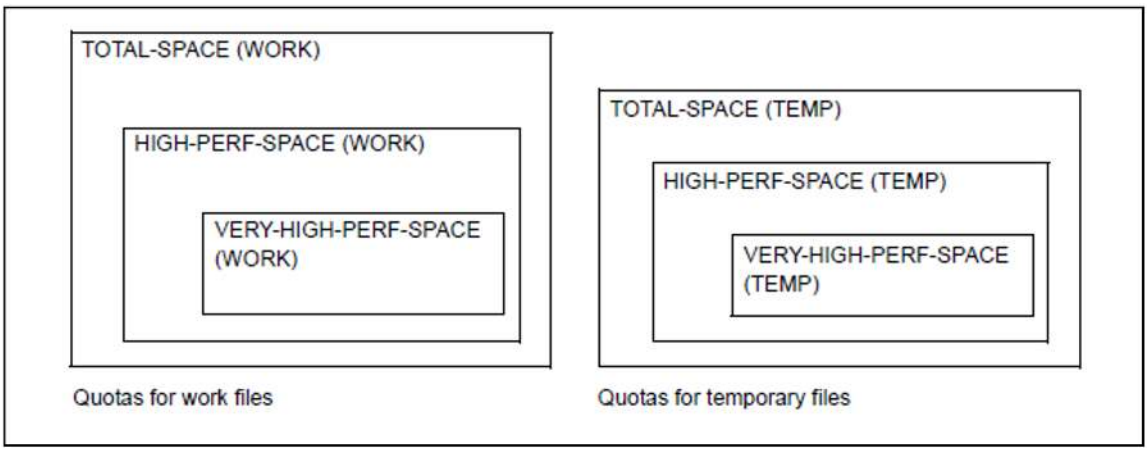


Figure 12: Quota structure for temporary and work files

Each quota is assigned a maximum and a current occupation value. The current quota occupation value (occupied value) is calculated from the user files and their file attributes. When files are created and enlarged, the system checks that the maximum values are not exceeded (if no PUBLIC-SPACE-EXCESS, which allows the permitted storage space to be exceeded, is set).

9.2 Pubset properties

This section describes the following properties:

- Physical properties
- Global properties
- User-specific properties
- Volume set and volume-specific properties

9.2.1 Physical properties

Physical properties include, for example, disk formats and I/O properties.

9.2.1.1 Disk formats

The formatting properties of an SF pubset or volume set are defined during initialization and are therefore unchangeable over the complete lifetime of the SF pubset or volume set. All volumes (disks) of an SF pubset /volume set have the same formatting property. Systems support defines it during volume initialization (with SIR or VOLIN):

- file processing with and without PAM key usage in processing mode: K and NK
- minimum allocation unit (min. AU) with which DMS assigns space for files on the volumes of the pubset: 6 KB, 8 KB or 64 KB
- minimum I/O transport unit (min. TU) in which volume inputs and outputs are to be made: 2 KB or 4 KB

Corresponding pubset formats are available for the listed volume formats and these can be retrieved with the SHOW-PUBSET-CONFIGURATION command.

The following volume format combinations are generally possible:

Volume format	Processing mode	min. AU (in KB)	min. TU (in KB)	Max. number of volumes per SF pubset/volume set
K format	K	6	2	32
NK2 format	NK	6	2	32
NK2 format	NK	8	2	255
NK4 format	NK	8	4	255
NK2 format	NK	64	2	255
NK4 format	NK	64	4	255

9.2.1.2 Physical properties of volume sets

Systems support must ensure that the volume set properties profiles are fulfilled by the physical volume set configuration.

Physical properties of the separate volumes

The physical properties of a volume set are determined by the properties of the volumes belonging to it and their I/O configuration (controllers, access paths, etc.). The volume configuration of the volume sets should be homogeneous so that meaningful properties profiles can be assigned to the volume sets, i.e. systems support should only group volumes together into a volume set whose properties harmonize with the intended properties profile. With specific properties, the volume set homogeneity is forced by the system (e.g. allocation unit size, format).

Maximum input/output length

The maximum input/output length is the maximum number of blocks which can be transferred for an I/O request. It is determined by the device type. The maximum input/output length is determined by the smallest value for all volumes in the volume set. If an additional volume is added to a volume set, the maximum input/output length of the volume set may be reduced. The maximum input/output length of a volume set is used by certain privileged applications (ARCHIVE, COPY) for optimizing I/O operations.

The maximum input/output length of a volume set can be displayed with the SHOW-PUBSET-CONFIGURATION command.

Volume sets as independent failure units

Volume sets of an SM pubset can (with the exception of the control volume set) be created as independent failure units.

For this purpose these volume sets must be mapped to physical configurations which are independent of each other with respect to failure. For example, the access paths (channels, controllers, etc.) to the volumes of volume sets which are to be independent with respect to failure, must be decoupled from each other. The cache configurations assigned to the volume sets must also be considered.

DRV functions can be used for providing volume sets which are particularly secure against failure. These functions provide a similar user interface for SM pubsets at the volume set level as for SF pubsets at pubset level. Shared pubsets cannot be operated with DRV.

9.2.2 Global properties

Pubset-global settings are used mainly for controlling pubset operation:

- CMS settings

  They mainly define the number and position of the CMS buffers which affect the catalog management.
- Operating mode, i.e. a pubset can be used:
  - in exclusive mode by a single system
  - as a shared pubset by several systems simultaneously, see "Shared pubsets"
  - exclusively by one user. In this case, only this ID and systems support under the TSOS ID can access this pubset.
- Default settings for pubset space and file format
  - The default values for the size of primary, secondary and maximum allocation of files can be set for SM and SF pubsets.
  - The default value for the file format can be controlled for an SM pubset.

9.2.3 User-specific properties

Systems support defines the following user-specific properties:

- Default catalog ID (default pubset)

  Each user is assigned a default catalog ID. This is also called the default pubset. He/She can create, process and delete files and job variables on this pubset without entering the catalog ID.

- Storage space limit

  On the default pubset systems support defines the storage space limit for the users.

- Pubset-specific access rights

  These include, for instance, the right to use tuning resources for file processing, to allocate storage space directly, to exceed the quota limit, and to use Net-Storage.

9.2.4 Volume set and volume-specific properties

The following properties can be defined for volume sets of SM pubsets and individual volumes:

- Availability and performance

  Systems support ensures increased availability by, for example, using RAID systems or by means of mirroring. The performance profile is controlled by using caches.

- ○
  ○
- Usage types:
  - Normal case (STD), the only possible usage type for control volume sets.
  - Work files (WORK).
  - HSMS usage at S1 level (HSMS-CONTROLLED)
- Usage restrictions
  - With volume sets there are usage restrictions for creating files and processing files.
  - Allocation restrictions can be set for the various volumes of an SF pubset or a volume set.
- Saturation threshold values for SF pubsets and volume sets

  Several threshold values can be defined. When threshold values are reached, console messages are issued. In the case of higher threshold values, jobs may be rejected.
- File services
  - Storage services (e.g. offered performance, availability) which are controlled with the help of storage classes and volume set lists.
  - HSMS management services (e.g. controlling file migration to background levels, controlling file backup settings) which are controlled with the help of HSMS.

9.3 Setting up pubsets

Various utility routines are used to set up pubsets. All the utility routines mentioned below are described in the “Utility Routines” manual [15].

9.3.1 Setting up SF pubsets

Setting up and extending SF pubsets

SF pubsets, including the home pubset, are set up and extended using the SIR utility routine.

Instead of extending pubsets with SIR, it is also possible to use the VOLIN utility routine to initialize the new volumes with the corresponding VSN and format. The next time the pubset is imported with IMPORT-PUBSET (not in the event of an implicit import during startup) the system asks whether the new disks are to be added to the pubset.

9.3.2 Setting up and replacing SM pubsets

SM pubsets can be set up by configuring new ones or by converting existing SF pubsets to SM pubsets.

9.3.2.1 Setting up and extending volume sets and SM pubsets

SM pubsets and volume sets are set up using the SIR utility routine.

SM pubsets and the associated volume sets can be set up together in a SIR run. However, each can also be configured in separate SIR runs. In this case, SIR actions which require an imported pubset (e.g. CREATE-PAGING-FILE) cannot be executed for a single volume set.

9.3.2.2 Converting SF pubsets to SM pubsets

Existing SF pubsets must be converted to SM pubsets to enable users to use SM pubsets for existing files. Multiple SF pubsets can be combined to form an SM pubset. The following fundamental aspects must be borne in mind here:

- The pubset ID is the catalog id part of the path name for objects cataloged in the pubset. If an SF pubset is converted into an SM pubset whose ID differs from that of the SF pubset, the addressing is changed for the objects.
- If there are objects in several SF pubsets, which are converted to the SM pubset, whose path names only differ by the catalog ID, name conflicts would occur when forming the SM pubset, i.e. the path names would no longer be unique. Users who only have files on one of the SF pubsets to be converted to the SM pubset cannot be affected by this since the user ID is part of the path name.
- If an SF pubset is converted into an SM pubset whose ID contains more characters than that of the SF pubset, the length of the path names for objects on the pubset are increased. This can result in path names which exceed the maximum permitted length.

The processing level, background levels and backup archives are to be included when converting SF pubsets to SM pubsets. A detailed description can be found in the “SMS” manual [28].

Only the procedure for the processing level is described below. Two options are available for this: Conversion with SMPGEN or saving and restoring.

Conversion with SMPGEN (in-place conversion)

The processing level can be converted in-place with the SMPGEN utility. It allows one SM pubset to be formed from one or more SF pubsets, where each SF pubset is transferred into a volume set. The user files on the SF pubsets remain unchanged after conversion and the SF pubset metadata files (e.g. file, user and guards catalogs, etc.) is converted into corresponding SM pubset metadata files. Specific pubset and volume set properties, user quotas, etc. are determined automatically for the SM pubset by SMPGEN. If systems support wishes to change these settings, they must post-process the SM pubset appropriately after generation.

SMPGEN provides the following statements for in-place conversion:

- CREATE-SYSTEM-MANAGED-PUBSET for checking and conversion to an SM pubset which is to be created
- MODIFY-SYSTEM-MANAGED-PUBSET for checking and adding additional SF pubsets to an existing SM pubset

Conversion by saving and restoring files

The transition from SF to SM pubsets via saving and restoring files is recommended in particular if a structure which differs from that resulting from in-place conversion is required for an SM pubset. This is, for example, the case if several SF pubsets are to be converted to one SM pubset with several volume sets.

The SF pubsets being converted into an SM pubset are completely backed up (possibly without the files on the background levels). The SM pubset is then created by SIR. User entries for the previous SF pubset users are created on the SM pubset. This is supported by HSMS, which allows backups of SF pubset user catalogs to be transferred to SM pubsets. The user attributes introduced for SM pubsets (e.g. user quotas for high availability files) are preset to ensure as high a degree of compatibility as possible. Finally, the guards catalog is then set up using backups made on SF pubsets if required. Finally, the user files are transferred.

9.3.3 Generating pubsets with PVSREN

The PVSREN utility routine can generate autonomous pubsets from the disk copies of an SM or SF pubset (pubset copy) within a system.

PVSREN uses the local replication functions of the disk storage systems to create autonomous pubsets by renaming pubset copies. In the case of SM pubsets the rules for renaming the volume sets are stored in a parameter file in which the new catalog ID for is defined for each volume set using the SET-NAME-OF-NEW-VOLUME-SET statement.

The CREATE-PUBSET-FROM-MIRROR statement is used to create an autonomous pubset. Here the MIRRORING-METHOD operand determines whether or with which function PVSREN generates a copy:

- The specification MIRRORING-METHOD=*SPLIT-MIRROR requires that the pubset’s copies have already been split and are available in double-point notation (see "Double-point notation for mirror disks"). In this case PVSREN only renames an autonomous pubset.
- PVSREN uses MIRRORING-METHOD=*CLONE(...)/*SNAP(...) to generate a copy on the basis of clone units or snap units and renames this as an autonomous pubset.

9.4 Configuring pubsets

Before a pubset is activated and can therefore be used, it must be configured, i.e.:

- Enter the pubset in MRSCAT.
- Define global properties
- Define user-specific properties
- Define volume-specific properties (for SM pubsets)

9.4.1 Managing MRSCAT entries

The maximum number of MRSCAT entries is limited and can be set to 128, 256, 512, 1024, 2048 or 4096 to optimize access performance. This maximum number can be set with the system parameter DMCMAXP.

Systems support can set up, modify and delete MRSCAT entries with the following commands:

- ADD-MASTER-CATALOG-ENTRY
  Creates new MRSCAT entries and simultaneously initializes the static pubset parameters, apart from the cache configuration.
- MODIFY-MASTER-CATALOG-ENTRY (EDIT-MASTER-CATALOG-ENTRY)
  Modifies the static pubset parameters, apart from the cache configuration.
-
- REMOVE-MASTER-CATALOG-ENTRY
  Deletes MRSCAT entries.

9.4.2 Defining global pubset properties

CMS settings

CMS buffer settings can be made with the RESIDENT-BUFFERS and NUMBER-OF-BUFFERS operands of the ADD-/MODIFY-MASTER-CATALOG-ENTRY or IMPORT-PUBSET command. These specifications are evaluated when the pubset is imported (IMPORT-PUBSET command), according to the following hierarchy:

1. Explicit parameter specification in the IMPORT-PUBSET command
2. Specifications via the ADD or MODIFY-MASTER-CATALOG-ENTRY command. If only one of the RESIDENT-BUFFERS or NUMBER-OF-BUFFERS parameters is specified, the default value applies to the other (RESIDENT-BUFFERS=*NO, NUMBER-OF-BUFFERS=32).
3. Definitions according to system parameters CATBUFR and BMTNUM

Operating modes

The operating modes can be set with the ADD-/MODIFY-MASTER-CATALOG-ENTRY command using the following operands:

- ACCESS-CONTROLLED: exclusive use by one user ID
- SHARED-PUBSET: access by several systems (shared pubset)

The shared pubset property can also be defined when the pubset is imported (IMPORT-PUBSET command, USE operand).

Default settings for pubset space and file format

The default values for the size of primary, secondary and maximum allocation of files can be set for SM and SF pubsets (MODIFY-PUBSET-SPACE-DEFAULTS (EDIT-PUBSET-SPACE-DEFAULTS) command).

The command MODIFY-PUBSET-SPACE-DEFAULTS ...,PUBSET-TYPE=*S-M(FILE-FORMAT=*K/*NK2/*NK4) can be used for setting the file format default for SM pubsets.

9.4.3 Defining user-specific settings

Systems support specifies the default catalog ID (default pubset) of a user and defines various limit values (storage space, number of job variables and files) for the default pubset and pubset-specific access rights.

9.4.3.1 Assigning the default catalog ID (default pubset)

Systems support assigns each user a default catalog ID with the ADD-USER command (DEFAULT-PUBSET operand). This defines the pubset accessed if the user accesses files or job variables without specifying a catalog ID.

The default catalog IDs are stored in the user catalog of the home pubset.

> If systems support adds a new user ID which is to have access to several pubsets, they must issue an ADD-USER command for each of these pubsets. This also applies to modifying and deleting user entries.

9.4.3.2 Defining limit values and access rights

Systems support defines user-specific limit values and access rights for a pubset with the ADD-USER and MODIFY-USER-ATTRIBUTES commands.

Operands for limit values:

- PUBLIC-SPACE-LIMIT and PUBLIC-SPACE-EXCESS

  Determines the maximum storage space which the user may occupy on the pubset. PUBLIC-SPACE-EXCESS specifies whether this limit may under certain circumstances be exceeded.
- TEMP-SPACE-LIMIT

  Determines the maximum storage space which the user may occupy on the pubset for temporary files.
- JV-NUMBER-LIMIT

  Maximum number of job variables which the user may create.
- FILE-NUMBER-LIMIT

  Maximum number of files which the user may create.

Operands for access rights:

- DMS-TUNING-RESOURCES
  Defines whether a user is allowed to use tuning resources for file processing. The pubset caches and high level storage classes are tuning resources.
- PHYSICAL-ALLOCATION
  Defines whether a user can request storage space directly by specifying the volume set, volume or block.
- NET-STORAGE-USAGE
  This right grants or denies the user use of Net-Storage.
  The space occupied on a Net-Storage volume is not counted toward the user’s public space limit.

For further information on managing the user catalog, see section "Managing an access authorization and user catalog"

9.4.4 Defining properties of a volume set

- Defining availability
- Defining the performance profile
- Defining the usage type
- Usage restrictions
- Defining saturation threshold values for volume sets and SF pubsets
- Defining file services
- Defining user quotas

9.4.4.1 Defining availability

Systems support defines the availability of a volume set with the AVAILABILITY operand in the MODIFY-PUBSET-DEFINITION-FILE command. Possible values are STD and HIGH. In this case the system does not check whether the actual property of the volume set matches the assigned values. It is up to systems support to select values which correspond to reality.

The availability value assigned by systems support corresponds to the long-term positioning which was intended for the volume set within the SM pubset. Changing the AVAILABILITY value from HIGH to STD requires considerable organizational effort as particularly the files on the volume set with the file attribute AVAILABILITY=*HIGH must first be moved to another volume set with high availability.

The availability of a volume set can be enhanced, for instance, by using a suitable RAID level in the disk storage system (RAID 1/0, RAID6) or volume mirroring.

9.4.4.2 Defining the performance profile

Systems support defines the performance profile of a volume set with the PERFORMANCE-ATTR operand in the MODIFY-PUBSET-DEFINITION-FILE command. Systems support is responsible for defining the volume set performance profile in such a way that it correctly defines the real circumstances. PERFORMANCE-ATTR defines a performance spectrum (PERFORMANCE=) and the specification of restrictions with respect to the write consistency (WRITE-CONSISTENCY=).

- Performance spectrum:
  A volume set with a cache can generally cover a spectrum of different user performance requirements (STD, HIGH, VERY-HIGH). It is therefore suitable for use with files requiring PERFORMANCE=*STD, *HIGH or *VERY-HIGH. The different performance values correspond to different types of cache usage which are possible when processing files:

  PERFORMANCE=*STD (no cache usage)

  PERFORMANCE=*HIGH (caching with migration)

  PERFORMANCE=*VERY-HIGH (caching without migration)

  The type of cache usage can be changed to suit the individual performance requirements of the separate files.

  > ℹ It is not possible to simultaneously assign several different caches to one volume set.

- Write consistency:
  The cache media differ in their write consistency (WRITE-CONSISTENCY=*BY-CLOSE/*IMMEDIATE). WRITE-CONSISTENCY=*IMMEDIATE means that enhanced performance can be used on the volume set without restrictions. WRITE-CONSISTENCY= *BY-CLOSE means that enhanced performance is only used for write operations if data consistency at CLOSE time is sufficient and is requested as such.
  A cache is considered to be secure if the risk of failure for output data buffered in the cache is not higher than if it were written immediately to disk.
  Volatile caches can generally be used for performance enhancement of read file accesses. They should only be used for writes if the user accepts the reduced write consistency and expresses this by setting the file attribute DISK-WRITE=*BY-CLOSE.

Recommendations for the performance profile

The settings recommended in the following table for the different cache media can therefore only be taken as an aid to orientation for the performance spectrum since they do not take the conditions of real configurations into account. It is, however, possible to make recommendations for write consistency with enhanced performance which can be used in any configuration.

Volume set configuration	Recommended performance profile for PERFORMANCE (Performance spectrum)	Recommended performance profile for WRITE CONSISTENCY with enhanced performance (PERF=HIGH/VERY-HIGH)
Volume set made up of volumes with no special performance properties and with no cache assigned	STD	irrelevant
Volume set which is assigned main memory as the cache medium	STD, HIGH, VERY-HIGH	not satisfied (BY-CLOSE)
Volume set whose volumes are connected to a cache controller	STD, HIGH	satisfied (IMMEDIATE)

Table 20: Recommendations for performance profiles

9.4.4.3 Defining the usage type

The volume set usage type of an SM pubset can be defined with the USAGE operand of the MODIFY-PUBSET-DEFINITION-FILE command:

- STD
  Normal setting. This is the only possible usage type for the control volume set.
- WORK
  Work files can only be stored on volume sets with this usage type.
- HSMS-CONTROLLED
  Volume sets with usage type are required for setting up the S1 level with HSMS. They are only used for this purpose.

9.4.4.4 Usage restrictions

In contrast to the usage type, usage restrictions can also be changed at short notice. There are different restrictions for creating files, file processing and for temporarily deactivating a volume set.

Usage restrictions are defined with the MODIFY-PUBSET-RESTRICTIONS command in the operands below:

- NEW-FILE-ALLOCATION
  Restrictions for creating files.
  Only physical allocation or a general lock are possible.

  A restriction may be revoked only if access to the volume set is not restricted.
- VOLUME-SET-ACCESS
  Restrictions for processing files on TSOS.
- ALLOCATION-ON-VOLUME
  Usage restrictions for single volumes.
  Only physical allocation or a general lock are possible.

9.4.4.5 Defining saturation threshold values for volume sets and SF pubsets

The various SF pubsets or volume sets of an SM pubset are assigned definable threshold values by systems support for monitoring storage bottlenecks. The highest exceeded level determines the actual saturation level, see also section “Monitoring storage space saturation”.

The threshold values for the various saturation levels are set with the SET-/MODIFY-SPACE-SATURATION-LEVEL command. The saturation levels can be modified temporarily or permanently with this command (operand SCOPE=*PERMANENT/ *TEMPORARY/*NEXT-PUBSET-SESSION).

The system storage space emergency reserve for a system start in ZIP mode can be set on a pubset-specific basis with the MODIFY-SPACE-SATURATION-LEVEL command. This is handled in the same way as saturation level 6.

The global system default for saturation level 4, which was set during system generation with the system parameter L4SPDEF, is effective for pubsets or volume sets for which no explicit definitions were made.

The saturation level settings can be retrieved with the SHOW-SPACE-SATURATION-LEVELS command for SF pubsets or with the operand VOLUME-SET=*ALL/<cat-id> for the volume sets of an SM pubset.

For compatibility reasons, the ADD- and MODIFY-MASTER-CATALOG-ENTRY commands provide restricted setting options for saturation level 4 for SF pubsets only.

9.4.4.6 Defining file services

From the viewpoint of the user, an SM pubset appears as a file container which provides specific file management services. These are called file services. They can be subdivided into

- storage services (e.g. performance, availability offered)
- HSMS management services (e.g. controlling file migration to background levels, controlling the creation of file backups) .

The user requests the file services he requires by means of file attributes for which appropriate values must be assigned to the files.

Storage services

Systems support can set up storage classes which represent a specific combination of values for the file attributes relevant to storage location (see the CREATE-STORAGE-CLASS and MODIFY-STORAGE-CLASS (EDIT-STORAGE-CLASS) commands).

The user selects the storage class for a file which fulfills the requirements for a file most closely (see the STORAGE-CLASS operand of the CREATE-FILE or MODIFY-FILE-ATTRIBUTES command). He/She does not have to enter the values individually.

The storage service requested by the user for a file is taken into account when:

- a file is created
- a file is moved from a background level to the processing level
- for an existing file the user modifies the file attributes relevant to storage location in such a way that it is incompatible with the previous storage location

The system automatically determines the most suitable volume set

Systems support can define a default storage class for every user. This can make some direct attributes ineffective when there are no rights to physically edit attributes.

Volume set lists

The system behavior can be influenced by volume set lists which systems support configures and links to storage classes (see the VOLUME-SET-LIST operand of the CREATE-VOLUME-SET-LIST and CREATE-STORAGE-CLASS commands).

If a user assigns a file a storage class to which a volume set list is assigned, the system stores the file with priority on a volume set belonging to this volume set list. If the file already occupies storage space on another volume set, it is, if possible, relocated from this volume set to one of the preferred volume sets. Files which are assigned no storage class or a storage class without a volume set list are stored with priority on volume sets which do not belong to any of the volume set lists configured by systems support.

HSMS management services

Management services are requested using HSMS management classes. They must be set up by systems support (HSMS statement CREATE-MANAGEMENT-CLASS). The HSMS management classes represent specific methods of data backup (e.g. backup frequency, lifetime of backup versions, etc.) and rules which control the migration to background levels (e.g. suitability for migration, dependent on the time since the last access).

When the user assigns a file to an HSMS management class (MANAGEMENT-CLASS operand of the CREATE-FILE or MODIFY-FILE-ATTRIBUTES command), it is submitted to the backup or migration methods which are represented by this HSMS management class. The user can exclude particular files from being migrated to background levels either generally or under specific conditions (MIGRATE operand of the CREATE-FILE and MODIFY-FILE-ATTRIBUTES commands). If necessary, the user must have allowance for physical allocation.

9.4.4.7 Defining user quotas

Systems support defines user quotas for a volume set with the MODIFY-USER-PUBSET-ATTRIBUTES command. The following operands are available for this purpose:

- PERM-SPACE-LIMITS
  Determines the maximum storage space which the user may occupy on the volume set for permanent files.
- TEMP-SPACE-LIMITS
  Determines the maximum storage space which the user may occupy on the volume set for temporary files.
- WORK-SPACE-LIMITS
  Determines the maximum storage space which the user may occupy on the volume set for work files.

The limit values can be differentiated according to the subquotas for all operands (TOTAL-SPACE, S0-LEVEL-SPACE, HIGH-PERF-SPACE, VERY-HIGH-PERF-SPACE and HIGH-AVAIL-SPACE).

9.5 Activating/deactivating pubsets

Data pubsets can only be activated and deactivated via the command interface. The home pubset is activated automatically during system startup and deactivated during shutdown.

Systems support can use the SHOW-PUBSET-IMPORT-EXPORT command to obtain information on the processing state of active import and export jobs.

Static pubset states

The pubset state changes each time it is activated and deactivated. This is also stored in the MRSCAT entry and provides information on pubset usage, see also the figure below.

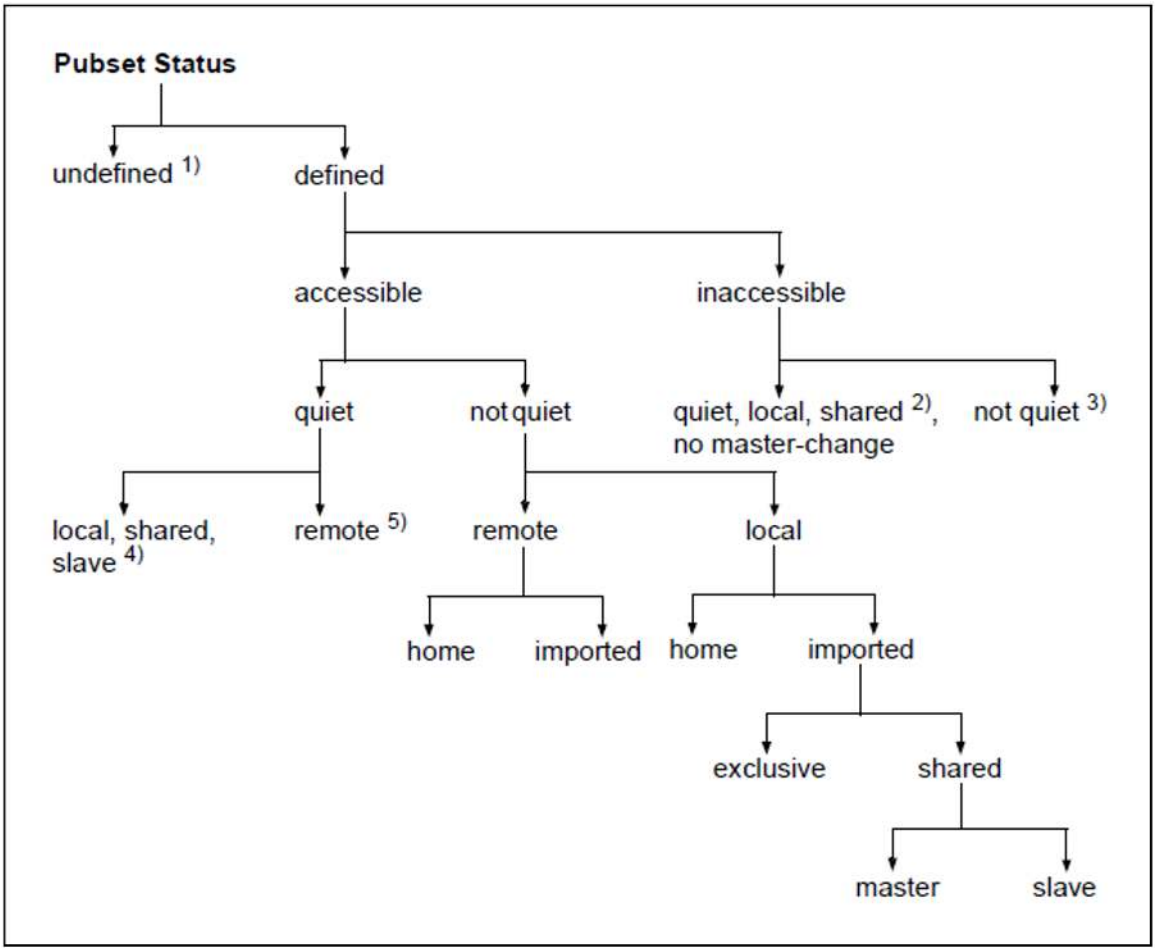


Figure 13: Static pubset states

1) This state means that no MRSCAT entry is defined

2) This state is assumed on a pubset slave when a shared pubset no longer has a pubset master after a master change has failed. The operator must start the master change explicitly with the IMPORT-PUBSET SHARER-TYPE=*MASTER(MASTER-CHANGE=*YES) command on one of the pubset slaves.

3) This state is assumed if the pubset is not imported.

4) This state is assumed by the pubset slave after the MSCF connection fails.

5) This state is assumed after the MSCF connection fails.

9.5.1 Activating a pubset

Pubset activation is started with the IMPORT-PUBSET command. A basic requirement for this is the hardware connections to the volumes concerned.

> A pubset in which files on Net-Storage are cataloged can also be activated if the Net-Storage concerned is not available. However, for reasons of availability you are recommended first to make the Net-Storage available and then to place the pubset in service, see the chapter “Net-Storage management”.

A further requirement is that an MRSCAT entry exists for the pubset and contains the following information:

- pubset ID
- pubset type (SF or SM pubset)
- control volume set ID (only for SM pubsets)

All further information can either be specified at the time of activation or determined from the storage locations themselves and stored in the appropriate MRSCAT entries. The MRSCAT entries assigned to the pubset must remain intact during the pubset session. The definitions in the entries can be modified, but the changes are only effective for the next pubset session unless they are activated with the pubset reconfiguration functions.

9.5.2 Special startup factors

Unforeseeable special situations can occur during activation, e.g.:

- internal system problems or bottlenecks (e.g. lack of storage space)
- connection problems
- volume defects, ambiguous volume identifiers
- inconsistent pubset metadata (e.g. invalid time stamp)
- faulty current master assignments
- difficulties in assigning the requested cache areas
- problems connecting cache areas still assigned

If problems occur during the import operation, systems support can abort the current import job before it has terminated by using the CANCEL-PUBSET-IMPORT command. In a shared pubset network, the main advantage of this is that an active import with master change can be aborted while the pubset remains accessible to at least one other system.

Checking homogeneity

The homogeneity of pubset mirroring for the pubset to be imported is checked by specifying the CHECK-PUBSET-MIRRORS=*YES operand in the IMPORT-PUBSET command. As soon as an inhomogeneity is determined in a pubset volume, the import operation is interrupted and systems support has the choice of aborting or continuing the import operation (see section "Checking the homogeneity in the case of pubset extension" in chapter "Modifying dynamic configuration data of SF and SM pubsets").

Excluding defective volume sets from import

In the IMPORT-PUBSET command systems support can specify volume sets (with the exception of the control volume set) which are removed automatically from the pubset before it is actually activated (DEFECT-VOLUME-SET operand). Only the remaining part of the pubset is then activated. In this case a list of the files on the removed volume set is created which can be used in a later RESTORE call (by HSMS).

Repairing defective F5 labels and TSOSCAT files

If there is a suspicion that a pubset's control structures are defective (e.g. the TSOSCAT file or F5 labels), the following operands can be specified in the IMPORT-PUBSET command:

- RECONSTRUCT-F5-LABEL=*YES: Initiate F5 label reconstruction explicitly
- REPAIR-TSOSCAT=*YES: Activate repair mode

F5 label reconstruction

If, during the reconstruction of the F5 labels in a TSOSCAT file, an error is encountered in the chaining of a user ID's TSOSCAT pages (defective user chain), then an analysis mode is activated in catalog management (CMS) for the remainder of F5 reconstruction. In this analysis mode, the chaining of the user IDs that remain to be processed is logged at the console for diagnostic purposes. An intact chaining is indicated by message `DMS13A7` (one or more per ID). If an error occurs in a user chain then this is logged by means of message `DMS13A8`.

This information can be used to assess the extent of the damage to the TSOSCAT file. Once analysis is complete, the pubset import is always terminated with an error in such cases. It is then not normally possible to import the pubset successfully until the TSOSCAT file has been repaired.

In such cases, Customer Support should always be contacted. The repair can be performed by system specialists manually following a suitable diagnosis or can be initiated automatically.

Repair mode

During the import phase, the command attempts to make the defective user chains accessible for catalog management again by unchaining defective blocks or detaching the remainder of the chain.

> **CAUTION!**
> Files whose catalog entries are located on pages that have been detached in this way are lost! No information concerning the affected files can be provided by the system. The service should always be contacted before automatic TSOSCAT repair is activated.

To protect against accidental activation, the repair mode must be confirmed once via the message `DMS13AE`. The implemented measures are logged via messages `DMS13A9`, `DMS13AA`, `DMS13AB` and `DMS13AC`. If the repair fails (for example, because of a permanent hardware error), message `DMS13AD` is displayed.

9.5.3 Deactivating a pubset

A pubset is deactivated with the EXPORT-PUBSET command. A pubset can only be deactivated if no user is using it as a resource.

The command SHOW-PUBSET-OCCUPATION provides information on the current pubset users. The export job should therefore only be started when no further occupation is displayed with SHOW-PUBSET-OCCUPATION.

If it is presumed that data blocks which have not yet been released exist (e.g. because of a preceding error or system crash), the operator can remove these with UNLOCK-DISK (disk by disk or for the complete pubset). The operator must make sure when doing this that no systems still working with the disk are removed.

Special shutdown factors

If the operator wishes to start the pubset export although the pubset is still occupied, the following options are available:

- The operator starts the export job and waits until all pubset users have released their occupancy. The actual pubset export begins after this.

  If the release takes too long, the operator can abort the export job with the CANCEL-PUBSET-EXPORT command. The pubset is then fully available again.
- The operator starts the export with the EXPORT-PUBSET ...,TERMINATE-JOBS=*YES command. All pubset users are forced to release their occupancy. This forces termination of all occupying tasks and pubset occupations are released during this termination.
- The FORCE-PUBSET-EXPORT command is the final alternative. Pubset export is started in spite of existing occupancy. This results in pubset management data not being correctly cleared, which in turn results in severe management data inconsistencies if it is activated again in the same BS2000 session. This option should only be used in extreme emergencies. It is logged with the message `DMS0379` on the console.

9.6 Administering pubsets

Pubset administration comprises the following tasks:

- Reconfiguring pubsets
- Monitoring storage space saturation
- Reorganizing disk storage space

9.6.1 Reconfiguring pubsets

This section describes how you can modify the configuration of pubsets and which commands you use to do this. For information on reconfiguring Net-Storage volumes, see section "Reconfiguring Net-Storage volumes" in chapter "Managing Net-Storage in BS2000".

9.6.1.1 Modifying static configuration data of SM pubsets

The static configuration data of an SM pubset includes the volume set configuration definition and the volume set service attributes.

The static configuration data of an SM pubset is modified with the MODIFY-PUBSET-DEFINITION-FILE (EDIT-PUBSET-DEFINITION-FILE) command.
The command enables:

- a new volume to be defined with its properties for the pubset
- the properties (service attributes) of an existing volume set to be modified
- or a defined volume set to be deleted from the pubset configuration (provided it is not activated).

The static configuration data is modified in both the pubset configuration file and in the MRSCAT.

9.6.1.2 Modifying dynamic configuration data of SF and SM pubsets

"Dynamic configuration data" comprises configurations of pubsets (volumes of SF and SM pubsets and volume sets of SM pubsets), as well as usage restrictions and the cache configuration.

The current pubset configuration of the system can be displayed using the SHOW-PUBSET-PROCESSING command.

The command SHOW-DEVICE-CONFIGURATION UNIT=*PUBSET-DEVICES(...) is used to output information on a selection of disks belonging to the pubset.

Modify the configuration of a pubset

The configuration of a pubset can be modified using the MODIFY-PUBSET-PROCESSING command. The following options are available:

- Individual disks can be added to or removed from an SF pubset.
- Individual volume sets can be added to or removed from an SM pubset.
- Individual disks can be added to or removed from a volume set which is the component of an SM pubset.

Pubres and Volres cannot be removed from the SF pubset/volume set.

An empty volume which is suitable for the pubset or volume set can be added with SIR, which is used for pubset generation.

Notes on volume set reconfiguration

- A volume set must be defined with the MODIFY-PUBSET-DEFINITION-FILE (EDIT-PUBSET-DEFINITION-FILE) command before it can be activated for an SM pubset.
- Before a volume set can be deactivated, it must be cleared of all files and locked against primary allocation (see "Restricting usage to remove disks and volume sets").
  A defective volume set is an exception to this rule and deactivation can be forced, which causes all the files it contains to be lost. Because of this, a list of all files on the defective volume set is created when it is deactivated and can be used directly as an input file to the HSMS subsystem for restoration. The list is stored in the file $TSOS.SYS.PUBSET.DEFECT.<volume-set-id>.<date.time>.
  The defective state of a volume set is either detected by the system itself, e.g. when it is no longer possible to access the metadata on a volume set, or it can be set with the MODIFY-PUBSET-RESTRICTIONS command.

Notes on disk reconfiguration

- When adding an empty disk, the disk properties must be compatible to those of the volume set or SF pubset.
- In the case of disk removal, a check is first made to ensure that the disk is empty. Then the disk data structures are cleaned up and the access rights released.
- Disks of a pubset can be attached and detached with the ATTACH-/DETACH-DEVICE UNIT=*PUBSET-DEVICES(...) commands. Since the names of the corresponding disks are managed in the SVL of the system disk, these must have been entered in MRSCAT . The entry is made each time the pubset is imported or exported, when changing the makeup of the pubset (MODIFY-PUBSET-PROCESSING), or explicitly with the ADD-, EDIT- or MODIFY-MASTER-CATALOG-ENTRY command.

  As many disks as possible are always attached. When mirroring with DRV, both disks are attached. When mirroring in a disk storage system controller, only the standard disks are attached. If the mirror disks are to be attached, then the mirror disk of the system disk (Pubres) must be specified in the PUBSET operand.

Checking the homogeneity in the case of pubset extension

When pubsets are extended using the MODIFY-PUBSET-PROCESSING command, a homogeneity check for mirroring is performed when the CHECK-PUBSET-MIRRORS=*YES operand is specified: If a volume which has different mirroring properties from the volumes which have already been processed is detected in the course of pubset extension, the answerable message `DMS1369` is output to SYSOUT.

Depending on the caller's answer, one of the following procedures is selected:

- Pubset extension is aborted
- Despite inhomogeneity being detected on the pubset volume which is being processed, pubset extension is continued. The message `DMS136B` is issued on the console for every further volume with different mirroring properties.

The subsystem SHC-OSD must be available for the homogeneity check.

> The CHECK-PUBSET-MIRRORS command also enables you to check the homogeneity of pubset mirroring before the pubset configuration is changed dynamically.
>
> When a pubset is placed in service, the homogeneity check can be requested using the operand of the same name in the IMPORT-PUBSET command.

Criteria for the homogeneity of pubset mirroring

A pubset is homogeneous with regard to local mirroring when the following conditions are satisfied:

- The same number of mirror units is assigned to all volumes of the pubset.
- All mirror units of the pubset are in the same operating state. By and large only the operating states `ESTABLISHED` and `SPLIT` are of importance here; the `ESTABLISHING` and `SPLITTING` states only occur temporarily as intermediate states.
- If remote mirror units are also being used, either all mirror units which are assigned to the pubsets are located in the local or in the remote disk storage system, or there is a complete set of mirror units in both the local and the remote disk storage system.

The pubset is homogeneous with regard to remote mirroring when the following conditions are satisfied:

- The same number of mirror units is assigned to all volumes of the pubset in the remote disk storage system.
- All remote mirror units are in the same state, `ACTIVE` or `IN-HOLD`.
- All remote mirror units are operated in the same mode (synchronous, semi-synchronous or adaptive).
- All remote mirror units are located in the same remote disk storage system.

Restricting usage to remove disks and volume sets

An object cannot be removed from the pubset configuration if it contains any data. The MODIFY-PUBSET-RESTRICTIONS command is an important tool for fulfilling this requirement. Systems support can use it to assign or release access and allocation restrictions.

- Allocation restrictions

  They can be granted for volumes or volume sets. Volumes can thereby be completely locked against allocation or only cleared for physical allocation. Volume sets can be locked against primary allocation or only cleared for physical allocation, where physical allocation means direct specification of the volume set ID.

- Access locks

  These exist exclusively for volume sets. When a lock is set, access to data in the files on the volume set is reserved exclusively for systems support. A volume set can simultaneously be completely closed, i.e. all access to the metadata on it is forbidden. This access lock is to be recommended if an error is predicted for the volume set, e.g. caused by an I/O request failure. The volume set is subsequently seen as defective if the access lock cannot be revoked.

  A lock to create files is not possible for the control volume set as this would cause undesired impedance of system functions.

Setting the volume-oriented restrictions reduces the number of freely available volumes and the amount of freely available space. It is not allowed if a severe storage bottleneck already exists (saturation level 4 exceeded) or would be caused by setting the restriction.

9.6.1.3 Replacing and removing SF pubsets with the same names

A pubset is replaced by another pubset with the same name as follows:

1. Save any information from the old pubset which may be required
2. Export the old pubset
3. Detach the volumes from the old pubset with the DETACH-DEVICE command (see the “Commands” manual [27]) or rename them
4. Generate the volumes for the new pubset with SIR
5. Import the new pubset with IMPORT-PUBSET...,ACTUAL-JOIN=*FIRST
6. Read any saved files in

A pubset is replaced by another pubset with the same name from a different system as follows:

1. Save any information from the old pubset which may be required
2. Export the old pubset
3. Detach the volumes from the old pubset with the DETACH-DEVICE command or rename them
4. Optional: Modify the MN of the Pubres in the MRSCAT entry of the pubset
5. Export the new pubset to the other system
6. Attach the volumes of the new pubset with the ATTACH-DEVICE command
7. Import the new pubset

If the new pubset is to be used as the home and paging pubset, the volumes from the old pubset with the same name must be renamed or physically detached before the next system initialization to ensure that the number of physically available pubsets remains unambiguous.

If this is accidentally forgotten, systems support must carefully select the required volumes interactively during system initialization.

9.6.1.4 Converting existing pubsets to large pubsets

Existing pubsets can be upgraded with the SET-PUBSET-ATTRIBUTES command to large pubsets with and without support for large files.

```
/SET-PUBSET-ATTRIBUTES ...,LARGE-VOLUMES=*UNCHANGED/*ALLOWED(
                           LARGE-FILES=*UNCHANGED/*ALLOWED)
```

The following upgrades are possible:

Small pubset	to	Large pubset without large files
Small pubset	to	Large pubset with large files
Large pubset without large files	to	Large pubset with large files

When importing the pubset these attributes are added to the MRSCAT and displayed by the corresponding informational functions:

- SHOW-MASTER-CATALOG-ENTRY command
- STAMCE program interface

To set up and expand (large) pubsets with SIR, see the “Utility Routines” manual [15].

> Permission to use large volumes and files cannot be removed from a pubset once it is granted.

All pubsets with the “large volumes” and “large files” attributes can be displayed with the SHOW-PUBSET-ATTRIBUTES and SHOW-MASTER-CATALOG-ENTRY commands.

9.6.2 Managing an access authorization and user catalog

The ADD-USER, MODIFY-USER-ATTRIBUTES, LOCK-USER, REMOVE-USER and UNLOCK-USER commands can be used by systems support to define which users may access a pubset and whether the access right has been revoked for a user. Since each pubset has its own user and file catalogs, systems support can group the user IDs on the pubsets according to the functions the users exercise. The following applies here:

- Entries for all users must be stored in the home pubset user catalog because the access checks with the SET-LOGON-PARAMETERS command are made in the user catalog of the home pubset.
- It is recommended that users be entered in the user catalog of the home pubset with PUBLIC-SPACE-LIMIT=0, FILE-NUMBER-LIMIT=0 and JV-NUMBER-LIMIT=0. The users can access files of this pubset without hindrance, but may not create any files or use any job variables.

  > **i** A user entry with PUBLIC-SPACE-LIMIT=0 has no effect on creating job variables (these occupy no storage space) or access to files of other user IDs (this is subject to the normal checks: shareability, passwords, read or write access, etc.). The creation of job variables and file entries on this pubset is prevented by JV-NUMBER-LIMIT=0 and FILE-NUMBER-LIMIT=0.

- When a user has no entry in the user catalog of the pubset concerned, he/she cannot access this pubset, not can he/she access shareable files or job variables held by other users on this pubset. This is recommendable, for example, when the pubset is required for production and user IDs are employed solely for test and training purposes.
- Access protection is rendered ineffective if the system parameter FSHARING was assigned the value 1 during system generation. All users can then access all pubsets and do not even require an entry in the user catalog concerned.

9.6.3 Monitoring storage space saturation

The system monitors the degree of saturation on pubsets in order to detect storage space bottlenecks on shared volumes in good time. Systems support can influence this monitoring by setting saturation levels, see section "Defining saturation threshold values for volume sets and SF pubsets". The saturation level is reached when less space is free on an SF pubset or volume set than is defined for the saturation level concerned.

Messages concerning the saturation level

If the saturation level reached changes, the messages `EXC044i`, `DMS141C` or `DMS1400` are output to the console:

`EXC044i` Message in the case of SF pubsets when saturation level i is reached. `i` is a digit from 0 through 5.

`DMS141C` Message in the case of SF pubsets when the zip level is exceeded.

`DMS1400` Message for volume sets of an SM pubset. The saturation level (0 through 6) is shown in the message, 6 standing for the zip level.

Meassage `EXC044E` plays a special role, see "Handling the storage space requests" below.

The table below shows the operator actions depending on the saturation level:

Level	Effects on system	Operator measures
0	Normal system processing	None
1-3	Warning of impending saturation for a pubset or volume set	Inform systems support request the dialog partner to delete files which are not required, using the INFORM-ALL-JOBS command
4	Storage space requests by user jobs for a pubset or volume set are rejected with message `DMS0441`, see below for exceptions	Inform systems support; respond to message `EXC044E` if necessary
5	As for saturation level 4 with additional restrictions for system tasks	Inform systems support
6 (=ZIP)	As for saturation level 5 with an additional restriction for a session which is initiated with a type ZIP system start	Inform systems support After sufficient space has been made available on the volume, initiate a new session with a normal type system start

Table 21: Saturation state of a pubset

Information on the saturation state of a pubset is also supplied by the SHOW-PUBSET-SPACE-ALLOCATION command.

Handling the storage space requests

- Requests by user jobs are rejected if their fulfillment would lead to saturation level 4 being reached unless the system parameter L4MSG is assigned the value 1. In this case the following message requiring a response is output to the console for each user storage space request which would lead to saturation level 4 being reached, but not saturation level 5:

```
EXC044E    SATURATION LEVEL 4 FOR PUBSET/VOLUMESET '(&00)' EXCEEDED.
           REQUEST FOR DISK SPACE ACCEPTED NEVERTHELESS? REPLY
           (Y=YES; N=NO)
```

Operator granting of storage space requests (reply Y) should be restricted.

- Requests for the TSOS user ID are accepted regardless of the reached saturation level as long as there is still a minimum storage space reserve left for the system after fulfillment. This emergency reserve is only used for a system start in ZIP mode.
- Requests from system tasks can also be fulfilled with saturation level 4 or 5.
- Requests for creating system dumps are rejected if their fulfillment would lead to saturation level 5.
- With SM pubsets, an attempt is made to fulfill requests, which do not refer to a specific volume set in the pubset, on the volume set with the lowest saturation level (as long as it complies with the other volume set selection criteria). The request is rejected if its fulfillment would lead to a saturation level (as described above) being reached on all volume sets which are suitable for the allocation.

Saturation prevention measures

Occurrence of saturation level 4 or 5 indicates that the pubset or volume set concerned is overloaded. A pubset or volume set which continuously reaches this level is insufficiently configured.

Systems support can employ the following measures:

- Redistribute separate user IDs to other pubsets
- Enlarge the pubset or volume set by adding volumes
- Migrate unused data with the software product HSMS
- Issue regular requests to users to release storage space which is not required, and move files
- Check storage space assignment with the SPCCNTRL utility or the SHOW-PUBSET-SPACE-ALLOCATION command
- Economical use with the “storage space violation” function (ADD-USER PUBLIC-SPACE-EXCESS=*NO /*ALLOWED command)
- Timely reorganization

9.6.4 Reorganizing disk storage space

The constant creation, deletion, extension and reduction of files during active operation means that the free storage space and the files created on the volumes of a pubsets become increasingly fragmented. This fragmentation increasingly impairs file access performance and the uniform distribution of the I/O load to the volumes on the pubset. The Extents list in the catalog entries becomes longer because of the unavoidable creation of large numbers of small file extents when files are extended.

SPACEOPT

The software product SPACEOPT makes it possible to reduce fragmentation by optimally reorganizing the file extents on a pubset's volumes. The purpose of SPACEOPT is to create the largest possible continuous free storage areas on a volume so that large files can be allocated with only a small number of extents. In addition, the number of extents is reduced wherever possible by combining multiple small extents to form a single large extent. This results in improved file access performance. Large, continuous areas of free storage space are created on a pubset's volumes and the danger of an extent list overflow in the catalog entries is eliminated.

SPACEOPT is described in the manual of the same name [52].

HSMS

A pubset can be reorganized by performing a backup and subsequent restoration.
The following HSMS statement can be used to write back all the files in a continuous form and the reorganize the storage space:

```
//RESTORE-FILES REPLACE-FILES-AND-JV=*YES(REORGANIZE-SPACE=*YES)
```

If the operand RELEASE-UNUSED-SPACE=*YES is also specified, further storage space is saved: the assigned (allocated), but as yet unused, pages after the file's "last page pointer" are released.

9.7 Shared pubsets

If the product HIPLEX MSCF is used with an appropriate hardware configuration, up to 16 systems (BS2000 native or guest systems under VM2000) may have simultaneous, shared access to a pubset. This type of pubset is known as a “shared pubset”.

In addition to the shared pubset network, HIPLEX MSCF provides enhanced network functionality: the XCS network (cross-coupled system).
The XCS network provides closer coordination of the participating systems than a shared pubset network. An “XCS pubset” is used as the central storage location for all data required throughout the network. XCS pubsets are imported automatically by the system.

The complete shared pubset concept (hardware configuration, pubset management, file access, watchdog mechanism and shared pubset and XCS network) is described in detail in the “HIPLEX MSCF” manual [32].

The SHOW-SHARED-PUBSET command can be used to display information on the shared pubset.

> ¡ More than one BS2000 system with BS2000/OSD-BC V9.0 or higher can jointly access Net-Storage volumes of shared pubsets.

9.7.1 Creating a shared pubset network

The master/slave principle is used for implementing the shared pubset network. A system within the network is named as the temporary pubset owner (“pubset master”) and handles all metadata management functions centrally. All remaining systems in the network (the pubset slaves or “slave sharers”) direct their management requirements to the pubset master via MSCF functions.

Assigning the system ID

The system ID (sysid) identifies the systems in a shared pubset network. In the case of shared pubset mode the system ID must be unique in a computer network.

When a system ID is assigned which is used internally as a synonym for the BCAM name of the system, the pubset notation must be distinguished:

- With a VSN in PUB notation (see "VSN in PUB notation"), the system ID is identical to the single-character catalog ID (sysid=catid).
- With a VSN in point notation (see "VSN in point notation"), the system ID can be a numeric value between 65 and 192. The default setting on the system is 250 (i.e. it is invalid).

The system ID is assigned when the home pubset is set up with SIR via the SYS-ID operand of the DECLARE-PUBSET statement. If the home pubset exists, the system ID can be reassigned with the SET-PUBSET-ATTRIBUTES command but the change is only effective after the next system start.

Selecting the pubset master

The desired pubset master of a shared pubset can be entered via the SET-PUBSET-ATTRIBUTES command. This command can also be used to enter a backup master which then takes over the function of the pubset master if this should fail. This process is known as a master change.

In the DMS record of the SVL of the Pubres of the SF pubset and, in the case of SM pubsets, in the DMS record of the SVL of the control volume set there are two fields which provide information on the desired pubset master and the current pubset master. These fields are evaluated when the shared pubset is activated, and selection takes place in the following order:

1. If a pubset master is currently entered, this external system is taken to be the pubset master and the system of the user is then inevitably a pubset slave.
2. Notification was made during startup that the own system is to be the pubset master.
3. The pubset entered in the SVL as the desired owner is made the pubset master.
4. If none of the above conditions are fulfilled, the first system to activate the shared pubset is made the pubset master.

The SHOW-PUBSET-ATTRIBUTES command displays the settings of the desired, current and backup masters.

Setting up and clearing down a pubset network

A shared pubset network is set up in two steps:

1. The MSCF subsystem is first started on all participating systems and the necessary connections are made. At least the connection between the future pubset master and pubset slave must exist and, to cater for a possible master change, the additional connection between the backup master and pubset slave.

2. The shared pubset is activated on all participating systems. The pubset master is selected after the owner selection described above. Startup of the pubset slave can only be fully completed after it is completed on the pubset master.

A shared pubset network is cleared down implicitly when the shared pubset is exported.

Configuration changes

The configuration of the shared pubset network can change dynamically. The causes of such a change are:

- an additional pubset slave has connected itself by activating the shared pubset
- a pubset slave has disconnected itself by deactivating the shared pubset
- a pubset slave or the pubset master has failed and the master change was concluded successfully.

9.7.2 Monitoring a shared pubset and correcting errors

- Canceling pubset locks manually
- Behavior in the event of a system crash

9.7.2.1 Canceling pubset locks manually

The locks of the pubset management (pubset locks) are used in the shared pubset network to synchronize pubset reconfiguration requests with each other and with import and export jobs. Pubset locks are also used when a change to a pubset's volume configuration is to be prevented, e.g. when creating a Snapset.

Types of pubset lock

Each pubset lock is recorded either on the pubset master or one of the pubset slaves in the form of a lock entry (see "Creating a shared pubset network"). The system on which the lock's lock entry is recorded is referred to as the lock location. The lock entry includes information on the type and holder of the pubset lock (task ID and system ID).

The following types of pubset lock exist:

- PUBSET-RECONFIGURATION
  Used to protect pubset reconfiguration requests from each other and from import and export jobs. This pubset lock can be set on the pubset master and on the pubset slave.
- SHARED-EXCAT (held by export tasks)
  Prevents pubset reconfiguration requests, but permits parallel pubset locks. This pubset lock can be set on the pubset master and on the pubset slave.
- SHARED-IMCAT (held by import tasks)
  Prevents pubset reconfiguration requests, but permits parallel pubset locks. This pubset lock can only be set on the pubset master.
- SHARED-MASTER-EXCAT
  Prevents pubset reconfiguration requests, but permits parallel pubset locks of the type SHARED-EXCAT. This pubset lock can only be set on the pubset master.

The SHOW-PUBSET-LOCKS and REMOVE-PUBSET-LOCK commands (TSOS privilege) are provided to permit lock states to be diagnosed and to correct faulty lock states.

As pubset locks are internal pubset management locks, manual release should only be necessary in abnormal lock situations, e.g. after the failure of an MSCF connection in the shared pubset network. Abnormal lock situations exist, for instance, when the pubset management can determine via internal interfaces that the lock holder's task is no longer active or is in the "pended indefinitely" state.

Examples of normal lock situations

	Existing pubset lock			
Requested pubset lock	**PUBSET-RECON-FIGURATION**	**SHARED-IMCAT**	**SHARED-EXCAT**	**SHARED-MASTER-EXCAT**
PUBSET-RECON-FIGURATION	Task waiting	Not permissible	Not permissible	Not permissible
SHARED-IMCAT	Not permissible	Permissible	Permissible	Not permissible
SHARED-EXCAT	Not permissible	Permissible	Permissible	Permissible
SHARED-MASTER-EXCAT	Not permissible	Task waiting	Permissible	Task waiting

Table 22: Combinations of pubset locks on the pubset master

In the examples below the shared pubset network consists of the pubset master with the SYSID “183” and two pubset slaves with the SYSIDs “184” and “185”.

1. The following lock situation can, for example, occur when executing /MODIFY-PUBSET-PROCESSING. The command was entered on the pubset master or on the pubset slave (in the latter case the entire command is sent to the pubset master). Under the protection of a PUBSET-RECONFIGURATION lock on the pubset master, PUBSET-RECONFIGURATION locks are set on the pubset slaves, and after the processing which needs to be performed there has been completed, they are released again.

```
/show-pubset-locks pubset-id=puba
LOCK-TYPE      LOCK-LOCATION               LOCK-HOLDER-INFORMATION
               HOSTNAME  SYSID  SHARER-    TID       SYSID  BS2000
                                TYPE                       Version
*PUBSET-RECONF D017ZE15  183    *MASTER    1000004F  183    V20.0
*PUBSET-RECONF D017ZE16  184    *SLAVE     2000007F  184    V20.0
*PUBSET-RECONF D017ZE17  185    *SLAVE     3000009A  185    V20.0
```

   2000007F is the TID of a MSCF server task on pubset slave “184”.
   3000009A is the TID of a MSCF server task on pubset slave “185”.

   Comment
   The PUBSET-RECONFIGURATION locks on the pubset slave do not need to be set at all times. Depending on the status of the processing operations, these locks need not yet be set or may already have been released again.

2. The following lock situation can occur when executing /EXPORT-PUBSET which was entered on pubset slave “184”.

```
/show-pubset-locks pubset-id=puba
LOCK-TYPE      LOCK-LOCATION               LOCK-HOLDER-INFORMATION
               HOSTNAME  SYSID  SHARER-    TID       SYSID  BS2000
                                TYPE                       Version
*SHARED-EXCAT  D017ZE16  184    *SLAVE     2000007E  184    V20.0
*SHARED-EXCAT  D017ZE15  183    *MASTER    2000007F  184    V20.0
```

   2000007F is the TID of an export task on pubset slave “184”.

   Comment
   The SHARED-EXCAT locks on the pubset slave and pubset master do not have to be set simultaneously at all times. Depending on the status of the processing operations, the SHARED-EXCAT lock might only be set on one of the two systems.
   In this example no SHARED-EXCAT lock is set on pubset slave “185” because this system is not affected by the export job on pubset slave “184”.

9.7.2.2 Behavior in the event of a system crash

If a system within the shared pubset network fails, all resources reserved by it must be released or recovery measures must be initiated. All systems involved in the shared pubset network are monitored by the MSCF subsystem.

Two checking mechanisms are used for system monitoring:

- the watchdog file $TSOS.SYS.PVS.SHARER.CONTROL to which all sharers periodically write time stamps (vital-sign reports). If a sharer fails, this can be detected by one of the other sharers because of the missing vital-sign report and appropriate measures can be initiated.
- If the vital-sign report does not appear, the system connection is checked by sending a request to the sharer concerned, which must be acknowledged within a specific time period.

A partner failure is only assumed if the missing vital-sign report is confirmed by an unsuccessful network/LAN check.

If the owner system fails, a pubset-specific job variable is set on all dependent systems.

Failure of a pubset slave

If a pubset master detects the failure of a participating slave, all resources reserved by the failed pubset slave are released.

Failure of a pubset master

If the pubset master fails, the watchdog mechanism initiates a master change. A prerequisite for the master change is that an active pubset slave is entered in the shared pubset SVL as the backup master which is to take over the new master functions.

The backup master is entered in the SVL DMS record with the command SET-PUBSET-ATTRIBUTES BACKUP-MASTER=.... If no backup master is entered or the entered backup master is not active, the value of the ALTERNATE-BACKUP operand decides whether the first active pubset slave in the SVL becomes the pubset master, the operator explicitly defines one of the active slaves as the new pubset master with the command IMPORT-PUBSET SHARER-TYPE= *MASTER(MASTER-CHANGE=*YES) or whether the master change with an alternate backup master is to be prohibited.

If no backup master is foreseen or the master change fails for another reason, then one of the following actions is necessary:

- All participating pubset slaves deactivate the shared pubset and rebuilt the shared pubset network completely.
- With the command SET-PUBSET-ATTRIBUTES the permission for a subsequent master change is given and it is initiated with the command IMPORT-PUBSET SHARER-TYP=*MASTER(MASTER-CHANGE=*YES).

Possible reasons for a master change failure:

- The entered backup master is not active.
- The connection to a participating slave is interrupted.
- One of the systems involved in the shared pubset network is using a version of HIPLEX MSCF which is not compatible with the network or an incompatible revision level.

All participating pubset slaves can resume normal operation after a successfully concluded master change. The master change itself is almost completely transparent to the users.

9.7.3 Shared pubsets in the XCS network

The XCS network (cross-coupled system) provides closer coordination of the participating systems. Each system has a consistent and complete view of the total network. The XCS network thus provides mechanisms for implementing distributed applications. It is conceived mainly as a high availability and load network for BS2000. Among other things, the user is offered the following functions which are important in the DMS environment:

- Distributed lock manager (DLM)

  Implements a global system lock management, thereby supporting global system synchronization and serialization. It is the basis of SFS.

- Shared file system (SFS)

  Allows global system updating of files on shared pubsets, which are not necessarily XCS pubsets, within the XCS network. In HIPLEX MSCF, this global shared update is supported for the block or byte stream-oriented access methods UPAM, FASTPAM and DIV.

An XCS network must fulfill more extensive requirements than a shared pubset network:

- A system can only belong to one XCS network
- The participants must be fully meshed, i.e. MSCF connections must exist between all systems in the network
- At least one XCS pubset must belong to the XCS network and all systems must have access paths to it

An XCS pubset is used as the central depository for data required throughout the network. XCS pubsets are imported automatically by the system.

The SHOW-XCS-PUBSET can be used to display information on the XCS pubset.

The SHOW-XCS-OCCUPATION command can be used to display the TU tasks currently using XCS.

9.8 Special information on standby pubsets

Pubsets can be mirrored to increase data security. When used in conjunction with the software product SHC-OSD, disk storage systems provide interfaces for the creation of standby pubsets. When an error occurs, the standby pubsets can take over the functions of a failed pubset. As a result, the downtimes are kept as low as possible and the availability of the pubset or system is increased.

Standby pubsets can be created for home and data pubsets. These are referred to as **standby home pubset** or **standby data pubset**:

- If the home pubset fails, the system is no longer operable and must be restarted with the standby home pubset. The failed home pubset can then be reconstructed.
- If a data pubset fails, applications which access this pubset may no longer be operable. Users of this pubset can be enabled to resume processing activities at a certain checkpoint by importing the standby data pubset. If necessary, the failed data pubset can be reconstructed by systems support.

> **i** Standby pubsets must not be used if HSMS migration is employed in the system. Migrated data is no longer available if standby pubsets are used!

Command	Meaning
ACTIVATE-CLONE	Activate clone pair
RESTART-CLONE-SESSION	Restart clone pair
RESTORE-FROM-CLONE	Restore unit from clone unit
SHOW-CLONE-SESSION-STATUS	Display status of clone pairs
START-CLONE-SESSION	Create clone pair
STOP-CLONE-SESSION	Delete clone pair

Table 23: SHC-OSD commands for the mirroring of standby pubsets

> **i** In the context of a high-availability concept with multiple disk storage systems at various locations, data pubsets can also be mirrored to the remote disk storage system.
>
> In addition, the Snapset function, which is based on Snap functions of the disk storage systems, can also be used to generate pubset copies for backup purposes.
>
> For information on Snapsets see section "Data saving using Snapsets".For information on replication functions of external disk storage systems see the “SHC-OSD” manual [48].

9.8.1 Creating standby pubsets

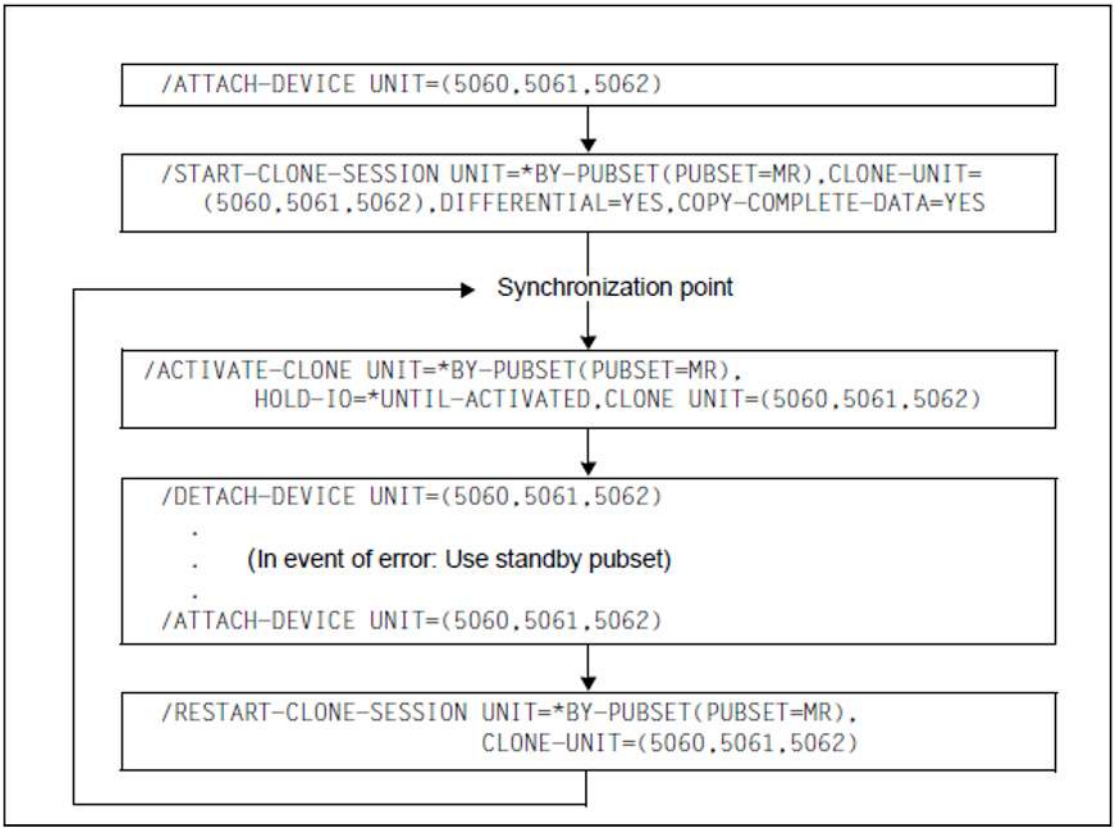
Standby pubsets are generated with the replication functions of the disk storage systems. The procedures here are analogous. A distinction must be made according to whether you are working with a standby pubset or two alternating standby pubsets.

9.8.1.1 Setting up an individual standby pubset

Procedure:

1. Attach the disks which are required for mirroring (ATTACH-DEVICE command).
2. Assign a mirror disk to each disk (unit) of the pubset (START-CLONE-SESSION command).
   As a result the data is copied from the pubset to the mirror disks and then kept synchronous.
3. Create a synchronization point.
   - For home pubsets this means:
     Write back the host buffer for the home pubset and stop or terminate applications on the home pubset (DAB write caches are not permitted on the home pubset and consequently do not need to be considered).
   - One of the following actions must be performed for data pubsets:
     - Close all files, terminate DAB write caches, and write back the host buffers of other applications. This can be achieved by briefly terminating all applications and exporting the pubset.
     - The applications define a synchronization point themselves without exporting the pubset (see the “SHC-OSD” manual [48]).
4. Split the disks and mirror disks (ACTIVATE-CLONE command)
   - If a home pubset consists of more than one disk, all the disks must be split simultaneously in order to ensure that the data remains consistent and can be reconstructed. As far as possible, metadata should be located on only one logical disk.
   - If a data pubset consists of more than one disk, the metadata (file, user, GUARDS catalogs, etc.) should, as far as possible, be located on only one logical disk. All the disks must be split simultaneously at the split time in order to guarantee data consistency and permit reconstruction.
5. Detach the mirror disks (DETACH-DEVICE command).
   This generates the standby pubset which can be used if a fault occurs.
6. Reattach the disks of the standby pubset (ATTACH-DEVICE command).
7. Initiate synchronization with the pubset again so that the data of the standby pubset is updated (RESTART-CLONE-SESSION command).
8. Continue with step 3.

Example with local mirroring using clones



Setting up alternating standby pubsets

When just one standby pubset is used, periods exist in which no standby pubset is available (during updating and renewed splitting). For critical applications it is therefore advisable to use two alternating standby pubsets which are available and updated on an alternating basis.

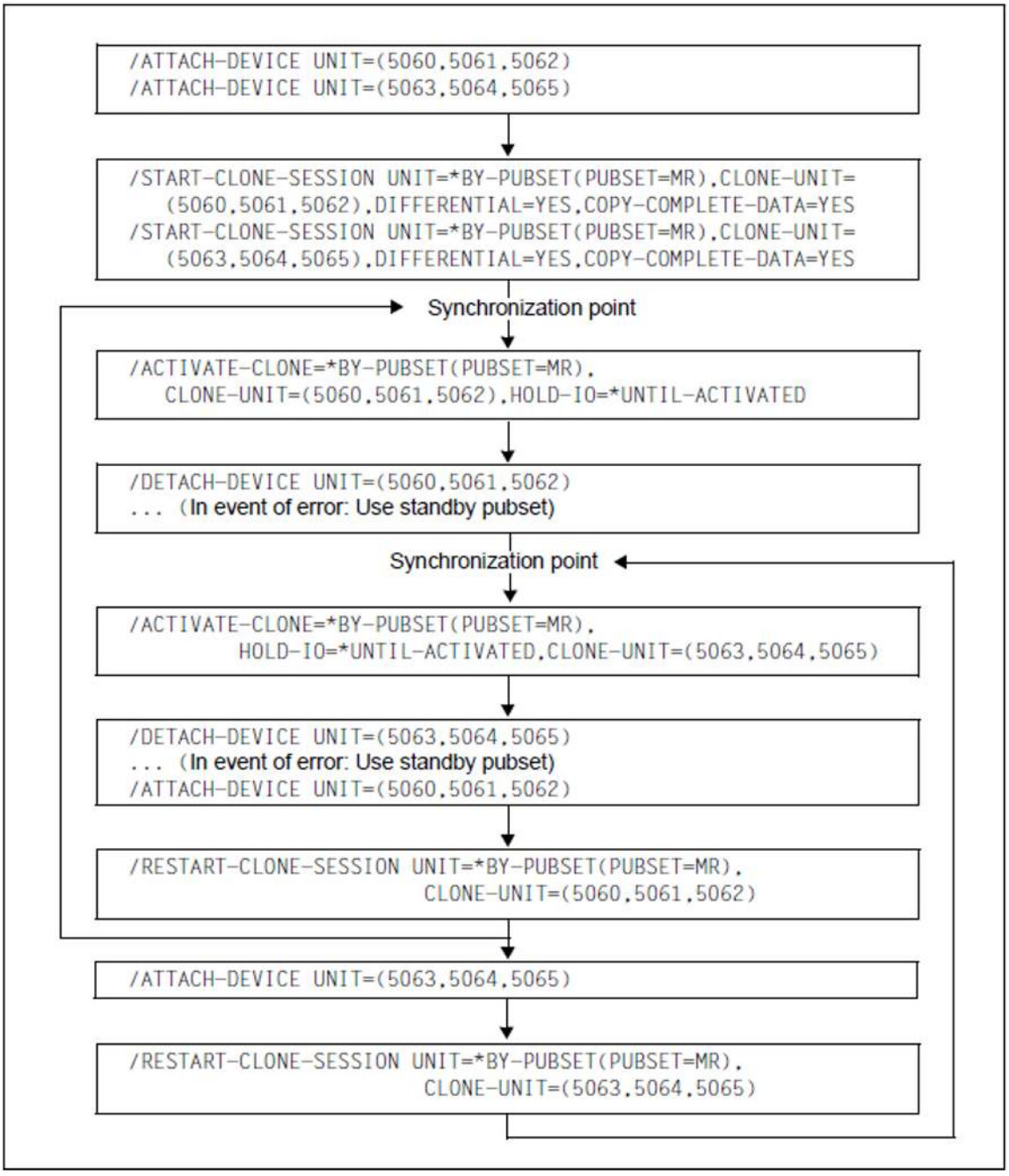
Procedure:

1. Attach the disks which are required for mirroring (ATTACH-DEVICE command).
2. Assign a mirror disk (clone unit) to each disk (unit) of the pubset twice (START-CLONE-SESSION command). As a result the data is copied from the pubset to the mirror disks and then kept synchronous.
3. Create a synchronization point for the first standby pubset.
   - For home pubsets this means:
     Write back the host buffer for the home pubset and stop or terminate applications on the home pubset (DAB write caches are not permitted on the home pubset and consequently do not need to be considered).
   - One of the following actions must be performed for data pubsets:
     - Close all files, terminate DAB write caches, and write back the host buffers of other applications. This can be achieved by briefly terminating all applications and exporting the pubset.
     - The applications define a synchronization point themselves without exporting the pubset (see the “SHC-OSD” manual [48]).
4. Split the disks and mirror disks for the first standby pubset (ACTIVATE-CLONE-UNIT command)

- If a home pubset consists of more than one disk, all the disks must be split simultaneously in order to ensure that the data remains consistent and can be reconstructed. As far as possible, metadata should be located on only one logical disk.
- If a data pubset consists of more than one disk, the metadata (file, user, guards catalogs, etc.) should, as far as possible, be located on only one logical disk. All the disks must be split simultaneously at the split time in order to guarantee data consistency and permit reconstruction.

5. Detach the mirror disks for the first standby pubset (DETACH-DEVICE command).This generates the first standby pubset, which can be used if a fault occurs.
6. Create a synchronization point for the second standby pubset.
7. Split the disks and mirror disks for the second standby pubset (ACTIVATE-CLONE-UNIT command)
8. Detach the mirror disks for the second standby pubset (DETACH-DEVICE command). As a result the second standby pubset is generated, and this has more up-to-date data than the first one and can be used if a fault occurs.
9. Reattach the disks of the first standby pubset (ATTACH-DEVICE command).
10. Initiate synchronization of the first pubset with the pubset again so that the data of the standby pubset is updated (RESTART-CLONE-SESSION command).
11. Continue with step 3 for the first standby pubset.
12. If the first standby pubset is split again, attach the disks of the second one again (ATTACH-DEVICE command).
13. Initiate synchronization of the second standby pubset with the pubset again (RESTART-CLONE-SESSION command).
14. Continue with step 6 for the second standby pubset.

Example with local mirroring using clones



Generating a home pubset for the standby system

In a high-availability network, a separate home pubset is required for the use of the standby system. This home pubset, which has the character of a standby pubset but has its own catalog ID, can also be created using the disk storage systems' local replication functions.

The following example illustrates how a home pubset with a new catalog ID is generated. The starting situation is as follows:

- The MR pubset contains a loadable BS2000 of the current version.
- The MR pubset consists of three disks (units) with the mnemonic device names 5070, 5071 and 5072.

- The active BS2000 system was started from the MR pubset.
- Three further mirror disks are available for mirroring (5073, 5074 and 5075).

Procedure

1. Generate copies for the MR pubset (clone units 5073, 5074 and 5075):

   ```
   /START-CLONE-SESSION UNIT=*BY-PUBSET(PUBSET=MR),
   CLONE-UNIT=(5073,5074,5075),DIFFERENTIAL=YES,COPY-COMPLETE-DATA=YES
   ```

2. Create a synchronization point. This means:
   Write back the host buffer for the home pubset and stop or terminate applications on the home pubset (DAB write caches are not permitted on the home pubset and consequently do not need to be considered).
3. Split or activate copies, at the same time generating the pubset with the new catalog ID WG.

   ```
   /ACTIVATE-CLONE UNIT=*BY-PUBSET(PUBSET=MR,NEW-PUBSET=WG,
   HOLD-IO=*UNTIL-ACTIVATED)
   ```

4. Call PVSREN and use the PVSREN statements to rename as required:

   ```
   /PVSREN
   . . .
   //RENAME-PUBSET-OR-VOLUME-SET NAME=MR,NEW-NAME=WG,
     SYSID=173,MODE=*COMPLETE-SHC-RENAME
   %  PVR0201 CHANGE DEFAULT CATALOG ID ENTRY 'TT' TO DEFAULT CATALOG ID
   ENTRY 'WG' IN USER CATALOG OF HOME PUBSET ? (Y=YES; N=NO)?N
   %  PVR0202 CHANGE DEFAULT CATALOG ID ENTRY 'TT' TO DEFAULT CATALOG ID
   ENTRY 'WG' IN USER CATALOG OF NEW PUBSET ? (Y=YES; N=NO)?Y
   %  PVR0206 CHANGE CATID ENTRY 'TT' IN STANDARD IMON SCI OF HOME PUBSET TO
   CATID 'WG' ? (Y=YES; N=NO)?N
   %  PVR0203 READY PUBSET IN THIS BS2000 SYSTEM ? (Y=YES; N=NO)?N
   //END
   %  PVR0145 PVSREN TERMINATED NORMALLY
   ```

For details on PVSREN see on section "Managing SYSEAM storage space on pubsets" or the complete description in the “Utility Routines” manual [15].

> The current passwords of important user IDs (in particular TSOS) should be retained since if the new home pubset is subsequently used on the standby system, the user catalog and consequently also the user passwords are “reset”.
>
> In addition, it should be noted that within procedures and programs, file accesses with an explicit specification of the catalog ID should be avoided since these can result in errors when the new pubset is used.

9.9 Managing SYSEAM storage space on pubsets

SYSEAM files temporarily store the EAM files of the users.

Temporary means that the files are only stored short-term in the SYSEAM system file. After the job terminates successfully the temporary EAM files created during the job are deleted again.

A SYSEAM file can be set up on all pubsets. The name is

`:<pvsid>:$TSOS.SYSEAM:`

`<pvsid>` designates the pubset. In this case there is exactly one SYSEAM file.

If a SYSEAM file with the corresponding name was created by systems support on the default pubset of a user, then the user accesses the SYSEAM file which is on his default pubset. However, if there is no SYSEAM file on the default pubset of a user, then the user accessed the SYSEAM file on his home pubsets.

Defining values for SYSEAM

Systems support can specify the following attributes for a SYSEAM file for a specific pubset. They apply equally to all SYSEAM files created on a pubset:

- Minimum size
- Value for secondary allocation
- Size of a cache for a SYSEAM file in class 4 memory (however, this is only used for the SYSEAM file on the home pubset).

These attributes are entered in the corresponding master catalog entry with the help of the ADD-MASTER-CATALOG-ENTRY and/or MODIFY-MASTER-CATALOG-ENTRY commands.

A SYSEAM file is created (at the latest during the first access using the EAM access method) whose size is at least as large as the minimum size defined for it. If there is not enough space anymore for a SYSEAM file, the SYSEAM file is extended dynamically by its secondary allocation value until its maximum size is reached.
Once the EAM load starts to drop again, the extra space now left in the SYSEAM file is freed dynamically piece by piece (in blocks the size of the secondary allocation value) until it reaches its lower limit.

Default setting by means of system parameters

If there are no entries stored in the master catalog entry of a pubset for the attributes of a SYSEAM file, then the system parameters EAMMIN, EAMSEC and EAMMEM are used for the minimum size, the value for secondary allocation and the size of the cache for the SYSEAM file of the home pubset. Furthermore, the space available for a single user alone in a SYSEAM file can be specified with the system parameter EAMSIZ.

They are defined in the startup parameter file in the section with the SYSOPT-CLASS2 identifier. The values of the system parameters named can be displayed with the help of the SHOW-SYSTEM-PARAMETERS command.

SYSEAM files on shared pubsets

The handling of SYSEAM files on a shared pubset is controlled by the system parameter EAMSPVS:

- either there is exactly one SYSEAM file on the affected shared pubset (SPVS) for the pubset master of the shared pubset (EAMSPVS=0). The pubset slaves participating in the shared pubset network cannot use this SYSEAM file.

- or every system participating in the shared pubset network (i.e. on the pubset master as well as on every pubset slave) can work with a SYSEAM file on this shared pubset (EAMSPVS=1).

  The name of the SYSEAM file concerned is `:<pvsid>:$TSOS.SYSEAM.<sysid>:.`

  `<sysid>` is a three-digit number and designates the identification of the importing system in the shared pubset network (“system identification”). It can be determined using the SHOW-PUBSET-ATTRIBUTES command.

  There is consequently a SYSEAM file with the suffix `<sysid>` in this case for every importing system on the pubset indicated by `<pvsid>`.

  If in this case systems support creates a file with the name `:<pvsid>:$TSOS.SYSEAM`, it is not used.

  The values for EAMMIN, EAMSEC and EAMSIZ equally apply then to all SYSEAM files created on a pubset (see above).

9.10 Using Speed Catalog Access (SCA)

SCA (Speed Catalog Access) is a software product for speeding up catalog management of SF pubsets. SCA replaces the sequential search of catalog entries by means of direct access. SCA does not change the catalog structure.

This procedure is implemented with tables:

- The freelist table provides information on free space in the catalog blocks. The freelist table is managed by routines which run under the calling task.
- The cross-reference table assigns the logical block number of the catalog block containing the entry to the file /job variable names. Irrespective of the variant set (see "SCA execution variants)"), the cross-reference table is managed either by the SCA task or by routines which run under the calling task.

The SCA tasks of different catalogs can be simultaneously started and terminated.

Using SCA enables the total system throughput to be increased and greater flexibility in data center organization:

- The runtime for programs which use the catalog intensively (e.g. ARCHIVE) and the response times of commands with a high percentage of catalog accesses (e.g. CREATE-FILE, MODIFY-FILE-ATTRIBUTES, IMPORT-FILE) can be considerably improved.
- It is no longer necessary to catalog files with frequent catalog accesses at the front of a user chain.

The following should be noted:

- Noticeable performance improvements can only be expected if the users make heavy use of the catalogs (> 60-100 file entries per user ID).
- SCA requires additional resources at runtime, such as virtual and real memory, tasks, etc.
- SCA use can be reduced to a few pubsets if the file catalog structures are optimum.

Command	Meaning
ADD-MASTER-CATALOG-ENTRY	Create an entry in the MRSCAT catalog list on the home pubset with definition of whether SCA is to be started automatically when IMPORT-PUBSET is called
ENTER-JOB	Start and terminate SCA
MODIFY-MASTER-CATALOG-ENTRY (EDIT-MASTER-CATALOG-ENTRY)	Modify an entry in the MRSCAT catalog list on the home pubset with definition of whether SCA is to be started automatically when IMPORT-PUBSET is called

Table 24: SCA command overview

SCA is installed using the installation monitor IMON and requires the following files under the TSOS ID

SYSLNK.SCA.<ver> or SKMLNK.SCA.<ver>	SCA module library for SUs /390 SCA module library for SUs x86
SYSREP.SCA.<ver>	REP file
SPEEDCAT.ENTER.START	Start procedure
SPEEDCAT.ENTER.STOP	Termination procedure
SPEEDCAT.START	Startup program
SPEEDCAT.STOP	Termination program

SCA execution variants

When SCA is started for a specific pubset, the user can specify which SPEEDCAT variant is to be loaded:

- SCA with task swapping

  A separate SCA task is set up when SCA is started. It generates the SCA tables and waits for jobs. The SCA task communicates with the user task via a double-chamber bourse, in order to exchange requests and results. Allowance must be made for performance degradation caused by task swapping with this variant. The load on class 4 memory is, however, reduced since the SCA directories are held in class 5 memory (where they are only read by the SCA task).

- SCA without task swapping

  All SCA tasks can be executed without task swapping. A separate SCA task is also set up in this case when SCA is started, but this is terminated once the tables have been built up.
  The cross-reference table must be accessible here to all user tasks and is created in class 4 memory for this reason. Access is regulated with exclusive locks via a signal bourse (single-chamber bourse) to prevent several tasks from simultaneously reading from and writing to the directories.

Starting SCA implicitly with IMPORT-PUBSET

If SCA is to be executable during IMPORT processing, this must be stored in the pubset's MRSCAT entry. Systems support can create an appropriate entry with the START-SPEEDCAT operand in the ADD-MASTER-CATALOG-ENTRY or MODIFY-MASTER-CATALOG-ENTRY command. The default value for starting SCA in the ADD-MASTER-CATALOG-ENTRY command is START-SPEEDCAT=*AUTOMATIC, i.e. SCA is to be started automatically.

Starting SCA explicitly during the session

1. Start with catalog ID specification:

   ENTER-JOB FROM-FILE=SPEEDCAT.ENTER.START,JOB-NAME=SCA<catid> where <catid> is the single-digit or multiple-digit pubset catalog ID. The SCA variant is set during response to message `DMS03FE`.

2. Start without catalog ID specification:

   ENTER-JOB FROM-FILE=SPEEDCAT.ENTER.START where input of the desired pubset catalog ID is prompted for on the console. The SCA variant is set by the additional response to message `DMS03FE`.

It should be noted that SCA should only be started for a pubset after the pubset is completely imported. The command for starting SCA during the session can also be integrated into the CMDFILE.

Terminating SCA

1. Termination with catalog ID specification:

   ENTER-JOB FROM-FILE=SPEEDCAT.ENTER.STOP,JOB-NAME=SCA<catid> where <catid> is the single-digit or multiple-digit pubset catalog ID.

2. Termination without catalog ID specification:

   ENTER-JOB FROM-FILE=SPEEDCAT.ENTER.STOP where the pubset catalog ID is prompted for on the console.

3. Implicit termination:

   The EXPORT-PUBSET and SHUTDOWN commands terminate the SCA task or release the class 4 memory used for the cross-reference table and freelist.

The command for terminating SCA can be stored in a RUN command sequence.

If the catalog ID is not specified (2), the commands may also be issued by the operator.

Error handling

SCA deactivates itself automatically and normal catalog access is reverted to if an error occurs in SCA or if hardware-related write errors in the catalog occur. A system dump is initiated, depending on the type of error.

SCA uses the SERSLOG function (see the “Diagnostics Handbook” [14]) to output diagnostic information. Error messages are also output to the console.

9.11 Detecting and rectifying hardware faults on pubsets

If a volume in an SF pubset or a volume set fails or is partially defective, this pubset or volume set can no longer be worked with continuously.

The error can only be eliminated by reinitializing the volume concerned. This generally means the loss of all data on the volume. The pubset can only be activated again after it has been completely reconstructed from the backup.
It is therefore necessary to detect such volume areas in good time to enable them to be excluded from future storage space requests. After the volume error is detected, suitable error elimination measures can be initiated.

Detecting hardware errors with DMS

With shared volumes (pubsets), the DMS checking mechanism ensures that permanent volume errors which occur during access by DMS are made known to the allocation components and excluded from further storage space assignments. This ensures that these registered volume areas are not used to fulfill any future requests for storage space and that defective volume areas do not spread to other files.
The information on defective volume areas is stored in the pubset-specific defect garbage file which is generated by the system.
The first time a volume error is registered within a pubset, the file is generated on this pubset under the following path name (dependent on the pubset type):

- for SF pubsets: :<catid>:$TSOS.SYSDAT.DEFECT.GARBAGE.<catid#>
- for SM pubsets: :<catid>:$TSOS.SYSDAT.DEFECT.GARBAGE.<volume-set-id#>

<volume-set-id#> designates the catalog ID of the volume set containing the file. If the catalog ID of the SF pubset or volume set is shorter than four characters, the <catid#> and <volume-set-id#> variables are padded to four characters.

The extent list of this file only consists of the extents which the reported defective blocks of the pubset encompass. This means that these defective blocks are allocated twice for the lifetime of the file, once in the file itself and once in the defect garbage file.

The defect garbage file may only be processed by the appropriate system components and all accesses are rejected. A file password is assigned and the file protection attribute is set to ACCESS=READ to prevent access. The defect garbage file is set to backup level E, i.e. it is not backed up by ARCHIVE or HSMS. This prevents inconsistent conditions from occurring.

As with other files, the defect garbage file extent list has a limited capacity. It can only record approximately 140 to 310 extents, depending on volume distribution, which corresponds to the maximum number of defective blocks per pubset. Message `DMS060C` is output to the console if this capacity is exceeded. If this occurs, it is imperative that the pubset is reorganized as soon as possible to eliminate the volume errors (e.g. with VOLIN).

An existing defect garbage file can only be deleted via the IMPORT-PUBSET processing of the pubset concerned. During IMPORT-PUBSET processing, a check is made to see whether a defect garbage file exists and, if it does, whether blocks have already been repaired. If they have been repaired, these blocks can be reintegrated into the normal storage space assignment process. This is done explicitly in the IMPORT-PUBSET processing via an F5 reconstruction. The defect garbage file is deleted as soon as no (registered) defective blocks exist for this pubset.

> **CAUTION!**
> The defect garbage file may, under no circumstances, be deleted manually. This must be done exclusively by the system mechanisms. Otherwise, this would result in data loss and the destruction of the F5 labels!

Double allocations between the defect garbage file and other files may occur with the F5 reconstruction of a pubset during the import process (messages DMS0461 and DMS0462). These messages are not, however, to be taken as an indication of system errors or inconsistencies in the F5 area, they only indicate defective volume areas in the allocation area of the reported files.

Error message DMS0608 may also appear on the console during pubset operation. This reports just one defective block. The following error occurs when an attempt is made to enter it in the defect garbage file: a file exists with a name structure belonging to a defect garbage file (see above) but its internal structure (catalog attributes) deviates from that of a normal defect garbage file. Message DMS0608 must be answered. An attempt can thereby be made to use the existing file as a defect garbage file. If this does not succeed, the function can be canceled, i.e. either the defective block has not been entered or the system generates a new defect garbage file and implicitly deletes the existing one.

Assigning a manual allocation lock

Allocation on defective volumes can be forbidden on a volume-specific basis for SF and SM pubsets as well as on a volume-set-specific basis for SM pubsets (see the MODIFY-PUBSET-RESTRICTIONS command).
Even when SM pubsets are imported, it is possible to prevent the allocation of volume sets with defective volumes. In this case, the volume sets can be set to the state “defect” or “in hold” (see the IMPORT-PUBSET command, operand DEFECT-VOLUME-SET or IN-HOLD-VOLUME-SET).

10 Net-Storage management

BS2000 permits UNIX file systems (UFS) to be accessed via NFS.

Files of BS2000 and of open systems can be stored and edited in the server network in the storage space released by file servers.

10.1 Overview

In BS2000 there are public and private volumes and also Net-Storage. Only disks can be public volumes. Net-Storage does not belong to the public volumes, but is used to extend them. With respect to the device type, Net-Storage in BS2000 is regarded as belonging to the family of disks.

When defining disks, DMS distinguishes between public volumes, which are combined to form pubsets, and private disks. Net-Storage volumes are used to extend pubsets.
Net-Storage volumes can be assigned to a pubset. They are not, however, a component part of the pubset (IMPORT-PUBSET is also possible when the assigned Net-Storage volumes cannot be accessed).

Each pubset contains a user catalog and its own file catalog. Specifying NET-STORAGE-USAGE=*ALLOWED in the user entry specifies that the user may occupy storage space on Net-Storage.

All volumes are identified in BS2000 by a name which may be up to six characters long. This name is referred to as the VSN (Volume Serial Number). The valid character set consists of the alphanumeric characters A..Z, 0..9 and the special characters period, @, # and $. Pubset volumes can be distinguished from other volumes by their VSN, where the VSNs of public volumes are subject to a convention: they must begin with the string “PUB” or contain a period in the third, fourth or fifth position. This VSN syntax may not be used for private disks and Net-Storage volumes.
Default names of Net-Storage volumes which are assigned to a pubset have a special format depending on the pubset notation, see the section “Notation of Net-Storage volumes” in chapter "Terms".

DMS handles storage space management on disks. Some control functions (access authorization, saturation control, etc.) can only be used on a pubset, not on private disks or Net-Storage.

After Net-Storage volumes have been assigned to a local pubset, BS2000 users can utilize them via the interfaces of DMS if this is permitted in their user entry. The available storage space on Net-Storage is then limited only by the size of the released storage space on the net server.
See the “Introductory Guide to DMS” [19].

Files on Net-Storage can be created both by BS2000 and by open systems and processed jointly by these, see section "Interoperability on Net-Storage". Details of the hardware and software requirements for this are provided in the current Release Notice for BS2000.

10.2 Terms

The following terms are used in BS2000 when working with Net-Storage:

Net server

A file server in the worldwide computer network which provides storage space (Network Attached Storage, NAS) for use by other servers and offers corresponding file server services.

Net-Storage

Storage space provided by a net server in the computer network and storage space released for use by foreign servers. Net-Storage can be a file system or also just a node in the net server’s file system.

Net-Storage is connected to a BS2000 system using the MOUNT-NET-STORAGE command. The Net-Storage (more precisely: the released directory) is actually mounted on the net client. The Net-Storage is disconnected from the BS2000 system using the UMOUNT-NET-STORAGE and unmounted from the net client.

You use the SHOW-NET-STORAGE command to request information on the Net-Storage available in the BS2000 system.

The following are approved as Net Storage systems: ETERNUS CS with NAS interface and NetApp ONTAP systems. Other net servers require special approval.

Net client

Implements access to Net-Storage for the operating system using it.
In BS2000 the net client, together with the BS2000 subsystem ONETSTOR, transforms the BS2000 file accesses to corresponding UNIX file accesses and executes them on the net server using NFS.

The net client on an SU /390 is the HNC, and on an SU x86 the carrier system X2000.

Net-Storage volume

Net-Storage volumes represent Net-Storage in BS2000 in the context of data pubsets.

As of BS2000 V21.0A, there is a second Net-Storage volume type NETVOL in addition to the previous Net-Storage volume type NETSTOR.

As of BS2000 V21.0A, the ADD-NET-STORAGE-VOLUME command is always used to set up Net-Storage volumes of the NETVOL type and assign them to a pubset. In this case a directory is configured in the released file system of the net server and assigned to a local data pubset (SF or SM) in the form of a Net-Storage volume. Existing Net-Storage volumes of the NETSTOR type will continue to be supported in BS2000 V21.0A.

Both Net-Storage volumes of type NETVOL and of type NETSTOR due to compatibility reasons are addressed by means of their Volume Serial Number (VSN) and the type NETSTOR. The directory name in the net server’s released file system and the VSN of the Net-Storage volume are the same.

A Net-Storage volume (more precisely: the directory with the name of the Net-Storage volume on the net server) contains the following files (see also the next page):

- A file system label (file name .FSL)
- The users' BS2000 files
- User-specific directories with the users' node files

Information on the Net-Storage available in a data pubset can be requested using the SHOW-PUBSET-NET-STORAGE command.

Notation of Net-Storage volumes

Irrespective of the pubset type, the Net-Storage volume's VSN is by default derived from the pubset name:

- In the case of a one-character pubset name, PUB notation (e.g. `PUBA00`) is assumed and the Net-Storage volume's VSN is formed by replacing the "U" in the "PUB" string by "@" (e.g. `P@BA00`).
- In the case of a multiple-character pubset name, point notation (e.g. `OTTO.0, ABC.00`) is assumed and the Net-Storage volume's VSN is formed by replacing the period by "@" (e.g. `OTTO@0, ABC@00`).

Only the standard Net-Storage volume in the pubset can be named using this notation.

Multiple Net-Storage volumes for a pubset

When multiple Net-Storage areas on one or more net servers are to be assigned to a pubset, the Net-Storage volumes must be assigned with different VSNs. As only one Net-Storage volume with a standard name can exist per pubset, further Net-Storage volumes must be assigned to the pubset with explicitly defined VSNs (as with private disks).

Net-Storage file

Designates a file which is created on a Net-Storage volume. On Net-Storage a distinction is made between the two file types BS2000 file and node file.

BS2000 file

Designates a file which is created and processed exclusively by BS2000. BS2000 files on Net-Storage (FILE-TYPE=BS2000) are supported. They reside directly on a Net-Storage volume. Open systems may only access them in read mode.

Node file

Designates a Net-Storage file (FILE-TYPE=NODE-FILE) which can be created and processed both by BS200 and by open systems. Node files are supported since BS2000 OSD/BC V10.0. They reside on a Net-Storage volume in a user-specific directory (name of the user ID), and the file names comply with the BS2000 naming conventions. BS2000 supports PAM node files and SAM node files.

10.3 Interoperability on Net-Storage

Both BS2000 and open systems can connect to the same Net-Storage, create node files there and process these alternately:

- BS2000 can create and process node files on Net-Storage - open systems can also process these node files
- Open systems can create and process node files on Net-Storage - BS2000 can import and also process these node files

This is referred to as **interoperability**.

Requirements

The following additional requirements must be satisfied for node files to be processed jointly:

- Net server: ETERNUS CS with NAS interface and NetApp ONTAP systems.

  > ¡ Other net servers require a special release.

- The POSIX user attributes UserID (USER-NUMBER, <uid>) and GroupID (GROUP-NUMBER, <gid>) must be entered in the pubset's user management for each BS2000 user who wishes to work with node files (see "Managing Net-Storage in BS2000").
- When Net-Storage is connected via NFSv4, the users must be entered in a directory service (openLDAP or AD) with the corresponding UserID and GroupID. The directory service must be configured on the net client and net server. When connected via NFSv3, it is not necessary to use a directory service.

Procedure

Node files are created in user-specific directories in BS2000 in the UNIX file system (UFS), see "Managing Net-Storage in BS2000".

BS2000 supports the following node file types:

- PAM node files (BLKCTRL=NO)
- SAM node files (BLKCTRL=DATA, RECFORM=V).
  SAM node files are used for text based processing.

  While the data are written from BS2000 to Net-Storage, the SAM structures are deleted from the data stream and, if desired, the code is converted from EBCDIC to ASCII (ISO).

  While the data are read from Net-Storage to BS2000, the SAM structures are inserted and the data converted to EBCDIC, so that the SAM access method encounters the familiar SAM data structures.
  The conversion is based on the settings the user has chosen for the coded character set and net coded character set of the file when he created the file. If the user has made no specifications, the default values are taken from the user entry in the user catalog of the pubset on which the file is being catalogued. These default values are taken from the HOSTCODE or NETCODE parameters whenever a new user is created.

Node files which are created by open systems during ongoing operation must be imported before they are used in BS2000. See the command description of IMPORT-NODE-FILE in the "Commands" manual [27].

Information on node files is provided by the LIST-NODE-FILES command.

Node files can be removed from the BS2000 file catalog using the EXPORT-NODE-FILE command.

General conditions

General conditions must be complied with to enable interoperability on Net-Storage, see section "Interoperability".

10.4 Connecting BS2000 to Net-Storage

BS2000 has access to external storage systems over Fibre Channel and a Storage Area Network (SAN).

With the support of Net-Storage, BS2000 also offers access to net servers and the Network Attached Storage (NAS) connected to them.

On SUs /390 the net servers are accessed via the net client HNC.
On SUs x86 X2000 performs this role.

Information on the NAS servers which have been released is provided in the current Release Notices.

10.5 Access to Net-Storage

The following systems can access Net-Storage:

- BS2000: full access to all imported BS2000 files and node files
- BS2000 as of V21.0A: access to Net-Storage type NETVOL
- Open systems: full access to node files, read access to BS2000 files

10.5.1 Access from BS2000 to Net-Storage

The BS2000 user accesses a file on Net-Storage via DMS interfaces and the net client on the net server in the following steps:

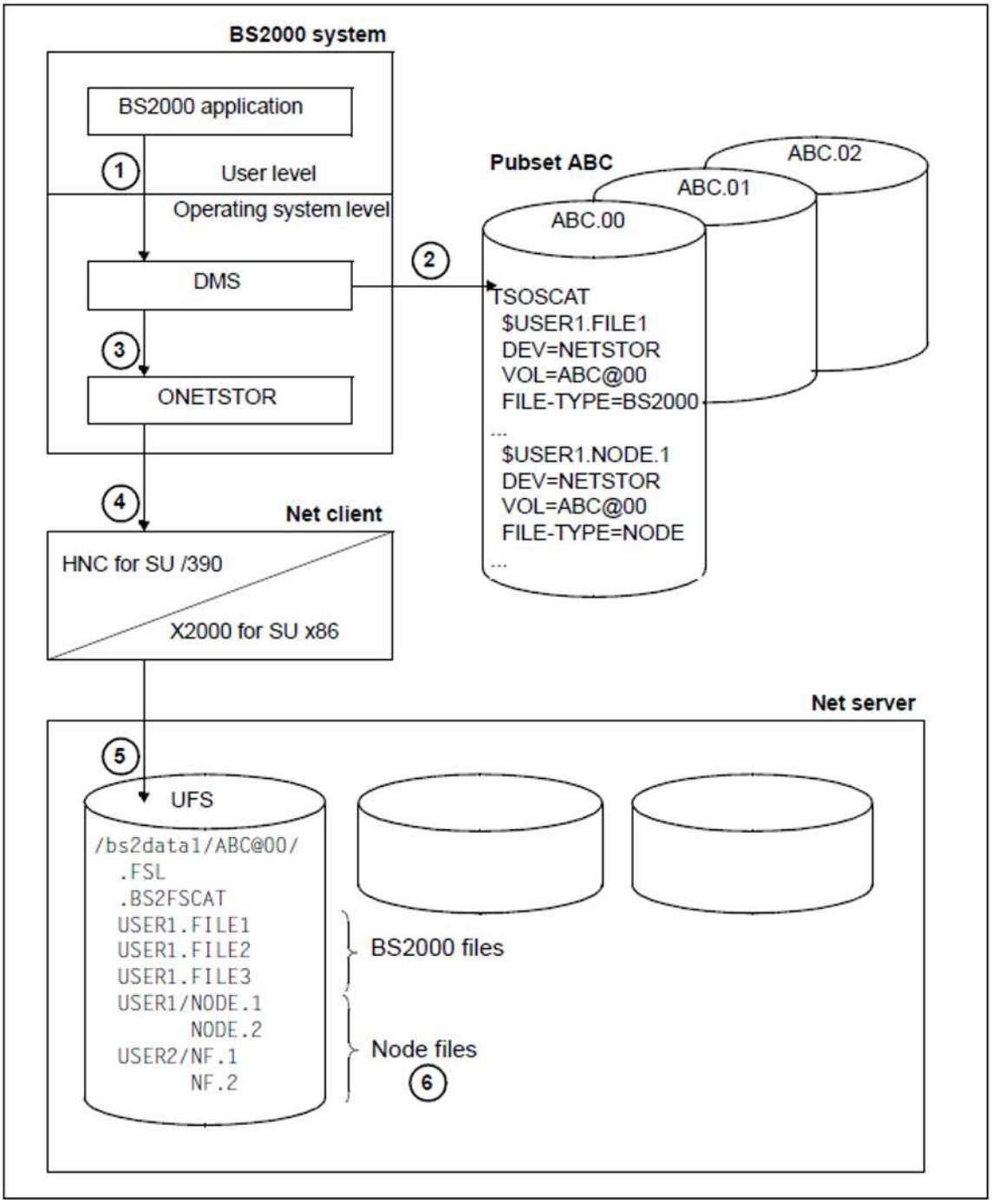


Figure 14: BS2000 access to Net-Storage of type NETSTOR

1. A BS2000 application under the user ID USER1 wishes to access the FILE1 file which resides on Net-Storage and is cataloged in the ABC pubset. This is done via the normal user interfaces of DMS.

2. DMS checks whether the file exists in the user and file catalogs of the ABC pubset. On the basis of the file attributes DEVICE=NETSTOR and VOLUME=ABC@00, DMS recognizes that the file concerned is contained in the ABC@00 directory on the Net-Storage released by the net server (see the section "Connecting Net-Storage to BS2000").
3. The FILE1 file is actually accessed via the BS2000 subsystem ONETSTOR and the net client.
4. The BS2000 subsystem ONETSTOR transforms the BS2000 file access to the corresponding file access in the UNIX file system (UFS) and forwards it to the net client.
5. The necessary preparations must be made on the net server and net client, see section "Preparations on the net server and net client". On the net client the `/bs2data1` directory (with the ABC@00 subdirectory) released by the net server is mounted, see section "Connecting Net-Storage to BS2000". File access takes place via NFS in the net server's UFS.
6. When a node file is accessed, e.g. the file NODE.1, DMS recognizes that a node file (FILE-TYPE=NODE) in the ABC@00 directory on the Net-Storage released by the net server is involved. Node files are contained in a user-specific directory which has the name of the user ID, in this case directory USER1. This directory is created automatically when BS2000 creates the user's first node file.
   To permit a user to work with node files, the POSIX user attributes UserID and GroupID must be entered in the pubset's user entry. For user USER1, for instance, `1005:100` is entered (see figure 15 in chapter "Access of open systems to Net-Storage").

10.5.2 Access of open systems to Net-Storage

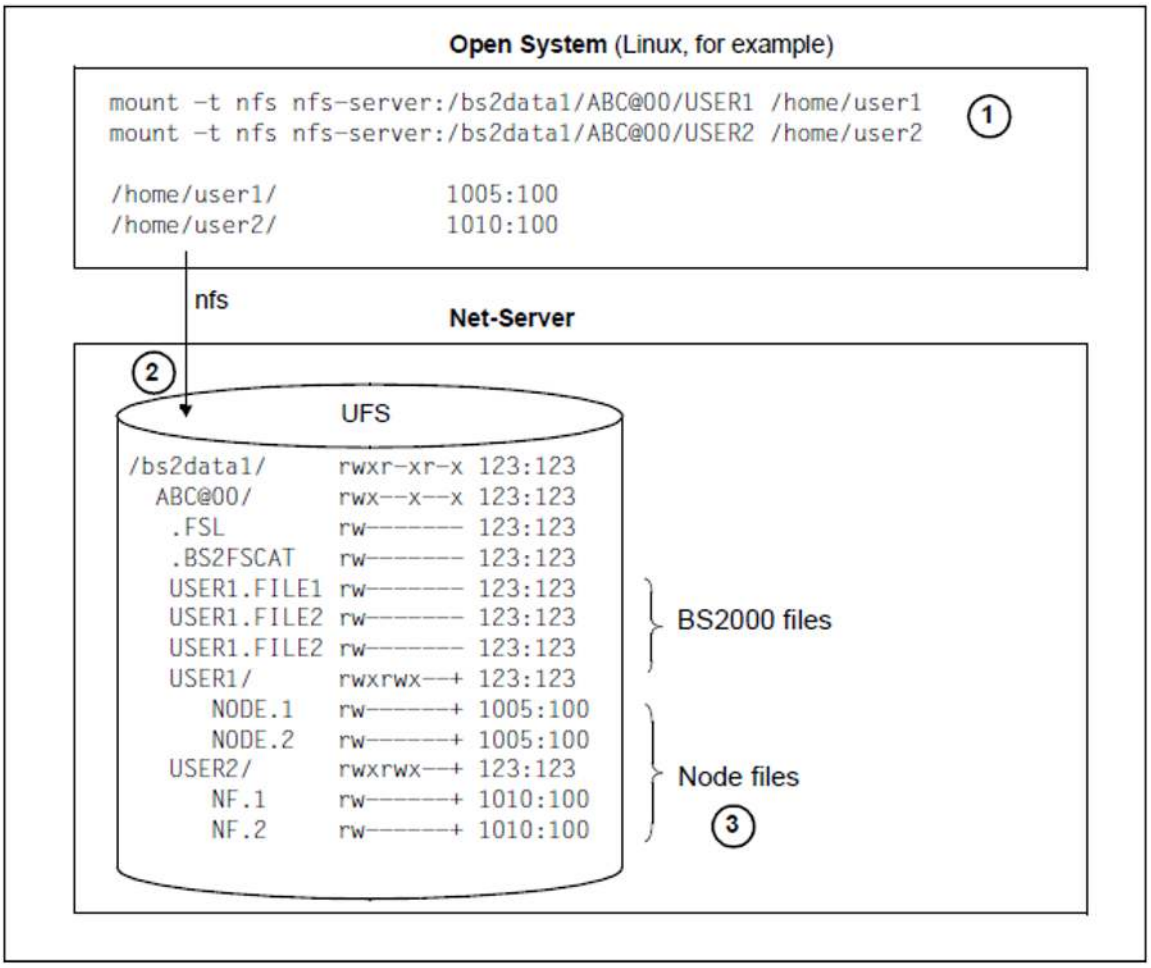


Figure 15: Access of open systems to Net-Storage of type NETSTOR

1. Open systems are granted access to BS2000 files and node files on the net server by means of mount commands.
2. When the user-specific directories are created, the POSIX ACLs are also provided with the necessary information by BS2000. Under Linux, the users `user1` and `user2` in the example above therefore obtain access to the directories USER1 and USER2.
3. To grant a user read and write access both from BS2000 and from the open system, the same UserId and GroupId must be entered for him/her on both systems. The same applies for creating and deleting files.

> When NFSv4 is used, this must be ensured by means of a directory service, e.g. openLDAP or Active Directory, to which the NFS server and the open system are connected.

10.6 Preparations on the net server and net client

The Net-Storage connection must be configured on both the net server and the net client.

The net server's system administrator must assign the directory which has been released for mounting to a user with UserID and GroupID (owner). This user has authorization to access the released directory.

Detailed information on this topic can be found in the configuration description of the net server.

> ¡ Write authorization is always required for BS2000 access.

The instance tasked by BS2000 to access the directory released in the net server is the bs2netsagent in the net client.

The SE server's administrator must use the SE Manager on the net client to set the access authorization (user ID and group ID) of the bs2netsagent so that the net server permits access to the released directory. Detailed information on this is provided in the “Operation and Administration” manual [57].

As the bs2netsagent can only be assigned one UserID and GroupID, on the net servers this one UserID and GroupID must be assigned to all directories released for this net client.

> ¡ When implementing maintenance work (e.g. software update or reinstallation) and configuration changes, make absolutely certain that Net-Storage which is operated via the net client concerned is beforehand disconnected from all connected BS2000 systems using UMOUNT-NET-STORAGE.

10.7 Connecting Net-Storage to BS2000

In order to be able to use Net-Storage in BS2000, systems support (following the preparations on the net server and net client) must initially connect the BS2000 system to the Net-Storage and create the Net-Storage volumes and assign them to a local pubset.

- Connect the BS2000 system to the Net-Storage, e.g. with

```
/MOUNT-NET-STORAGE DIRECTORY='/bs2data1',
                   SERVER=*IP-ADDRESS(IP-ADDRESS=172.10.1.110),
                   CLIENT=system1
```

  This command performs the following processing steps:

  - Mount the released directory on the net client
  - Record the released directory, i.e. all existing 6-character directory names below this directory are added to the NDM tables as volumes
  - If necessary occupy the volume(s)

Two options are available to you for creating and assigning Net-Storage volumes:

1. Create a new Net-Storage volume and assign it to a local pubset, e.g. with

```
/ADD-NET-STORAGE-VOLUME VOLUME=*STD, PUBSET=A,
                        DIRECTORY='/bs2data1',
                        SERVER=*IP-ADDRESS(IP-ADDRESS=172.10.1.110)
```

   While in earlier BS2000 versions Net-Storage volumes of the NETSTOR type were created, as of the current BS2000 V21.0A only Net-Storage volumes of the NETVOL type are set up. The Net-Storage volume is now available for use in BS2000 operation as a volume with the default VSN `P@BA00` of the specified pubset. A new `P@BA00` subdirectory is created in the Net-Storage's `bs2data1` directory. The file system label has also been created. Files can now be written to the Net-Storage volume.

2. Add an existing directory to a Net-Storage volume for BS2000 operation, e.g. with

```
/ADD-NET-STORAGE-VOLUME VOLUME=MYNETS, PUBSET=A,
                        DIRECTORY='/bs2data1',
                        SERVER=*IP-ADDRESS(IP-ADDRESS=172.10.1.110)
                        IMPORT=*YES(OLD-VOLUME-NAME=*SAME)
```

   The `MYNETS` subdirectory in the `bs2data1` directory on the net server is now available for use in BS2000 operation as a Net-Storage volume of the specified pubset. The `MYNETS` directory still contains the management data and the BS2000 files created on it. With type NETVOL Net-Storage volumes, the CEs are already cataloged in TSOSCAT after the Net-Storage volume has been successfully imported. However, in case of Net-Storage volumes of type NETSTOR (created under BS2000 < V21) these have not yet been imported.
   Thus import the BS2000 files of the Net-Storage volume (type NETSTOR only), e.g. with

```
/IMPORT-FILE SUPPORT=*DISK(VOLUME=MYNETS, DEVICE-TYPE=NETSTOR,
                           PUBSET=A, FILE-NAME=*ALL)
```

The files in the MYNETS directory have now been entered in the catalog of pubset A and can be used. This applies for BS2000 files. It also applies for node files when they have already been entered in the volume's . BS2FSCAT file catalog.

The IMPORT-NODE-FILE command can be used to create a catalog entry for node files which are not yet contained in the .BS2FSCAT catalog entry.

10.8 Managing Net-Storage in BS2000

Systems support can manage Net-Storage via the DMS interfaces. To permit this, the Net-Storage must be connected to the BS2000 system.

- Net-Storage volumes are managed in the context of the data pubsets.
- You use the SHOW-NET-STORAGE command to obtain information on the Net-Storage available in the BS2000 system.
- You use the SHOW-PUBSET-NET-STORAGE command to obtain information on the Net-Storage volumes assigned to a pubset.
- You use the SHOW-NET-STORAGE-OCCUPATION command to obtain information on the tasks of the local system which occupy the Net-Storage.
- Information on the directories which are released by a net server for a net client and BS2000 is provided by the LIST-NET-DIRECTORIES command.
- The SET-NET-CLIENT-ALTERNATE command is used to specify the assignment of net clients for high availability. A symmetrical relationship between two net clients is created which ensures that if one of the two net clients fails the other net client takes over the Net-Storage connections. The assignment is displayed using the SHOW-NET-CLIENT-ALTERNATE command. It can be canceled with SET-NET-CLIENT-ALTERNATE.
- After the Net-Storage has been connected to the BS2000 system, all users of the pubset can use it because the default setting in the user catalog is `NET-STORAGE-USAGE=*ALLOWED`. The space occupied on a Net-Storage volume is not counted toward the user's public space limit.

  Permission for users to create and edit the files on the pubset's local disks can be withdrawn, as can permission for them to create and edit files on the pubset's Net-Storage volumes, e.g. with

  ```
  /MODIFY-USER-ATTRIBUTES ...,NET-STORAGE-USAGE=*NOT-ALLOWED
  ```

  The SHOW-USER-ATTRIBUTES command displays this setting.
- Speed Catalog Access for Net-Storage of type NETSTOR (SCANET) supports high-speed access to information in the file catalog (file name `.BS2FSCAT`) with the metadata stored on Net-Storage.
  You are recommended to start the SCANET subsystem (e.g. in the CMDFILE) when Net-Storage of type NETSTOR is used or planned to be used. Net-Storage of type NETVOL does not use SCANET.
- For users who wish to work with node files, the POSIX user attributes UserID and GroupID must be entered in the pubset's user catalog, e.g. with the following command:

  ```
  /MODIFY-POSIX-USER-ATTRIBUTES USER-ID=<bs2000-userid>; PUBSET=<cat-id>;
                                USER-NUMBER=<uid>; GROUP-NUMBER=<gid>
  ```

  These IDs must be coordinated with the users of the open system who want to work with the same node files.
- The IMPORT-NODE-FILE command imports individual node files to the BS2000 system. The user can specify whether the file to be imported is to be processed as a PAM or SAM file.
- Information on node files is provided by the LIST-NODE-FILES command. Here only those files are displayed which comply with the BS2000 naming conventions for files.
- Node files can be removed from the BS2000 file catalog using the EXPORT-NODE-FILE command.
- Before the shutdown in the BS2000 system is initiated, the pubsets to which Net-Storage volumes have been assigned must be exported (see also the section "Disconnecting Net-Storage from BS2000").

Reconfiguring Net-Storage volumes

Systems support can reconfigure Net-Storage volumes in BS2000, i.e. remove them from a pubset and, without deleting the files on the Net-Storage volume, assign the volumes to another pubset.

- Remove the Net-Storage volume from the (imported) pubset, e.g. with

  ```
  /REMOVE-NET-STORAGE-VOLUME ...,FILES-ON-VOLUME=*EXPORT
  ```

  The files' catalog entries are then deleted only in the local pubset. The files and management data on the Net-Storage are retained.
  With Net-Storage type NETVOL, a file catalog (file name .SCAN_TSOSCAT_CES) is created.
- Assign the Net-Storage volume to another pubset, e.g. with

  ```
  /ADD-NET-STORAGE-VOLUME ...PUBSET=B,IMPORT=*YES(...)
  ```

  The management data on the Net-Storage volume is updated. If the Net-Storage volume to be assigned was created under BS2000 <V21, it is a volume of the NETSTOR type. This can simultaneously be converted into a Net-Storage volume of the NETVOL type: ..., `IMPORT = *YES (CONVERT = *TO-NETVOL)`. A reverse conversion to the NETSTOR type is not possible!
- With Net-Storage type NETVOL, all CEs are cataloged and the file catalog (file name .SCAN_TSOSCAT_CES) is deleted.
- Create the catalog entries for the files on the Net-Storage volume of type NETSTOR in the "new" pubset, e.g. with

  ```
  /IMPORT-FILE SUPPORT=*DISK(VOLUME=<net-storage-volume>,
                             PUBSET=B,DEVICE-TYPE=NETSTOR,FILE-NAME=*ALL)
  ```

  The files of the Net-Storage volume are now entered in the catalog of pubset B and can be used.
- You can also simulate the import in advance, e.g. with:

  ```
  /CHECK-IMPORT-DISK-FILE VOLUME=P@BA00,DEVICE-TYPE=NETSTOR,...
  ```
- For Net-Storage of type NETSTOR:
  In certain cases, e.g. when rectifying inconsistencies after a Net-Storage has failed, it can make sense to export individual files from the pubset:
  To do this a Net-Storage volume or the storage type *NET-STORAGE must be specified in EXPORT-FILE. (Otherwise EXPORT-FILE will only apply for files on private disks.)

  ```
  /EXPORT-FILE ...,SELECT=*BY-ATTRIBUTES(STORAGE-TYPE=*NET-STORAGE,...)
  ```

  Correspondingly you can import files which are not yet known in BS2000, i.e. create a catalog entry in the local pubset, e.g. with:

  ```
  /IMPORT-FILE SUPPORT=*DISK(VOLUME=P@BA00,DEVICE-TYPE=NETSTOR,
                             FILE-NAME= ... )
  ```

10.9 Disconnecting Net-Storage from BS2000

Net-Storage can be disconnected from the BS2000 system using UMOUNT-NET-STORAGE. Applications which are running can then no longer access files residing on Net-Storage.

Before the shutdown in the BS2000 system is initiated, the data pubsets to which Net-Storage volumes are assigned must be exported. This ensures that no further open files exist on the net server.

BCAM may be terminated only after the pubset has been exported. If BCAM is terminated beforehand, open files on Net-Storage (including the catalog) can no longer be closed and the connection between the net client and net server cannot be cleared. In this case the Net-Storage volume can only be accessed again when another pubset import is performed after the message `DMS1326` has been answered.

Net-Storage can also be disconnected during ongoing BS2000 operation, e.g. when the net server is switched:

- Disconnect the Net-Storage from the BS2000 system, e.g. with

  ```
  /UMOUNT-NET-STORAGE DIRECTORY='/bs2data1',
                  SERVER=*IP-ADDRESS(IP-ADDRESS=172.10.1.110),
                  CLIENT=system1
  ```

  The command is executed only if none of the Net-Storage is still occupied. You can check whether it is occupied by the local system using the SHOW-NET-STORAGE-OCCUPATION command. SHOW-NET-STORAGE-OCCUPATION shows all the tasks of the local system which are occupying storage space for a mount ID. For shared pubsets, too, only the tasks of the local system which occupy storage space are output.

  In an emergency you can force the UMOUNT-NET-STORAGE command to be executed with the FORCE=*YES option: the Net-Storage is also disconnected if tasks are still occupying it. However, inconsistencies can occur as open files are not closed properly.

You also use the UMOUNT-NET-STORAGE command to unmount the released directory on the net client.

10.10 General conditions

- BS2000 in general
- Shared pubsets
- Disk mirroring
- Interoperability
- Net-Storage-Volumes from BS2000 V21

10.10.1 BS2000 in general

The following general conditions must be observed in BS2000 when working with Net-Storage:

- A Net-Storage volume can only be assigned to one data pubset. No Net-Storage can be assigned to the home pubset.
- A file on Net-Storage consists of just one extent. Specification of a volume list is accepted, but only the first volume is used.
- Storage space assignment can be specified in the SPACE operand of the CREATE-FILE and MODIFY-FILE-ATTRIBUTES commands, but no storage space of this size is occupied on Net-Storage. The (maximum) storage space assignment takes place only when the file is stored in the corresponding size.
- With respect to allocation, a Net-Storage volume behaves like an NK2 disk with the minimum allocation unit of 8 KB.
- Absolute allocations are not possible for files on Net-Storage.
- Files with the following attributes **cannot** be created on a Net-Storage volume:
  - Files with a PAM key
  - Work files
  - Temporary files
  - File generation groups
- An SMPGEN run is aborted if pubsets with files on Net-Storage are edited. Catalog entries of files on Net-Storage must be removed or exported from the pubsets concerned before the SMPGEN run takes place, e.g. with

  `/REMOVE-NET-STORAGE-VOLUME ...,FILES-ON-VOLUME=*EXPORT`
- When a pubset to which Net-Storage volumes are assigned is renamed using PVSREN, any volume which exists is also renamed accordingly with the standard name (P@BX00, XY@000, etc.). Net-Storage volumes with user-defined names are not renamed.
- The administrator of the file server can grant access to public files on Net-Storage for applications which run on different operating systems. The following must be borne in mind here:
  - Applications of foreign operating systems (not BS2000) have only write access to files of the type *NODE-FILE.
  - Data access control in BS2000 is effective only if access takes place from BS2000. Otherwise it has no effect.
    Example
    Setting up shareability of a file in BS2000 with `/MODIFY-FILE-ATTRIBUTES <file name>,USER-ACCESS=*ALL-USERS` is only valid in BS2000. For a foreign system, access is still restricted.

10.10.2 Shared pubsets

Several BS2000 systems can jointly access Net-Storage volumes from shared pubsets.

All sharers must connect themselves to the corresponding Net-Storage (MOUNT-NET-STORAGE).

In the shared pubset network a new Net-Storage volume can be created or deleted only on the master computer. If a Net-Storage volume was created on the master, this Net-Storage volume must also be made known and assigned on each slave using the ADD-NET-STORAGE-VOLUME command. Only then can the Net-Storage volume also be accessed from a slave system computer (IN-USE status). This assignment is retained until the pubset is exported.

Subsequently the Net-Storage volume is immediately available again (provided a connection exists to the Net-Storage), also when IMPORT-PUBSET is issued again on a slave computer.

When the master is changed, the system updates the master SYSID entered on the associated Net-Storage volumes. However, this is only possible for available Net-Storage volumes (status: IN USE). If a Net-Storage volume was not available when the master was changed (not connected or not occupied (status: ONLINE)), the first time the volume is accessed message `DMS1326` is displayed on the console of the master computer and must be answered. For Net-Storage volumes of the NETVOL type, the master can also be a BS2000 <V21.0, since with this type the master does not need to have access to the Net-Storage volume itself.

Net-Storage volumes of the NETVOL type are not supported in BS2000 <V21.0A. Under BS2000 versions <V21, the data from Net-Storage volumes of the NETVOL type cannot be backed up or restored to Net-Storage volumes of the NETVOL type. On the other hand, it is possible to restore files that were backed up from volumes of the NETVOL type under BS2000 V21 to volumes of the NETSTOR type. To do this, the appropriate volume must be specified under NEW-SUPPORT (see HSMS manual instructions). This is particularly important in shared pubset operation, since HSMS jobs are usually sent from the slave system to the master or backup server and processed there.

10.10.3 Disk mirroring

Pubsets on which Net-Storage files are cataloged can be mirrored using DRV or SHC-OSD. However, the Net-Storage files themselves are not mirrored because they are stored on a different medium. After the mirroring has been terminated, files on Net-Storage can therefore not be reached via a split mirror.

A distinction must be made between the following cases:

- Splitting the mirror without renaming

  The original pubset is imported again. Net-Storage files can still, as before, be created, deleted or edited using the original pubset.

  When the split mirror is synchronized again with the original pubset, the catalog entries of the Net-Storage files are also implicitly updated.

  It must be ensured that when the mirrors are split no changes are made to the management entries of the Net-Storage volumes. In other words the pubset which is imported first obtains access to the Net-Storage concerned (provided a connection exists to the Net-Storage).

  If the split mirror is imported on another system, the answerable message DMS1326 is issued when the assigned Net-Storage volumes are accessed because the volume contains a foreign SYSID. When the message is answered with “yes”, access to the volume is permitted. It must be borne in mind that if the volume is actually assigned in another system, answering the question incorrectly can lead to inconsistencies and loss of data.

  If you plan to process the split mirror in this way, you are recommended to proceed as follows:

  1. Remove the Net-Storage volumes which were assigned to the pubset:

     ```
     /REMOVE-NET-STORAGE-VOLUME ..., FILES-ON-VOLUME=*EXPORT
     ```

  2. Export the pubset and disconnect the mirror.
  3. Import the required pubset, original or mirror.
  4. Assign the Net-Storage volumes to the imported/active pubset again:

     ```
     /ADD-NET-STORAGE-VOLUME ..., IMPORT=*YES
     ```

  5. Add the catalog entries of the files of the Net-Storage volume to the TSOSCAT of the imported pubset again using IMPORT-FILE.
  6. The split mirror now contains no further Net-Storage files and can be processed further on another system.

- Splitting the mirror with renaming

  You are recommended to remove the Net-Storage volumes from the pubset before splitting the mirror:

  ```
  /REMOVE-NET-STORAGE-VOLUME ..., FILES-ON-VOLUME=*EXPORT
  ```

  After the mirror has been split, the Net-Storage can once again be assigned to the desired pubset mirror:

  ```
  /ADD-NET-STORAGE-VOLUME ..., IMPORT=*YES
  ```

  The files must subsequently be imported from the Net-Storage volume.

  If the files are to be permanently available on Net-Storage, the following procedure can be used as an alternative:

  1. The mirror is split using a DRV or SHC-OSD statement and at the same time renamed. When the mirror is split, the Net-Storage volume entries on it are implicitly declared invalid.
  2. When this pubset is imported, the Net-Storage volume entries and the catalog entries of the associated Net-Storage files are automatically deleted.
  3. The pubset can now either continue to be used either without Net-Storage or new Net-Storage volumes can be created. Existing Net-Storage volumes can also be placed back in service:

     ```
     /ADD-NET-STORAGE-VOLUME ..., IMPORT=*YES
     ```

  4. If the Net-Storage files and volumes of the original pubset are to be added to the split and reorganized mirror, these must be exported from the original beforehand:

     ```
     /REMOVE-NET-STORAGE FILES-ON-VOLUME=*EXPORT
     ```

10.10.4 Interoperability

The following additional general conditions must be observed for Node-Files:

- Only files of the type *NODE-FILE, the so-called node files, are permitted for interoperability with open systems.

  > ! Files of the type *BS2000 must not be changed by open systems!

- Node files are structureless. They end at byte boundary.
- As seen from BS2000, only the following files can become node files:
  - PAM files without a PAM key (BLKCTRL=NO)
  - SAM files with block control information and variable record length (BLKCTRL=DATA, RECFORM=V)
- Node files are created in user-specific directories. These are generated automatically by BS2000 the first time a node file is created and supplied with the necessary POSIX ACLs. Subsequently authorized users can access the directory and the node files it contains from both BS2000 and from open systems.
- Access rights and protection attributes in BS2000 can also be set for node files as usual (access right (ACCESS), shareability (USER-ACCESS), access control, passwords, GUARDS, coownership), see, for example, the MODIFY-FILE-ATTRIBUTES command in the "Commands" manual [27]. They apply for BS2000, but not for open systems.
- Node files which are created by open systems must be imported before they are used in BS2000 using command IMPORT-NODE-FILES.
- The BS2000 command LIST-NODE-FILES informs a user of the node files in his/her user-specific directory.
- The owner of the files is the respective BS2000 user under whom the files are cataloged in BS2000 or in the TSOSCAT of the pubset to which the Net-Storage volume is assigned.
  In this context, it must be ensured that a meaningful user and group number is entered in the user catalog of the pubset using MODIFY-POSIX-USER-ATTRBIUTES:
  - Users from open systems receive access to node files using their user and group number (uid, gid). The pubset-specific POSIX user and group number of a user must be coordinated with the uid, gid of the same user in the open world so that access from systems in the open world is possible. This can be ensured by a directory service (e.g. openLDAP, AD).
  - Make sure that different users in BS2000 receive different and unique user and group numbers. If Net-Storage is configured on multiple pubsets, the user and group numbers of the same user must be identical in the different user catalogs (see SHOW or MODIFY-POSIX-USER-ATTRIBUTES).
- In UFS, node files have the access rights `rw- --- ---`.
  These are the minimum access rights which BS2000 requires for a user of this type. A user of the open system may change these access rights. A BS2000 user may not do this.
- Users from open systems obtain access to node files through their user and group numbers (uid, gid).
- BS2000 access rights are managed using BS2000 tools alone. Access rights from the perspective of open systems that go beyond the standard must be managed using the tools of the open systems, i.e.:
  If a user of an open system has extended the access rights for his group, for example: chmod 660 <file-name>, this cannot be reversed by means of BS2000.

10.10.5 Net-Storage-Volumes from BS2000 V21

As of BS2000 V21.0A, there is a second Net-Storage volume type NETVOL in addition to the previous Net-Storage volume type NETSTOR.

As of BS2000 V21.0A, only Net-Storage volumes of the NETVOL type are created with the ADD-NET-STORAGE-VOLUME command. As before, a directory is set up in the shared file system of the net server and assigned to a local data pubset (SF or SM pubset) in the form of a Net Storage volume.

Existing Net-Storage volumes of the NETSTOR type are still supported in BS2000 V21.0A. You can continue to use them as usual.

When creating files, the user specifies the NETSTOR volume type as before in order to create a file on Net-Storage:

```
/CREATE-FILE FILE-NAME=:NET1:$UA.NET-FILE.1,VOLUME=NEWNET,DEVICE-TYPE=NETSTOR
```

In the following example, the specified Net-Storage volume is of the NETVOL type (see DEVICE-TYPE):

```
/sh-file-attributes :NET1:$UA.NET-FILE.1,inf=*par(alloc=*yes)
%0000000004n:NET1:$UA.NET-FILE.1
%  ------------------------------ ALLOCATION   ------------------------------
%  SUPPORT    = PUB         S-ALLOC    = 4           HIGH-US-PA = 0
%  EXTENTS     VOLUME     DEVICE-TYPE      EXTENTS     VOLUME     DEVICE-TYPE
%      1       NEWNET       NETVOL
%  NUM-OF-EXT = 1
%:NET1: NET   :      1 FILE  RES=         4 FRE=         4 REL=         4 PAGES
```

Net-Storage volumes are treated as disk devices, so specifying NETSTOR, STDDISK or D3435 in the DEVICE operand of the CREATE-FILE command leads to the same result if the user specifies the VSN of the data carrier.

For files that are allocated on NETSTOR type net volumes, NETSTOR is output under DEVICE-TYPE; for files on Net-Storage volumes of the NETVOL type NETVOL is output under DEVICE-TYPE respectively (see example above).

Net-Storage volumes of the NETSTOR type can be converted into Net-Storage volumes of the NETVOL type. For this purpose, the volumes must not be assigned to a pubset (previous REMOVE-NET-STORAGE-VOLUME ... FILES-ON-VOLUME = * EXPORT required). The conversion function is contained in the ADD-NET-STORAGE-VOLUME command:

```
/ADD-NET-STORAGE-VOLUME VOLUME=NET001, PUBSET=NET1, ... , IMPORT=*YES(CONVERT=*TO-
NETVOL)
```

> ! The conversion cannot be reversed! After the conversion, BS2000 systems <V21 can no longer access the respective Net-Storage volumes and the files stored on them. See also the information in the section on shared pubsets.

Net-Storage volumes can be detached from a pubset with REMOVE-NET-STORAGE-VOLUME VOLUME = ..., FILES-ON-VOLUME = * EXPORT and later attached to another pubset . In the case of Net-Storage volumes of the NETVOL type, the files contained are automatically imported into the TSOSCAT of the pubset. If file name conflicts occur during the import, the import is canceled and any catalog entries that have already been created are removed. In this case the system administrator must clean up the pubset and repeat the ADD-NET-STORAGE-VOLUME. (In the case of Net-Storage volumes of the NETSTOR type, the files must be imported explicitly with the BS2000 command IMPORT-FILE.)

Net-Storage volumes of the NETVOL type are not supported in BS2000 <V21.0A. When SHOW-NET-STORAGE is output, these are marked as NOT-SUPPORTED ("NO SUP").

```
/show-net-storage
************************************************************************
- NET-DIRECTORY: /nas/200/fgBS2    (MOUNT-ID = 1)
------------------------------------------------------------------------
-   NET-SERVER:  IPV4: 172.17.11.111
-   NET-CLIENT:  NETCLH3
-                IPV4: 192.168.153.12
------------------------------------------------------------------------
-   NET-VOLUMES: NET1@0    (IN USE, PUBSET = NET1)
-                NEW0SM    (NO SUP, PUBSET = SM  )
-                NETK00    (ONLINE, PUBSET = K   )
************************************************************************
```

11 Job and task management

This chapter describes job management, which handles all the waiting jobs, and task management which handles all running jobs (tasks).
The JMP program (Jobpool Management Program) uses the information stored by JMS concerning accepted batch jobs in the batch pool to reconstruct ENTER-JOB commands. The section "PCS: Performance Control System" discusses the relationships between job and task management on the one hand and the performance monitoring system on the other.

A basic distinction can be made between the following user jobs:

- dialog jobs
- batch jobs (ENTER job or SPOOLIN)
- SPOOLOUT jobs

Except for SPOOLOUT jobs, all user jobs are initiated with the SET-LOGON-PARAMETERS command and terminated with the EXIT-JOB command. SPOOLOUT jobs are either initiated with the PRINT-DOCUMENT command or made available for output upon task termination.

A job is assigned to a job class by job management and placed in the appropriate queue. As soon as a job has been accepted it receives a task sequence number (TSN) by which it can be addressed during its time in the system. Job management controls and manages all jobs.

A job becomes a task when system resources (CPU, storage, devices) are assigned to it. Task management sets up a task control block (TCB). Task management controls and manages all tasks.

The Job Management System (JMS) is responsible for accepting and handling user jobs.
The main functions of JMS are as follows:

- job acceptance and checking of access authorization
- job selection for start clearance
- job initialization

These jobs are implemented by the components of job management (job classes and job streams).

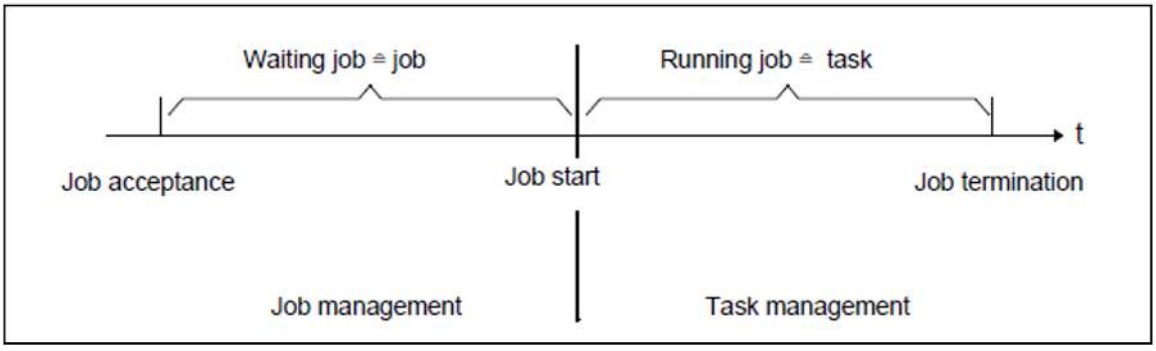


Figure 16: Relationship of job and task

11.1 Job management

Job management enables a differentiated and thus more economical job control. This effect mainly relies on the following two key features of job management:

Subdivision of jobs into job classes.

Job classes describe a job profile, combining jobs that share certain characteristic traits.

Assignment of job classes to job schedulers.

Systems support couples the job classes with up to 16 job schedulers, which may pursue different strategies and which determine which job is to be started next. System job class $SYSJC is permanently assigned to system job scheduler $SYSJS.

Normally all jobs released by the job schedulers are immediately passed to task management by the class scheduler and started.

After an overload or memory saturation situation has been cleared, preference is given to jobs of the class whose optimum has not yet been reached and which is the most urgent (depending on the number of jobs already running in the job class and the class weight). The class scheduler represents the interface to task management. It decides whether or not a job that has been released for execution is actually started.

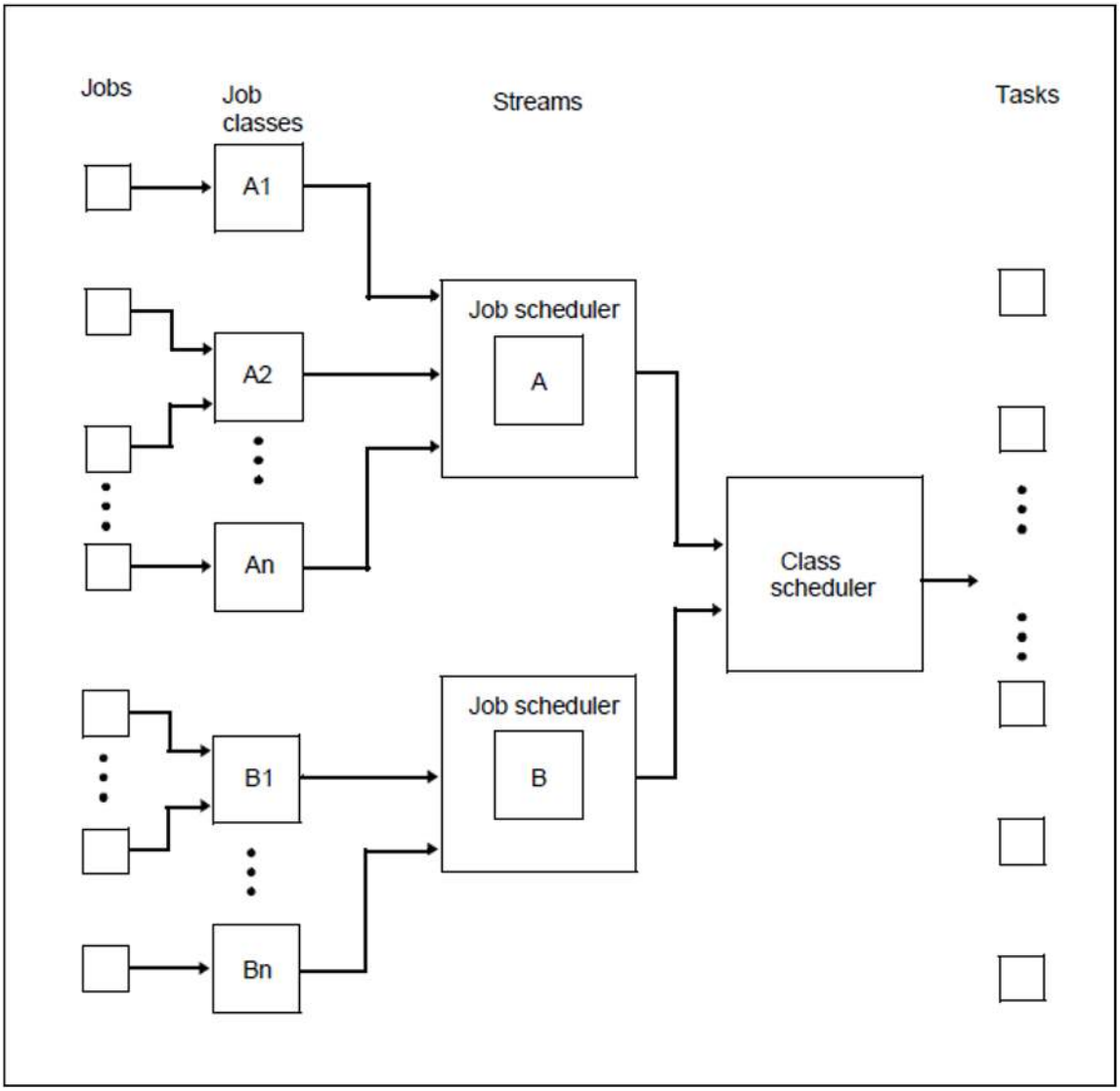


Figure 17: Job management components

JMS support for the central calendar

It is possible to specify the start times for batch jobs via symbolic data from the BS2000 central calendar. This has resulted in the introduction of a new type of batch job, the calender job. The following three types are defined:

Scheduled job	a one-off batch job whose start time is specified by the user.
Repeat job	a job which is repeated and whose repetition cycle depends on a repeat attribute specified by the user.
Calender job	a scheduled job with the start attribute *AT(...), but which can be repeated according to specifications in a calendar. A calendar job is thus also a repeat job with a time limit imposed by specifications in the calendar (or in the LIMIT operand of the ENTER-JOB and MODIFY-JOB commands).

The first execution and subsequent repetitions of a calendar job are handled equally by the system. The system does not distinguish - as it does for repeat jobs - between the “current” and “next image” of a calendar job. All runs of a calendar job have the same TSN. This allows better identification and monitoring of the job. Each individual run is managed by the job scheduler until the start time is reached: Once the preceding run has terminated, the next one is set up (same TSN) and passed on to the scheduler with the start attribute *AT(...) and the next start time of the symbolic date from the calendar.

The possible specifications for the start time depend on the stipulations of the central calendar (e.g. the repetition cycle is based only on days and not on hours). For further details, see the “CALENDAR” manual [7].

In contrast to repeat jobs, calendar jobs can be monitored with the aid of monitoring job variables (MONJVs). The MONJV remains open for the entire lifetime of the calendar job. Coupled with the fact that the TSN remains the same, this guarantees access to the job via the MONJV at any time.

Failed repetitions of a calendar job are repeated with the expired start date - in much the same way as the simple scheduled job. Afterwards the next start time is ascertained from the calendar.

The LIMIT operand of the ENTER-JOB and MODIFY-JOB commands can be used to limit the lifetime of a calendar job. Limits might be exceeded due to missing repetitions or delays in the scheduler, for example, but this is permissible. The lifetime can also be limited by setting a run counter. Once the limit is reached, the entire repeat job is terminated.

Command	**Meaning**
CANCEL-JOB [1]	Cancel user job
CHANGE-TASK-PRIORITY [1]	Change the job or task priority of an interactive, batch or print job
ENTER-JOB	Start the command sequence stored in an ENTER file as a batch job, specifying its job class and job priority
ENTER-PROCEDURE	Start the command sequence stored in a procedure as a batch job
EXIT-JOB	Cancel the task
FORCE-JOB-CANCEL	Cancel user job
HOLD-JOB	Put a user job in the wait state
HOLD-JOB-CLASS	Put a job class in the wait state
HOLD-JOB-STREAM	Put a job stream and job scheduler in the wait state
MODIFY-JOB [1]	Change the attributes of a user job
MODIFY-JOB-CLASS	Change the limits and weighting of job classes
MODIFY-JOB-OPTIONS	Change the values entered with ENTER-JOB, SET-LOGON-PARAMETERS and MODIFY-JOB-OPTIONS

MODIFY-JOB-STREAM	Change the execution priority of the stream task and stream-specific parameters

MODIFY-JOB-SWITCHES	Change the settings of job switches
MOVE-JOBS	Export or import job descriptions
RESUME-JOB	Cancel the wait state of a user job
RESUME-JOB-CLASS	Cancel the wait state of a job class
RESUME-JOB-STREAM	Cancel the wait state of a job stream and a job scheduler
SET-LOGON-PARAMETERS	Start an interactive or batch job, specifying the job class and/or job priority
SHOW-JOB-CLASS	Output information on job class attributes; this supports output in S variables
SHOW-JOB-OPTIONS	Request information about the values set with ENTER-JOB, SET-LOGON-PARAMETERS and MODIFY-JOB-OPTIONS
SHOW-JOB-STATUS [1]	Request information about a user job
SHOW-JOB-STREAM	Request description of all job streams
SHOW-SYSTEM-STATUS	Request information on job classes and job streams
SHOW-USER-ATTRIBUTES	Request information about authorization for the job classes
SHOW-USER-STATUS [1]	Request information on groups of user jobs
START-JMP	Start program for the reconstruction of ENTER-JOB commands
START-JOB-STREAM	Start a job stream and a job scheduler
STOP-JOB-STREAM	Terminate a job stream and a job scheduler
Macro	**Meaning**
DJINF	Create a DSECT or data area for the JINF macro
ENTER	Process an ENTER job
JINF	Request job information
JMGDJP	Create a DSECT or data area for the JMGJPAR macro

JMGJPAR	Request job parameters
JOBINFO	Request job information for selected jobs

JSATTCH	Attach a job scheduler to JMS
JSDETCH	Detach a job scheduler from JMS
JSEXPCT	Request information about JSS events
JSINFO	Request information about stream parameters
JSRUNJB	Transfer a job for starting
JSWAKE	Initiate a timer event for the job scheduler
LGOFF	Terminate job
SWITCH	Request information about and modify user and job switches
TMODE	Request information about job attributes

Table 25: Overview of commands for job management

[1] With these commands it is also possible to process batch jobs which have been created by the calling user ID but which run under a different user ID

11.1.1 Concept of job classes

The use of job classes enables systems support to classify user jobs.

The job class concept satisfies various demands.

1. The definition of job classes and their allocation to job schedulers permits the job mix to be optimized (e.g. many short-running jobs, few long-running jobs). This promotes **well-balanced system utilization**. In addition, it is possible to distribute job management privileges among users by issuing authorizations for access to job classes.
2. An **additional access** control instrument is provided by specific job class authorizations which systems support grants to users at its discretion. If the user does not specify a job class in the SET-LOGON-PARAMETERS or ENTER-JOB command, the user job is included in a standard job class. Systems support can define, modify or delete job classes. This can be done with the aid of the JMU utility routine, which is described in the “Utility Routines” manual [15].
3. Since the user requirements for jobs differ, the job schedulers must ensure differentiated treatment within job classes.

The description of a job class is based on a variety of parameters. The number of potential combinations of job characteristics is very large, and theoretically a separate job class could be defined for each of these combinations. However, such an approach is bound to have an extremely adverse effect on transparency.

Therefore systems support should define the job classes in accordance with the criteria dominating the day-to-day production runs.

Using the JMU routine (DEFINE-JOB-CLASS statement), systems support can define the following attributes and declarations for a job class:

- job class name
- responsible stream or default stream
- urgency (weighting) of the job class
- maximum number of jobs that can execute concurrently within the job class
- number of jobs which should ideally run in the job class
- job type
- job and task scheduling priority
- authorization for the starting of repeat jobs
- repetition cycle for repeat jobs
- maximum CPU time to be used
- start attributes

Whereas the **job scheduling priority** influences the starting of the job, the **task scheduling priority** affects the execution of the started job (= execution priority).

The possible values for job management are priorities 1 through 9, and for task management priorities 30 through 255.

The lower the value, the higher the priority.

The term **job type** refers to the difference between batch jobs and interactive jobs. Job classes may be defined for either job type, with the restriction that interactive job classes are not subject to job scheduling. Here attention is paid only to compliance with the class limits, i.e. the number of interactive jobs held in a job class and the access authorization for this job class are checked.

Example 1

Job classification according to CPU time used:

- Job class JCSHORT
  for jobs using no more than 5 CPU seconds
- Job class JCNORMAL
  for jobs using no more than 500 CPU seconds
- Job class JCLONG
  for jobs using more than 500 CPU seconds

Example 2

Job classification according to the start time:

- Job class JCEXPRES
  for jobs with the start attribute IMMEDIATE
- Job class JCNORMAL
  for jobs with no specific start time
- Job class JCTERMIN
  for jobs to be started at a specific point in time (date/time of day), i.e. scheduled jobs

In addition to the job classes for user jobs, there is a predefined system job class called $SYSJC for system jobs. This system class should not be available to users, as $SYSJC allows all job types and does not impose any limitations on class attributes.

The definition, modification and deletion of job classes may take place in two different ways:

1. The **static** definition is stored in the $TSOS.SJMSFILE. This file is the basis of every session. It is generated and managed using the JMU utility routine.

   The following JMU statements are available:

   ```
   DEFINE-JOB-CLASS
   MODIFY-JOB-CLASS
   DELETE-JOB-CLASS
   GRANT-JOB-CLASS-ACCESS
   SET-JOB-CLASS-DEFAULT
   SET-POSIX-JOB-CLASS-DEFAULT
   ```

   If systems support modifies the attributes of a job class or alters the allocation of a user ID to a job class in the $TSOS.SJMSFILE these changes will not take effect until the next session is started.
2. The **dynamic** definition refers to the current session only and is implemented via the command interface. It is also managed using the JMU utility, where the editing mode `SET-MODIFICATION-MODE=*SYSTEM` must be set. The same JMU statements are available as above. If `SET-MODIFICATION-MODE=*ALL` is used, the changes are also made to the file $TSOS.SJMSFILE.

In addition, you can make modifications via the command interface. The following commands enable systems support to react promptly to overload situations without having to change the $TSOS.SJMSFILE.

```
HOLD-JOB-CLASS
MODIFY-JOB-CLASS
RESUME-JOB-CLASS
```

11.1.2 Job streams, job and class scheduler

The job classes are coupled with different job schedulers, using the JMU utility routine (see the “Utility Routines” manual [15]).

Each job scheduler executes within a separate job called the “job stream”.

In accordance with the stipulated job scheduling strategies, the job schedulers decide which of the jobs is to be released for execution.

Job streams, and therefore also job schedulers, control the selection of user jobs by means of specific job scheduling algorithms.

Example

Most data centers have certain central production tasks:

1. Jobs involving the compilation, linkage and loading of programs do not require any special resources.
2. Jobs from end-user departments are usually time-consuming, high-priority jobs with high resource demands.
3. The data center creates a number of jobs for administrative purposes which run at specific times and must be repeated.

Hence the obvious approach is to install three job streams (job schedulers) with appropriate job scheduling strategies for these three production tasks:

for 1) A FIFO-based job scheduler (FIFO = First In, First Out) is chosen for jobs from the programming department.
Jobs for program compilation, linkage, etc. may be combined into job classes that are assigned to a job scheduler which is active during the night hours only.
This serves to lighten daytime workloads in the data center.

for 2) For jobs from end-user departments, a job scheduler based on priority and usage of resources is the best choice.

for 3) An algorithm covering scheduled jobs is available for data center jobs.

By means of the JMU statement DEFINE-JOB-STREAM systems support specifies the criteria on which the job stream is to be based and in accordance with which the job scheduler will control the jobs. For example, the job priority or the required CPU time can be specified as the criterion with the aid of the S-PAR operand.

A job stream is characterized by a few other qualities in addition to the scheduling algorithm:

- A job stream need not be continuously active. For example, systems support has to decide whether a job stream should be active immediately after BS2000 loading or only during the night. Jobs read in for an inactive stream are collected in a special queue (TYPE1/DO) until the stream is active and the job scheduler takes over management.
- During job stream definition a task scheduling priority is set which governs the stream from the start. Since each job stream runs in its own job, this ensures an unambiguous order of precedence. This order can be altered by means of the MODIFY-JOB-STREAM command.
- A job stream manages batch jobs only. Interactive jobs are not subject to job scheduling; they are started if and when the user is authorized to start interactive jobs in the relevant job class and the job class has not yet reached the specified limit.

The definition of job streams, and hence job schedulers, may take place in two different ways:

1. The **static** definition is stored in the file $TSOS.SJMSFILE and is the basis of every session. It is generated and managed using the JMU utility routine.

   The necessary JMU statements are:

   ```
   DEFINE-JOB-STREAM
   DELETE-JOB-STREAM
   MODIFY-JOB-STREAM
   ```

   Changes made to this file do not take effect until the next session is started.

2. The **dynamic** definition refers to the current session only and is implemented via the command interface. It is also managed using the JMU utility, where the editing mode `SET-MODIFICATION-MODE=*SYSTEM` must be set. The same JMU statements are available as above. If `SET-MODIFICATION-MODE=*ALL` is used, the changes are also made to the file $TSOS.SJMSFILE.

   In addition, you can make modifications via the command interface.

   The following commands enable systems support to react promptly to overload situations without having to change the definitions stored in the file $TSOS.SJMSFILE.

   ```
   HOLD-JOB-STREAM
   MODIFY-JOB-STREAM
   RESUME-JOB-STREAM
   START-JOB-STREAM
   STOP-JOB-STREAM
   ```

The name of the ENTER file which defines the job stream job is stored in the file $TSOS.SJMSFILE. In principle, this may be any name, but it is recommended for reasons of consistency that SYSENT.JOBSCHED.nnn is used (nnn is the identifier for the operating system version).

The job scheduler is called up from this file by a START-EXECUTABLE-PROGRAM, which specifies its name. Again, any name may in principle be used for the job scheduler; but on grounds of consistency, for which systems support is responsible, the name SYSPRG.JOBSCHED.nnn is recommended (nnn is the identifier for the operating system version).

Agreed logon and logoff procedures are not performed for the stream ENTER job (for information on logon/logoff procedures, see the “SDF Dialog Interface” manual [43]).

There are three kinds of scheduler with different functions:

- the system/emergency job scheduler $SYSJS
- the standard job schedulers
- the class scheduler

11.1.2.1 System job scheduler

In contrast to the standard scheduler, the system/emergency job scheduler is a constituent part of the $SYSJS job stream, which is coupled with the operating system.

It executes in the TPR state and manages only jobs that are assigned to the system class $SYSJC. Its scheduling strategy, which is based on the LIFO (Last In First Out) principle, cannot be changed, i.e. the START parameter is ignored.

This is an advantage above all when errors occur when the job management system is started and when errors are displayed when the $TSOS.SJMSFILE or
$TSOS.SJMSFILE.WORK file is opened or accessed.

It is thus always possible to reconstruct the files after an error and start the corresponding jobs.

The following commands can also be used to process the $SYSJC system class and the $SYSJS system job scheduler.

```
HOLD-JOB-STREAM
HOLD-JOB-CLASS
RESUME-JOB-STREAM
RESUME-JOB-CLASS
SHOW-JOB-STREAM
SHOW-JOB-CLASS
```

The $SYSJS and $SYSJC definitions are constant and not addressable by the JMU routine.

The system job scheduler permits jobs to be started under the TSOS user ID at any time during the session and independently of the standard job scheduler. Unless systems support has defined further job classes for the users, the $SYSJC system job class is the standard job class for all users, which means that all user jobs are then processed by the $SYSJS system job scheduler.

11.1.2.2 Standard job scheduler

The standard job scheduler decides which of the jobs of the assigned job class is to be released for execution.

For this purpose the standard job scheduler employs a certain job scheduling strategy, which is defined by means of the JMU utility routine (static definition) and can be
dynamically changed using the MODIFY-JOB-STREAM command. Up to 16 standard job schedulers with differing scheduling strategies may be installed at any given time. A standard job scheduler executes under a separate job (job stream). These are user programs in the TU state under the system administration ID.

If the control capabilities of the standard job scheduler are not sufficient for the data center, a self-developed job scheduler may be used (see section "Data center job management").

In either case the scheduler is started under the system administration ID. This takes place either automatically or with the START-JOB-STREAM command, depending on what was specified for the START operand in the stream definition of the scheduler in the SJMSFILE.

Job scheduling parameters

During stream definition with the JMU statement DEFINE-JOB-STREAM, the operand STREAM-PARAMETER (or S-PAR) is used to define scheduler-specific parameters.

This operand is ignored by BS2000; it is interpreted only by the job scheduler and provides the basis for calculating the scheduling algorithm.

For the standard job scheduler, the following parameters are specified with the S-PAR operand:

S-PAR =	'JOB-PRIORITY = YES / NO
	,CPU-TIME = NO / YES
	,WAIT-TIME = NO / YES
	,JOB-QUOTA = 1 / <integer 1..255>
	,LOGGING = YES / NO
	,CATID-LIST = (catid1,...)
	,CAT-TIME = min
	,MSG-OUTPUT = YES / STD'

Notes

1. The definition variants of the first three parameters correspond to the scheduling strategies listed in table 26.
2. From the jobs, those jobs are selected for which the lowest precedence value is determined in the calculation of the scheduling algorithm. The number of jobs selected depends on the JOB-QUOTA parameter. The job scheduler then attempts to start these jobs in accordance with their sequence in the queue. Following the start, jobs are again selected from the set of jobs. This process is repeated until all the jobs have been started or the job class limit is reached.
3. The LOGGING parameter controls output of a log for the standard job scheduler. In addition, it defines whether messages generated by the job scheduler are to be output on the console.

   These messages include:

- JMS0302:Error on execution of an ASSIGN-SYSFILE command
- JMS0303:Log aborted owing to lack of storage space
- JMS0304:Unrecoverable error
- JMS0305:Invalid stream parameter
- JMS0306:Job stream waiting for specified pubsets to become available
- JMS0307:Job stream terminated owing to missing pubsets
- JMS0308:Job stream waiting for the GET-TIME subsystem to become available

4. The parameter CATID-LIST defines the public volume sets which must be imported before the job scheduler is started up. The parameter is interpreted only at stream
   startup time; it delays starting of the job scheduler until all the pubsets specified under CATID-LIST are available. If the CATID-LIST parameter is omitted, the job scheduler is started irrespective of which pubsets have been imported.
   The CAT-TIME parameter is used to control this waiting time. If a pubset cannot be imported within the time specified via CAT-TIME, the job scheduler will not be started. If CAT-TIME is omitted from the stream definition, the system waits for however long it takes to import the required pubsets. The time specification for CAT-TIME is in minutes.
5. The MSG-OUTPUT parameter controls the behaviour of the job scheduler when outputting messages. If the parameter is set to the value YES, then JMS messages generated by the job scheduler are output on the console even if logging is turned off (LOGGING = NO). If the parameter is set to the value STD, the message is output as before depending on the LOGGING parameter.
6. Systems support can change the S-PAR setting dynamically via the command interface (see the MODIFY-JOB-STREAM command in the “Commands” manual [27]).

Job scheduling algorithm

The job schedulers pass the jobs they have released for execution to the class scheduler. The sequence in which the jobs are released for execution is calculated with the following formula:

$M= (S^a * P^c * R^d) / (W^b + S^{(a*b)})$

where:

M	Precedence
S	CPU time (in seconds) that the job has requested
P	Job scheduling priority
W	Waiting time (in minutes) of job after job acceptance
R	Time (in minutes) remaining at job start until the latest starting time is reached. This is based on the assumption that the start attribute START=WITHIN or START=LATEST is use
a,b,c	Job scheduler parameters (may assume the value 0 or 1) defined by systems support during stream definition. The following parameters correspond to the following variables: • CPU-TIME to variable a • WAIT-TIME to variable b • OB-PRIORITY to variable c
d	d=1 if the job is supplied with the start attribute START=WITHIN or START=LATEST by the user. Otherwise d=0 applies.

> The smaller the value for M, the better the position of the job, irrespective of the job class.

If the same value M is calculated for several jobs, the sequence of the jobs that are released for execution is determined on the basis of the FIFO principle.

The various scheduling procedures resulting from different combinations of the exponential values are described on the next few pages. The values for exponents a, b, c are assigned by means of the S-PAR operand in the stream definition.

In the table, abbreviations are used for the scheduling strategies. A description follows the table.

Scheduling strategy	CPU-TIME (a)	WAIT-TIME (b)	JOB-PRIORITY (c)
(1) FIFO	NO	YES	NO
(2) HPF	NO	NO	YES
(3) HPA	NO	YES	YES
(4) SJF	YES	NO	NO
(5) SJP	YES	NO	YES
(6) HRN	YES	YES	NO
(7) HRP	YES	YES	YES

Table 26: Scheduling strategies

(1) *Selection by arrival time (FIFO)*

This strategy is recommended when the CPU time requests of the jobs to be released do not differ greatly. If they do differ considerably, the exclusive use of this strategy causes time-consuming jobs to be given preference.

Selection of jobs with the option START=WITHIN or START=LATEST in the ENTER-JOB command (FIFO and R (waiting time)):

The STREAM parameter WAIT-TIME=YES is evaluated on the basis of the FIFO selection principle. If the jobs arrived in the system with the start option START=WITHIN, the jobs will not be included in the selection process until the start interval has commenced. For jobs with START=LATEST the probability of being started increases as the latest desired start time approaches.

With WAIT-TIME=YES, FIFO cannot be guaranteed if a stream start is executing and a TSN overflow has occurred. If two jobs were started in the same minute and a TSN overflow then occurred, the job with the TSN 0AAA is started before the job with the last assignable TSN (e.g. 9999).

(2) *Selection by priority (HPF)*

The jobs are selected according to their externally assigned job scheduling priority. Since the job with the highest priority is always released first, the preference of privileged jobs is guaranteed.

Selection of jobs with the option START=IMMEDIATE or START=AT in the ENTER-JOB command:
These functions cause the scheduling algorithm to set the value M (precedence) for jobs with the option START=IMMEDIATE automatically to 0. For jobs with the option START=AT, M is set to 0 if the specified start time has been reached. Jobs for which M=0 is detected are immediately released for starting and cannot be overtaken by jobs with M>0.

(3) *Selection by priority and aging (HPA)*

By including the waiting time in the selection strategy, jobs with a lower priority can also be passed to the class scheduler, even when there is a permanent supply of high-priority jobs.

(4) *Selection by runtime (SJF)*

This strategy prefers short jobs and, in contrast to the FIFO method, reduces the mean waiting time. As batch jobs are aborted when the requested CPU time has expired, this strategy cannot be circumvented. SJF delays the start of time-consuming jobs; in extreme cases, namely where there is a sufficient amount of short jobs, this may result in long jobs not being released for starting.

(5) *Selection by CPU time and priority (SJP)*

This strategy combines the “selection by runtime” and “selection by priority” methods.For jobs with identical CPU time requests, the priority becomes the selection criterion; jobs with identical job scheduling priorities are selected on the basis of the requested CPU time.

(6) *Selection by response ratio (HRN)*

The response ratio is expressed by (waiting time + CPU time) / CPU time and constitutes a combination of the HPA and SJF strategies. This gives preference to short jobs, but takes the waiting time into account.

(7) *Selection by throughput and priority (HRP)*

Short jobs are given preference. The order or precedence of the jobs is determined by the respective job scheduling priorities.

Log of the standard job scheduler

The standard job scheduler logs certain events of the stream tasks to the log file SYS.SCHEDLOG.<yyyy.mm.dd>.<hh.mm.ss>.<streamname>, provided logging has been activated in the stream definition (in the JMU statement DEFINE-JOB-STREAM or in the S-PAR='...,LOGGING=YES' operand of the MODIFY-JOB-STREAM command). The log provides an overview of the scheduler's activities during a session. Moreover it serves as a support document for potential error analysis.

Extract from a log file (the number (1., 2., ...) do not form part of the log file but act as explanatory texts).

```
FTS BS2000   JOB SCHEDULER JSSTD1 VERSION <version> <date> <time>    PAGE 1
**               STREAM STARTED
** JSSTD1         JSATTCH 1.
**               EVENT         10    (JS_HOLD) 16.
**               EVENT         11    (JS_RELEASE) 17.
**               EVENT          9    (JC_AVAILABLE) 15.  JCLASS=JCDSTD
**               JSWAKE 4.                                WAKE-TIME=2147483647
**               EVENT          9    (JC_AVAILABLE) 15.  JCLASS=JCDSTD
**               EVENT          9    (JC_AVAILABLE) 15.  JCLASS=JCDSTD
**               EVENT          9    (JC_AVAILABLE) 15.  JCLASS=JCDSTD
**               EVENT               (J_INTRODUCTION) 6. TSN=0FSM
**               JSRUNJB 3.                               TSN=0FSM, JCLASS=JCBSTD,
                                                          WAIT-TIME=0, JOB-PRIO=9,
                                                          CPU-TIME=32000,
                                                          MERIT=45DBBA000000000000000001
**               EVENT          1    (J_TERMINATION) 7.  TSN=0FSM
**               EVENT          9    (JC_AVAILABLE) 15.  JCLASS=JCBSTD
**               EVENT               (J_INTRODUCTION) 6. TSN=0FSN
**               JSRUNJB 3.                               TSN=0FSN, JCLASS=JCBSTD,
                                                          WAIT-TIME=0, JOB-PRIO=9,
                                                          CPU-TIME=32000,
                                                          MERIT=45DBBA000000000000000002
**               EVENT          9    (JC_AVAILABLE) 15.  JCLASS=JCDSTD
**               EVENT           1    (J_TERMINATION) 7. TSN=0FSN
**               EVENT          9    (JC_AVAILABLE) 15.  JCLASS=JCBSTD
**               EVENT               (J_INTRODUCTION) 6. TSN=0FSQ
**               JSRUNJB 3.                               TSN=0FSQ, JCLASS=JCBSTD,
                                                          WAIT-TIME=0, JOB-PRIO=9,
                                                          CPU-TIME=32000,
                                                          MERIT=45DBBA000000000000000003
**               EVENT               (J_INTRODUCTION) 6. TSN=0FSR
**               JSRUNJB 3.                               TSN=0FSR, JCLASS=JCBSTD,
                                                          WAIT-TIME=0, JOB-PRIO=9,
                                                          CPU-TIME=32000,
                                                          MERIT=45DBBA000000000000000004
**               EVENT               (J_INTRODUCTION) 6. TSN=0FSS
**               JSWAKE 4.                                WAKE-TIME=0
** JOB_HOLD      JOB HELD                                 TSN=0FSS
**               EVENT          2    (J_HOLD) 8.         TSN=0FSS
**               JSWAKE 4.                                WAKE-TIME=10111560
**               EVENT          4    (J_CANCEL) 10.      TSN=0FSS
**               JSWAKE 4.                                WAKE-TIME=2147483647
**               EVENT               (J_INTRODUCTION) 6. TSN=0FSS
**               JSWAKE 4.                                WAKE-TIME=0
**               EVENT         16    (JS_TIMER) 21.      DATE=<date>, TIME=<time>
**               JSWAKE 4.                                WAKE-TIME=10111500
**               EVENT          3    (J_RELEASE) 9.      TSN=0FSS
**               EVENT          1    (J_TERMINATION) 7.  TSN=0FSQ
**               EVENT          9    (JC_AVAILABLE) 15.  JCLASS=JCBSTD
**               EVENT          1    (J_TERMINATION) 7.  TSN=0FSR
**               EVENT          9    (JC_AVAILABLE) 15.  JCLASS=JCBSTD
**               EVENT          4    (J_CANCEL) 10.      TSN=0FSS
**               JSWAKE 4.                                WAKE-TIME=2147483647
**               EVENT         13    (JS_CLOSE_IMMEDIATE) 19.
**               STREAM CLOSED
** JSSTD1         JSDETCH 2.
```

Explanation of the job scheduler functions listed in the log (steps evident from the logging information are printed in bold):

1. **JSATTCH**
   The job stream issues a message indicating that it has started successfully.
2. **JSDETCH**
   The specified job stream is terminating. The job management system no longer supports the job scheduler. In addition, the task in which the job stream executes is terminated.
3. **JSRUNJB**
   The job scheduler requests the class scheduler to start the specified job. If the CLASS-LIMIT is reached for the job class, information to this effect is returned to the standard job scheduler.
4. **JSWAKE**
   The job scheduler informs the job management system of when it wants to be reactivated.
5. JSEXPCT
   The job management system is requested to pass the next event to the job scheduler. Such an event may be (see items 6 - 21):
6. **J_INTRODUCTION**
   The job scheduler is requested to include a job in the set of jobs for which it is responsible. The following information is supplied to the job scheduler for this purpose:

   Job number, name of job class, repeat indicator, CPU time, job scheduling priority, job arrival time, start attribute, job parameters.
7. **J_TERMINATION**
   A job managed by the relevant job scheduler is terminated after completion of its runtime.
8. **J_HOLD**
   A job is halted and placed in the hold queue.
9. **J_RELEASE**
   A job in the hold queue is released. The job scheduler is requested to consider the job for selection.
10. **J_CANCEL**
    A job not yet started by the job scheduler is aborted.
11. J_EXPRESS
    Acceptance of a job that is to be started as soon as possible. The job is assigned the precedence M=0 (see "Job scheduling algorithm").
12. J_RESCHEDULE
    Indicates that the job attributes have been modified by means of the MODIFY-JOB command.
13. JC_HOLD
    The specified job class is placed in the wait state. The job scheduler is requested to skip the jobs of this class during the selection process.
14. JC_RELEASE
    The job class and hence (implicitly) all its jobs are removed from the wait state.
15. **JC_AVAILABLE**
    The value defined with CLASS-LIMIT has not been reached for the specified job class.
16. **JS_HOLD**
    The specified job stream is placed in the wait state.

17. **JS_RELEASE**
    The job stream is removed from the wait state. The job stream is requested to process all jobs within its sphere of activity.
18. JS_CLOSE QUIET
    The system is in the termination phase. This event has the same effect as JS_HOLD, i.e. no further jobs are started.
19. **JS_CLOSE IMMEDIATE**
    The job scheduler is terminated immediately after the STOP-JOB-STREAM command has been processed.
20. JS_CHANGE
    Information on modified STREAM-PARAMETERs is output.
21. **JS_TIMER**
    The job scheduler is requested at one minute intervals to check the start attributes of its jobs.

11.1.2.3 Class scheduler

Within the framework of job management the class scheduler handles the following functions:

Starting of the batch jobs

As shown in figure 17 in section "Job management", all job schedulers pass the jobs they have released for execution to the class scheduler.
In collaboration with the task management system, a job released by the job scheduler is started immediately by the class scheduler.

Monitoring job class limits

The JMU statement DEFINE-JOB-CLASS and the CLASS-LIMIT operand serve to define an admission limit per job class (maximum number of jobs to be started in this class). Relative to the entirety of job classes, systems support thus specifies a job mix. The class scheduler is responsible for checking that this mix is not violated. When a job is passed to it, the class scheduler checks whether the CLASS-LIMIT has been reached. If this is the case, the job is rejected (unless it is a job with the start attribute IMMEDIATE).
As soon as the situation changes, e.g. as a result of job termination or a MODIFY-JOB-CLASS command to increase the limit, the relevant job scheduler is notified of this event and a job is passed to the class scheduler.

Buffering jobs in the case of system saturation

If the paging area or the main memory is saturated, the class scheduler still accepts jobs from the various schedulers, but buffers them in class-specific queues.
The inclusion of jobs in these queues is based on the FIFO principle. This principle does not apply to jobs with the start attribute IMMEDIATE (known as express jobs). These jobs come first in any queue. If the class scheduler receives more than one express job, the job that has arrived last takes precedence in the queue.

Starting these jobs after clearing the saturation state

Once the saturation state has been eliminated, the class scheduler starts the jobs by selecting a job class with the lowest M value whose class optimum has not yet been reached, using the following algorithm:

M = (C + 1) / W

where:

M	Precedence
C	Number of jobs currently running in the job class
W	Weight (urgency) of job class

The class scheduler starts the job that comes first in the queue.

After every job start and job termination the class scheduler must recalculate the precedence.

If the software product PCS is used, the class scheduler is linked to the performance controller.

11.1.3 Job management activities during system initialization

In the system initialization phase, the following functions are executed for the job management system:

- The job stream/class definitions and class access authorizations are read in from the SJMSFILE.
- A copy of the SJMSFILE (type ISAM) is made and stored under the name SJMSFILE.WORK (type SAM).
- The system scheduler and the class scheduler are activated.
- The number of objects (streams, classes, access lists etc.) read from the SJMSFILE or SJMSFILE.WORK file is output on the console.

If the SJMSFILE should be defective or not present during startup, the SJMSFILE.WORK file generated during the previous session functions as a substitute. Since the SJMSFILE has to be recreated, it is advisable to ready a batch job with the corresponding JMU run.

11.1.4 Data center job management

The job management system offers systems support additional options for job scheduling tailored to datacenter needs:

1. Installing a self-defined job scheduler

   Via the Assembler interface a customized job scheduler can be generated which executes as a user program under the system administration ID. The macros required for this purpose are described in the “Executive Macros” manual [30].
   By means of the JMU statement DEFINE-JOB-STREAM (S-PAR operand) systems support defines additional parameters, the syntax and semantics of which are freely selectable. They are interpreted solely by the customized job scheduler.

2. Using a system exit routine

   The JMU statement DEFINE-JOB-CLASS (JOB-PAR operand) enables systems support to define additional parameters, whose syntax and semantics are freely selectable. These parameters are interpreted only by the system exit routine (system exit 32).

The data center might find it expedient to define additional parameters because, for example, the standard job scheduler does not distinguish between the following within the job classes:

very important jobs (class attribute VIMP)

important jobs (class attribute IMP)

less important jobs (class attribute NIMP)

Systems support thus defines the following for the job classes concerned:
`DEFINE-JOB-CLASS...,JOB-PAR='VIMP'/'IMP'/'NIMP'`

11.1.5 Interrupt-free clock resetting

From the point of view of job management, the system no longer needs to be shut down and restarted when switching between summer and winter time:

Job management works internally with UTC time, which is wholly incremental and which does not recognize time changes. Externally, job management uses the time set for the system installation, known as the system time, which is based on local time (LT, the legally valid time at the place of installation). This means that the user is always able to work with the official local time.

A precondition for correct time switching by job management is the correct system parameter files with specifications for time conversions (see "System time control (GTIME)"). Time specifications are thus always interpreted as legally valid local time, regardless of time changes, e.g. start times for jobs with deadlines.

Consistent interpretation of time specifications as official time also results in a change in the way repeat jobs with the “DAILY” and “WEEKLY” attributes are started: the start time for the daily or weekly repeat is determined when such repeat jobs are accepted. This start time is now consistently interpreted as official time. Thus, for instance, a job that was executed daily at 5 p.m. during summer time will also be started daily at 5 p.m. after the change to winter time.

11.1.6 JMP: reconstruction of batch jobs

The JMP program (Jobpool Management Program) reconstructs ENTER-JOB commands on the basis of the information which JMS has stored concerning accepted batch jobs in the job pool (in the system file SYSTEM.JOBPOOL). JMP writes the commands to a file. From the file they can be accessed, modified as necessary and returned to the system.

instruction	Meaning
CREATE-PROCEDURE-FILE	Generate procedure file using the ENTER-JOB commands
OPEN-JOBPOOL-FILE	Open job pool file
SHOW-JOBPOOL-STATUS	Output information on the contents of a job pool file
END	Exit JMP

Table 27: Summary of the JMP statements

For a detailed description of JMP and examples of reconstructed ENTER-JOB commands, refer to the “Utility Routines” manual [15].

11.2 Task management

While the job management system handles all jobs, all tasks are controlled and managed by the task management system.

The central functions of task management are:

- creating tasks
- interrupt analysis and processing
- management of register save areas
- supply of data to central tables of the Executive
- activating tasks
- initiating tasks
- moving tasks within and between queues
- deactivating tasks
- terminating tasks

The name PRIOR designates those task management routines which control:

- tasks by means of categories and task scheduling priorities
- system utilization through internal control functions

Tasks are generally vehicles for interruptible BS2000 processes. A distinction is made between user and system tasks.

User task

All user jobs, including privileged ones, are managed as user tasks which were passed by the job management system to the task management system as:

- interactive tasks (timesharing mode; task type X'40')
- batch tasks (batch mode; task type X'20')

System task

System tasks assume a wide range of system functions. Some system tasks are permanent (TCB pregenerated in class 1 memory), but most are dynamically generated and terminated as required.
These system tasks are assigned specific task identifiers (TIDs) and TSNs, by means of which they can be unambiguously identified.
The most important system tasks, both permanent and dynamically (during startup or as required) generated, are described in the tables below.

TID	TSN	Functions
0001 0001	TSC	Emergency job scheduler $SYSJS for job class $SYSJC, in which in particular all other job schedulers run.
0001 0003	RMM	Main memory check during startup

0001 0005	PT5	Periodic task 5: every 10 seconds it checks, among other things, the lifetime (waiting time excess) in the bourse mechanism
0001 0006	PT6	Periodic task 6 of job management: for operations that are to execute at a defined time (“AT”); checks repeat jobs (interval elapsed?)
0001 0007	PGE	Paging error recovery task
0001 0008	UCO	Universal console UCON: distributes console commands and messages to the responsible agency (operator task for normal operator commands, $CONSOLE application for special operator commands); implements the application $CONSOLE
0001 0009	REK	Responsible for reconfiguration
0001 000A	VMM	Memory management task: manages the slot pools, e.g. for stacks, bourse control blocks
0001 000B	MSG	Responsible for message handling
0001 000C	KTT	Console driver task: I/O for console devices

Table 28: Permanent (preallocated) system tasks

TSN	belongs to ...	**Functions**
RUNT		Operator task: processes normal operator commands from a RUN file
Xxxx		General system work tasks, e.g. MSCF server task, import/export task, operator task (processing and execution of the normal operator commands)
CLOG		Console logging (SYS.CONSLOG)
RP01		SRPM task; manages the user catalog of the home pubset
RP02		SRPM task; manages the user catalog of all other pubsets
HELT		Manages the hardware error file SYS.HEL.<date>.<time>
HERS		HERS task; initiates HEL entries in the case of machine errors
SERS		Manages the error file SYS.SERSLOG....
REPL		Maintains the logging file REPLOG and logs REP before-image
DSSM	Tasks of dyn. subsystem management	Dynamic subsystem monitor
HTnn		Help tasks for DSSM

NKA	Task of nucleus device management NDM	Resource management

DM	Disk monitor

TM	Tape monitor

NKS		Resource allocation
ACCT PACT	Tasks of the accounting system	Processes the accounting file Task for the Periodic Accounting function
BCAM	Tasks of the openNet server system component	Message pool management; processing of /BC commands
BCAH		Holder task for data which survives a /BCEND
BCAF		File access task (e.g. reading the processor file)
BCAC		Cryptographic task; handles encrypted and decrypted requests when IPSec is used
BCA0		Transfer task for data communication using TCP/IP
BCAT		Transfer task for data communication using other protocols
DIAA		TIAM task; generates dialog tasks (for virtual hosts also: DIAB, DIAC, ...)
DBxx		Tasks of the disk access buffer DAB
DRV		Task of the software product DRV
SNAP		SNAP task for the implementation of the SNAP dump function
S	System tasks of SPOOL	SPOOL scheduler; generates SPOOL tasks and supplies them with jobs, responsible for SPOOL commands
SPAx		SPOOL output writer; responsible for SPOOLOUT
SPMG		spool message task; responsible for message output
xxxx		SPOOL task (device controller task); assigned to a SPOOL device (TSN is allocated by the system)
RSO		Remote SPOOL (RSO-Spool)
RSAx		RSO output writer
EQS		Management of file $TSOS.EQUISAMQ
M		New message handler
ISDS		Management of the ISAM pools
IOTD IOTT		Tasks for input/output management when disks (IOTD) and tapes (IOTT) are reconfigured
JOBP		Job pool controller task: management of the job pool file for batch jobs
TSNA		TSN file manager: management of the file TSOJOIN.TSNN
TIME TIMC		Timer management: clock synchronization with external timers for setting summer/winter time
DRTx		Domain reconfiguration tasks on SU x86

MSCF MSCM WDGS	System tasks of the software product MSCF	Task for the implementation of the MRS link
FTCP	Task of the openFT software product	File Transfer Control Process
SM2G	System tasks of the openSM2 software product	Software monitor scanning task
SM2W		openSM2 write task
SM2U		openSM2 task for user monitoring operations
SATP	System tasks of the software product SECOS	SAT parameter file management
PRxx		GUARDS server task, manages the guards catalog of a pubset
SATT		SECURITY-AUDIT-TRAIL task for SAT file management

Table 29: Important dynamically generated system tasks

Task categories

Every task is assigned to a category. In addition to the four standard categories SYS, TP, DIALOG and BATCH, the task management system supports twelve further categories, the names of which are specified during job class definition.

The following applies for the four standard categories:

SYS	for system tasks only
TP	for transaction tasks (inquiry and transaction processing)
DIALOG	for interactive tasks (timesharing mode)
BATCH	for batch tasks (batch mode)

In addition, there are four task attributes whose names are identical to the standard category names SYS, TP, DIALOG and BATCH. Special runtime parameters important for task scheduling are assigned to the task attributes.

As opposed to the other task attributes, the TP task attribute features optimized memory management especially geared to the requirements of inquiry and transaction processing. This attribute can be obtained either by definition in the job classes allocated to the jobs or by calling the TINF macro (see the “Executive Macros” manual [30]). The required authorization must be contained in the user catalog.

The task category feature is available only to the system management (in exceptional cases to the operator as well) via the command interface. Using the MODIFY-TASK-CATEGORIES command the system administration specifies the relative importance of each category for the activation process (= allocation of main memory).

For this purpose the following category attributes are specified:

WEIGHT	Importance
MINMPL	Minimum number of tasks to be kept active

MAXMPL Maximum number of tasks to be kept active

IO-PRIO I/O priority of all tasks in this category

Whether a task is activated, deactivated or preempted depends not only on the category attributes, but also on the system workload and the priority of the task concerned.

As a category is usually assigned several tasks, category control is always **group-specific**. In an evaluation of the effects of category control, the following situations have to be distinguished:

- In the underload and normal-load ranges, category control is negligible.
- In the full-load and overload ranges, i.e. during resource bottlenecks, category control is of major importance in load limitation (categories of minor significance are restrained.

The MOVE-TASK-TO-CATEGORY command, IO-PRIORITY operand, can be used to determine the I/O priority of all tasks in the specified category. This value is evaluated only if the IORM subsystem has been started. The default *NONE means that all tasks in this category are assigned an I/O priority derived from the task priority (implicit I/O priority). The I/O priorities are evaluated by the IORM subsystem solely for the purpose of task-specific control of inputs/outputs, with runtime control both by PRIOR and by the PCS subsystem.

Systems support can use the MOVE-TASK-TO-CATEGORY command to modify the assignment of a task to a category if, for example, different (better) support of this task or a reduction of the load on a category is required (with or without the use of PCS). Target categories must be JMS categories and may not be successor categories. The SYS category is not supported as a source or target category.

Choosing the right category values is not an easy undertaking. Therefore it is advisable - particularly for large servers - to carry out performance studies over a prolonged period of time. The necessary data can be obtained with the aid of the software monitor SM2 (see the “openSM2” [49] and “SM2-PA” [50] manuals). Further information is contained in the “Performance Handbook” [37].

Command	**Meaning**
ADD-USER	Create an entry in the user catalog, indicating whether or not the user tasks may be deactivated
CANCEL-JOB	Cancel a user job
CHANGE-TASK-CPU-LIMIT	Increase the maximum CPU time of a batch job
CHANGE-TASK-PRIORITY	Change the job or task priority
ENTER-JOB	Start the command sequence stored in an ENTER file as a batch job, specifying the task priority
FORCE-JOB-CANCEL	Cancel a user job
HOLD-TASK	Hold a user job

MODIFY-TASK-CATEGORIES	Control allocation of CPU, main memory and I/O processor to the individual categories, define priorities of the categories, and specify I/O priorities for all tasks of a category
MOVE-TASK-TO-CATEGORY	Modify the category assignment of a task

RESUME-TASK	Release a user job on hold
SET-LOGON-PARAMETERS	Start an interactive or batch job, specifying the task priority
SHOW-JOB-STATUS	Request information about a user job
SHOW-SYSTEM-STATUS	Request information on job classes and job streams
SHOW-USER-ATTRIBUTES	Request information on maximum task priority
SHOW-USER-STATUS	Request information on groups of user jobs
Macro	**Meaning**
PASS	Wait one second
TINF	Read and change task attributes
TSPRIO	Output run priorities
VPASS	Wait

Table 30: Interface overview for task management

11.2.1 Priority concept and queues

The execution priorities may be subdivided as follows:

Priorities 0 - 29	Priorities for system tasks
Priorities 30 - 127	Fixed task scheduling priorities
Priorities 128 - 255	Variable task scheduling priorities

Except for the system priorities, priorities are specified during job class definition and, for each user, in the user catalog.

If the user specifies a priority in the SET-LOGON-PARAMETERS or ENTER-JOB command, this priority is checked both in the job class assigned to the user and in the user catalog.

The priority of a task is considered during both activation and initiation (allocation of CPU).

The system parameter ETMFXLOW also offers the option of setting up a lower area of fixed priorities.

Variable priorities

Variable priorities are characterized by dynamic priority adaptation using the HRN (Highest Response ratio Next) algorithm

HRN based on the ratio residence time / CPU time used and taking into account the start priority in the SET-LOGON-PARAMETERS or ENTERJOB command or an externally assigned priority (CHANGE-TASK-PRIORITY command).

The HRN algorithm causes tasks taking little CPU power and I/O-intensive tasks to be given preference, without creating extreme disadvantages for compute-intensive tasks. Moreover, appropriate handling is also ensured for tasks with a low start priority.

The variable priority is recalculated at the following times:

- every time activation is performed
- with every PEND/UNPEND following inclusion in the Q5 queue
- on expiration of a micro time slice (a time quantum dependent on the CPU speed and on the immediately preceding I/O behavior of the task)
- periodically (once per second)
- when a MODIFY-TASK-CATEGORIES command is issued
- when a CHANGE-TASK-PRIORITY command is issued

Fixed priorities

Fixed priorities never change.

A fixed priority is a great advantage for the task concerned.
Fixed priorities have been conceived for applications with extreme real-time demands. They can bring about performance improvements with the following restrictions:
Fixed priorities greatly reduce the system's leeway for decisions. Therefore, in order to achieve positive results, the resource requests of all tasks must be known. In the case of high resource utilization, fixed-priority tasks waiting for activation lead to the immediate preemption of other tasks, which may cause an overload situation. This in turn has repercussions on performance in the long run.

Fixed priorities should only be used after meticulous analysis of system load and resource utilization data, and in conjunction with load-reducing measures such as the limitation of the category multiprogramming level by means of the MAXIMUM-ACTIVE-TASKS operand in the MODIFY-TASK-CATEGORIES command.

Summary

- Priorities of any kind influence the sequence of queued tasks.
- Any priority better than the “normal” priority 255 affects the system, regardless of the load situation.
- Due to the increased importance of priority, systems support can easily influence the performance of individual tasks just by changing their priority. Especially in the case of low-performance systems with only a few user tasks it is helpful to assign different variable priorities to the tasks. This ensures that each task receives a share of system capacity commensurate with its importance.
- If nearly all tasks are working with priority 255, a slight difference in priority (e.g. 5) is sufficient to give strong preference to a task. If the range of priorities is greater, the difference in priority must likewise be greater. Depending on the load, the priority should be varied (e.g. in increments of 5 or 10) until the desired performance target is reached. As a rule it is not necessary to assign fixed priorities, since medium-range variable priorities (around 200) already yield satisfactory performance values. Larger priority intervals are required in the case of consistently high task loads or predominantly I/O-intensive tasks.
- The allocation of the resources main memory and CPU to user/system tasks is controlled in accordance with the predefined category/priority concept.

Queues

Tasks may exist within the system in any of five states:

- task is occupying a CPU
- task is active and executable
- task is active but not executable
- task is inactive and executable
- task is inactive and not executable

The basis for this is the queuing concept of the task management system, which ensures that each task is included in one of the following queues (depending on the relevant task state).

Q0	Running task	Task in control
Q1	Waiting for CPU allocation	a c t i v e and executable

Q2	Write tasks of openSM2	a c t i v e and not executable
Q3	Waiting for paging	a c t i v e and not executable

Q4	Waiting during rapid input/output Explicit synchronization functions (VPASS, SOLSIG, REVNT, locks and job relations in system) Waiting for new input for TP tasks	a c t i v e and not executable
Q5	Per category: waiting for activation	i n a c t i v e and executable
Q6	Per category: waiting for permission from PCS (only if PCS is activated)	i n a c t i v e and executable
Q10	Waiting in HOLD state	i n a c t i v e and not executable
Q11	Waiting system tasks	i n a c t i v e and not executable
Q12	Waiting during lengthy I/O operations and lengthy synchronization functions, especially terminal I/O in interactive mode	i n a c t i v e and not executable
Q13	Waiting for n seconds (PASS/VPASS)	i n a c t i v e and not executable

Table 31: Queues in task management

If a task is not waiting for input, its primary goal is to occupy the CPU. This usually presupposes several changes of state.

Let us assume a user task (e.g. a batch task) is in Q10 following a HOLD-TASK command.As soon as systems support releases the task via the RESUME-TASK command, its state changes.
The task leaves Q10 and is entered in Q5 (the subqueue for the category involved, e.g. batch). There it waits for PRIOR to access it in the context of the activation process and move it to Q1 and eventually to Q0.

These transitions from one state to another are implemented by means of the PEND/UNPEND routines:
The UNPEND routines move the task towards the CPU, while the PEND routines move the task away from a high-ranking queue and include it in a lower-ranking queue.

11.2.2 Allocation of resources

Task management has to make a number of decisions before a task can execute. Resource management plays a vital role in this connection.

When a task is given the right to occupy space in main memory, this means that the task is activated. Subsequently the task must be assigned a CPU i.e. the task is initiated.

Activation decisions are based on the following criteria:

- multiprogramming level per category
- priority
- resource utilization (main memory, CPU, paging)
- system services provided (CPU time, number of I/O operations).

Time slices

Time slices are used by task management to make CPU and main memory resources available to all tasks in as uniform a manner as possible in accordance with their priority

- micro time slice (MTS) for optimum utilization of the CPU and
- system service time slice (SSTS) for efficient allocation of main memory

Micro time slice

The micro time slice MTS is the maximum time a task may occupy the CPU without interruption if no task with a higher priority is waiting for allocation of the CPU.
No later than after the micro time slice has elapsed, an interrupt is issued in order to prevent a task from blocking the CPU. If the task has a variable priority (128-255), this will be modified in accordance with the I/O behavior and CPU utilization of the task.

System service time slice

The system service time slice SSTS does not document an absolute time. Instead, it corresponds to a maximum quantum of system services (CPU utilization, I/O rate) that an active task may demand. After the system service time slice has elapsed, the task is deactivated and loses its claim of main memory if tasks of the same category are waiting for activation.

Allocation of the main memory resource

- Activating a task

  Task activation depends on permission from the internal control facility known as ACF (Activate Control Function). For this purpose ACF measures the utilization of the resources main memory and CPU, and also the paging activities; if an activation sequence is to be performed, it only measures main memory utilization. Depending on the degree of resource utilization and the predefined category/priority concept, a matrix-based decision is then made as to whether a further activation is permissible or a forced deactivation, a preemption or no action should be performed.

  The following example shows how PRIOR proceeds during task activation:

  Total active: 19 TP, 9 DIALOG and 5 BATCH tasks

  Waiting for activation: 2 TP, 2 DIALOG and 2 BATCH tasks

  Systems support defined the categories as follows; the IO-PRIORITY operand is set to *NONE (default):

```
/MODIFY-TASK-CATEGORIES CATEGORY-NAME=DIALOG,-
       WEIGHT-CODE=100,MINIMUM-ACTIVE-TASKS=25,MAXIMUM-ACTIVE-TASKS=35
/MODIFY-TASK-CATEGORIES CATEGORY-NAME=TP,-
       WEIGHT-CODE=500,MINIMUM-ACTIVE-TASKS=30,MAXIMUM-ACTIVE-TASKS=40
/MODIFY-TASK-CATEGORIES CATEGORY-NAME=BATCH,-
       WEIGHT-CODE=3,MINIMUM-ACTIVE-TASKS=5,MAXIMUM-ACTIVE-TASKS=15
```

The selection/activation decision by task management is taken in two steps:

1. Selection of the category from which a task is to be activated.
2. Within this category, the task with the highest priority is selected; in the case of identical priorities the task occupying the first position in the queue is activated.

For category selection the following decisions are required	**With reference to the example this means:**
Only those categories are considered for which **at least one task is waiting for activation**	This condition is satisfied for all categories
Among these, the following categories are given priority: ○ all categories (if present) which have not yet reached MINMPL. MAXMPL has no meaning here. ○ when all categories have reached MINMPL: all categories which have not yet reached MAXMPL.	This condition is satisfied for categories TP and DIALOG
If such preferred categories exist, all other categories are excluded from further selection: ○ z=0 n (number of active tasks < MIN z=MIN) ○ z=MIN <= number of active tasks < MAX) ○ z=MAX (number of active tasks > MAX)–	Explanation: z=0 for TP (19 < 30) z=0 for DIALOG (9 < 25) z=5 for BATCH (MIN=5)
From these categories the category for which the lowest index is calculated is selected for activation. Index calculation is based on the following algorithm: *Index* = (*NAT* +1 -*z*) / *WT* where: ○ NAT = number of active tasks in the category for which the index is calculated ○ z = status indicator ○ WT = weight (urgency)	Index calculation is necessary for categories TP and DIALOG: $Index_{TP}$ = (19+1-0) / 500 = 20 / 500 = 0,04 $Index_{DIA}$ = (9+1-0) / 100 = 10 / 100 = 0,01
In the case of identical indices, activation is carried out in the sequence TP - DIALOG - BATCH	

Result: The highest-priority task of category TP is activated.

Exception: Tasks with fixed priority are given preference.

- Deactivating a task

  The purpose of deactivating active tasks is to obtain main memory space for executable inactive tasks.
  This takes place when

  - further processing is impossible
    (e.g. due to program-controlled wait calls, waiting for input from display terminal in timesharing mode)
  - the waiting time for events has been exceeded
    (e.g. waiting for input from display terminal in inquiry and transaction mode)
  - defined system services are utilized
    (e.g. CPU time and number of inputs/outputs) have been used.

  Exception

  Deactivation is suppressed if systems support has made provisions in the user catalog that preclude deactivation of a user's tasks (see the ADD-USER command, PRIVILEGE=INHIBIT-DEACTIVATION).

- Forced deactivation of a task

  If the ACF function senses bottlenecks in the course of its survey of resource utilization, forced deactivation of a task is carried out.
  Since forced deactivation relies on the category attributes MIN, MAX and WT, systems support can choose appropriate presettings for these values so that forced deactivation is performed only for tasks of less important categories.

  Exceptions

  - fixed-priority tasks
  - tasks in the TPR state
  - tasks that hold a lock
  - tasks under a user ID for which PRIVILEGE=INHIBIT-DEACTIVATION was declared (ADD-USER command); forced deactivation is not possible for these.

- Preempting a task

  Preemption may occur if an activation request has been received, but the ACF function does not permit activation because it has sensed resource bottlenecks. There are two ways of preempting a task:

  1. An active task is preempted by an inactive task from a different category.
  2. The active and inactive tasks belong to the same category.

  The preemption rate is proportional to the duration of the overload situation (resource bottleneck) in the system.

- Preemption control function (PCF)

  PCF periodically monitors the preemption rate. Message: `EXC0455 TASK PREEMPTION LEVEL=i`

  `i` is a number from 0 through 3 and designates the preemption rate.

Level	Effects on system	Operator action
0	Normal system processing	None
1, 2	Short-term overload conditions	None
3	Long-term overload conditions The MAX-MPL value of a category is not exceeded. The system reserves main memory for privileged tasks.	Reduce the MIN-MPL values of categories

Measures for preventing excessive preemption rates

- Modify the MIN-MPL and MAX-MPL values for separate categories such that fewer activation requests are made.
- Increase main memory or reduce the load.

Allocation of the CPU resource

After successful activation, each task waits for allotment of the CPU (central processing unit) resource, i.e. for **initiation**.

After the task has been allocated the CPU resource, the following interrupt causes may lead to **deinitiation**:

- task wants to wait for synchronization event
- waiting for termination of an I/O operation
- execution of a paging operation
- expiration of the micro time slice

Preemption may also occur on the initiation level. This takes place when another active and executable task with a higher priority is waiting for initiation. This can happen, for example, when an I/O operation for a task with a higher priority has terminated.

Assignment of the I/O resources

Assignment of the I/O resources depends on a task's I/O priority. IORM is responsible for control here, see section "IORM: Control of I/O resources" and the "Utility Routines" manual [15].

11.2.3 Management of affinity task groups (TANGRAM)

The TANGRAM subsystem (task and group affinity management) plans the allocation of affinity task groups to CPUs on the basis of their performance requirements. For certain multitasking applications there are groups of tasks which often need write access to large amounts of shared data. These task groups are known as “affinity” task groups.
To reduce losses on multiprocessors due to co-hits in the CPU caches, it is best to restrict such task groups to a subset of CPUs.

Tasks groups are set up with the TINF macro (see the “Executive Macros” manual [30]). Management of the task groups is handled by the TANGBAS subsystem, which will have started automatically on “System ready”, regardless of whether or not the TANGRAM control function has been started.

The TANGRAM subsystem measures the utilization of the individual CPUs, the performance needs of the relevant task groups and of the other tasks, and specifies the respective CPUs permitted to initiate a task for each control interval.

The attachment of tasks to groups, their subsequent detachment and the allocation of processors to the individual groups can be logged with COSMOS hooks. COSMOS is an optional component of openSM2 and is available only via special release (see measurement programs in the “openSM2” manual [49]).

The performance gain that can be achieved with TANGRAM depends on the server (multiprocessor configuration, cache architecture), on the load profile (proportion and structuring of the TP applications) and on server utilization. The BS2000 TP products openUTM, UDS/SQL and SESAM/SQL create TANGRAM task groups. Multitasking applications should be checked to establish whether creating TANGRAM task groups leads to an improvement in performance.

The way in which TANGRAM works can be controlled by specifying parameters in parameter file SYSSSI.TANGRAM.<version>.
The defaults come into effect if any of the parameters cannot be found in the parameter file or if a parameter file is not available. In this latter case, the following console message is also displayed:

```
NTG0002 Warning: Opening of TANGRAM parameter file (&00) failed.
        DMS-Error-Code (&01). Processing continued with default parameters.
```

In order to change parameters, TANGRAM must be stopped and restarted with the modified parameter file. Throughout this process all the tasks groups already created are retained.

The following parameters control the operation of TANGRAM. The parameters may appear in the parameter file in any order, but each parameter must appear in a separate line.

PERIOD=<integer 1..100>
Determines the interval (in seconds) between two consecutive runs of the periodic TANGRAM routines. Default: 10 seconds

CLEARENCE=<integer 0..100>
Provides a percentage value for calculating the maximum CPU load of a task group. The load is calculated as follows: load = (1-value). Default: 20 %, i.e. no CPU should be more than 80 % occupied by a single task group.

THRESHOLD=<integer 0..100>
Determines the overall utilization threshold on all CPUs at which a task group actively takes part in the process of processor allocation by TANGRAM. Default: 10 %

11.3 Time limits in BS2000

In BS2000 there are three different time limits which affect the user in timesharing mode: a user ID-specific, a task-specific and a program run-specific time limit.

A user ID-specific time limit is specified by systems support in the JOIN entry (user entry). When a task terminates, the BS2000 accounting system deducts the consumed CPU time from this specified time allowance.

When a task starts, the maximum CPU time (task time limit, TTL) can be specified.

- Batch jobs:
  The CPU time that can be specified is the minimum from the time quota of the user entry and the maximum value from the job class definition.
- Dialog jobs:
  The maximum CPU time is defined by the maximum value from the job class definition, if the CPU time value in the user entry is greater than zero.

If no maximum CPU time is specified, the default value from the job class definition or, if applicable, the lower time allowance of the JOIN entry is taken over as the maximum CPU time of the task. Users for whom the no-time-limit (NTL) privilege is specified in their JOIN entry, and the user ID TSOS can start tasks without time limits (NTL tasks).

Since the time allowance in the JOIN entry is not updated until the task is completed, the maximum CPU time (and if applicable the actual time consumed) of several tasks of one user ID can together exceed the time allowance.

When a program is loaded or started, a maximum program runtime can be specified for it (program time limit, PTL).

Interactive and batch jobs behave differently with regard to the maximum CPU time of the task (TTL) and the maximum program runtime (PTL):

TTL or PTL reached in batch job

When the batch job has used up the maximum CPU time, no program is loaded and TTL < 300 seconds, the task is terminated with the following message on the console:

```
EXC0736 ABNORMAL TASK TERMINATION.
        ERROR CODE '(CMD1011)': /HELP-MSG (CMD1011)
```

When the batch job has used up the maximum CPU time, a program is loaded and TTL >= 300 seconds, the following message is output to the console:

```
EXC0070 Batch job with TSN (&00) has reached time runout.
```

When the system parameter DIATTL=Q is set (see section "TTL or PTL reached in interactive job"), the batch job is terminated.

Otherwise the job is now stopped for 20 minutes if a program is loaded. If there is no program loaded, then the job's TTL is increased by 5 minutes. During this time (5 CPU minutes or 20 minutes of time) the systems support then has a maximum of 20 minutes in which to increase the maximum CPU time of the batch job via the CHANGE-TASK-CPU-LIMIT command. This prevents important batch jobs from terminating prematurely because of a timeout.

If the maximum CPU time of the batch job is not increased within the wait time of 5 or 20 minutes, the batch job is terminated (finally). A distinction is made between the following two cases:

1. If the TTL or PTL was reached during a program run, message `EXC0072` is output or, if applicable, a STXIT routine is processed which is allowed to consume up to a further 30 CPU seconds. When the STXIT routine has been processed or at the latest after 30 CPU seconds, the program is terminated with message `EXC0073`.

The task itself is then terminated (TTL) or the spin-off is triggered (PTL). If error processing is planned, further processing continues with the next SET-JOB-STEP or IF-BLOCK-ERROR command. Otherwise, processing terminates on the next EXIT-JOB or LOGOFF command (PTL).

```
EXC0072 TIME LIMIT FOR TASK/PROGRAM EXCEEDED. PROGRAM TERMINATED
EXC0073 MAXIMUM PROGRAM RUNTIME EXCEEDED. 'STXIT' ROUTINE OR
        PROGRAM RUN USED UP 30 SECONDS.
```

2. If the TTL was reached during the processing of BS2000 commands, the task is terminated with the following command:

```
EXC0736 ABNORMAL TASK TERMINATION.
        ERROR CODE '(CMD1011)': /HELP-MSG (CMD1011)
```

TTL or PTL reached in interactive job

The CPU time for an interactive task is not usually limited. However, systems support can use the system parameter DIATTL (DIAlog Task Time Limit) to restrict the CPU time depending on the TTL. DIATTL can have the value Y, N or Q:

1. DIATTL=N (default)
   The interactive task is terminated on task time runout.

   If TTL is exceeded when no program is loaded then message EXC0067 is output and the user can continue to work for 30 CPU seconds until the process is repeated.

```
EXC0067 CPU TIME SPECIFIED IN /SET-LOGON-PARAMETERS EXCEEDED.
        TASK CONTINUED
```

   If a program is loaded, the following applies if TTL or PTL is exceeded:

   - In procedure mode, message EXC0068 is output and execution continues. The time limit is increased by 30 CPU seconds, i.e. the process subsequently repeats itself.

```
EXC0068 CPU TIME SPECIFIED AT PROGRAM START EXCEEDED. PROGRAM CONTINUED
```

   - In interactive mode, message EXC0075 is output for a program without a defined STXIT routine for the “end of program runtime” event class. The user can specify whether execution is to be resumed or terminated. If execution is resumed, the process repeats itself after 30 CPU seconds.

```
EXC0075 TIME LIMIT FOR PROGRAM RUN EXCEEDED. PROGRAM TO BE CONTINUED?
        REPLY (Y=YES; N=NO)
```

   - If a program has an STXIT routine, it is started. The task is given 30 additional CPU seconds to process the routine. If the STXIT does not terminate the program, it passes control back to the program. If the allowed 30 CPU seconds have not been used, the program continues running until this new time limit is exceeded. The following two messages are then output:

```
EXC0073 MAXIMUM PROGRAM RUNTIME EXCEEDED. 'STXIT' ROUTINE OR
        PROGRAM RUN USED UP 30 SECONDS.
EXC0075 TIME LIMIT FOR PROGRAM RUN EXCEEDED. PROGRAM TO BE CONTINUED?
        REPLY (Y=YES; N=NO)
```

     The user can specify whether execution is to be resumed or terminated. If execution is resumed, the process repeats itself after 30 CPU seconds.

2. DIATTL=Y
   The interactive task is terminated with a task time runout.

If the TTL is exceeded, message EXC0076 is first output:

```
EXC0076 TASK CPU TIME LIMIT for DIALOG TASK exceeded.
        Task will be terminated after 30 sec extra CPU-time.
```

If there is a program run with STXIT for timeout, this STXIT routine is started after the expiration of the period provided for by the message. By default, a further 30 CPU seconds is allowed for the STXIT routine.

3. DIATTL=Q (quick and quiet)
   The interactive task is terminated with a task time runout (as with DIATTL=Y).

   In addition (also for a batch job), for both TTL and PTL all aforementioned additional times and/or wait times will be reduced to one second (“quick”) and the message EXC0070 which is waiting for an answer is suppressed on the console (“quiet”).

11.4 PCS: Performance Control System

The optional PCS subsystem (Performance Control System) supports systems support in achieving the optimum configuration of the computer system. It enables the performance of a computer system to be distributed among the various task categories and tasks according to user requirements.

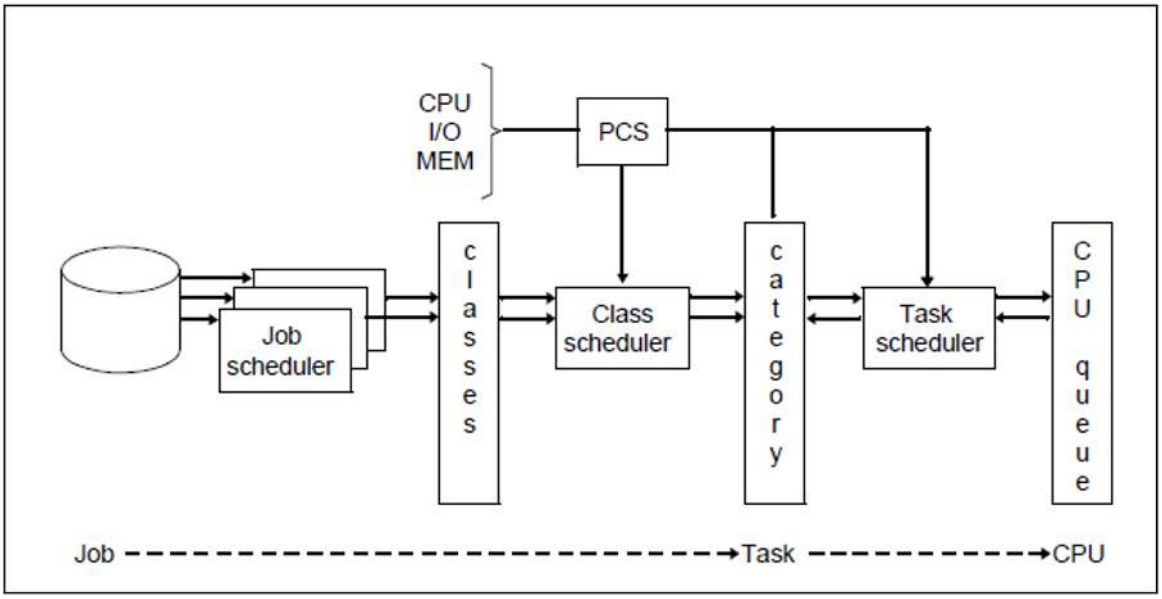


Figure 18: Job management – PCS – Task management

In the case of complex mixed mode PCS regulates the various shares of the load in accordance with the customer-specific requirements. At short intervals PCS adjusts the control variables in such a way that predefined optimization strategies (response time/throughput time optimization) and the optimum operating point for these are complied with at all times.

During operation, PCS monitors: the performance requirements of the tasks and categories and the utilization of the CPU, the main memory and the I/O system.

If these variables do not match the setpoint values defined by systems support (PCS option), PCS corrects the performance levels by changing manipulated variables within BS2000 (such as task priority, MIN-MPL and MAX-MPL values of the categories).

When PCS intervenes for control purposes, in the following cases messages are issued on the console. The messages are also logged in the CONSLOG file.

- When the job scheduler is made to start no more jobs globally or in a category.
- When task activation upon reaching the MAX-MPL value is set.

PCS measures its control variables over periods of around 10 seconds and adjusts the manipulated variables to the current load situation in this period.

PCS works with PRIOR, the task priority control function of which is disabled when PCS is started, and with job management.

For further information on PCS, see the “PCS” manual [36].

12 Security

This chapter describes the most important BS2000 security mechanisms.

These include the BS2000 OSD/BC system access control and data access control mechanisms and the additional mechanisms available with the software product SECOS. For further information on SECOS, please refer to the section "Meeting security requirements using SECOS" and the “SECOS” manuals [46] and [47].

12.1 System access control

System access control covers all the methods that serve to protect a DP system against unauthorized access.

BS2000 OSD/BC

System access control using a LOGON password in the SET-LOGON-PARAMETERS command is currently the most widespread authentication procedure. The MODIFY-USER-PROTECTION command is used to declare an 8- or 32-byte password for the user ID.

With the help of organizational functions provided by SECOS, password protection in BS2000 can be substantially improved. To this end, user administration defines explicitly for each user which of the following stipulations that user must observe:

- minimum length of a password
- complexity of a password
- lifetime of a password.

Extended system access control with SECOS

SECOS offers the following additional options for system access control:

- Separation of access paths
- Access restrictions via terminal sets
- Guards access protection
- Personal identification
- Logging access attempts
- Locking terminals or user IDs after a defined number of unsuccessful access attempts
- Locking user IDs in the case of inactivity

12.2 Data access control

Data access control refers to the rules regulating the access of subjects to the objects of a DP system, as well as to the methods used to ensure that these rules are actually observed.

BS2000 OSD/BC

BS2000 OSD/BC provides access protection mechanisms for objects which can be administered by DMS (file management system). DMS provides object management for files which are DMS objects.

The following protection mechanisms exist for files:

- Limited pubset access
  By distributing user IDs to different pubsets it is possible to protect against unauthorized access to objects such as files on other pubsets (see also chapter "BS2000 user management" and chapter "Pubset management").
- Standard Access Control (USER-ACCESS/ACCESS)
  The standard access control mechanisms (with the ACCESS and USER-ACCESS operands of the CREATE-FILE and MODIFY-FILE-ATTRIBUTES commands) allow users to define access rights for themselves and others (see "Standard access control (ACCESS/USER-ACCESS)").
- Basic Access Control List (BACL)
  The BACL access protection mechanism allows users to define access rights to files for complex sets of subjects (see "Basic Access Control List (BACL)"). The read, write and execute (exec) acess privileges can be assigned separately for every one of the user classes owner, group and others.
- Password
  It is possible for the user to define passwords (read, write and execute passwords) for every one of his files. The corresponding password must be entered before password-protected files can be processed. Passwords may be encrypted if required (see "Passwords and retention periods").
- File encryption
  The user can convert a file into an encrypted format, specifying a crypto password (like a “normal” password, but 8 characters long). Access to the file contents is then possible only when the correct crypto password has been specified (see "File encryption").
  Encrypted files do not have a read or execute password, but can otherwise be combined with all the other access control mechanisms.
- Retention period
  A user can assign his file a retention period in which the file may not be changed (see "Passwords and retention periods").

Security with Cryptography

The BS2000 subsystem CRYPT offers cryptographic functions for BS2000 users7. It uses standardized open interfaces for encryption and decryption of data, see the “CRYPT” manual [9]. CRYPT does not require special hardware.

Extended data access control with SECOS

The software product SECOS (GUARDS component) is a convenient, flexible administrative tool with which it is possible to define and evaluate access conditions for objects such as files, libraries, library elements or other objects of various object management systems. The access protection for the named objects is performed by so-called guards in which the access conditions are stored. Access conditions are queried by object management systems (DMS, LMS, JVS, HSMS, SRPM, etc.). The object management system then grants or refuses access depending on the result of the evaluation.

Using protection mechanisms

The following table shows, which object types can be protected by which protection mechanism:

Protection mechanisms / Object		Limited pubset access	ACCESS USER-ACCESS	BACL	PASSWORD	File encryption	Retention period	GUARDS
File[1]	**public**	+ [2]	+	+	+	+	+	+
	temporary	- [3]	-	-	-	+	-	-
	private	-	+	+	+	-	+	-
	tape	-	+	-	+	-	+	-
File generation group	**Index public, FGen public**	-	+	+	+	+	+	+
	Index public, FGen tape	-	+	+	+	-	+	+
	Index private, FGen private	-	+	+	+	-	+	-
Job variable	**permanent**	+	+	+	+	-	+	+
	temporary	-	-	-	-	-	-	-
Library element[4]		-	-	+	-	-	-	+
FITC port		-	-	-	-	-	-	+
Storage classes		-	-	-	-	-	-	+
HSMS management classes		-	-	-	-	-	-	+

Table 32: Protection mechanisms for objects

[1] If the file is a library, see "Special cases when accessing libraries"
[2] + means: Protection mechanism can be used
[3] - means: Protection mechanism cannot be used
[4] see "Special cases when accessing libraries"

As the table shows, some objects can be protected with multiple protection mechanisms. Of the ACCESS/USER-ACCESS, BACL and GUARDS protection mechanisms, only one can be in effect for an object at any time (see below). All other protection mechanisms are also valid.

Hierarchy of file access protection mechanisms

When the ACCESS/USER-ACCESS, BACL and GUARDS protection mechanisms are applied to the same object at the same time, contradictory situations can arise. The following hierarchy is used to prevent this from happening:

- If protection with a guard is specified for an object, then only the access conditions specified for the guards apply. Any BACL conditions defined for the object as well as the protection attributes ACCESS/USER-ACCESS are ignored.
- If no protection with a guard is specified for an object, but BACL is, then the protection settings specified in the BACL are valid. The protection attributes ACCESS and USER-ACCESS are ignored.
- If the object is not protected with guards or BACL then the ACCESS and USER-ACCESS protection attributes are used as the protection mechanism.

Limited pubset access, password protection, file encryption and the retention period apply additionally in all cases.

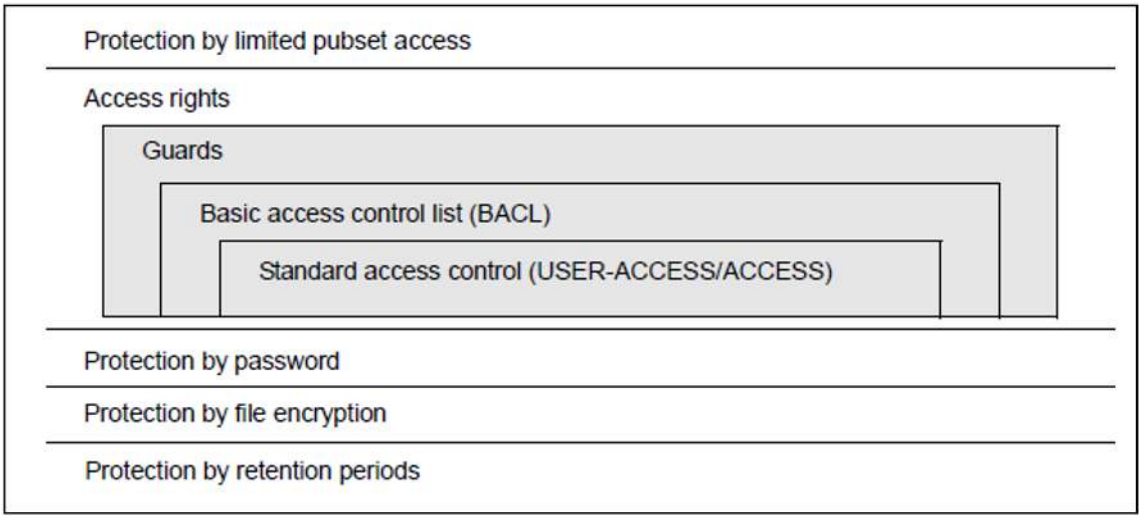


Figure 19: Protection mechanisms for files on a pubset

Every file and every job variable can be protected using one or more protection mechanisms. The access rights have the following priorities:

1. GUARDS
2. BACL
3. Standard access control

One and only one of the three protection mechanisms is used for checking access authorization. This is namely the one that is the highest in the hierarchy **and** is active.

Special cases when accessing libraries

PLAM library **files** can be protected as a whole like a file. Library **elements** can be protected using the LMS statement MODIFY-ELEMENT-PROTECTION regardless of the library file protection.

You must therefore note the following when accessing libraries and library elements:

- Access to individual library elements is controlled by the protection mechanism specified with MODIFY-ELEMENT-PROTECTION. Regardless of the element protection, this type of access is only possible, however, when permitted to read the entire library file.
- When accessing an entire library (with ARCHIVE, with File Transfer or with the DMS command COPY-FILE), the following applies:
  - If the library is not protected by a BACL or guards, then you can access it just like any other file.
  - The access rules in the following table apply when accessing a library that is protected by a BACL or a guard:

		Library contains at least one element that is protected by a BACL or a guard	Library contains **no** elements that are protected by a BACL or a guard
Access by	**Owner**	* [1]	*
	Co-owner	*	*
	Others	Access prohibited	*

Table 33: Access rules when accessing libraries

[1] * means: Access depends on access rules of the library as a whole

12.3 Data access control in BS2000

Command	Meaning
ADD-CRYPTO-PASSWORD	Crypto password for decrypting encrypted file contents is stored in the task's password table
ADD-PASSWORD	Local task entry of a password in the password table to permit access to password-protected files
CREATE-FILE	Define the name and attributes of a new file
CREATE-FILE-GROUP	Define the name and attributes of a new file generation group
CREATE-JV [1]	Create a new job variable
DECRYPT-FILE [2]	Convert an encrypted file into an unencrypted file
ENCRYPT-FILE [2]	Convert an unencrypted file into an encrypted file and define crypto password
MODIFY-FILE-ATTRIBUTES	Define the protection attributes of a file
MODIFY-FILE-GROUP-ATTRIBUTES	Define the protection attributes of a file generation group
MODIFY-JV-ATTRIBUTES [1]	Define the protection attributes of a job variable
REMOVE-CRYPTO-PASSWORD	Crypto password is removed from the current task's password table
REMOVE-PASSWORD	Remove password from the password table
SHOW-FILE-ATTRIBUTES	Request information on the protection attributes of a file
SHOW-JV-ATTRIBUTES [1]	Request information on the protection attributes of a job variable

Table 34: Overview of commands for access control in BS2000

[1] These commands require the software product JV
[2] These commands require the BS2000 subsystem CRYPT

12.3.1 Passwords and retention periods

When a file is generated or edited using the command CREATE-FILE or MODIFY-FILE-ATTRIBUTES then it is possible to assign a password for each access mode (READ, WRITE, EXEC) (Operand `PROTECTION=*PAR (<accessmode> PASSWORD=...)`).

`READ-PASSWORD=`	specified for read access(`*NONE`: no password necessary)
`WRITE-PASSWORD=`	specified for write access(`*NONE`: no password necessary)
`EXEC-PASSWORD=`	specified for execute access(`*NONE`: no password necessary)

When file attributes are modified using the MODIFY-FILE-ATTRIBUTES command, the previously specified passwords must be entered in accordance with the applicable password hierarchy (WRITE-PASSWORD – READ-PASSWORD – EXEC-PASSWORD).

When a file is edited (commands ADD-FILE-LINK-EXPIRATION-DATE=... or MODIFY-FILE-ATTRIBUTES..., PROTECTION=*PARAMETERS(EXPIRATION-DATE=...), it is possible to assign or modify a retention period.

Until this period has expired, any write access (by means of the “w” attribute in the file’s Basic Action Control List or because of an ACCESS= *WRITE in the case of standard access control) is ineffective. The file is therefore only available for read access.

> When a file protected by a read and/or execute password is converted, the file loses the protection by the read and/or execute password in the process.

12.3.2 File encryption

File encryption with a crypto password enables the content of a file to be protected against unauthroized access – also against people with the TSOS privilege and also against physical access to disks and backup tapes.

An encrypted file is created by converting a normal file using the ENCRYPT-FILE command. In the process the crypto password is defined and the encryption method set in the system parameter FILECRYP is transferred to the file's catalog entry.

The DECRYPT-FILE command reconverts encrypted files to unencrypted format. This can only be done after the correct crypto password has been specified.

Crypto password

A crypto password is up to 8 characters long and is not case-sensitive.

To specify the access authorization for an encrypted file, the associated crypto password is entered in the local task crypto password table using the ADD-CRYPTO-PASSWORD command.

A crypto password is explicitly removed from the password table using the REMOVE-CRYPTO-PASSWORD command (otherwise implicitly when the session is terminated).

> ¡ Decryption is impossible if the correct crypto password is not specified. To guard against the crypto password being lost, the following measures are recommended:
>
> - Keeping the specified crypto passwords in a safe place
> - Keeping the number of the used crypto passwords small. These include:
>   - Always protect associated encryption files with the same crypto password (users: ENCRYPT-FILE with a reference file entry).
>   - In addition to specific encrypted sample files, only allow encrypted files with the same crypto passwords as these sample files (system parameter FREFCRYP, see "Options for management via system parameters").

Working with encrypted files

In the event of **homogeneous transfer** of an encrypted file, the encrypted content is transferred one-to-one to a target file which has the same encryption attributes as the source file.

This homogeneous transfer is used for:

- homogeneous COPY-FILE
- saving and restoring (SAVE/RESTORE) with HSMS/ARCHIVE
- migrating and recalling (MIGRATION/RECALL) with HSMS
- exporting and importing with HSMS/ARCHIVE
- moving files within an SM pubset to another volume set
- homogeneous file transfer

Thus in the event of homogeneous transfer no decryption takes place, and no key and no crypto password are required. For transfer actions this means that in particular systems support can work in the same way with encrypted files as with unencrypted files.

The SECURITY section in the ENCRYPTION field of the SHOW-FILE-ATTRIBUTES command's output displays whether or with which encryption method (AES or DES) a file is encrypted. A file selection is offered in accordance with the values of the ENCRYPTION operand.

In the event of remote access via RFA , the ADD- and REMOVE-CRYPTO-PASSWORD commands are automatically forwarded to all connected RFA partner processes by the calling task.

DAB supports read caching of encrypted and unencrypted files, but not write caching. Read caching is advantageous for encrypted files in order to shorten the access times which are increased by the encryption method.

Restrictions and special aspects

- The link to the openCrypt subsystem means that access to encrypted files is possible as of "System Ready".
- Job variables, tape files, EAM files, files on private disk and TSOS files on the home pubset are not encrypted.
- Encrypted files cannot be printed. An encrypted file must first be decrypted beforehand.
- Encrypted files have no read/execute password. However, they can have a write password and can also be combined with the other access control mechanisms.
- It is not possible to modify the crypto password with the MODIFY-FILE-ATTRIBUTES command. Modifications to READ- or EXEC-PASSWORD are ignored.
- When PAMCONV is used or the REPAIR-DISK-FILE command is invoked for an encrypted file, it is necessary to specify the crypto password.

Options for management via system parameters

1. FILECRYP

   The system parameter FILECRYP determines the encryption method for conversion into an encrypted file using the ENCRYPT-FILE command. The encryption methods supported are AES (default) and DES.

   - With ENCRYPT-FILE the current value of the system parameter FILECRYP is taken over into the file's encryption attributes.
   - When a file that has already been encrypted is accessed, the encryption method is not taken over from the system parameter FILECRYP, but from the file's catalog entry.
   - A change to the system parameter FILECRYP only becomes effective for a file that was already encrypted at the time of the change when the file is decrypted and then encrypted again.

   In the case of shared pubset mode with encrypted files, the selection of the particular encryption methods should be uniform in the system parameter FILECRYP of the systems involved.

2. FREFCRYP

   The system parameter FREFCRYP is available for controlled assignment and limiting the number of the crypto passwords used . If not empty, it contains a selected user ID. It is then only possible for files from this user ID to be converted into encrypted files (ENCRYPT-FILE command) with free definition of a crypto password. To convert files from other user IDs, a reference file which is already encrypted must be specified. The set of crypto passwords used is thus limited to that of the encrypted files from the selected user ID.

3. PWACTIVE

The system parameter PWACTIVE is used to define the maximum number of crypto passwords which a crypto password table may contain. If the threshold value is reached, message `DMS0691` is issued and no further crypto password can be entered for the current task before at least one of the crypto passwords in the crypto password table has been removed.

4. PWENTERD

   The system parameter PWENTERD is used to define how many crypto passwords may be entered per task. If the threshold value is reached, message `DMS0692` is issued and no further crypto password can be entered for the current task.

5. PWERRORS

   The system parameter PWERRORS is used to define the maximum number of invalid crypto access attempts are tolerated under a task. If the threshold value is reached, a SAT entry may be written, the message `DMS0693` issued on the console and the task terminated abnormally.

6. PWPENTI

   The system parameter PWPENTI is used to define the time penalty for the invalid crypto access attempts tolerated.

> **i** All the threshold values mentioned above apply both for the entry of crypto passwords and for the entry of file passwords (READ, WRITE, EXEC). In contrast to the file passwords, for whose entry privileged systems support possesses the special right to exceed the defined threshold values, no special rights are granted for the entry of crypto passwords.
>
> Crypto password tables and crypto password counters are maintained separately from the file password tables and file password counters. If, for example, the maximum possible number of READ, WRITE and EXEC password entries under a task has been reached, further crypto passwords can be entered, and, by the same token, the same applies in this situation for file passwords.
>
> The ENCRYPT parameter only applies for the encryption of file passwords, not for the encryption of crypto passwords. The latter are always stored in the crypto password table with one-off encryption.

For further information on using encrypted files, please refer to the “Introductory Guide to DMS” [19].

12.3.3 Standard access control (ACCESS/USER-ACCESS)

Standard access control is regulated by means of the ACCESS and USER-ACCESS operands in the CREATE and MODIFY commands (see above).

Standard access control via the protection attributes ACCESS and USER-ACCESS offers itself whenever higher-level access protection by means of an access control list or basic access control list is not desired.

If neither a BACL nor a guard is defined, standard access control automatically becomes effective. In addition, the passwords and retention period are always checked.

Protection attribute **ACCESS**

Write access or just read access for a file can be specified by setting the ACCESS protection attribute. Write access also implies read access.

Protection attribute **USER-ACCESS**

The protection attribute USER-ACCESS can be used to specify whether only the owner (*USER-ONLY) or all other users (*ALL-USERS, including or excluding the user with the privilege of HARDWARE-MAINTENANCE for online maintenance) may access a file.

Note for the user ID with the privilege HARDWARE-MAINTENANCE

File access is granted to the user ID with this privilege only under the following conditions:

- If the file is protected by guards, then access conditions must be specified in the guard that allow the privileged user ID access.
- If the file is not protected with guards but instead using a basic access control list (BACL), then this list must allow privileged users access.
- When the file is not protected by guards or BACL, then USER-ACCESS=*SPECIAL must be set.

12.3.4 Basic Access Control List (BACL)

The basic access control list (BACL) is one level above the ACCESS/USER-ACCESS protection attributes in the hierarchy of access protection mechanisms. It takes effect for an object when no guards protection is defined for the object. The password protection and retention period are also in effect.

Different access rights can be defined for the owner of the object, members of its user group and for all other users with a BACL . However, it is not possible to define access rights for individual user IDs with this access protection mechanism.

A basic access control list for files is defined using the BASIC-ACL operand of the CREATE-FILE or MODIFY-FILE-ATTRIBUTES command.

Basic access control lists for job variables can be defined accordingly with the CREATE-JV or MODIFY-JV-ATTRIBUTES commands.

User classes

Building on the concept of user groups, user classes are defined for access to objects. User classes subdivide the set of all users into the subsets OWNER, GROUP and OTHERS.

OWNER	The owner of an object, meaning the user ID under which the file or job variable is cataloged as well as co-owners specified using the co-owner protection mechanism
GROUP	All user IDs of the user group to which the owner belongs except for the owner and any co-owners
OTHERS	All other users except for the co-owners

The definition of the group structure on the home pubset is used to define the user class.

Notes on the GROUP user class

All users that are not explicitly assigned to any group are automatically members of the implicitly defined group *UNIVERSAL. This is especially true when no groups at all were set up. In this case all users of the system are members of the same group. When a BACL is evaluated, all user IDs except for the owner himself are granted the access rights of the OTHERS entry.

It is therefore urgently recommended for members of the *UNIVERSAL user group to assign the same access rights to the GROUP and OTHERS user classes.

Access rights

Nine access rights for a file are specified in a BACL. For each of the three user classes OWNER, GROUP and OTHERS three access types can be separately assigned:

- read (R),
- write (W) and
- execute (X)

None of these access rights automatically includes the other access rights.

Evaluation of the basic access control list

1. If the user ID requesting access is the owner of the object, a co-owner or the TSOS, the access rights stored under OWNER apply.

2. If the user ID belongs to the owner's user group, the access rights stored under GROUP apply.
3. For all other user IDs the access rights stored under OTHERS apply.

Example

OWNER = R W X

GROUP = R W -

OTHERS = R - -

The owner of this file may perform read, write and execution operations on the file. The file owner's group may read from and write to the file. All other users can only read the file.

12.4 Privileges

System privileges (privileges for short) describe the right to handle certain systems support tasks together with the required system functions required for this under a particular user ID.

Privileges can be allocated to various user IDs. On the one hand, this allocation reduces the load on systems support. On the other, it increases security in the context of systems support because, for example, the number of individuals who need to know the TSOS password in order to perform routine operations is smaller.

By default the privileges are allocated to predefined user IDs, see table 36 in section "Allocation of privileges". The default allocation of privileges can only be changed with SECOS.

Each command must be declared in one of the activated system syntax files and explicit permission must be given for its use. Before it is processed, each command (user, system administration or operator command) passed to SDF is checked to see whether the user who issued it has the privilege required for its execution. In the case of operator commands, for example, this is the privilege OPERATING.

The figure below takes a few commands as examples to illustrate how authorization to issue user, operator and system administration commands to BS2000 is assigned:

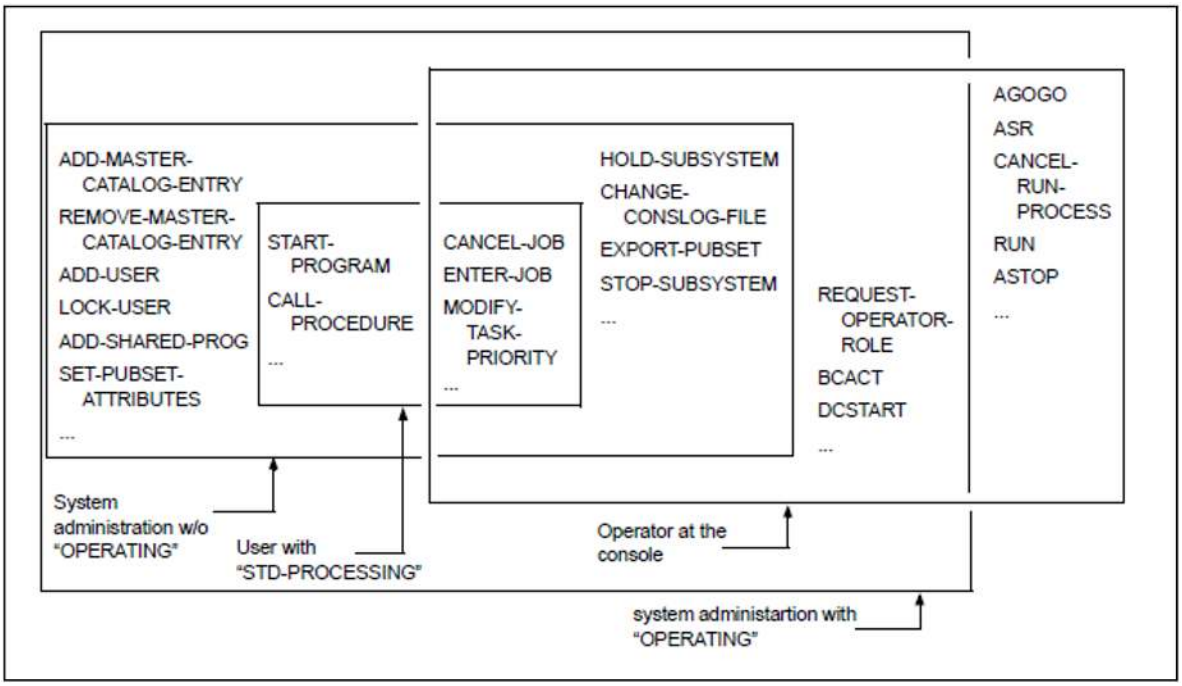


Figure 20: Authorization to enter commands

Certain commands can be given both by the operator (via the console) and by system administration (from a data display terminal under TSOS user ID). Thus there is no rigid division of functions between system administration and operator. There is a certain latitude in the organization of the data center. But close cooperation between system administration and operator is always required.

Privilege allocation using SECOS

Each user ID in BS2000 is allocated at least one privilege. By default this is the STD-PROCESSING privilege, i.e. the right to execute the user commands.

A user ID can have more than one privilege, and SECOS can be used to allocate a privilege to more than one user ID.

SECOS enables individual privileges to be grouped for certain tasks. A grouping of this type is implemented by defining “privilege sets” to which the various (individual) privileges can be allocated.

The privileges of a user ID are stored in the user catalog (SYSSRPM). Privilege allocation in the user catalog of the home pubset is effective throughout the system. A pubset’s user catalog is opened when the pubset concerned is imported and remains open until the pubset is exported.

The various privileges are described in the section "Description of the privileges".

12.4.1 Privileges of the user ID TSOS

The **TSOS** privilege is immutably linked to the user ID TSOS and can neither be withdrawn from this user ID nor assigned to another user ID. The privileges that are assigned as standard to the TSOS user ID are shown in the table 36 in section "Allocation of privileges".

> According to the BS2000 convention, product files or files created by the system under the TSOS user ID start with the string “SYS” (SUs /390), “SKM” or “SKU” (for SUs x86). It must be ensured that they are not inadvertently overwritten or deleted.

User commands

The TSOS user ID is always the co-owner of all the files and job variables of all the users as long as the users themselves have not denied the TSOS user co-ownership.

The full scope of user commands is available to systems support.

TSOS is authorized to make read and write accesses to all entries in the file/user catalog. The password protection of user files is suspended for systems support inasmuch as TSOS is authorized either to ignore the passwords or to obtain them if necessary. When SECOS is used the owner can use a condition guard to protect any data requiring security in order to deny the TSOS user access.

Access to all files also covers temporary files that systems support may create under any desired catalog/user ID. These files, however, are **not** automatically deleted upon LOGOFF or EXIT-JOB. Systems support itself is responsible for deleting these files. Upon a partially qualified specification of file names, temporary files are taken into account in accordance with their internal representation.

Commands with privileges other than TSOS

The TSOS user ID can only execute the commands for which it has the associated privilege.

In particular, no operator commands can be issued under TSOS because by default TSOS does not have the OPERATING privilege. The default allocation of privileges can only be changed with SECOS.

Macros

Privileges for creating and modifying files at the command level also apply to the corresponding macro calls. The privileged operands of the macros are described in detail in the “Executive Macros” [30] and “DMS Macros” [20] manuals.

Job Variables

The following commands are available only in conjunction with the software product JV and are described in detail in the “Job Variables” manual [26].

The TSOS user ID is always the co-owner of all the files and job variables of all the users as long as the users themselves have not denied the TSOS user co-ownership.

Systems support can use the following commands to process job variables of other user IDs:

```
CREATE-JV
DELETE-JV
MODIFY-JV
MODIFY-JV-ATTRIBUTES
MODIFY-JV-CONDITIONALLY
REMOVE-JV-LINK
SET-JV-LINK
SHOW-JV
SHOW-JV-ATTRIBUTES
SHOW-JV-LINK
SHOW-CJC-STATUS
```

Any file protection in the form of passwords is displayed only to the privileged caller under the TSOS user ID.

12.4.2 Privileges for operating

The **OPERATING** privilege is by default assigned to the SYSOPR user ID. The default allocation of privileges can only be changed with SECOS.

Operating is thus possible from both physical and virtual consoles and from user tasks with the OPERATING privilege.
The command authorization concept via authorization codes remains unchanged for consoles. For user tasks it is now irrelevant (exception: commands which are processed via the UCON task and to whom the authorization code $ is assigned see also "Command definition").

Certain commands that are usually always available to operators may be locked in the system syntax file by systems support (see also the “SDF-A” manual [44]).
A task with the OPERATING privilege offers most operator commands, including those that could previously only be entered via the console. Among the commands not offered are CONSOLE, ASR, RUN and AGOGO.

Each operator command entered at a console is managed by an OPR task that identifies itself to SDF and to the system as privileged when a command is executed. Before it is processed, each operator request (command passed by the OPR task to SDF for syntax and privilege verification) is checked to see whether the caller has the necessary OPERATING privilege.
The OPR task of a physical console (with NBCONOPI=NO) and a virtual console always has the OPERATING privilege.

The operator can enter commands via the console. The abbreviated form, if any, is given in the command syntax. For the structure of the command entries and messages, see chapter "Operator functions" and chapter "Automation of operator functions".

The commands are described in the “Commands” manual [27]. The BCAM commands for the operator (e.g. BCACT, DCSTART) are described in the “BCAM” manual [4].

Under certain conditions the operator may also use some system administration commands and user commands (see section "System administration functions performed by the operator").

12.4.3 Description of the privileges

BS2000 knows the following privileges.
They are described after this table.

Administration area	Name of the privilege
ACS administration	ACS-ADMINISTRATION
(Not predefined; determined by systems support)	CUSTOMER-PRIVILEGE-1 : CUSTOMER-PRIVILEGE-8
File transfer administration	FT-ADMINISTRATION
FTAC administration	FTAC-ADMINISTRATION
Global guard administration	GUARD-ADMINISTRATION
Online hardware maintenance	HARDWARE-MAINTENANCE
HSMS administration	HSMS-ADMINISTRATION
Network administration (obsolete)	NET-ADMINISTRATION
Notification service administration	NOTIFICATION-ADMINISTRATION
Operating	OPERATING
POSIX user management	POSIX-ADMINISTRATION
SPOOL administration	PRINT-SERVICE-ADMINISTRATION
PROP-XT administration	PROP-ADMINISTRATION
Audit file analysis	SAT-FILE-EVALUATION
Audit file administration	SAT-FILE-MANAGEMENT
Security administration	SECURITY-ADMINISTRATION
Execution of user commands	STD-PROCESSING
Subsystem management	SUBSYSTEM-MANAGEMENT
Software monitor administration	SW-MONITOR-ADMINISTRATION
Tape administration	TAPE-ADMINISTRATION
Encryption key administration for tapes	TAPE-KEY-ADMINISTRATION
TSOS	TSOS
User administration	USER-ADMINISTRATION

VM administration	VIRTUAL-MACHINE-ADMINISTRATION
VM2000 administration	VM2000-ADMINISTRATION

Table 35: Administration areas and associated privileges

ACS administration

The user ID which has the **ACS-ADMINISTRATION** privilege may exercise the following rights within the scope of the ACS (Alias Catalog Service) function:

- define system-wide preset values and restrictions for the use of an alias catalog
- make and amend the declarations relating to the ACS system files
- exercise the extended functions of individual ACS commands

By default this privilege is assigned to the TSOS ID.

The extent of the rights and functions for the user ID with the ACS-ADMINISTRATION privilege are described in section "ACS: Alias catalog system".

CUSTOMER-PRIVILEGE-1..8

By assigning the privileges **CUSTOMER-PRIVILEGE-1** or CUSTOMER-PRIVILEGE-2 or CUSTOMER-PRIVILEGE-3 etc. up to **CUSTOMER-PRIVILEGE-8**, systems support can make access to commands and statements flexible for certain user IDs.

By default, these privileges are not assigned to any ID.

File transfer administration

The file transfer administration (the user ID with the privilege **FT-ADMINISTRATION**) may manage the job and network description journal of the software product openFT (see the “openFT” manual [23]).

FTAC administration

The FTAC administration (the user ID with the privilege **FTAC-ADMINISTRATION**) may manage the protection functions of the software product openFT-AC (see the “openFT” manual [23]).

Global guard administration (GUARD-ADMINISTRATION)

The global guard administration with the **GUARD-ADMINISTRATION** privilege has the right to administer guards of any type on all local pubsets and to save and restore guards for any number of users using the GUARDS-SAVE program. This means that a user ID with this privilege is co-owner of all guards in the system.
By default this privilege is assigned to the TSOS ID.

Online hardware maintenance

The **HARDWARE-MAINTENANCE** privilege includes the right to execute online hardware maintenance. In particular, it covers the following functions:

- maintenance of the hardware fault statistics file
- running of statistics and trace programs under the control of BS2000 simultaneously with the user programs

By default this privilege is assigned to the user ID SERVICE.

If the HARDWARE-MAINTENANCE privilege is assigned to any chosen user ID, the following applies:

- A user ID with the HARDWARE-MAINTENANCE privilege is authorized to access files of other user IDs when the following applies:
  - If the file is protected by guards, then access conditions must be specified in the guard that allow the privileged user ID access.
  - If the file is not protected by guards but is protected by a basic access control list, then this list must allow the privileged user ID access.
  - When the file is not protected by guards or BACL, then USER-ACCESS=*SPECIAL must be set.
- Make sure that this user ID has all the necessary access rights.

For further information on online hardware analysis, see the “Diagnostics Handbook” [14].

HSMS administration

The user ID with the **HSMS-ADMINISTRATION** privilege may perform actions in BS2000 relating to the administration of the Hierarchical Storage Management System (see the “HSMS” manual [24]).

By default this privilege is assigned to the user IDs TSOS and SYSHSMS.

Network administration

A user job with the **NET-ADMINISTRATION** privilege may perform network administration functions, including the redefinition of station names.

By default this privilege is assigned to the TSOS ID.

Notification service administration

The **NOTIFICATION-ADMINISTRATION** privilege enables the Notification Service to be configured, i.e. it permits definition of the products which may use the Notification Service and of which methods are supported here for notification.

By default this privilege is assigned to the TSOS and SYSSNS user IDs.

The Notification Service in BS2000 is a mechanism which can be employed to notify users when certain events occur. Currently this functionality is used by SPOOL. Users can be notified by e-mail if certain events occur with their print jobs, for example job termination. See also the SNS manual [51].

Operating

The **OPERATING** privilege allows operator tasks to be performed. This means that operator functions can be performed from user workstations. For basic system operation, however, operation from the (normal) console is still essential.

By default this privilege is assigned to the user ID SYSOPR. See also the section "Privileges for operating".

POSIX user management

The **POSIX-ADMINISTRATION** privilege guards against access to POSIX attributes which are managed by BS2000 user administration and to the tool for installing additional products in POSIX. The POSIX attributes can be protected as follows:

- by managing the POSIX user attributes of all the user IDs on all local pubsets. This authorization is a subset of the USER-ADMINISTRATION privilege.
- by using privileged subfunctions of POSIX-SVC

By default this privilege is assigned to the user ID SYSROOT and cannot be withdrawn from it.

See also the chapter "POSIX user administration".

SPOOL administration

The **PRINT-SERVICE-ADMINISTRATION** privilege allows SPOOL administration tasks to be performed. These include:

- starting and stopping SPOOL devices such as printers or tapes
- changing the SPOOL parameters with the SPSERVE utility routine
- changing print control files with the PRM utility routine
- distributed access to networked printed with Dprint
- managing the print jobs of all users

By default this privilege is assigned to the TSOS, SYSSNS and SYSSPOOL IDs.

PROP-XT administration

The **PROP-ADMINISTRATION** privilege allows PROP-XT system commands to be executed. For PROP-XT administration, see also the “PROP-XT” manual [40].

By default this privilege is assigned to the TSOS ID.

Evaluation of the audit files

The **SAT-FILE-EVALUATION** privilege grants the right to evaluate the SATLOG files and the CONSLOG files. SAT logging is always active for the owner of the SAT-FILE-EVALUATION privilege but can be explicitly disabled.

By default this privilege is assigned to the user ID SYSAUDIT.

Management of the audit files

The management of audit files requires the privilege **SAT-FILE-MANAGEMENT** which permits the user:

- to manage the files generated by SAT (Security Audit Trail) (in particular, to switch the SATLOG file using the CHANGE-SAT-FILE command)
- to request the current status of the REP logging file, using the SET-REPLOG-READ-MARK command
- to call up information about the current status of the EVENTING selection
- to evaluate the SATLOG files and the CONSLOG files.

SAT logging is always active for the owner of the SAT-FILE-MANAGEMENT privilege but can be explicitly disabled. By default this privilege is assigned to the user ID SYSAUDIT.

Security administration

Security administration (the security administrator) is authorized to manage privileges, to manage the operator roles and to control SAT logging (auditing). For the user ID with the **SECURITY-ADMINISTRATION** privilege, logging via SAT is always active, and cannot be deactivated.

The privilege administration may manage the individual privileges, i.e.:

- assign privileges to user IDs on all pubsets
- withdraw privileges from user IDs on all pubsets
- assign and withdraw privilege groups on all pubsets
- request information on the current allocation of privileges to privilege groups

In connection with the management of operator roles, the security administrator can:

- define, modify and delete operator roles
- assign operator roles to user IDs, and withdraw them
- request details of the current definition and distribution of the operator roles

The SECURITY-ADMINISTRATION privilege cannot be assigned to or withdrawn from a user ID by a command. In a system which does not have SECOS, the privilege is permanently linked to the SYSPRIV user ID (which, like TSOS, is always contained in the SYSSRPM user catalog); in a system with SECOS, the startup parameter service can be used to assign the privilege to any required user ID (see the “SECOS” manual “Access Control” [46]).

Execution of user commands

The **STD-PROCESSING** privilege assigns the right to execute the commands described in the “Commands” manual [27], and the nonprivileged commands of software products which do not form part of BS2000 OSD/BC.

When a user ID is first set up (using ADD-USER), the system assigns it the STD-PROCESSING privilege by default. The STD-PROCESSING privilege is also issued by default to the user IDs created by the system at the time of first startup (except for the SERVICE, SYSAUDIT and SYSPRIV user IDs).

Although it is a valid rule that a user ID which has a privilege cannot be deleted, when a user ID is deleted (using REMOVE-USER) the fact that it “possesses” the STD-PROCESSING privilege is disregarded; i.e. a user ID can be deleted if there is no privilege other than this one assigned to it.

Subsystem management

The user ID which has the **SUBSYSTEM-MANAGEMENT** privilege may perform the tasks for global management of the subsystems of the dynamic subsystem management **DSSM** system:

By default this privilege is assigned to the TSOS ID.

Software monitor administration

Any user ID which has the **SW-MONITOR-ADMINISTRATION** privilege may start, terminate and administer the software monitors openSM2 and COSMOS (see the “openSM2” manual [49]).

By default this privilege is assigned to the TSOS ID.

Tape administration

The tape administration (the user ID which has the **TAPE-ADMINISTRATION** privilege) may execute the administrative functions for the tape software product MAREN. I.e. it may manage the MAREN catalog (see the “MAREN” manual [31]).

By default this privilege is assigned to the user IDs TSOS and SYSMAREN.

Encryption key administration for tapes

The encryption key administration (this is the user ID which has the **TAPE-KEY-ADMINISTRATION** privilege) may execute the statements of the MARENEKM (MAREN Encryption Key Manager) program. This means that it may administer the encryption keys for tapes (see the “MAREN” manual [31]).

By default this privilege is assigned to the user ID SYSMAREN.

TSOS

The **TSOS** privilege enables the TSOS user ID to execute system administration functions.

The TSOS privilege is immutably linked to the user ID TSOS and can neither be withdrawn from this user ID nor assigned to another user ID.

User administration

The system-global user administration (i.e. all the user IDs with the **USER-ADMINISTRATION** privilege) may perform actions relating to user or user-group administration on all local pubsets (for all users or user groups). There are no restrictions on the allocation of resources and privileges of the user catalog entry (e.g. START-IMMEDIATE, NO-CPU-LIMIT,...) to user IDs and user groups.

By default this privilege is assigned to the TSOS ID.

VM administration

The **VIRTUAL-MACHINE-ADMINISTRATION** privilege gives the right to operate a virtual machine (VM), except for the monitor VM. It is allowed to issue VM administrator commands for the privilege owner's own VM (see the “VM2000” manual [60]).

By default this privilege is assigned to the TSOS ID.

VM2000 administration

The **VM2000-ADMINISTRATION** privilege gives the right to administer the monitor VM and to enter all the VM2000 commands (see the “VM2000” manual [60]).

By default this privilege is assigned to the TSOS ID.

12.4.4 Allocation of privileges

If a system is started with a first startup, a new user catalog SYSSRPM is created. By default, certain predefined user IDs have certain privileges:

Privilege	TSOS	SERVICES	SYSAUDIT	SYSHSMS	SYSMAREN	SYSOPR	SYSPRIV [1]	SYSROOT	SYSSNS	SYSSPOOL	SYSWSA	Others [2]
ACS-ADMINISTRATION	X [3]	- [4]	-	-	-	-	-	-	-	-	-	-
CUSTOMER-PRIVILEGE-1...8	-	-	-	-	-	-	-	-	-	-	-	-
FT-ADMINISTRATION	X	-	-	-	-	-	-	-	-	-	-	-
FTAC-ADMINISTRATION	X	-	-	-	-	-	-	-	-	-	-	-
GUARD-ADMINISTRATION	X	-	-	-	-	-	-	-	-	-	-	-
HARDWARE-MAINTENANCE	-	X	-	-	-	-	-	-	-	-	-	-
HSMS-ADMINISTRATION	X	-	-	X	-	-	-	-	-	-	-	-
NET-ADMINISTRATION	X	-	-	-	-	-	-	-	-	-	-	-
NOTIFICATION-ADMINISTRATION	X	-	-	-	-	-	-	-	X	-	-	-
OPERATING	-	-	-	-	-	X	-	-	-	-	X	-
POSIX-ADMINISTRATION	-	-	-	-	-	-	-	X	-	-	-	-
PRINT-SERVICE-ADMINISTRATION	X	-	-	-	-	-	-	-	X	X	-	-
PROP-ADMINISTRATION	X	-	-	-	-	-	-	-	-	-	-	-
SAT-FILE-EVALUATION	-	-	X	-	-	-	-	-	-	-	-	-
SAT-FILE-MANAGEMENT	-	-	X	-	-	-	-	-	-	-	-	-
SECURITY-ADMINISTRATION	-	-	-	-	-	-	X	-	-	-	-	-
STD-PROCESSING	X	-	-	X	X	X	-	X	X	X	X	X
SUBSYSTEM-MANAGEMENT	X	-	-	-	-	-	-	-	-	-	-	-
SW-MONITOR-ADMINISTRATION	X	-	-	-	-	-	-	-	-	-	-	-
TAPE-ADMINISTRATION	X	-	-	-	X	-	-	-	-	-	-	-
TAPE-KEY-ADMINISTRATION	-	-	-	-	X	-	-	-	-	-	-	-
TSOS	X	-	-	-	-	-	-	-	-	-	-	-
USER-ADMINISTRATION	X	-	-	-	-	-	-	-	-	-	-	-

VIRTUAL-MACHINE-ADMINISTRATION	X	-	-	-	-	-	-	-	-	-	-	-
VM2000-ADMINISTRATION	X	-	-	-	-	-	-	-	-	-	X	-

Table 36: Allocation of privileges after first startup (default allocation of privileges)

[1] If an ID other than SYSPRIV has been specified in the startup parameter file as the user ID for the security administrator, this column applies to precisely this user ID. In this case, the SYSPRIV user ID must be entered under “Others”.

[2] the system IDs SYSDUMP, SYSGEN, SYSNAC, SYSSOPT, SYSSNAP, SYSUSER

[3] X means: The privilege is assigned to the user ID by default

[4] - means: The privilege is not assigned to the user ID by default

Allocation of privileges after non-first startup with SECOS

The default allocation of privileges can only be changed with SECOS.

If after a shutdown in a system with BS2000 a startup occurs in the system with the same version by means of cold start, warm start, selective start or ZIP start, the distribution of privileges is the same as before the last shutdown.

However, if simultaneously with the startup there is a first-time change of version from a lower version to the current version of BS2000 OSD/BC (version upgrade), the new privileges are distributed to the IDs to which they would also have been assigned on first start.

Privilege allocation after pubset import

Privilege allocation after a pubset import with ACTUAL-JOIN=*FIRST is the same as after a first startup; after a different type of pubset import it is the same as after a non-first startup.

If the user catalog is reconstructed at the same time, the privileges are restored to all users who are transferred from the old user catalog (saved in the $TSOS.SYSSRPM.BACKUP file) to the new one. When the restoration is complete, therefore, these users are assigned the same privileges as they had at the time of the save. Which user IDs are ultimately reconstructed depends on the reconstruction type.

If the old user catalog belongs to a system with SECOS functionality but is to be reconstructed on a system without SECOS, the corresponding default values for the privilege allocation are entered in the new user catalog for all users. This is the same as the privilege allocation after a first startup.

12.5 Limiting resources for users

The user administration can impose stipulations on user groups and user IDs in order to limit the use or prevent the misuse of the following resources:

- utilization of disk storage capacity on the pubsets
- utilization of server main memory
- utilization of server performance (CPU capacity).

User-related allocation of these resources is always performed at the command level with ADD-/MODIFY-USER.

The actual control and checking of the allocation of these resources with regard to the specified values is performed in the operating system (e.g. administration of task categories, control by PCS, management of job streams and job classes, etc.).

> ¡ Analogous control for user groups is possible with SECOS.

Use of the system-global resources can be limited for user IDs by the commands shown in the following table.

Resources used		**Command**	**Operands**
Use of disk storage capacity on the pubsets	Public-Space	ADD-USER/ MODIFY-USER-ATTRIBUTES	PUBSET=..., PUBLIC-SPACE-LIMIT=..., PUBLIC-SPACE-EXCESS=..., FILE-NUMBER-LIMIT=...., JV-NUMBER-LIMIT=..., TEMP-FILE-SPACE=..., DMS-TUNING-RESOURCES=...
Use of server main memory	Address space	ADD-USER/ MODIFY-USER-ATTRIBUTES	ADDRESS-SPACE-LIMIT=...
	Main memory	ADD-USER/ MODIFY-USER-ATTRIBUTES	RESIDENT-PAGES=...
	Task (de-) activation	ADD-USER/ MODIFY-USER-ATTRIBUTES	ACCOUNT-ATTRIBUTES= (MAX-ALLOWED-CATEGORY=..., PRIVILEGE=NO / PARAMETERS (INHIBIT- DEACTIVATION=...))

Use of server performance (CPU capacity)	CPU limit	ADD-USER/ MODIFY-USER-ATTRIBUTES	ACCOUNT-ATTRIBUTES= (CPU- LIMIT=..., PRIVILEGE=NO/PAR- (NO-CPU LIMIT=...))
	Allowed execution priority	ADD-USER/ MODIFY-USER-ATTRIBUTES	ACCOUNT-ATTRIBUTES= (MAXIMUM-RUN-PRIORITY=...)
	Allowed task categories	ADD-USER/ MODIFY-USER-ATTRIBUTES	ACCOUNT-ATTRIBUTES= (MAX-ALLOWED-CATEGORY=...)
	Scheduling right	ADD-USER/ MODIFY-USER-ATTRIBUTES	ACCOUNT-ATTRIBUTES= (PRIVILEGE=NO/PARAMETERS- (START- IMMEDIATE=...))
	Performance measures	ADD-USER/ MODIFY-USER-ATTRIBUTES	DMS-TUNING-RESOURCES=...

12.6 Meeting security requirements using SECOS

Use of the software product SECOS extends BS2000 OSD/BC by functions which guarantee secure operation in accordance with the required security criteria.

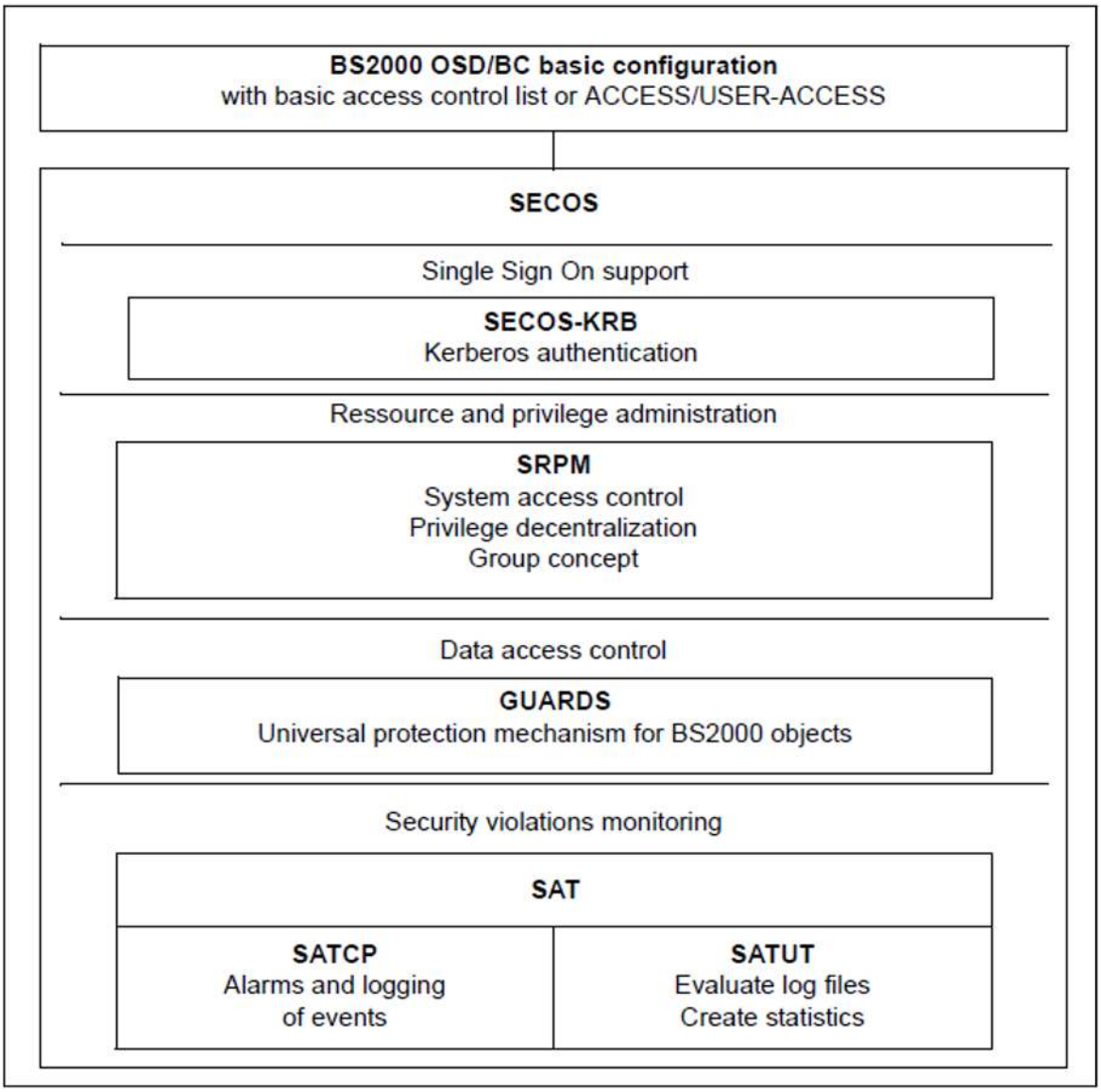


Figure 21: Functional units in the security package

Components of SECOS

SRPM (System Resources and Privileges Management) offers the following basic facilities:

- Decentralization of systems support privileges. This makes it possible to bundle or unbundle systems support tasks, i.e. task assignments can be tailored to each individual data center. Privileges are assigned by the security administrator under a special user ID (see privilege SECURITY-ADMINISTRATION, in "Security administration" in section "Description of the privileges").
- Identification and authentication of users by means of extended access protection (separation and control of access paths, terminal sets) and password protection (limited lifetime, minimum length and complexity).

- Establishment of user groups, with the help of which differentiated access protection is possible. At the same time, administrative tasks can be carried out in decentralized form by group administrators. This means a reduction of trivial tasks for systems support.

GUARDS (Generally Usable Access Control Administration System) allows access to a user's data objects to be made dependent on “access conditions”. To prevent illegitimate access to the data objects, a user can specify access conditions which must be satisfied by any subject before it may access the data objects.

GUARDDEF (default protection) is used to assign default attribute values for files and job variables. Optionally, these values may be predefined for the creation or modification of objects. The settings may be made for the entire pubset by systems support (TSOS) or by individual users under their own user IDs for their own objects. GUARDDEF uses GUARDS to store the settings.

GUARDCOO represents a refinement to the mechanism for co-owner protection (files and job variables can be created under a different user ID and this user ID is also able to administer them).
It is possible to assign co-owner rights for different name areas of the objects, remove them for the TSOS user ID and grant certain privileges to other user IDs or co-owners. GUARDCOO uses GUARDS to store the settings.

SAT (Security Audit Trail) permits auditing by logging security-relevant events in a specially protected file (SATLOG file). The security administrator has the exclusive right to activate/deactivate SAT logging and to switch auditing of user IDs and loggable events on and off.

With the help of the SATUT component, SATLOG files can be evaluated by a specially authorized user having the “SAT-FILE-MANAGEMENT” or “SAT-FILE-EVALUATION” privilege. Evaluation can be set in such a way, for example, that a trace of special processing steps or actions performed by certain user IDs is output. It is thus possible to detect misuse of the system or unauthorized access to secured data.

Kerberos can be used for SSO (Single Sign On) in BS2000.
SSO is implemented in BS2000 with **SECOS-KRB** (Kerberos authentication). Kerberos is an SSO (Single Sign On) security system based on cryptographic encryption methods. When authentication takes place with Kerberos, no passwords are sent over the network in plaintext. This prevents passwords from being intercepted on the network.

The software product SECOS, its individual components and their installation and integration in BS2000 are described in detail in the “SECOS” manuals [46] and [47].

13 Data backup

This chapter presents criteria for selecting backup concepts in BS2000, data backup functions in BS2000 and software products for data backup (ARCHIVE, HSMS and FDDRL), and includes information on how to reconstruct files and volumes.

The chapter concludes with a brief description of the archive systems supported by BS2000 with the ROBAR software product that control them.

13.1 Criteria for selecting backup concepts

The question of data saving is of particular importance to all data centers. Major arguments for a reliable data backup strategy are

- the demand for high data availability
- the possibility of accessing data sets which cannot be stored on public volumes due to limited capacity
- the improvement of response times by reorganizing both public and private volumes
- the desire/necessity to transport data sets to another data center.

In order to meet these requirements, preventive and regular saving of all important data must be carried out at all data centers.

Selection of data

Selection of data for saving generally depends on its so-called save worthiness. Thus a distinction is made between production data, which is required for ongoing production and is subject to continuous modification, and pure test data, which can be reproduced at any time. In addition, system data which normally does not change during operation can be excluded from regular saves. In such cases, a complete backup copy is sufficient to reconstruct the current status if necessary.

Time and frequency of data saving

The data saving methods must be devised in such a way that both the requirement for data security and the requirement for availability of the applications are met. For example, time-consuming saves - logical or physical total saves - should be strategically shifted to times of low workload in order to minimize the loss of usable processor time.
Furthermore, the data saving method used must ensure that redundant inventories of saved data are avoided. This means that each update status of a file should be saved once only.

Types of save

- **Total save**

  In a total save all of the files which are designated by the selection criteria and which are closed at the time of the backup are marked. These files are completely saved, irrespective of whether or not they have changed since the last backup.

  A total save is sometimes called a “full save”. Physical saves are always full saves.

- **Incremental saves**

  In an incremental save, only those files that have changed or been created since the last backup are saved. These files are completely saved. The same directory file must be used in which the information on the file version was stored.

- **Partial saves**

  In a partial save (a special form of the incremental save), only those PAM pages that have been changed since the last complete backup are saved for selected files. The other files are not saved at all. In order to reconstruct a partially saved file, the last partial save and the last total save are required.

  A partial save is frequently also called a “partial backup“.

Scope of data saving

The scope of data saving is determined by systems support on the basis of the criteria data inventory, consistency of the data, server load and configuration.

- **Data inventory**

  The number and size of the data to be managed and saved has a direct influence on the scope of data saving. If the data inventory is small, systems support can forego partial or incremental saves and provide for regular logical or physical total saves of the system. Although unchanged data is saved time and again, the overall “data packet” is always consistent and will not have to be reconstructed from various saved data inventories. In the case of extensive data inventories, systems support must devise a strategic concept for saving data. All files can be saved successively by arranging a sequence of several partial and incremental saves; At weekly intervals, for example, a logical or physical total save can be performed.
  Files can be assigned different backup levels or management classes. These assignments can then be used by systems support to limit the scope of the backup.

- **Consistency of the data**

  The data inventory must also be subject to a qualitative check with respect to the scope of the save.

  The quality attributes of system and user files include the following:

  - the number of accesses
  - the frequency of updates
  - the scope of updates
  - the frequency with which program versions are exchanged
  - the assignment to a backup level or management class

  If the data inventory is limited for the most part to constant, stable versions that are rarely or never updated, the current status can also be quickly reconstructed from a saved version made some time earlier. On the other hand, for program versions or file inventories that are changed or updated frequently, systems support must prevent data loss by means of a suitable, layered data saving strategy.

- **Server load**

  With the help of a detailed analysis of the server load, systems support can not only determine the time and frequency of saves but also draw conclusions regarding the scope of the data to be saved.
  Ideally, the data saving strategy could be devised in such a way that comprehensive full saves are performed in periods of relatively low load, partial and incremental saves in periods of moderate load, and no saves at all during periods of full load.

- **Configuration**

  The configuration of the server and thus the hardware available for data saving purposes is an important consideration when devising the data saving strategy and is a factor which determines the scope of the individual saves.
  If enough peripheral equipment is available, systems support can shorten the save duration and increase the scope of the saved data.

  - By using the subtask function of the ARCHIVE and FDDRL utility routines, subtasks can be directed to different devices and saves can be performed in parallel.
  - Distribution of user groups to individual pubsets (MPVS) facilitates strategic access to subsets of the data to be saved.
  - At system generation time systems support can create the prerequisites for accelerated saving of even large data volumes by means of a performance-boosting configuration of peripherals, channels and IOPs.

Down time due to backup

When the data of an application is backed up, the application itself is not available for a while. This down time is determined mainly by the scope of the backup and how long its takes to back up the data. The down time can be reduced to the time it takes to create a copies of the files by just backing up copies (e.g. on mirror disks). The application itself can continue to work with the original files after the copy phase while the more time-consuming backup of the copy is being performed.

Logical and physical save

In a **logical save**, individual files and job variables are read from one or more volumes and written contiguously (i.e. in logical units) to other volumes.

The software products ARCHIVE (see the “ARCHIVE” manual [3]) and HSMS (see the “HSMS” manual [24]) are available in BS2000 for logical data saving.

HSMS provides the user with the four basic functions

- data saving (backup)
- long-term archiving
- migration
- data transfer (export/import)

HSMS is an extension of the software product ARCHIVE. Most functions that were formerly called via ARCHIVE are now available in compatible form in HSMS.

To reduce downtime, backup of a copy with CCOPY (Concurrent Copy) is offered in HSMS.

In a **physical save**, individual files are not saved; instead, entire volumes are saved according to their physical structure. All the files on a volume, including the volume labels, are written block by block in physical sequence to a second volume. This is then identical to the original volume.
The FDDRL software product (see the “FDDRL” manual [22]) is available in BS2000 for physical data saving.

The **Snapset save** is a mixture of the physical and logical saves: It is a pubset backup in which a copy of each pubset disk is created on a snap unit. Individual files and job variables can be read from this pubset backup as logical units. Depending on the storage system, the entire pubset can also be restored. The functions for the Snapset save and access to the saved data (at pubset, file and job variable level) are available in BS2000.

Encrypted files are stored in encrypted format for all save types.

The table below shows the basic differences between backing up volumes (physical saves), backing up files and job variables (logical saves), and a Snapset save.

	Logical save	**Physical save**	**Snapset save**
What is saved?	Files, catalog entries, job variables	Entire volumes, i.e. private and public disks	Complete pubset (disk by disk)
Who saves it?	Systems support or HSMS administration: all user and system files; Users: only their own files	Systems support or HSMS administration	Systems support or HSMS administration
When is it saved?	At regular intervals (the applications are stopped)	Regularly in the case of pubsets which are exported with EXCAT	Regularly during ongoing operation (imported pubset)
Facilities used	ARCHIVE and HSMS utility routines	FDDRL utility routine	Snapset commands in BS2000
Facts to be noted	• Full and incremental backups; comfortable methods for selecting objects • Reduction of the downtime by backing up clone units	• Only the used blocks are backed up • Reduction of the downtime by backing up clone units	• Access to saved files, catalog entries, job variables (for information or reconstruction purposes) is possible “online” for all users in accordance with their DMS access rights • For systems support, reconstruction of the complete pubset • No downtime • Save/reconstruction in a matter of minutes
Backup medium	Tape cartridge, disk and Net-Storage volume	Tape cartridge and disk	Snap units (in the storage system)

Table 37: Logical and physical save

13.2 Data saving using Snapsets

BS2000 offers an integrated solution for saving a pubset to so-called snap units of an external storage system during ongoing operation. This function is offered under the name **Snapset**. It is available for the storage systems ETERNUS DX/AF from FUJITSU.

BS2000 also supports data backup by replication functions of external storage systems. For details, see the “SHC-OSD” manual [48].

Principle of the Snapsets

A Snapset is the directly available backup of a pubset to so-called snap units of a storage system. A Snapset is used to restore lost data (e.g. after inadvertent deletion). The complete pubset can also be restored to the status at the time the Snapset was created. Both SM and SF pubsets can be backed up. Snapsets are generated by systems support or the HSMS administrator during ongoing pubset operation and deleted again later as required. Up to 52 Snapsets can exist for a pubset (provided this maximum number is supported by the storage system).

Within the storage system, snap units use a block-by-block incremental procedure, and only require storage space of their own for blocks which differ from the original.

This means that saving to disk using the snap technique also makes economic sense: When, for example, 10 backups are created in one day from a pubset with a capacity of 100 GB, if cloning were used 1 TB of additional disk storage space would be required. However, when Snapsets are used, the amount of additional disk storage space required depends on the volume of changes on the pubset. If the volume of changes per day is 20%, in the example above additional disk storage space of only 0.2 TB (20% * 10 backups * 100 GB) would be required.

Snapsets are available exclusively for restore purposes (i.e. read-only access): They are placed in service automatically when they are generated. Existing Snapsets are generally placed in service when the associated pubset is imported.

Users can access Snapsets using the following functions:

- Display information on existing Snapsets (i.e. pubset backups)
- Display information on restorable files and job variables
- Restore files and job variables
- Restore pubset (privileged users only)

The following software products also enable Snapsets to be accessed:

- LMS enables members of PLAM libraries to be selected and copied to Snapsets (see the “LMS” manual [29]).
- HSMS enables files and job variables to be transferred from a Snapset and to a backup archive (see "Saving a Snapset to a backup archive" in section "Restoration of files and job variables" and the “HSMS” manual [24]).

The Snapset functions are available in shared pubset mode on all the systems involved. The requirements for this (BS2000 versions of the pubset sharers and of SHC-OSD) are described in the current BS2000 Release Notices.

In BS2000 you can use up to 52 Snapsets. In addition, program interfaces which in functional terms correspond to the command interfaces mentioned above are offered for accessing the Snapsets (see the “DMS Macros” manual [20]).

VM2000 supports the use of Snapsets by the guest systems (see the “VM2000” manual [60]).

Application area of Snapsets

Snapsets are used on a short-term basis as a backup when the files or job variables of a pubset are lost. An entire pubset can also be restored quickly to a particular status with the RESTORE-PUBSET-FROM-SNAPSET command. They are not suitable for long-term backups or data transfer. The snap units used by Snapsets are normally local copies within a storage system and thus offer no protection when the storage systems fail. You are therefore urgently recommended also to back up Snapsets with (see "Saving a Snapset to a backup archive" in section "Restoration of files and job variables").

Private disks (or files on private disks) cannot be saved on Snapsets.

The operation of Snapsets can be combined with the simultaneous use of disk copies on the basis of local mirrors in the same pubset provided this is possible in the storage system concerned.
Mixed mode of Snapsets (/CREATE-SNAPSET) and Snapshots (/START-SNAP-SESSION, /ACTIVATE-SNAP) is not supported .
These other usage modes are described in the “HSMS” [24] and “SHC-OSD” [48] manuals.

Restrictions:

- The pubset cannot be restored from a Snapset if a clone session is still active for this pubset.

Overview of the interfaces for working with Snapsets

All the commands for Snapset mode, saving to Snapsets and restoring from Snapsets are offered in the SDF application area “SNAPSET”. These commands are described in the “Commands” manual [27].

Comman	Function
ADAPT-SNAPSET-ACCESS [1]	Adapt access to Snapsets in the case of remote replication
CHECK-SNAPSET-CONFIGURATION [2]	Check the Snapset configuration and activate available Snapsets
CREATE-SNAPSET [2]	Create a Snapset
DELETE-SNAPSET [2]	Delete a Snapset
LIST-FILE-FROM-SNAPSET [3]	Provide information about files on a Snapset
LIST-JV-FROM-SNAPSET [3]	Provide information about job variables on a Snapset
RESTORE-FILE-FROM-SNAPSET [3]	Restore files on the basis of a Snapset
RESTORE-JV-FROM-SNAPSET [3]	Restore job variables on the basis of a Snapset
RESTORE-PUBSET-FROM-SNAPSET [2]	Restore a complete pubset on the basis of a Snapset
SET-PUBSET-ATTRIBUTES [2]	Determine the maximum permitted number of Snapsets for the pubset (SNAPSET-LIMIT operand)
SET-SNAPSET-PARAMETER [2]	Define the Snapset environment
SHOW-PUBSET-ATTRIBUTES	Display information about, among other things, the Snapset limit

SHOW-SNAPSET-CONFIGURATION	Display information about existing Snapsets and the Snapset configuration

Table 38: Overview of the interfaces for working with Snapsets

[1] Privilege TSOS required

[2] Privilege TSOS or HSMS-ADMINISTRATION required

[3] There is also a program interface for these commands, see the “DMS Macros” manual [20].

13.2.1 Preparing Snapset mode

Starting Snapset mode for a pubset means that copies (backups) of the pubset are created on Snapsets which will be available for restoring files/job variables or the entire pubset.

Terms

Snap unit

The local replication functions of the storage systems create a "Snapshot" of a logical unit (also of more than one if required). The Snapshot, which is called a snap unit, is a logical copy of the original unit at a particular time ("point-in-time copy"): while the data on the original unit is changed, the snap unit retains the status of the data at the time the Snapshot was created.

In ETERNUS DX/AF systems for snap units are used Thin Devices (TDEV) or Flex Volumes (FDEV). For Snapset mode, the planned snap units have to be initialized on Thin Devices (TDEV) or Flex Volumes (FDEV) in advance, using the VOLIN utility routine (see “Utility Routines” manual [15]). The special notation `S#<mn>` is expected as the VSN for these volumes, e.g. `S#5234`, where <mn> is the device mnemonic. Due to its flexibility this type of snap units is strongly recommended.

Using the Snapset mode specially configured units, called snap units or snap device volumes (**SDVs**) can be used alternatively. In most cases, these are set up directly on the storage system by a qualified technician. Original unit and snap unit must be placed inside the same storage system.

In BS2000, the snap units are generated like normal disk devices (type D3435), with a capacity equal to the respective original unit.

Save Pool

The storage space actually required for the original data to be saved is, if necessary, provided from a so-called save pool. Save pools are configured in the storage system with a suitable capacity.

The implementation of save pools depends on the storage system:

ETERNUS DX/AF Systems

- **in S#-Volume mode**
  In this case, there is ***no*** separate save pool.
  The required storage space is provided by the respective snap units, which are subject to the overflow control of the respective “Thin Provisioning Pool”.
- **in SDV mode**
  In this case, there is only ***one single*** save pool for all Snapsets.
  The required storage space is initially provided by the respective snap units. Only when the capacity of those is reached (about 10 % of the original unit), storage space on the save pool is used.
  In the case of a save pool overflow, ***all*** Snapsets using storage space on the save pool are corrupted. The SHC-OSD functions of for monitoring save pools should therefore consequently be used

Usually, the actually required storage space (for the original data to be saved) comes from a storage area specifically provided on the storage system for this purpose, a so-called save pool. The required capacity of a save pool depends on the following factors:

- Data volume of the applications for which Snapsets have been generated
- Number of snap sessions per original
- Change volume on the original (and the associated snap units)

When a Snapset is deleted, the area concerned in the save pool is released again.

Save Pool / Thin Pool Monitoring

SHC-OSD provides functions for monitoring the utilization level of a save pool or the thin pool respectively. As soon as a specific value is reached or changed, warning messages are output to the console.
In any of these cases, system administration should react appropriately and create storage on the save pool or, if necessary, extend the save pool. The threshold values are set in the storage system or in the SHC-OSD parameter file.

> **CAUTION!**
> An overflow will lead to loss of the existing Snapsets.
> Specifically, the respective snap sessions will enter the “FAILED” state; After that, they can only be deleted.

Example of preparations

Before Snapset mode can start for a pubset, corresponding preparatory measures are required. The procedure is illustrated using the example below.

- Initial situation and requirements planning

  Pubset A consists of 8 disks (PUBA00 through PUBA07).
  It is to be saved twice each workday (e.g. at midday and in the evening).
  Access to the backups of the last 6 workdays should be ensured.

  12 Snapsets must then be available for 2 x 6 pubset backups. For each Snapset the same number of snap units is required as the number of disks in pubset A. In total 8 x 12 = 96 snap units are required.

- Providing snap units in the storage system

  The 96 snap units required must be provided in the same storage system as the original disks of pubset A with the same type and the same size.
  If it is already clear that a pubset will be extended, additional snap units should be included for this purpose in sufficient number.

  > ETERNUS DX/AF systems can be used with pre-configured snap units or S#<mn> volumes. See "Terms".

- Configuring snap units in BS2000

  All 96 snap units must be included in the hardware generation of the BS2000 systems involved like normal disks (type D3435), in other words in practically all hardware generations in which the original disks of pubset A are also entered (see the “System Installation” manual [55]).

  On SUs /390, disks can also be included in the active I/O configuration while operation is in progress.
  The inclusion of emulated D3435 disks in the X2000 configuration of an SU x86 is described in the “Operation and Administration” manual [57].

  Note for VM2000

  Snap units created in the storage system are also recognized as snap units by VM2000. They are automatically attached when a pubset is imported, and under VM2000 implicitly assigned to the VM if the VM has the privilege AUTO-SNAP-ASSIGNMENT. When initialization takes place using the VM2000 command CREATE-VM, a VM is assigned this privilege by default. It permits the guest system on the VM to implicitly assign itself the snap units of a Snapset without the VM and device being prepared for implicit device assignment (i.e. the VM privilege and the device attribute ASSIGN-BY-GUEST are not necessary in this case).

  Exception

  Volumes S#<mn> used as snap units of VM2000 are not recognized as snap units and are also not output. Despite of that, they can be assigned implicitly to a VM with AUTO-SNAP-ASSIGNMENT privilege.

- Setting the Snapset limit and, if necessary, assigning the save pool (in BS2000)

  The Snapset limit is entered in the SVL of the Pubres. It defines the maximum number of Snapsets permissible for the pubset. For a pubset without Snapset mode the Snapset limit is 0.
  As a maximum of 12 Snapsets are to be created for pubset A, systems support sets the Snapset limit to 12:
  `/SET-PUBSET-ATTRIBUTES PUBSET=A,SNAPSET-LIMIT=12`

Snapset mode in the case of disaster protection with remote replication

If, because of disaster protection, pubset A is operated in a local storage system **and** in a remote storage system, the following must be borne in mind for Snapset mode:

- Normally the Snapsets are configured **only** in the local storage system. If the local storage system fails, they are no longer available.
- If the Snapsets are also to be available in the remote storage system should a disaster occur, they must be created in **both** storage systems. To permit this, the same number of snap units must be provided in the remote storage system as in the local storage system and included in the hardware generation of the BS2000 systems involved.

  In addition, systems support must set the processing environment accordingly for the Snapsets:
  `/SET-SNAPSET-PARAMETER PUBSET=A,REMOTE-COPY=*YES`

  When disk access is switched between the source and target controllers, access to the Snapsets assigned to the pubset is not also switched automatically.
  The ADAPT-SNAPSET-ACCESS command ensures that the assigned Snapsets are still available after such a switchover without the pubset having to be exported:
  `/ADAPT-SNAPSET-ACCESS PUBSET=A`

13.2.2 Saving using Snapsets

Once the preparations have been completed, Snapset mode (in the example for pubset A) can commence, i.e. systems support or HSMS administration can create pubset copies (=Snapsets) at the planned times.

Example

Pubset A is to be saved twice each workday (e.g. at midday and in the evening).

Creating a Snapset

At the required save time systems support or HSMS administration uses the CREATE-SNAPSET command to create a Snapset:
`/CREATE-SNAPSET PUBSET=A`

This command can be entered on all systems involved in shared pubset mode. Only one CREATE-SNAPSET command can be issued at a time for a pubset. Parallel calls are serialized.
The default for this command is that the oldest Snapset is implicitly deleted when the Snapset limit is reached. Optionally the DELETE-EARLIEST operand can also be used to specify that the oldest Snapset should always be implicitly deleted or that it should never be implicitly deleted.
The newly created Snapset is automatically placed in service and is then available to all users for restoration purposes.

Snapset identification

Each newly generated Snapset is assigned a unique Snapset identification. Firstly the lower case letters “a” through “z”, then the upper case letters “A” to “Z” in alphabetical order are used here for the 52 maximum possible Snapsets. After “Z” has been reached, assignment begins again with “a” (the chronological order of Snapset creation consequently does not always match the alphabetical order of the Snapset IDs). Case sensitivity must be observed at the command interface.

Each snap unit contains a VSN which is formed from the VSN of the original disk and the Snapset identification.

Example illustrating the mapping rule

VSN in PUB notation and Snapset ID as lowercase letter, e.g. "a": PUBX12 --> PaaX12

VSN in PUB notation and Snapset ID as uppercase letter, e.g. "A": PUBX12 --> aaBX12

VSN in dot notation and Snapset ID as lowercase letter, e.g. "a": AB.123 --> ABa123; ABC.12 --> ABCa12; ABCD.1 --> ABCDa1

VSN in dot notation and Snapset ID as uppercase letter, e.g. "A": AB.123 --> AB123a; ABC.12 --> aABC12; ABCD.1 --> AaBCD1

Managing and starting up Snapsets

Each newly created Snapset has a unique Snapset identification (see "Snapset identification"). The descriptions of all the Snapsets of a pubset are kept and managed in the Snapset catalog. The Snapset catalog is stored in the associated pubset under the name $TSOS.SYSCAT.SNAPSET and contains the following information:

- the processing environment for the pubset’s Snapsets, such as the usage of remote replication and details of the save pool

- Descriptions of the various Snapsets with details such as the Snapset identification, creation date and MNs of the snap units in the local storage system and, if required, also of the snap units in the remote storage system.

If the Snapset catalog does not yet exist, it is configured when the Snapset is created. In the case of an SM pubset the Snapset catalog is located on the Volres of the control volume set, and in the case of an SF pubset on the associated Pubres.

Starting and shutting down Snapsets

Existing Snapsets are generally started when the associated pubset is imported.

They are not started if the SHC-OSD subsystem is not yet active at the time the pubset is imported (existing Snapsets of the home pubset in particular are not automatically started when the system starts up). As soon as SHC-OSD is active, the Snapsets are activated retroactively by issuing the SHOW-SNAPSET-CONFIGURATION command.

When Snapsets cannot be placed in service implicitly because of hardware or configuration problems, they initially remain in the "not available" status. After the error has been corrected, these Snapsets can be made accessible again using the CHECK-SNAPSET-CONFIGURATION command.

When a Snapset is created with the CREATE-SNAPSET command, it is automatically placed in service.

The following actions are performed in this case:

- Open the Snapset catalog (by means of GCF)
- Assign all snap units (ATTACH)
  In a configuration with remote replication, the “suitable” snap units are assigned. In the normal status with active remote replication, these are the snap units in the local storage system. In the case of split mirrors, the selection is made on the basis of the imported original pubset:
  - In the case of the pubset on the local storage system, only the snap units on the local storage system are used.
  - In the case of the pubset on the remote storage system, only the snap units on the remote storage system are used.

  Under VM2000, constraints must be observed for implicit device assignment, see section “Note for VM2000” in chapter "Preparing Snapset mode".
- Creating a Snapset in the storage system
- Set up a CCOPY session for the Snapset

When a pubset is exported, the associated Snapsets are shut down. The actions performed during the import are performed in reverse order.

Deleting a Snapset

Irrespective of snap creation, systems support or HSMS administration can also explicitly delete an existing Snapset for an imported pubset:
`/DELETE-SNAPSET PUBSET=A[,SNAPSET=*EARLIEST]`

Here the oldest Snapset of pubset `A` is deleted. The specification `SNAPSET=*EARLIEST` can also be omitted (default).

The Snapset to be deleted can also be specified by means of its Snapset identification or as a relative specification relating to the current status of the pubset (with -n, where -1 designates the latest Snapset). Deleting a Snapset means that the associated pubset backup status is no longer available. Restore operations which are currently active for a Snapset which is to be deleted are aborted when the Snapset is deleted.

When operating with ETERNUS DX/AF, only the oldest Snapset can be deleted. This must be borne in mind when a Snapset to be deleted is specified explicitly by means of the Snapset ID or the relative age.

Terminating Snapset mode

The SNAPSET=*ALL specification causes all Snapsets to be deleted and Snapset mode to be terminated for the pubset. This call incorporates the following measures:

- Delete all Snapsets
- Close and delete the Snapset catalog
- Set the pubset's Snapset limit to zero

Displaying Snapsets

The SHOW-SNAPSET-CONFIGURATION command displays information on the relevant Snapsets, see the "Commands" manual [27]:
`/SHOW-SNAPSET-CONFIGURATION PUBSET=<cat-id>,SNAPSET=*ALL`

In this case both cross-pubset and Snapset-specific information is displayed.

If the privileged user (privilege TSOS, OPERATING, HSMS administrator) requests information on a particular Snapset, the VSNs of the pubset volumes and the MNs of the assigned snap units of the local storage system are also output (in the case of remote replication, possibly also for the remote storage system).

Notes and restrictions

No Snapset mode

Snapset mode for a pubset – like all other replication functions offered via SHC-OSD – is not compatible with DAB write caching for this pubset. The CREATE-SNAPSET command is rejected in the event of DAB write caching.

Snapset mode and DRV mirroring for a pubset are mutually exclusive: DRV operation cannot be started on a pubset in Snapset mode (Snapset limit not equal to 0) and vice versa.

Snapset mode with remote replication

When Snapsets are also to be operated on the remote storage system (see section "Snapset mode in the case of disaster protection with remote replication" in chapter "Preparing Snapset mode"), the following restrictions must be borne in mind:

- Snapset operation on the remote storage system is not possible if remote replication is interrupted (either by the HOLD-REMOTE-COPY command or as a result of interrupted links) if the remote mirror has not yet been synchronized or in the case of asynchronous remote replication.
- If Snapset creation fails on one of the storage systems involved (local or remote), no Snapset is created on the other storage system either.

- When access to the pubset's disks is switched dynamically between the source and target systems (e.g. by SHC-OSD commands), access to the Snapsets allocated to the pubset is not also switched automatically.
  The ADAPT-SNAPSET-ACCESS command ensures that the allocated Snapsets are still available after such a switchover without the pubset having to be exported.

  When the command is called, a check is made to see whether access to the allocated Snapsets takes place in the same controller as access to the pubset's disks. If this is not the case, the switchover for the Snapsets allocated to the pubset is emulated:

  - The Snapsets currently attached are taken out of service.
  - The Snapsets of the local controller are then attached.

Resource bottleneck and its consequences

The storage space required for the original data to be saved is initially provided by the respective snap units and, if necessary, by the save pool. Information on this and the possible consequences in the event of an overflow is provided in section "Preparing Snapset mode".

The measures below should be taken to avoid high memory requirements for the original data to be saved:

- Do not generate paging files in pubsets which use Snapset mode
- Reorganize the pubset as little as possible using the software product SPACEOPT

Changing the pubset configuration

Before one of the following changes to the pubset configuration is performed, Snapset mode must be terminated (this means deleting all Snapsets):

- before a pubset is renamed
- before an SF pubset is transferred to a volume set of an SM pubset
- before switching to other pubset disks, i.e. when switching from K to NK format, or in the event of a physical relocation using FDDRL (save/restore)

When the pubset is extended by the addition of further disks, the set of suitable pubsets must be extended accordingly (see section "Preparing Snapset mode").

When the pubset is reduced by removing disks, existing Snapsets become invalid. They can no longer be used for restoration. After the remote disks have been reinitialized, the Snapsets will be removed from the Snapset catalog the next time the pubset is imported. Existing Snapsets must therefore be saved before disks are removed from the configuration using HSMS if the data stored on them is still required. A new Snapset should be created immediately after a configuration change so that it is still possible to restore files and job variables and also the entire pubset.

Migrated files

To ensure that migrated files are still usable after Snapsets have been restored, the associated tapes must still exist. They may not have been released or already have been reused.
When existing Snapsets cover a period of **n** days for restoration, restoration of the migrated files using HSMS should always only be performed for a period preceding these **n** days.

13.2.3 Restoration of files and job variables

As soon as Snapset mode has started for a pubset, the pubset copies on the created Snapsets are available for all users to restore. The Snapsets are operable if the pubset has been imported. The nonprivileged user has access to files and job variables in accordance with the DMS access rights.

Users obtain information on all the operable Snapsets by means of the SHOW-SNAPSET-CONFIGURATION command; Privileged users are also provided with more detailed information on Snapset disks:

```
/SHOW-SNAPSET-CONFIGURATION PUBSET=A[,SNAPSET=*ALL]
```

Before a file or job variable is restored, the user can determine which save status, i.e. which Snapset, is to be used.

Displaying information on saved files/job variables

Users obtain a list of the files and job variables which are contained in a Snapset backup and can therefore be restored from this backup using the LIST-FILE-FROM-SNAPSET or LIST-JV-FROM-SNAPSET command. These commands can be executed in parallel and synchronously on all pubset sharers.

Nonprivileged users obtain information on files and job variables which they can access in accordance with their DMS rights (comparable to SHOW-FILE-ATTRIBUTES and SHOW-JV-ATTRIBUTES). Systems support has extended rights.

Partial qualification and the specification of wildcards in file/job variable names are permitted when selecting a set of files/job variables, but not in the catalog and/or user ID.

The information output contains not only the name, but also the attributes required for permitting the restore operation: size, special identifier for migrated files and tape files, creation date and date of the last access (or expiration date in the case of job variables), plus the file status. The file status indicates whether the file was open in write mode at the time it was saved (STATE=OPENED or CLOSED). STATE=NOREST is displayed in the case of files which cannot be restored (e.g. files with the save frequency BACKUP-CLASS=E or special system files).

Restoring files/job variables

Users can restore files and job variables from a Snapset using the RESTORE-FILE-FROM-SNAPSET or RESTORE-JV-FROM-SNAPSET command. These commands can be executed in parallel and synchronously on all pubset sharers.

Nonprivileged users can access files and job variables in accordance with their DMS rights (read authorization for the original file and owner rights for the target file). Systems support has extended rights.

Partial qualification and the specification of wildcards in file/job variable names are permitted when selecting a set of files/job variables, but not in the catalog and/or user ID.

The files/job variables specified are restored from a particular Snapset (the latest Snapset is preset by means of the SNAPSET=*LATEST operand). However, a range of Snapsets (SNAPSET=*INTERVAL operand) or all Snapsets (SNAPSET=*ALL operand) can also be included in the restore operation. In this case the files/job variables are restored to their most recent status among these Snapsets.

As with HSMS restoration from backup archives, files and job variables are restored with their original catalog attributes. In particular, the creation and modification times and also the protection attributes are the same as those of the original at the time it was saved, i.e. at the time the Snapset was created. Optionally a log can list either just the files/job variables which were not restored owing to errors or also all the files/job variables which were restored.

Additional options for restoration

When a Snapset is generated, all the files and job variables of the pubset are saved simultaneously. All the pages of a file are also saved simultaneously. Consequently files which are open in write mode are saved crash-consistently. By default these files are not restored (except for files with the ONLINE-SAVE indicator). Restoration of such files can be requested explicitly (RESTORE-OPEN-FILES=*YES operand). In the case of ISAM files, postprocessing with the REPAIR-DISK-FILE command may be necessary. System files for SRPM, GUARDS and GCF which are constantly open are made consistent when the Snapset is created in order to permit proper restoration later.

Files and job variables can be restored under a new name (NEW-FILE-NAME or NEW-JV-NAME operand). They are renamed either by specifying a different user ID or a file name prefix. Nonprivileged users can specify a different user ID if they are co-owners of the ID concerned. They can specify their own user ID when they want to restore a foreign file (read access is required) on their own ID.

Existing files and job variables are by default not overwritten, i.e. not restored. Restoration of these files/job variables can be explicitly requested (REPLACE=*YES operand). Delete authorization (e.g. write password) is required to overwrite them. Privileged callers can explicitly ignore the write protection (IGNORE-PROTECTION=*YES operand).

Restrictions

Individual file generations cannot be restored, only entire file generation groups can.

Only the catalog entries are restored for migrated files and tape files. Renaming is not possible in this case (exception: a privileged user under TSOS can specify a different user ID when a tape file is involved).

Restoring PLAM library members

In LMS, the OPEN-LIBRARY statement enables a PLAM library on a Snapset to be opened in read mode. Members of this library can then be displayed, copied or selected using the relevant LMS statements.

Saving a Snapset to a backup archive

The HSMS statement BACKUP-FILES enables files/job variables which have been saved to a Snapset to be transferred to a backup archive. With the BACKUP statement below you can, for example, transfer all the files and job variables of the user ID USER1 which were saved in the last Snapset backup of pubset A to a backup archive:

```
//BACKUP-FILES FILE-NAMES=$USER1., JV-NAMES=$USER1.,
               ARCHIV-NAME=<archiv>,CONCURRENT-COPY=*YES(
                WORK-FILE-NAME=*FROM-SNAPSET(PUBSET-ID=A,SNAPSET-ID=-1)
```

This function is also available to nonprivileged users.

A Snapset can be transferred to a backup archive at any time between the creation and deletion of the Snapset. This takes place practically “offline” with regard to the ongoing pubset operation. The new save version generated in HSMS is then not assigned the current date of the BACKUP-FILES call as the Save time, but the creation date of the Snapset. However, no newer save version may exist in the backup archive.

13.2.4 Restoration of pubsets

Systems support can use the RESTORE-PUBSET-FROM-SNAPSET command to restore the entire pubset from an existing Snapset of the pubset on a volume basis. This function cannot be executed in active pubset mode. The pubset must be exported, i.e. it may not have been imported into another system either locally or remotely. It must also still exist on the original disks and have the original pubset attributes (see also section “Changing the pubset configuration” in chapter "Saving using Snapsets").

In the storage system, the pubset is restored to the status of the relevant Snapset. The pubset then has the same status as immediately before this Snapset was generated.

The pubset is protected against being placed in service by /IMPORT-PUBSET until the restore operation has been concluded.

While restoration is in progress, a temporary copy of the Snapset catalog of the pubset which is to be restored is created on the home pubset and assigned the file name $TSOS.SYSWRK.SNAPSET.<catid>. This file is only required for restoration and is only deleted once restoration has been completed. While restoration is in progress it is protected against being deleted because it is used for restarting an interrupted restore operation (see below). If the pubset is restored in another way, the file is deleted when the next pubset import takes place.

After a session is aborted while restoration is in progress, the pubset is in an intermediate status which corresponds to a partial restoration. In this case restoration can be completed by issuing the command again. However, restoration can only be started again in the same runtime environment, i.e. with the same home pubset, because the copy of the home pubset which is required is located there.

With ETERNUS DX/AF systems reconstruction of a pubset can be executed on any Snapset of that pubset. All these Snapsets including that one used for reconstruction, are retained and can continue to be used.

Note on the SNAPSET operand of the RESTORE-PUBSET-FROM-SNAPSET command

- SNAPSET = *LATEST
  While the function is being executed, the pubset is *not* implicitly imported. F5 label restoration takes place when the restored pubset is imported.
- SNAPSET = <name 1..1 with-low> / <integer -52..-1> (explicit entry)
  While the function is executed, the pubset is implicitly imported and at the end exported again. In this case F5 label restoration also takes place implicitly.

Changing the pubset configuration

When the pubset has been reduced by the removal of disks following the last Snapset save and these disks have been reinitialized using VOLIN, it is no longer possible to restore the pubset. In this case the RESTORE-PUBSET-FROM-SNAPSET command is rejected.

The option of restoring the pubset following a pubset reduction can be retained by means of one of the following measures:

- Do not reinitialize removed disks
- Generate a new Snapset immediately

> VOLIN does not permit snap units to be initialized and responds to the initialization of original disks by inquiring if snap units are still concatenated.

13.3 Reconstruction of files and volumes

Data can be saved and reconstructed at both the file level and the volume level. There are various reasons why files or disks might have to be restored:

- One or more files have to be read in from backup media because they could not be kept online due to limited capacity.
- The entire system has to be reconstructed following a breakdown.
- A disk has to be reconstructed.

The purpose of reconstruction is to retrieve the saved files as quickly as possible. Different measures must be taken, depending on whether the volume in question is a public or private disk.

Reconstruction of individual files

If a user loses a file as a result of an operating error (e.g. inadvertent deletion), there are two ways of reconstructing it by retrieving the most recent backup copy.

- Retrieval of the logical backup copy of the file

  The file has been logically saved on a private volume. The backup copy has to be read in and the updates made to the original file have to be repeated.
- Retrieval of the physical backup copy of the disk or tape containing the file (exceptional case)

  The last physical backup copy produced with FDDRL is either present on a backup disk or must be read in from tape to a disk. The file in question is then copied from this disk to the original disk. Any modifications made to the replaced file in the interim must be subsequently entered in the backup copy.

Reconstruction of a public disk

After failure of a public disk, an operable disk mounted on a standby device must first be provided.

When a public disk fails, the entire pubset has to be reconstructed. The reconstruction outlay depends on the range and currency of the most recent physical backup.
This backup copy must first be read in from tape. Depending on the status of the physical backup copy, logical backup copies may also have to be read in order to obtain the most recent status.

Subsequently the catalog entries of the user files on private disks have to be imported, or the most recent backup copy has to be read in. This can be done with a command sequence containing an IMPORT-FILE command for each private disk mounted.

Reconstruction of an SM pubset

If the control volume set of an SM pubset cannot be accessed anymore, then the entire SM pubset will fail.

In order to continue working with the data on the other volume sets, a volume set can be imported exclusively in BS2000/OSD-BC V5.0 and higher and converted to an SF pubset with the same catalog ID. The `USE=*EXCLUSIVE (CONVERT-VOLUME-SET=*YES)` operand of the IMPORT-PUBSET command is used for this purpose.

Control volume sets are excluded from conversion.

The data and metadata of the failed control volume set must be reconstructed from the backup after an additional SF pubset has been provided. After that, the SMPGEN utility routine can create a new SM pubset from the converted SF pubset and the reconstructed control volume set also available as an SF pubset.

Reconstruction of a pubset

If a pubset (SM or SF) which was saved in Snapset mode can no longer be imported, it can be reconstructed to the status of the latest Snapset using the RESTORE-PUBSET-FROM-SNAPSET command (see section "Restoration of pubsets").

Reconstruction of a private disk

After providing an operable disk device, the following steps must be taken:

Initialize the private disk using the utility routine VOLIN (see the “Utility Routines” manual [15]). When reconstruction is performed using FDDRL, the backup must be read in from tape. Depending on the state of the physical backup, it may also be necessary to read in logical backups in order to restore the current state. Then erase all catalog entries of files for which an entry for the volume in question was made in the catalog (with the EXPORT-FILE command).

Thereafter the files can be restored using ARCHIVE. It should be borne in mind that in the case of files stored on more than one private disk, all of the volumes concerned have to be reconstructed. Otherwise, every ARCHIVE backup is logical, i.e. each file can be reconstructed individually. It is recommended that private disks be selectively saved, with a FILES statement in ARCHIVE for each disk; reconstruction can then be performed by means of the associated SVID. In addition, the ORIGIN operand can be used in the FILES statement, by means of which the VSN of the volume whose files are to be reconstructed is specified.

The following diagram shows reconstruction after a system error.

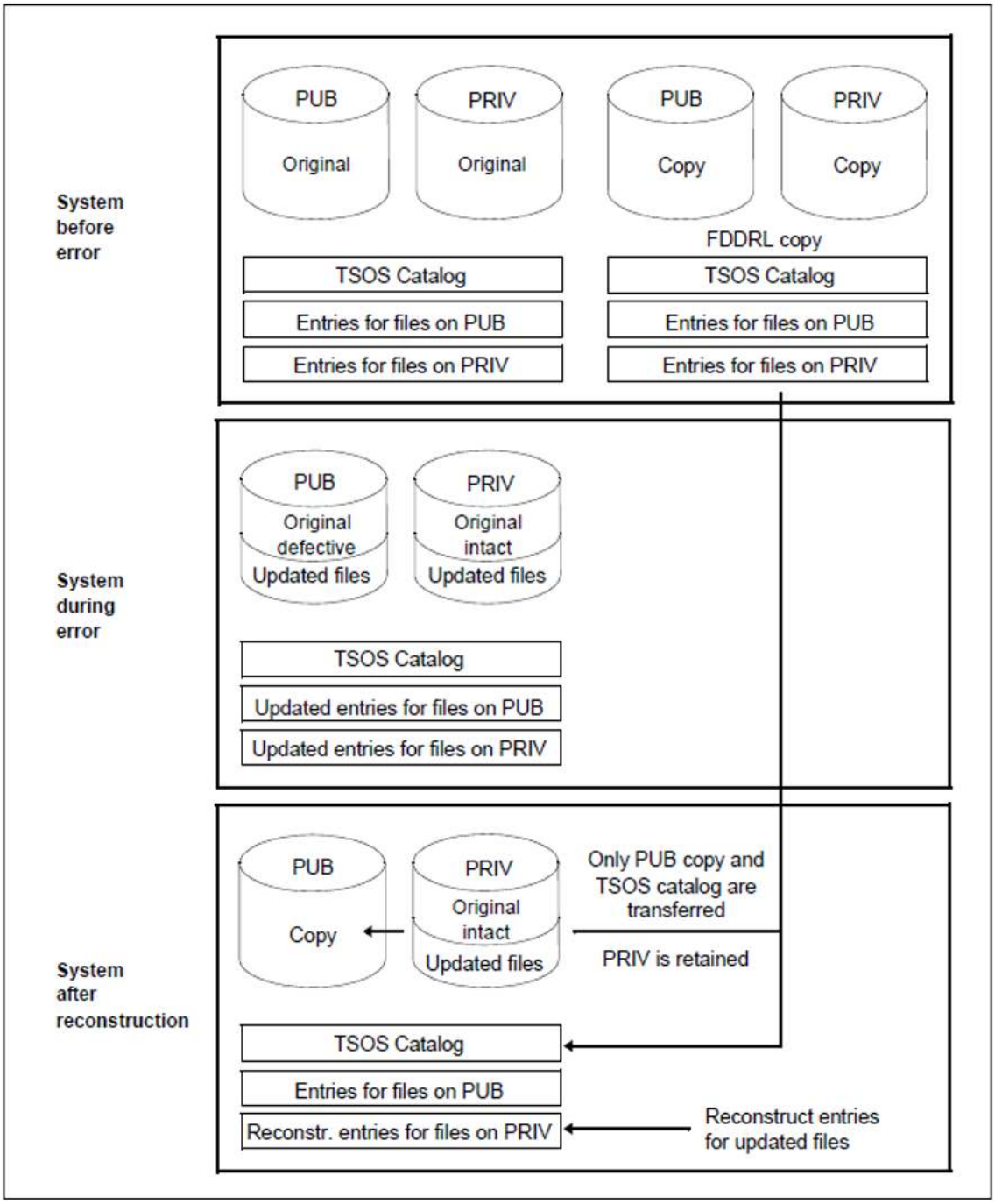


Figure 22: Reconstruction after a system error

13.4 Archiving systems

Robot-supported automation of peripherals operation features the following advantages:

- transfer of time-consuming backup runs to unmanned operating times
- efficient use of cartridge drives
- increased data security thanks to installation of the archiving system in secure and access-controlled areas
- standardization of actions and responses to tape requests
- reduced burden on disk peripherals with fully automatic access to external data resources thanks to relocation of little-used online data from disk to cartridge

BS2000 supports real archiving systems. They are controlled by means of the software product ROBAR (see the “ROBAR” manual [41]).

The tapes can be managed by MAREN. The decision as to which archiving system is involved is taken by MAREN on the basis of the storage location (for storage locations, see also "Device selection mechanisms for tape devices").

For MAREN, see also section "Tape management with MAREN" and the “MAREN” manual [31].

In addition to real archiving systems, the virtual archiving system **ETERNUS CS** can also be used. With ETERNUS CS, a virtual tape robot system is placed in front of the real tape robot system (with the real drives and cartridges). In this way the server and the real archive are fully decoupled. The virtual tape robot system knows what are referred to as virtual (logical) drives and virtual (logical) volumes. The core element here consists principally of a disk system as data cache, guaranteeing not only extremely high-speed access to the data, but also, thanks to the large number of virtual drives (up to 1024) and logical volumes (up to 1,500,000) which can be generated, that the bottlenecks which occur in a real archiving system can be cleared.

For details on ETERNUS CS, please also refer to the “ETERNUS CS” manual [8].

14 Accounting

The accounting system of BS2000 is responsible for logging and storing data for user accounting and operations accounting.

The data acquired for user accounting enables the data center to bill users for resources and services received. The information collected for operations accounting provides an uninterrupted survey of the utilization and availability of the entire server system.

The accounting routine writes the collected data in the form of different types of accounting records to a file especially provided for that purpose. This accounting file is then further evaluated using special programs.

At the end of the chapter there is an overview of all the BS2000 accounting records (for a detailed description of the accounting records and their structure please refer to the “Accounting Records” manual [1]).

14.1 Control of the accounting systems

The entire accounting system of BS2000 is controlled by systems support. Systems support determines the time at which the accounting system is to be started, defines the name of the accounting file and specifies the names and number of accounting records and record extensions to be stored in the accounting file. Systems support also determines the cycle and scope of periodic accounting for specific accounting records and job classes.

The specified sequence can be modified with the accounting exit routine in such a way that the accounting record is altered, suppressed or supplemented by further user-defined accounting records (see the “System Exits” manual [56]). The accounting exit routine is activated before an accounting record is written to the accounting file.

The START-ACCOUNTING and STOP-ACCOUNTING commands start and stop the accounting system, respectively.

The ADD-USER and MODIFY-USER-ATTRIBUTES commands can be used to specify the number of user-specific data records that may be written to the accounting file for each user.

The accounting system enables systems support to dynamically activate and deactivate accounting records, either completely or partially, and to influence the scope of the individual accounting records. The MODIFY-ACCOUNTING-PARAMETERS command can be used to deactivate records and record extensions that are not needed.

Systems support can use the SHOW-ACCOUNTING-STATUS command to obtain information on the following data:

- the status of the accounting system (activated or deactivated)
- the name of the current accounting file
- the names of the continuation files
- the activated and deactivated records and record extensions
- the frequency of periodic recording of specific accounting records
- the names of the cyclically monitored job classes

Command	Meaning
ADD-USER	Create accounting-specific user ID entries
CHANGE-ACCOUNTING-FILE	Change the accounting file
CREATE-FILE	Define the file attributes of the accounting file or create the accounting file on tape or private disk
MODIFY-ACCOUNTING-PARAMETERS	Specify the system accounting parameters
MODIFY-DEFAULT-ACCOUNT	Define default account numbers for BS2000 and POSIX access
MODIFY-FILE-ATTRIBUTES	Define the file attributes of the accounting file
MODIFY-USER-ATTRIBUTES	Change accounting-specific entries for a user ID

SHOW-ACCOUNTING-STATUS	Request information on the accounting system
SHOW-USER-ATTRIBUTES	Request accounting information from the user catalog
START-ACCOUNTING	Activate the accounting system
STOP-ACCOUNTING	Terminate the accounting system
WRITE-ACCOUNTING-RECORD	Write user accounting record
Macro	**Meaning**
ARDS	Create DSECT for defining the accounting records
AREC	Write user accounting record
ASPC	Measure the current storage space allocation on public or private volumes in the form of accounting records
SRMUINF	Read data for accounting from the user catalog

Table 39: Overview of interfaces for system accounting

14.2 Accounting file

The name of the accounting file may be determined either by means of the parameter service at system initialization or during the session.

- System initialization

  The accounting system can be activated during system initialization if the necessary statements are issued via the parameter service. Activation of the accounting system in connection with the parameter service is described in section "Starting the accounting system (ACCOUNT)".

- BS2000 session

  Systems support can activate the accounting routine during a BS2000 session by means of the START-ACCOUNTING command. This may be useful, for example, if the accounting system was temporarily deactivated for testing purposes. If the current accounting file is to be analyzed, the CHANGE-ACCOUNTING-FILE command is issued to close the current accounting file and open a new one.

Automatic file name generation

Accounting file names may be automatically generated by the system. The file name consists of the prefix $TSOS.SYS.ACCOUNT. and the suffix yy.mm.dd.xxx.nn or yyyy-mm-dd.xxx.nn (these being valid for all logging procedures, see the name of the logging file CONSLOG and the SERSLOG file)

where:

yyyy-mm-dd or yy.mm.dd	is the date on which the accounting file is created (the system parameter FMTYFNLG determines whether the year is specified in two or four-digit form)
xxx	is the number of the current session
nn	is the serial number of accounting file (01-99; always 01 at startup)

Systems support can change the $TSOS.SYS.ACCOUNT prefix setting if specific data center naming conventions have to be observed.

If a partially qualified file name is specified, the following applies:

- The suffix described above is automatically appended to any partially qualified file name. Taking into account the suffix length, a partially qualified file name (without user ID) must not exceed 26 characters; if a catalog ID has more than one character, this maximum length is reduced by the additional number of characters in the catalog ID.
- If the user ID is omitted, $TSOS. is automatically prefixed to the file name.
- If a partially qualified file name includes just the user ID, it is extended to $userid.SYS.ACCOUNT.<date>.xxx.nn (<date>= yy.mm.dd or yyyy-mm-dd).

Possible errors on activation of the accounting system

- If syntax errors occur in the parameter records during system initialization, an error routine of the parameter service issues a console message requesting the operator to either correct or ignore the error.

- The accounting system issues a FILE macro for the file to be opened. If this results in an error, an accounting file with the default name $TSOS.SYS.ACCOUNT.<date>.xxx.nn and appropriate default attributes is created.
- This also applies if the file to be opened already exists and has been cataloged with the protection attribute ACCESS=READ. This is important when converting earlier operating system versions, as older accounting files were always cataloged with ACCESS=READ.
- After a maximum of three unsuccessful attempts to open the file with the default name, a message appears on the console (no response required) and BS2000 is loaded without the accounting system. Systems support can then activate the accounting system during the session by issuing an appropriate command.

14.2.1 Characteristics and contents of the accounting file

The accounting file is a SAM file with variable record length. Systems support can specify the remaining file attributes both via the CREATE-FILE and MODIFY-FILE-ATTRIBUTES commands and via the accounting commands START-ACCOUNTING and CHANGE-ACCOUNTING-FILE.

The following points should be noted:

- The following default values apply:

SPACE:	48 PAM blocks for primary allocation and 48 PAM blocks for secondary allocation
BUFFER-LENGTH:	1 PAM block, or 2 PAM blocks for NK4 pubsets
VOLUME:	determined by DMS

- If the VOLUME operand is specified with the accounting commands, only those volumes can be used which are automatically supported by the BS2000 Data Management System and which require no additional device information.
  If the accounting file is to be created on **tape** or **private disk**, it must be generated with the CREATE-FILE command before the accounting system is started. If a **public disk** is used that does not belong to the default pubset of the designated user ID, the catalog ID of the volume must be included in the file name.
- The accounting file is opened in EXTEND mode. This means that an existing file is extended.
- New accounting files are cataloged with the protection attributes ACCESS=WRITE and USER-ACCESS=OWNER-ONLY.
  No change of attributes is performed for any existing accounting files. Therefore systems support must make sure that the accounting file is assigned the appropriate attributes.
- If only a catalog entry is set up for a file but no storage space is reserved, (CREATE-FILE command with the operand SUPPORT=NONE), then any attempt to open this file as an accounting file will result in an error.
  To prevent this happening, systems support must assign primary storage space to the file before an attempt is made to open it.

The accounting routine writes the following information to the accounting file:

- task accounting
- program accounting
- device and volume accounting
- storage space accounting
- SPOOLOUT accounting
- user-specific accounting
- operations accounting
- resource availability
- service accounting
- accounting of unbundled products
- accounting of subsystems and subsystem tasks

In addition, information relating to program accounting and storage space accounting is periodically recorded by the accounting routine. In this way accurate accounting is possible for long-running programs and for certain job classes that are subject to such monitoring.

The accounting data is stored in the accounting file in individual accounting records. The accounting records have a variable length; the maximum record length amounts to 492 bytes without the record length field (496 bytes with the record length field).

For the structure of an accounting record see the “Accounting Records” manual [1].

14.2.2 Types of accounting data

Program accounting with ledger marks

The resource usage of a program run is recorded in two accounting records at the start and at the end of the program. These records contain the current state of the usage values (*ledger mark*) since the start of the task run. The accounting system divides each task run into the events “task start”, “program start”, “program end”, and “task end”. By determining differences between these event data, an analysis program can calculate both the resource usage of the program and the usage data of the command mode occurring between two programs. For accurate accounting of long-running programs, the accounting system also provides a third record which periodically writes data to the accounting file and gives information on the average usage values during runtime.

Accounting data for user accounting

For user accounting the accounting system makes available a set of accounting data which are explained in the detailed description of the records in the appendix. The following information is provided for a user ID:

- Data on the basic resources
  - CPU time
  - I/O system (number of transported blocks or bytes)
  - main memory (working set integral)
  - service units (sum of the individual values of CPU time, I/O and main memory, takes into account CPU performance)
- Extended device usage data with time stamp and device type
- Volume usage data
- Job information and task execution information
- Extended program information
  - internal program name
  - reason for termination
  - memory usage integral for class 5 and class 6 memory
- SPOOLOUT information
  - type of output device
  - number of printed lines and pages
- Accounting of storage space changes

Space accounting

For accounting of permanent storage space usage on public and private disks, the accounting system provides an interface that permits recording on the basis of samples. These samples can be controlled by a user program (via the ASPC macro) under the privileged user ID of systems support. The ASPC macro writes one or more accounting records in the accounting file for each locally available pubset, every record containing

- the catalog ID of the pubset
- the time the record was written
- an indicator of record integrity
- the user IDs of the pubset
- the number of reserved PAM blocks per user ID.

The ASPC macro is described in the “Executive Macros” manual [30].

These data records can also be periodically recorded in the accounting file. The parameters for such cyclical monitoring and recording are controlled by systems support with appropriate specifications at the command level.

operations accounting

The availability of the entire system is documented by an accounting open record and an accounting close record. The open record is written after an accounting file has been opened, the close record is written before the accounting file is closed in the prescribed way. The identification section and basic information of these records contain:

- the designation of the server
- the system name and system version
- an identifier for startup or shutdown

Whenever the accounting file is changed, the new file is given a reference with respect to its predecessor. The availability of the CPU resource is documented in records which are written periodically and which contain the task (TU + TPR), interrupt handling (SIH) and IDLE times of the CPU. The main memory size is recorded in the first accounting open record, either during system initialization or after a START-ACCOUNTING command.

Accounting of system services in computer center tasks

A task is the largest unit that is monitored for purposes of user accounting.

System functions which are executed in a system task or on another server and which use an appreciable amount of resources are recorded in separate accounting records.

The following services requested by the user are recorded in appropriate service units:

- spoolout
- User dumps
- RFA links

For system functions whose resource usage for a given job is relatively small but which are called frequently, the service units are integrated in the entries of the task and program accounting records, e.g.:

- number of disk input/output operations
- number of terminal input/output operations
- number of catalog accesses

The accounting records are shown in the tables starting on "Overviews of accounting records".

14.3 Overviews of accounting records

The accounting records are arranged by target group (see below) and also in al alphabetical order by group (see following pages).

Accounting record for user accounting	**max. length in bytes**
JOBS (Job start accounting record)	314
PACC (Periodic program accounting record)	402
PDMP (User dump accounting record)	78
PRGS (Program start accounting record)	476
PRGT (Program termination accounting record)	528
SPLO (SPOOLOUT accounting record)	474
TASK (Task accounting record)	400
TDEV (Device and volume accounting record)	492
TATR (Task attribute update record)	70
Accounting record for space accounting	
DALC (Space allocation record)	492
DSPC (Space stocktaking record)	492
DSPP (Space stocktaking record for private disks)	492
User-specific accounting records	
UACC (User ledger-mark record)	382
UDAT (User data record)	326
Freely definable user accounting records)	496 (with SLF)
Accounting record for operations accounting	
ACLS (Accounting close record)	610 (with PUT)
AOPN (Accounting open record)	698 (with PUT)
RCPU (CPU availability record)	70
RSRV (Service accounting record)	[1]
Accounting record for DSSM	
ESMC (Subsystem initialization accounting record)	54

ESMD (Subsystem termination accounting record)	54

Table 40: Overview of the BS2000 accounting records

[1] The record length depends on the type of service account

In the following table means “default setting”: IE/IA = implicitly activated/deactivated; EE/EA = explicitly activated /deactivated.

Record ID	Extension ID	Meaning	Can be deactivated?	Default setting
Group A: Accounting data of the accounting system				
ACLS		Accounting close record	No	IE
ACLS	FN	Name of successor file	No	IE
AOPN		Accounting open record	No	IE
AOPN	FN	Name of preceding file	No	IE
AOPN	MM	Main memory configuration and working storage size	No	IE
Group C: reserved				
Group D: DMS accounting data				
DALC		Space allocation record	Yes	EA
DALC	FN	Space modification data	No	IE
DRFA [1]		RFA session accounting record	Yes	EA
DRFA [1]	ST	RFA session termination cause	No	IE
DRVR [1]		DRV accounting data	Yes	EA
DSPC		Space stocktaking record	Yes	IE
DSPC	SP	Space occupancy data	No	IE
DSPP		Space stocktaking record for private disks	Yes	IE
DSPP	PS	Space occupancy data for private disks	No	IE
Group E: DSSM accounting records				
ESMC		Subsystem initialization accounting record	Yes	IE
ESMD		Subsystem termination accounting record	Yes	IE
Group F: Accounting data of file transfer				

FTRO [1]			Yes	IE
FTRO [1]	FN	Name of the transferred filei	Yes	IE
FTRO [1]	MN	Name of the library	Yes	IE
FTRO [1]	YY	Jahrhundertanteil bei Zeitangaben	Yes	IE
FTRO [1]	MS	Computer time used	Yes	IE
Group G: reserved				

Group H: Accounting data of the hierarchic storage management system				
HSMS [1]		HSMS accounting record	Yes	EA
HSMS [1]	IO	Inputs/outputs per device grou	Yes	EA
Group I: reserved				
Group J: Job management accounting data				
JOBS		Job start accounting record	Yes	IE
JOBS	JO	Job origin	Yes	IE
JOBS	JD	Job definition	Yes	EA
JOBS	JR	Job resource request	Yes	EA
JOBS	JP	Job parameters	Yes	EA
Group K reserved				
Group L: reserved				
Group M: reserved				
Group N: reserved				
Group O: reserved				
Group P: Program accounting data				
PACC		Periodic program accounting record	Yes	EA
PACC	PD	Time of preceding PACC record	Yes	IE
PACC	MA	Background storage allocation	Yes	IE
PACC	IO	I/O and data volume per device group	Yes	IE
PACC	TI	Terminal inputs/outputs	Yes	IE
PACC	CA	Number of catalog accesses	Yes	IE
PACC	PC	Performance controller data	Yes	IE
PACC	ID	User identification	Yes	IE
PDMP		User dump accounting record	Yes	IE
PRGS		Program start accounting record	Yes	IE
PRGS	PN	Internal program name	Yes	IE
PRGS	MA	Background storage allocation	Yes	IE
PRGS	IO	I/O and data volume per device group	Yes	IE

PRGS	TI	Terminal inputs/outputs	Yes	EA
PRGS	CA	Number of catalog accesses	Yes	EA

PRGS	PC	Performance controller data	Yes	EA
PRGS	ID	User identification	Yes	EA
PRGT		Program termination accounting record	Yes	IE
PRGT	PT	Program termination cause	Yes	IE
PRGT	MA	Background storage allocation	Yes	IE
PRGT	IO	I/O and data volume per device group	Yes	IE
PRGT	TI	Terminal inputs/outputs	Yes	EA
PRGT	CA	Number of catalog accesses	Yes	EA
PRGT	PC	Performance controller data	Yes	EA
PRGT	EI	External program identification	No	IE
PRGT	ID	User identification	Yes	EA
Group Q: reserved				
Group R: Resource availability data				
RCPU		CPU availability record	Yes	EA
RSRV		Service accounting record	Yes	EA
RSRV	RD	Resource description	Yes	IE
RSRV	SV	Performance data	Yes	IE
Group S:SPOOL accounting data				
SPLO		SPOOLOUT accounting record	Yes	IE
SPLO	OT	Reason for SPOOLOUT termination	Yes	IE
SPLO	OC	SPOOLOUT creation	Yes	IE
SPLO	OI	SPOOLOUT initiation	Yes	IE
SPLO	IN	SPOOLOUT input tape	Yes	IE
SPLO	OM	Output medium	Yes	IE
SPLO	FN	SPOOLOUT file name	Yes	IE
Group T: Task accounting data				
TASK		Task accounting record	No	IE
TASK	TT	Task termination cause	Yes	IE
TASK	MA	Background storage allocation	Yes	IE

TASK	IO	I/O and data volume per device group	Yes	IE
TASK	TI	Terminal inputs/outputs	Yes	EA
TASK	CA	Number of catalog accesses	Yes	EA

TASK	PC	Performance controller data	Yes	EA
TASK	ID	User identification	Yes	EA
TATR		Task attribute update record	Yes	EA
TDEV		Device and volume accounting record	[2]	IE
TDEV	D	Unit-record device allocation data	Yes	IE
TDEV	DV	Volume-device allocation data	Yes	IE
TDEV	VU	Volume allocation data	Yes	EA
TDEV	ID	User identification	Yes	EA
Group U: Nonprivileged user accounting data and UTM accounting data				
UACC		User ledger-mark record	Yes	IE
UACC	ID	User identification	No	EE
UACC	MA	Background storage allocation	Yes	IA
UACC	IO	I/O and data volume per device group	Yes	IA
UACC	TI	Terminal inputs/outputs	Yes	IA
UACC	CA	Number of catalog accesses	Yes	IA
UACC	PC	Performance controller data	Yes	IA
UDAT		User data record	Yes	IE
UDAT	' '[3]	User data string	No	IE
UTMA [1]		UTM accounting record	Yes	IE
UTMK [1]		UTM calculation record	Yes	IE
Group V: Accounting data for virtual machines				
VACD [1]		VM accounting record for occupied devices	Yes	IE
VACD [1]	DV	Device extension	No	IE
VACM [1]		VM accounting record for occupied processors	Yes	IE
Group W: reserved				
Group X: reserved for user accounting records				
Group Y: reserved for user accounting records				
Group Z: reserved for user accounting records				

[1] For a description of these accounting records, see the manuals for the relevant products
[2] This record type is implicitly deactivated if all record extensions of these accounting records are deactivated
[3] Two blanks is the extension identifier

> The “default setting” distinction between explicit/implicit activation or deactivation applies to record extensions only. If an accounting record extension is explicitly activated (by means of a command), then all the other record extensions are implicitly deactivated, and vice versa.

The individual accounting records are described in in detail the “Accounting Records” manual [1].

15 System messages

This chapter provides an introduction to the system messages of BS2000.

The BS2000 system messages are an important aspect of communication between the user and the operating system. They inform users about error situations that occur during job processing, tell users when a function has been executed successfully and prompt them to respond or make an entry.
BS2000 system messages have a defined format, are stored in message files and are administered by the BS2000 message system.

Command	**Meaning**
HELP-MSG-INFORMATION	Request the text of a system message and explanations relating to this message
MODIFY-MIP-PARAMETERS	Create or modify the MIP parameter file and /or attach or detach message files to /from the system
MODIFY-MSG-ATTRIBUTES	Declare language for message output (task-local)
MODIFY-MSG-FILE-ASSIGNMENT	Determine message files for the session
SHOW-MIP-PARAMETERS	Display entries of the MIP parameter file or information on the message files for the session
SHOW-MSG-FILE-ASSIGNMENT	Display information on the message files for the session
Macro	**Meaning**
MSG7 / MSG7X	Output a message
MSGSINIT / MSGSMOD	Lock the message file or add it to the message system
MSGSHOW	Output information on system-specific or task-specific messages

Table 41: Overview of interfaces for message handling

15.1 BS2000 message system

The core of the BS2000 message system is the system component MIP (Message Improvement Processing). A module of another system component, subsystem or user program which is currently executing issues a prompt to MIP via the MSG7/MSG7X macro. This prompt tells MIP to send a specific message to a defined output location (console, data display terminal) (see the “Executive Macros” manual [30]) or in an S variable (see section "Output of messages in S variables").

The MIP system component performs the following message processing tasks:

1. It uses the message code and the language identifier to search the available message files for a particular message. The message is stored as a “message framework” in the message file.
2. It fills in the “message framework” by replacing the inserts in the message text with current values. The MSG7 /MSG7X macro is used to inform MIP of these.
3. It sends the completed message to the specified output location.

15.2 Message files

Messages are stored in message files. These files are organized as ISAM files in which the message code is also the ISAM key.

Message files are created and edited with the MSGMAKER utility routine, see the “Utility Routines” manual [15]. This applies both for message files which are supplied by the manufacturer and user message files.

15.2.1 System message files

Already at system initialization, it is possible to make various specifications for the message handling system via the parameter service:
In addition to the number of message files that can be processed via the DLAM access method (MSGDLAM) and the language in which messages are to be output by default (MSGLPRI), systems support also determines in the SYSOPT-CLASS2 parameter record the number of system message files and the file names to be defined by the parameter service. The following default settings apply to the system parameters MSGFIL01 and MSGFIL02: MSGFIL01=SYSMES.BS2CP.version or MSGFIL02=SYSMES.EKP.01. These files are created automatically during installation with IMON.

Systems support can specify a further 13 fully qualified message files (MSGFIL03-MSGFIL15).

The rest of the message files can be kept dynamically in an MIP parameter file. In contrast to the specifications in the SYSOPT-CLASS2 parameter record, this parameter file can have message files added or removed. The MIP file is created/managed by systems support with the MODIFY-MIP-PARAMETERS command and is given the name SYSPAR.MIP.<version> by default. It is stored under the TSOS ID and is structured as follows:

```
MIPPAR=SYSTEM
* this must be the first line
*
* comment lines begin with an "*"
*
MSGFILE=$.SYSMES.product_1.version
MSGFILE=$.SYSMES.product_2.version
:
```

The MIP parameter file can contain any number of declarations. However, since BS2000 can only manage a total of 255 message files, specifications in excess of this are ignored. A corresponding warning is output on the console.

The MIP parameter file can be empty but must always be cataloged at the time of system initialization. (If it is empty, the system asks for a new name. In this case it is advantageous to have an MIP parameter file with “MIPPAR=SYSTEM” in the first line.) A message file inserted with the MODIFY-MIP-PARAMETERS command is written at the beginning of the MIP parameter file, immediately after the opening line and any subsequent comment lines. The MIP parameter file can be read by means of the SHOW-MIP-PARAMETERS command.

The command MODIFY-MSG-FILE-ASSIGNMENT or MODIFY-MIP-PARAMETERS ..., SCOPE=*TEMPORARY can be used to temporarily influence the scope of the message files during the BS2000 session.

Systems support thus has three choices for defining the message files for the BS2000 session:

- at startup in the parameter service with the system parameters
- during startup but after MIP is loaded, with the MIP parameter file
- during the BS2000 session using the MODIFY-MIP-PARAMETERS and MODIFY-MSG-FILE-ASSIGNMENT commands

When the system run ends, all message files are closed.

15.2.2 User message files

Nonprivileged users can add or delete message files for their tasks (/MODIFY-MSG-FILE-ASSIGNMENT) and define the language for message output. They can therefore respond to error situations with their own messages.

The commands and macros concerned here are described in the “Commands” [27] and “Executive Macros” [30] manuals.

15.3 Guaranteed messages

For certain system messages in future subsystem versions or versions of BS2000 OSD/BC, the message code and the inserts (number and semantics) are guaranteed as invariable components of the message. These are known as warranty messages or **guaranteed messages**. Messages can be designated as guaranteed by means of the utility routine MSGMAKER.
MIP creates S variables for warranty messages; these variables can be used to access specific message data directly without having to know the output layout of the messages. For further information, see section "Output of messages in S variables".

The message attribute “warranty” (see "Warranty" in section "Message unit attributes") documents guaranteed messages.

Example

Message `JMS0066` is a warranty system message which is output when the ENTER-JOB command is successfully executed. The message contains four inserts and has the following structure:

```
% JMS0066 JOB '(&00)' ACCEPTED ON (&01) AT (&02), TSN = (&03)
```

Insert (&00) contains the job name, (&01) the date, (&02) the time and (&03) the TSN of the job. Allocation of the insert numbers to their contents is guaranteed, i.e. a TSN of the job (&00) is always output via the insert number (&03) of this message. In addition, the user can assume that message `JMS0066` is always sent when an ENTER-JOB command is successful.
The message text, on the other hand, can change in a subsequent system version:

```
% JMS0066 TSN (&03): JOB '(&00)' ACCEPTED AT (&02) ON (&01)
```

The order of the inserts has changed in the message text but their logical allocations remain the same. The meaning of the message, i.e. the information it contains, is also retained.

The following messages of BS2000 OSD/BC are guaranteed:

ACS0000	BLP0163	BLP0467	BLP0980	BLS0152	BLS0451
ACS0001	BLP0165	BLP0468	BLP0981	BLS0153	BLS0452
ACS0014	BLP0166	BLP0469	BLP0990	BLS0160	BLS0453
ACS0019	BLP0167	BLP0470	BLP0991	BLS0162	BLS0454
ACS0032	BLP0168	BLP0500	BLP0992	BLS0163	BLS0455
ACS0046	BLP0170	BLP0501	BLP0993	BLS0165	BLS0456
ACS0047	BLP0171	BLP0502	BLP0994	BLS0166	BLS0457
ACS0048	BLP0230	BLP0505	BLP0995	BLS0167	BLS0461
ACS0049	BLP0231	BLP0506	BLP0996	BLS0168	BLS0462
BLP0002	BLP0232	BLP0507	BLP0997	BLS0170	BLS0464
BLP0005	BLP0233	BLP0508	BLS0002	BLS0171	BLS0465
BLP0006	BLP0234	BLP0509	BLS0005	BLS0230	BLS0467
BLP0007	BLP0235	BLP0510	BLS0006	BLS0231	BLS0468
BLP0008	BLP0310	BLP0511	BLS0007	BLS0232	BLS0469
BLP0009	BLP0311	BLP0512	BLS0008	BLS0233	BLS0470
BLP0010	BLP0330	BLP0513	BLS0009	BLS0234	BLS0500
BLP0011	BLP0331	BLP0514	BLS0010	BLS0235	BLS0501
BLP0012	BLP0332	BLP0515	BLS0011	BLS0310	BLS0502
BLP0013	BLP0333	BLP0516	BLS0012	BLS0311	BLS0503
BLP0014	BLP0334	BLP0517	BLS0013	BLS0330	BLS0505
BLP0016	BLP0335	BLP0518	BLS0014	BLS0331	BLS0506
BLP0034	BLP0337	BLP0520	BLS0016	BLS0332	BLS0507
BLP0035	BLP0338	BLP0521	BLS0017	BLS0333	BLS0508
BLP0040	BLP0339	BLP0522	BLS0018	BLS0334	BLS0509
BLP0041	BLP0340	BLP0523	BLS0034	BLS0335	BLS0510
BLP0053	BLP0341	BLP0524	BLS0035	BLS0337	BLS0511
BLP0054	BLP0342	BLP0529	BLS0040	BLS0338	BLS0512
BLP0055	BLP0345	BLP0530	BLS0041	BLS0339	BLS0513
BLP0060	BLP0420	BLP0532	BLS0053	BLS0340	BLS0514
BLP0061	BLP0421	BLP0533	BLS0054	BLS0341	BLS0515
BLP0100	BLP0422	BLP0534	BLS0055	BLS0342	BLS0516
BLP0101	BLP0424	BLP0536	BLS0060	BLS0345	BLS0517
BLP0102	BLP0425	BLP0540	BLS0061	BLS0354	BLS0518
BLP0104	BLP0450	BLP0550	BLS0100	BLS0355	BLS0520
BLP0139	BLP0451	BLP0551	BLS0101	BLS0356	BLS0521
BLP0140	BLP0452	BLP0552	BLS0102	BLS0357	BLS0522
BLP0142	BLP0453	BLP0553	BLS0104	BLS0358	BLS0523
BLP0150	BLP0454	BLP0554	BLS0106	BLS0420	BLS0524
BLP0151	BLP0456	BLP0555	BLS0139	BLS0421	BLS0525
BLP0152	BLP0457	BLP0560	BLS0140	BLS0422	BLS0526
BLP0153	BLP0461	BLP0570	BLS0142	BLS0424	BLS0527
BLP0160	BLP0462	BLP0977	BLS0150	BLS0425	BLS0529
BLP0162	BLP0464	BLP0979	BLS0151	BLS0450	BLS0530

BLS0531	CMD0201	DMS03BE	DMS1D08	EMM2822	EMM2895
BLS0532	CMD0211	DMS051A	DMS1D09	EMM2823	EMM2897
BLS0533	CMD0214	DMS051B	DMS1D0A	EMM2824	EMM2898
BLS0534	CMD0216	DMS051C	DMS1D0B	EMM2825	ESM0201
BLS0535	CMD0221	DMS0533	DMS1D0C	EMM2826	ESM0202
BLS0536	CMD0300	DMS053C	DMS1D0D	EMM2828	ESM0203
BLS0537	CMD0302	DMS053F	DMS1D0E	EMM2829	ESM0204
BLS0538	CMD0490	DMS0574	DMS1D0F	EMM2830	ESM0205
BLS0539	CMD0500	DMS059C	DMS1D10	EMM2832	ESM0206
BLS0540	CMD0508	DMS05BF	DMS1D11	EMM2834	ESM0207
BLS0550	CMD0509	DMS05C3	DMS1D12	EMM2835	ESM0208
BLS0551	CMD0555	DMS05C6	DMS1D13	EMM2836	ESM0209
BLS0552	CMD0557	DMS05F8	DMS1D1F	EMM2837	ESM0210
BLS0553	CMD0601	DMS060A	DSP0006	EMM2838	ESM0211
BLS0554	CMD0671	DMS060B	DSP0007	EMM2839	ESM0212
BLS0555	CMD0672	DMS0666	ECCCOPY	EMM2840	ESM0213
BLS0560	CMD0674	DMS0681	EMM2303	EMM2842	ESM0214
BLS0570	CMD0678	DMS0682	EMM2304	EMM2843	ESM0215
BLS0977	CMD0679	DMS0683	EMM2305	EMM2844	ESM0216
BLS0979	CMD0680	DMS0684	EMM2307	EMM2846	ESM0218
BLS0980	CMD0681	DMS06D5	EMM2314	EMM2847	ESM0219
BLS0981	CMD0682	DMS0800	EMM2316	EMM2848	ESM0220
BLS0990	CMD0687	DMS0801	EMM2317	EMM2850	ESM0221
BLS0991	CMD0688	DMS0808	EMM2320	EMM2851	ESM0222
BLS0992	CMD0689	DMS0811	EMM2350	EMM2853	ESM0223
BLS0993	CMD0690	DMS0812	EMM2800	EMM2855	ESM0224
BLS0994	CMD2203	DMS0815	EMM2801	EMM2857	ESM0225
BLS0995	CMS0002	DMS0816	EMM2802	EMM2858	ESM0226
BLS0996	DMS0350	DMS0949	EMM2803	EMM2860	ESM0228
BLS0997	DMS0351	DMS09B5	EMM2804	EMM2863	ESM0230
CJC0002	DMS0352	DMS09B6	EMM2805	EMM2865	ESM0231
CJC0010	DMS0355	DMS0E49	EMM2806	EMM2867	ESM0232
CJC0011	DMS035A	DMS1343	EMM2807	EMM2870	ESM0233
CJC0020	DMS035B	DMS1373	EMM2808	EMM2877	ESM0235
CJC0021	DMS035C	DMS1374	EMM2809	EMM2878	ESM0237
CJC0022	DMS035D	DMS1376	EMM2810	EMM2880	ESM0238
CJC0025	DMS0362	DMS13C9	EMM2811	EMM2881	ESM0239
CJC0050	DMS0363	DMS13E2	EMM2812	EMM2882	ESM0240
CJC0051	DMS0364	DMS1D01	EMM2813	EMM2885	ESM0241
CJC0070	DMS036B	DMS1D02	EMM2814	EMM2886	ESM0242
CJC0072	DMS036C	DMS1D03	EMM2815	EMM2888	ESM0244
CJC0076	DMS036D	DMS1D04	EMM2817	EMM2889	ESM0245
CMD0001	DMS036E	DMS1D05	EMM2818	EMM2891	ESM0246
CMD0002	DMS037B	DMS1D06	EMM2819	EMM2892	ESM0249
CMD0093	DMS0382	DMS1D07	EMM2821	EMM2894	ESM0252

ESM0253	EXC0315	IDA0300	IMO9029	JMS0024	LMC0093
ESM0254	EXC0316	IDA0301	IMO9030	JMS0025	LMC0095
ESM0256	EXC0317	IDH0101	IMO9031	JMS0043	LMC0102
ESM0257	EXC0318	IDH0102	IMO9032	JMS0044	LMC0129
ESM0259	EXC031A	IDH0103	IMO9033	JMS0045	LMC0151
ESM0260	EXC031F	IDH0104	IMO9034	JMS0046	LMC0163
ESM0262	EXC032A	IDH0105	IMO9035	JMS0066	LMC0199
ESM0267	EXC032B	IDH0106	IMO9036	JMS0067	LMC0201
ESM0269	EXC0354	IDH0107	IMO9037	JMS0100	LMC0211
ESM0288	EXC0371	IDH0108	IMO9038	JMS0101	LMC0213
ESM0292	EXC0419	IDH0109	IMO9039	JMS0105	LMC0214
ESM0322	EXC0707	IDH0110	IMO9090	JMS0110	LMC0238
ESM0332	EXC0713	IDH0111	IMO9091	JMS0111	LMC0274
ESM0432	EXC0714	IDH0112	IMO9092	JMS0113	LMC0286
ESM0600	EXC0755	IDH0113	IMO9100	JMS0130	LMC0301
ESM0603	EXC0773	IDH0114	IMO9101	JMS0131	LMC0302
ESM0604	EXC0861	IDH0115	IMO9103	JMS0133	LMC0303
ESM0610	EXC0862	IDH0116	IMO9200	JMS0143	LMC0304
ESM0612	EXC0863	IMO9001	IMO9201	JMS0171	LMC0310
ESM0613	EXC0896	IMO9002	IMO9202	JMS0264	LMC0311
ESM0614	EXC0897	IMO9003	IMO9203	JMS0265	LMC0312
ESM0615	EXC0908	IMO9004	IMO9204	JMS0268	LMC0411
ESM0670	EXC0926	IMO9005	IMO9205	JMS0522	LMC0412
ESM0671	EXC0927	IMO9006	IMO9206	JPM0204	LMC0413
ESM0672	EXC0930	IMO9007	IMO9207	JPM0205	LMC0509
ESM0673	EXC0931	IMO9008	IMO9208	JPM0500	LMC0510
ESM0674	EXC0936	IMO9009	IMO9209	JVS04A2	LMC0712
ESM0675	GCF1014	IMO9011	IMO9210	JVS04A3	LMC0714
EXC0012	HEL0001	IMO9012	IMO9211	JVS04D0	LMC0721
EXC0123	HEL0002	IMO9013	IMO9212	JVS04D1	LMC1002
EXC0155	HEL0003	IMO9014	IMO9213	JVS04D2	LMC1003
EXC0156	HEL0004	IMO9015	IMO9214	JVS04D3	LMC1004
EXC0269	HEL0005	IMO9016	IMO9215	JVS04D4	MCA0005
EXC0300	HEL0006	IMO9017	JDS0301	JVS04D5	NAM0001
EXC0303	HEL0010	IMO9018	JDS0302	JVS04D6	NAM0002
EXC0304	HEL0011	IMO9019	JDS0303	JVS04D8	NAM0003
EXC0305	HEL0012	IMO9020	JDS0304	JVS0465	NAM0004
EXC0307	HEL0100	IMO9021	JDS0308	JVS0468	NAM0005
EXC0309	HEL0101	IMO9022	JDS0309	JVS0469	NAM0006
EXC030B	HEL0102	IMO9023	JDS0310	LMC0020	NAM0007
EXC030D	HEL0105	IMO9024	JDS0314	LMC0035	NAM0008
EXC0310	HEL0106	IMO9025	JDS0322	LMC0036	NAM0009
EXC0311	HEL0108	IMO9026	JMS0021	LMC0041	LMC0071
EXC0312	HEL0109	IMO9027	JMS0022	LMC0053	LMC0081
EXC0313	HEL0110	IMO9028	JMS0023	LMC0064	LMC0084

NAM0012	NBR0961	NKD0026	NKR0173	NKV0001	SDP0099
NAM0013	NBR0962	NKD0027	NKR0174	NKV0004	SDP0140
NAM0014	NBR1070	NKD0028	NKR0175	NKV0005	SDP0517
NAM0016	NBR1073	NKD0030	NKR0176	NKV0006	SDP1008
NAM0020	NBR1074	NKD0031	NKR0177	NMH1102	SDP1018
NAM0021	NDI0541	NKD0032	NKR0178	NMH1103	SDP1030
NAM0022	NER0000	NKD0033	NKR0179	NMH1104	SPF0300
NAM0023	NER1040	NKD0034	NKR0180	NMH1105	SPF0680
NAM0024	NER1060	NKD0035	NKR0181	NMH1106	SPF0681
NAM0025	NER1500	NKD0036	NKR0182	NMH1108	SPF0682
NAM0026	NER1510	NKD0037	NKR0188	NMH1109	SPF0688
NAM0027	NKA0094	NKD0038	NKR0189	NMH1116	SPF0689
NAM0028	NKA0096	NKD0039	NKR0191	NMH1117	SRM0003
NAM0029	NKD0002	NKD0040	NKR0192	NMH1120	SRM0004
NAM0030	NKD0005	NKD0041	NKR0193	NMH1123	SRM0016
NAM0031	NKD0006	NKR0018	NKR0194	NMH1124	SRM2101
NAM0032	NKD0007	NKR0073	NKR0195	NMH1154	SRM2103
NAM0033	NKD0009	NKR0074	NKR0196	NMH1155	SRM2151
NAM0034	NKD0010	NKR0075	NKR0197	NMH1165	SRM2210
NAM0035	NKD0011	NKR0124	NKR0201	NMH1166	SRM2211
NAM0036	NKD0013	NKR0125	NKR0202	NMH1180	SRM2401
NAM0037	NKA0094	NKR0147	NKS0008	NMH1181	SRM3102
NAM0039	NKA0096	NKR0148	NKS0010	NMH1183	SRM3999
NAM0040	NKD0002	NKR0149	NKS0011	PDT0200	SRM6001
NAM0042	NKD0005	NKR0150	NKS0012	PDT0201	SRM6010
NAM0043	NKD0006	NKR0151	NKS0014	PDT0202	SRM6020
NAM0044	NKD0007	NKR0160	NKS0015	PDT0203	SRM6030
NAM0045	NKD0009	NKR0161	NKS0036	PDT0204	SRM6040
NAM0048	NKD0010	NKR0162	NKS0046	PDT0205	SSM2012
NAM3001	NKD0011	NKR0163	NKS0047	PDT0206	SSM2039
NAM3003	NKD0013	NKR0164	NKS0053	PDT0207	SSM2052
NBR0031	NKD0014	NKR0165	NKS0055	PDT0208	SSM2053
NBR0032	NKD0016	NKR0166	NKS0056	PDT0209	SSM3056
NBR0033	NKD0018	NKR0167	NKS0057	PDT0210	SSM3100
NBR0034	NKD0019	NKR0168	NKS0062	PDT0213	SSM3101
NBR0200	NKD0021	NKR0169	NKS0063	PDT0214	SSM3284
NBR0724	NKD0023	NKR0170	NKS0068	SDP0090	
NBR0874	NKD0024	NKR0171	NKS0069	SDP0091	
NBR0960	NKD0025	NKR0172	NKS0071	SDP0092	

15.4 Message search

The message search begins in the task-specific memory area; the search criteria are the message code and the language identifier specified in the macro. If the search is not successful in the task area, it continues at system level. If the message with the specified message code and language identifier is still not found, the message search is first continued with the language set for the task, then with the first language set for the system and finally with the second language. If the search is unsuccessful, the message MESSAGE UNDEFINED is issued.
This process is outlined in figure 23, although not all details mentioned in the description are shown.

Message search at task level

At task level, the search begins in the task-specific message buffer in which the messages last output are stored. If the requested message is stored there, it is edited and output.If this is not the case, the class list is checked as to whether it contains the specified message class. For each message class known to the task, the class list contains the names of the corresponding message files. Messages that are output particularly frequently are specified in the message file with DLAM access. They are transferred to the task DLAM area with the task at the time of linking the message file. If the message class exists, the next step is to search the message file for the message code and the language identifier.Once the message has been found, the task-specific message buffer is updated with the “message framework”, the “message framework” is supplemented by the current inserts transferred by the macro MSG7/MSG7X, and the message is output.

Message search at system level

If the message is not found in any task-specific message file, the search is continued at system level in the system message buffer. If there is still no message that matches the specifications, the MIP task is called to continue the message search in the system message files. Here, too, in the same way at the task level, the DLAM area is checked first before the message files are searched.

Overview:

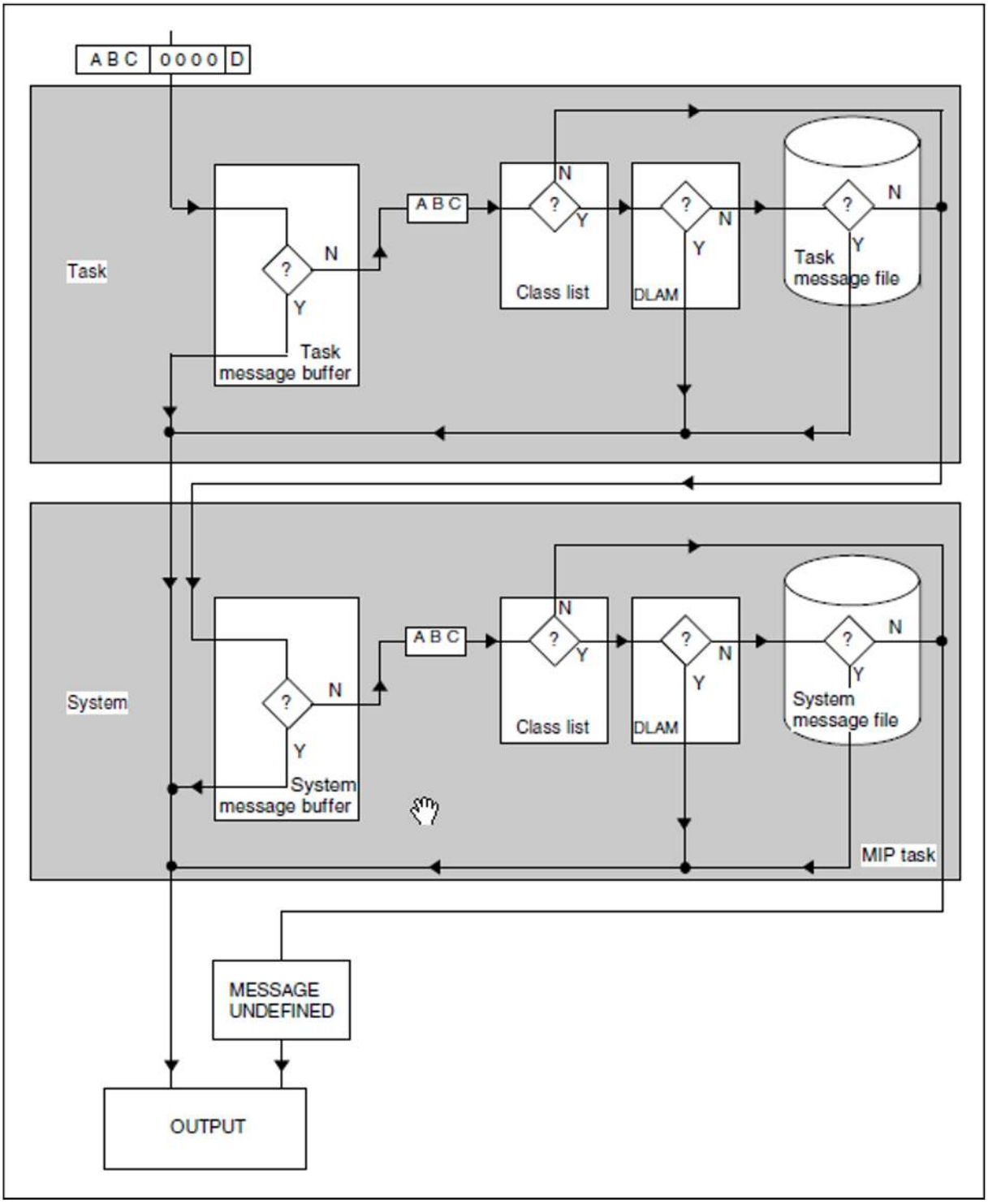


Figure 23: Message search in BS2000

15.5 Structure of a message unit

Each message file created using the MSGMAKER utility routine consists of a number of message units, each of which is accessed via a seven-character message code. The message unit comprises message attributes, message text and, if appropriate, a meaning and response text that explains in more detail the problem indicated in the message and indicates possible responses. All texts can be entered in up to eight languages. Each language is represented by a single letter, the language identifier.
Message output can be controlled using the message attributes assigned.

The structure of a message file as can be seen with the `/SHOW-FILE <message file>` command is shown below. After the identification line, messages appear which are specified for DLAM access. They are transferred to the system/task DLAM area together with the system/task at the time of linking the message file.

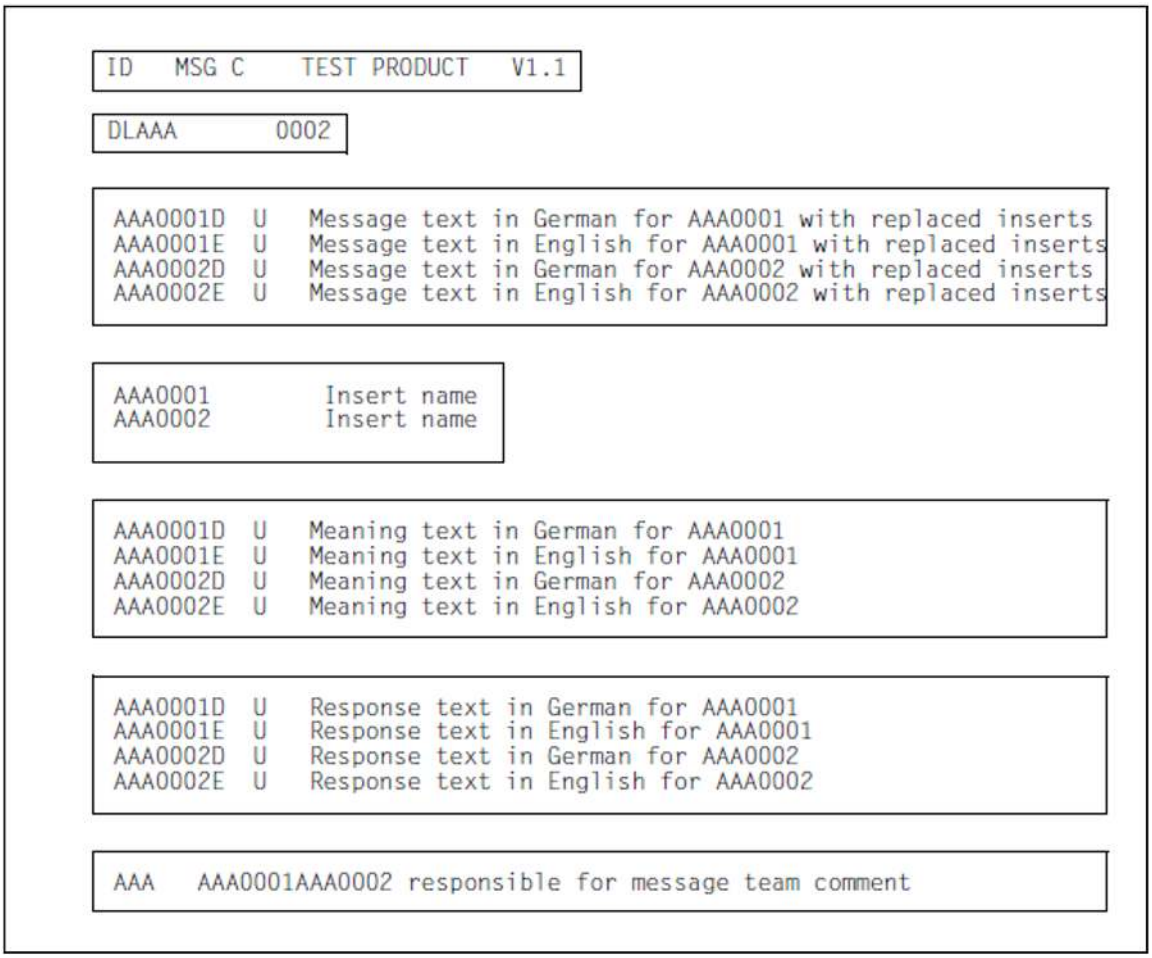


Figure 24: Output of a message file with the command /SHOW-FILE

15.6 Message code

The message code consists of seven characters. The first three letters are the message class and the final four characters are the message number.
The message class designates, for example, a BS2000 component, a subsystem or a module; the message number is used to number messages consecutively within a message class.
The message number usually consists of four numeric characters. Messages whose message number also contains letters are placed in the message file in front of messages whose message number contains only numeric characters.

Example

The message codes ABC123A through ABC123Z appear before the message codes ABC1230 through ABC1239 in the message file.

15.6.1 Language identifier

A single alphabetic character is used to identify the language in which the message, meaning and response texts are written. D is used for German and E for English; all other languages can have any abbreviation. A maximum of eight languages may be defined. In conjunction with the message code, the language identifier identifies a message text, meaning text and response text within the message unit.

15.6.2 Message unit attributes

MIP access method for messages

There are the following MIP access methods for messages: ISAM, DLAM, LOCAL-DLAM, BAMR and MINIMIP. The access methods LOCAL-DLAM, BAMR and MINIMIP are reserved for internal use.

Message output destination

Documents one or more output destinations for the message. Although this specification is not evaluated, it should correspond to the DEST operand of the MSG7/MSG7X macro.

Routing Code

If the message output destination is a console, the message is assigned an additional single-digit identifier, the routing code (RC). The routing code is analyzed to ascertain the message destination if subsidiary consoles are used in addition to the main console.If multiple consoles are used, they are assigned to specific task areas. The routing codes are then used by the system to correlate specific messages with the appropriate consoles.

The routing code defined with MSGMAKER is used only if no value is specified for the UCDEST operand of the MSG7/MSG7X macro.

AC/RC	Task area
A	System administration
C	Operation of the data communication system
D	Operation of disks
E	General tasks and responsibilities
W	Device management
H	Hardware maintenance
J	Job control
K	OPR command management
N	Monitoring of remote SPOOL
O	Operation of printers
P	Control of tasks
R	Monitoring and control of system behavior
S	Control of SPOOLOUT operation, load SPOOL
T	Operation of magnetic tape devices
U	File management
W,X,Y,Z	Available to the user

0	Messages with this authorization code cannot be requested. Messages without a response are simply stored in the CONSLOG file. Messages with a response (queries) are delivered to the main console.
9	VM2000 administration
#	POSIX
*	Main console
@	Main console (only for messages requiring a response; messages requiring no response are stored in the CONSLOG file but not output)
$	Special meaning; not used for message outputs
B,F,I,L,M,Q,V, 1,2,3,4,5,6,7,8	Reserved for future system expansion

Table 42: Task areas and the routing codes (RCs) of messages

See also the table 45 in section "Functional areas and their allocation to consoles".

Weight Code

The weight code determines the message priority.

If the output destination of a message is a console, a weight code must be specified; otherwise any weight code assigned has no meaning. Messages with a destination other than a console can be assigned WEIGHT=*NONE. If a message not intended for a console is sent to a console nevertheless, it is assigned the default weight code 99.

Weight codes can be assigned within the range 0 through 99. A message with the weight code 99 has the highest priority. This message cannot be suppressed.
A message that requires a response (REPLY operand of the MSG7/MSG7X macro) should be assigned the weight code 90 or 98.

To control message suppression, four filter levels can be set for each command at the console. These filter levels correspond to the weight codes of the messages as shown below:

Filter level	Weight Code	Standard values
1	0 - 19	None
2	20 - 39	30
3	40 - 59	50
4	60 - 79	70
Cannot be suppressed	80 - 99	90,98,99

The assigned weight codes (default) in the delivered message files can be changed by the systems support staff with the MASMAKER statement //MODIFY-MSG.

Filter level 1

The user can subsequently assign the weight of this range to messages which are considered irrelevant and allocate filter level 1 to the appropriate console or authorized user job to prevent the messages from being output there.

Filter level 2

A message with a weight code from this range is for information only and may be suppressed without jeopardizing safe operation.

Filter level 3

A message with a weight code from this range is for information only. This message may be suppressed without jeopardizing safe operation. The increased value is based on an additional aspect which may make output advisable, e.g. logging for data protection reasons or processing by an authorized user job.

Filter level 4

A message with a weight code from this range is for information only and does not require a response. However, the information is important for the operator. Its suppression might have adverse effects on the session.

Weight code range 80 - 99

A message with a weight code from this range either requires a response or is of utmost importance.

- Weight Code 90
  This weight code is assigned to acknowledgments for which, in conjunction with secured automation functions, suppression is conceivable in the future. At present, such acknowledgments cannot be suppressed.
- Weight Code 98
  A message with this weight code requires a response and may not be suppressed. The priority of weight code 98 is equivalent to that of weight code 99.
- Weight Code 99
  A message with this weight code has top priority and must not be suppressed under any circumstances (default value).

Warranty

The “warranty” message attribute is evaluated by MIP.

The attribute guarantees that for a message with such an attribute both the message key and the inserts (numbers and semantics) remain unchanged in future subsystem versions and versions of BS2000 OSD/BC.

MIP creates S variables for warranty messages (guaranteed messages). For detailed information, see section "Output of messages in S variables".

15.6.3 Message text

A message text must be written for each defined message code. Up to 30 inserts (see below) can be entered in the message text.
The message text is stored in the message file together with the message code and language identifier (see figure 24 in section "Structure of a message unit").
The following special rules must be observed when creating a message text:

- Separator “^”

  The message text is entered as a continuous line of text. To obtain a structured output, the user can insert the separator “^” at any position in the message text. Text that follows this separator is output on the next line. The SHOW function can be used to check how the message text will look.

- Character set restriction

  Messages may be output to different 7-bit terminals. Using other code tables results in duplicate character assignment.

  > The following characters may not be used in the message text, meaning text or response text if the texts are written in the standard languages, German and English. MSGMAKER issues a warning if one of these characters has been used. The restrictions do not apply to any other additionally defined languages.

Code	**Character**						
Hex code X' __'	4F	BB	BC	BD	FB	FD	FF
International EBCDIC.DF.03	|	\	[	]	<	>	~
German EBCDIC.DF.03	ö	Ö	Ä	Ü	ä	ü	ß

  Trailing blanks in a message text are suppressed. The last sentence or part sentence of a message text is ended without punctuation.

- Inserts

  Definition of inserts

  Up to 30 different inserts can be defined on a language-independent basis in the message texts of a message unit.
  The strings (&00), (&01) through (&29) are inserted in the message text; 00 through 29 are the numbers of the inserts. They can appear in any order in the text (e.g. 00, then 02, then 01) and do not have to be defined in an unbroken sequence (00, 08, 11).To ensure correct message output, it is essential for the defined inserts to be listed in the MSG7/MSG7X macro. If the message text contains more inserts than are specified in the macro call, the default text is used for each extra insert. If there is no default text, one of two things may occur:

  1. The number of the extra insert (e.g. (&02)) lies between the insert numbers specified in the macro call (e.g. (&00),(&01),(&03)). If this is the case an empty string is output.
  2. The number of the extra insert (e.g. (&03)) is higher than the insert numbers specified in the macro call (e.g. (&00),(&01),(&02)). The insert is not replaced, i.e. (&03) is output.

  If an entry is specified in the MSG7/MSG7X macro for an insert number that is not defined in the message text, MIP ignores the specification in the MSG7/MSG7X macro. No error message is issued.
  The default texts specified with MSGMAKER in the message replace the defined inserts only if no current text is defined in the MSG7/MSG7X macro.

  Insert names

  A mnemonic insert name can also be assigned to each insert. Insert names are used to create S variables.
  The assigned insert names are always converted to uppercase letters. For further information on the creation of S variables, see section "Output of messages in S variables".

  Default text for insert

  Text that is used instead of insert (&nn) in the output text. The default text is the contents of the I0 S variables (through I29).

  Automatic help

  The automatic /HELP-MSG-INFORMATION function is activated for the (part of) message code delivered as an insert value.

  > **i** *Presentation of inserts in statements*
  > If an insert is defined in a message text using the statement //ADD-MSG or //MODIFY-MSG, the & character must be entered twice.

- Length of the message text

  The message text may be up to 220 characters long, including the strings (&00)... (&29). This length specification does not include the current contents that are to replace these strings when the message is output. When completed, therefore, the message text can contain more than 220 characters.

  Sample message text:

  ```
  MSG-TEXT =THIS MESSAGE WILL BE SENT TO THE DESTINATION "(&04)",^
  IN LANGUAGE "(&00)", "(&01)" OR "(&02)"
  ```

  Maximum text length of messages displayed on data display terminals

  The maximum text length output by MSG7/MSG7X must not exceed 4079 characters. This includes the message text (without message code), insert values (possibly the default values), date and time (if required).

  In general the following applies: if a message text is too long, it is truncated when output. MSGMAKER issues no warning.

  Maximum text length of messages output to the console

  The maximum text length output by MSG7/MSG7X on the console is 230 characters, including the prefixed percentage sign % and the message code.

  MSGMAKER checks the length of the message text and of the default inserts. If the text length of 230 characters is exceeded, MSGMAKER issues a warning.

  The text length may vary depending on the current insert values or on the options of the MSG7/MSG7X macro. If TEXT-ONLY is specified, the message code is not output. The date and time entry preceding the message text may vary if DATESTAMP and TIMESTAMP are specified.

15.6.4 Meaning/response text

In order to explain the message text, an explanatory text can be formulated and issued on request. Answers and measures can also be offered which help solve the problem described in the message text. Meaning/response texts are given in the predefined language.

The meaning and response text is output to SYSOUT with the /HELP-MSG-INFORMATION command (see "HELP-MSG-INFORMATION command").
Both texts may consist of up to 256 lines of 74 characters each. Inserts or the separator “^” may be included in the text but do not have the special meaning that they have in the message text proper.
The last line of the meaning text and the response text should be concluded with a period.

The character set restrictions described for the message text also apply here.

15.7 Message output

- Message output forms
- HELP-MSG-INFORMATION command
- Messages that require a response
- Output of messages to user programs
- Output of messages in S variables

15.7.1 Message output forms

The following output forms can be selected for messages:

- **Normal form**
  comprises the message code, message text, if appropriate with current inserts
- **Short form**
  comprises the message code and, if appropriate, the current inserts, but no additional message text
- **Long form**
  comprises the normal form plus an explanatory text (meaning and response text)

The user can change the message output form with the aid of the INFORMATION-LEVEL operand in the MODIFY-JOB-OPTIONS command.
Declarations that have been made with SET-LOGON-PARAMETERS or ENTER-JOB and affect the logging of a job are redefined.

INFORMATION-LEVEL=*MEDIUM causes messages to be output in unabbreviated form (normal form).

INFORMATION-LEVEL=*MINIMUM causes messages to be output in coded form (short form).

Example

In order to demonstrate the different output forms of system messages, an attempt is made in the JOB.TEST job to output the attributes of the non-cataloged file ABCD. The system messages are logged in normal form and short form in the SYSOUT.JOB.TEST file.
The job is started with the /ENTER-JOB FROM-FILE=JOB.TEST command.

Job JOB.TEST

```
/SET-LOGON-PARAMETERS
/ASSIGN-SYSOUT TO=SYSOUT.JOB.TEST
/SHOW-FILE-ATTRIBUTES FILE-NAME=ABCD
/SET-JOB-STEP
/MODIFY-JOB-OPTIONS INFORMATION-LEVEL=*MINIMUM
/SHOW-FILE-ATTRIBUTES FILE-NAME=ABCD
/EXIT-JOB
```

SYSOUT log SYSOUT.JOB.TEST

```
/SHOW-FILE-ATTRIBUTES FILE-NAME=ABCD

%  DMS0533 REQUESTED FILE NOT CATALOGED IN PUBSET '2OSG'. COMMAND TERMINATED
%  CMD0205 ERROR IN PRECEDING COMMAND OR PROGRAM AND PROCEDURE STEP
TERMINATION                                                              1.

/SET-JOB-STEP

/MODIFY-JOB-OPTIONS INFORMATION-LEVEL=*MINIMUM                           2.

/SHOW-FILE-ATTRIBUTES FILE-NAME=ABCD

%  DMS0533 2OSG
%  CMD0205                                                               3.

/EXIT-JOB
%  EXC0419 /LOGOFF AT <time> ON <date> FOR TSN '2EZD'
%  EXC0421 CPU TIME USED: 0.5752
```

1. Output of system messages in normal form, because INFORMATION-LEVEL=*MEDIUM is the default setting.
2. Output form is set to INFORMATION-LEVEL=MINIMUM.
3. Output of system messages in coded short form.

15.7.2 HELP-MSG-INFORMATION command

This command is used to output explanatory texts (meaning and response text) relating to a message that has been received. The command can be entered either with or without operands, see the “Commands” manual [27].

HELP-MSG-INFORMATION without operands outputs the message last output by the system once more (if appropriate with current inserts) together with the meaning and response text in the same language.

HELP-MSG-INFORMATION with operands allows the user to specify a message code (MSG-IDENTIFICATION operand), the extent of information (INFORMATION-LEVEL operand), and the language (LANGUAGE operand).

Example

```
/help-msg-information msg-id=dms0533
%  DMS0533 REQUESTED FILE NOT CATALOGED IN PUBSET '(&00)'. COMMAND TERMINATED
%  ? This message is issued by DMS commands. The requested file is not
%    cataloged in the requested pubset.
%  RESPONSE : NONE %                                                    1.
/help-msg-information msg-id=dms0533,language=d
%  DMS0533 ANGEGEBENE DATEI IN PUBSET '(&00)' NICHT GEFUNDEN. KOMMANDO
%  BEENDET
%  ? Diese Meldung wird von DVS-Kommandos ausgegeben. Die angeforderte Datei
%    ist im gewuenschten Pubset nicht katalogisiert.
%  MASSNAHME : KEINE                                                    2.
/help-msg-information msg-id=dms0533,inf-level=*medium
%  DMS0533 REQUESTED FILE NOT CATALOGED IN PUBSET '(&00)'.
%  COMMAND TERMINATED                                                   3.
/help-msg-information msg-id=dms0533,inf-level=*minimum
%  DMS0533 (&00)                                                        4.
```

1. Output of the system message in long form. INFORMATION-LEVEL=*MAXIMUM is the default setting.
2. English was defined as the default language. If the message is required in German, this has to be specified separately.
3. Users can specify the language in which the messages are output for their tasks with the /MODIFY-MSG-ATTRIBUTES command.
4. Output of the system message in normal form, English as default language.
5. Output of a system message in coded short form.

15.7.3 Messages that require a response

For messages that require a response, MIP uses an interrogation mark as a standard answer. The interrogation mark is a reserved keyword in MIP. When the user enters an interrogation mark, MIP outputs the meaning and the response texts for this message and repeats the question.

```
%  CMD0213 TERMINATE PROGRAM? (Y=YES; N=NO)
?                                                                  1.
%  ? The user requested program termination by hitting a
%    function key or by NEXT line input.
%  ! Y: causes the END statement to be executed
%    N: termination request is ignored
%  CMD0213 TERMINATE PROGRAM? (Y=YES; N=NO)                        2.
```

1. The user enters a question mark. MIP outputs the meaning and the response texts for this message.
2. The message text is displayed a second time so that the user can then hopefully respond more easily.

15.7.4 Output of messages to user programs

Users have the option of having messages written by the system to a memory area belonging to their user program with the macro call MSG7/MSG7X (BUFFER operand). For more information refer to the “Executive Macros” [30] manual.

15.7.5 Output of messages in S variables

Messages are normally output via SYSOUT to the terminal or a temporary spoolout file. The software product SDF-P offers the additional option of outputting **guaranteed** messages in structured S variables (see section "Guaranteed messages"). SDF-P assumes control of the S variable stream SYSMSG, by means of which the guaranteed messages are diverted to S variables. More detailed information can be found in the “SDF-P” manual [45].
The system component MIP creates the S variables whose values can be used as input data in S procedures.

The figure below shows both the output to SYSOUT by the system file manager and the diversion of the output information to the S variable streams SYSMSG (and SYSINF) controlled by SDF-P. Further information on SYSINF can be found in the “Commands” manual [27].

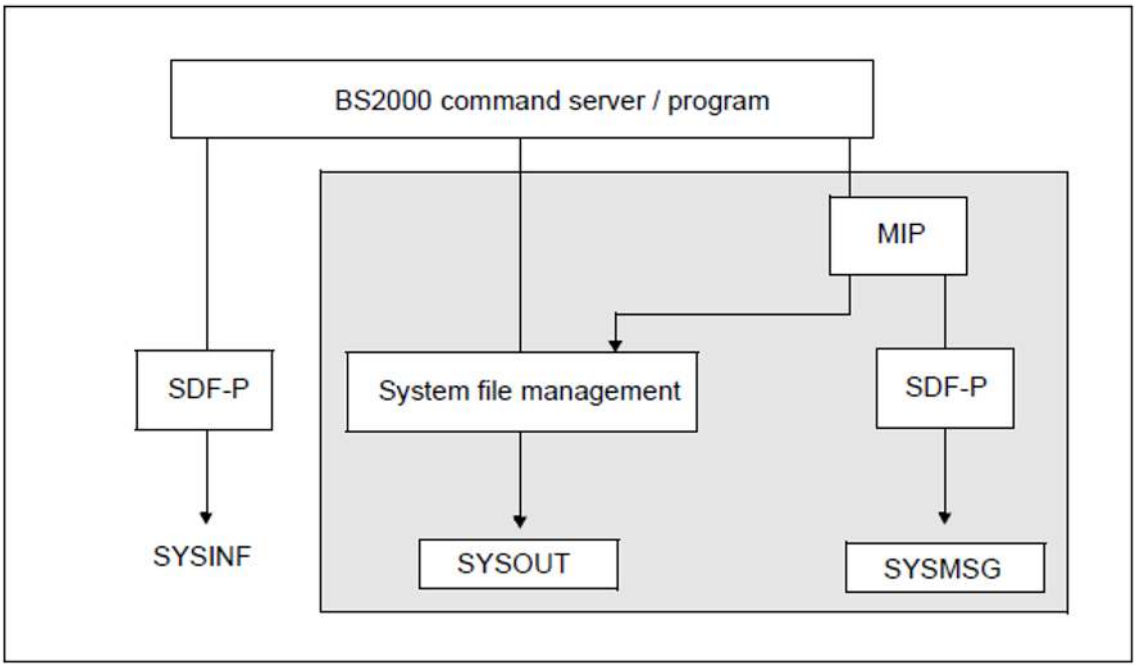


Figure 25: Output of messages to the S variables stream SYSMSG and to SYSOUT

The use of S variables offers users considerable advantages:

- The S variables permit simple access to the messages because the information is stored in structured format. The output layout of the message is immaterial for the user.
- The S variable name and the allocation of its contents are also guaranteed to remain the same in future subsystem versions or versions of BS2000 OSD/BC.
- The contents of the S variable can be used as input in S procedures.

The following entries are necessary in order to output guaranteed messages in structured S variables:

1. The user must declare a list variable of the type structure.

   `/DECLARE-VARIABLE <var-name>(TYPE=*STRUCTURE),MULTIPLE-ELEMENT=*LIST`

2. The user must assign the S variable stream SYSMSG for the structured output in S variables. One of the following declarations must be made:

   At command level:

   - The EXECUTE-CMD command declares the structured output in S variables for a command.
     `/EXEC-CMD (<cmd-name>),MSG-STRUCTURE-OUTPUT=<var-name>`

- The ASSIGN-STREAM command; the assignment of the S variable stream applies to all subsequent commands until the declaration is retracted (`ASSIGN-STREAM SYSMSG,TO=*STD`).
  `/ASSIGN-STREAM STREAM-NAME=SYSMSG,TO=*VARIABLE(<var-name>)`

At program level:

The CMD macro declares the structured output in S variables for two or more commands.
`CMD '<cmd-name>',...,VER=4,MSGVAR@=<adr>,MSGVARL=<len>,[,MSGEXT=YES]`

`<addr>` is the symbolic address contained in the S variable name.
CMD macro, see the “Executive Macros” manual [30].

3. The destination of the message output must be declared in the MSG7X macro. If DESTINATION=SYSOUT is specified, guaranteed messages will always be output in S variables, provided the user has declared an S variable.
   In addition, the BUFFER operand can be assigned a user-specific memory and BUFFUSE=EXTERNAL.
   MSG7X macro, see the “Executive Macros” manual [30].
4. The message must be identified as follows by MSGMAKER:
   a. The message to be output must be identified as guaranteed (//ADD-MSG MSG-ID=...,WARRANTY=YES). MIP creates the following S variables for warranty messages (guaranteed messages):
      - MSG-ID:
        contains the message code
      - REPLY:
        contains the reply to the message, if it requires a reply from the user.
      - I0 through I29:
        default names of the inserts; by default MIP allocates I0 through I29 as S variable names if the user declares no insert names using MSGMAKER (see b)).
      - MSG-TEXT:
        contains the message text including all replaced inserts.
   b. Names are declared for the inserts of the message to be output (//ADD-MSG MSG-ID=...,INSERT-ATTRIBUTES=...(NAME=...)). If the message is not guaranteed and a name is nevertheless assigned to at least one insert of the message to be output, the following S variables are created:
      - <insert-name>:
        name declared with MSGMAKER is converted to uppercase and used by MIP as S variable name
      - MSG-ID:
        contains the message code
      - REPLY:
        contains the reply to a message.
      - MSG-TEXT:
        contains the message text including all replaced inserts.

      > The S variables created by MIP are of the data type STRING.

The final S variable output by MIP is made up of two names:

- Name of the variable declared by the user: <var-name> (<var-name> is the first part of the S variable name)
- MIP supplies the second part of the name (MSG-ID, I0 to I29, <insert-name>, REPLY, MSG-TEXT)

Together the two parts of the name, separated by a period, make up the S variable name

- <var-name>.MSG-ID
- <var-name>.I0 ... <var-name>.I29
- <var-name>.<insert-name>
- <var-name>.REPLY
- <var-name>.MSG-TEXT

All the variables (MSG-ID, I0, ...) that exist for a warranty message are part of one list element. Any other message is included as a new list element. This applies when there are several warranty messages for one command (the command is called by means of EXECUTE-CMD) or the output of warranty messages is diverted to an S variable by the /ASSIGN-STREAM command.

Example

The example below illustrates how a message output is diverted to an S variable. The attributes of the file called TEST are to be output, but there is no corresponding file. Error message DMS0533 is a warranty message and is output in the S variable MSG:

```
/DECLARE-VARIABLE NAME=MSG(TYPE=*STRUCTURE(DEFINITION=*DYNAMIC)),
   MULTIPLE-ELEMENTS=*LIST                                          1.
/EXECUTE-CMD CMD=(SHOW-FILE-ATTRIBUTES FILE-NAME=TEST),TEXT-OUTPUT=*NONE,
   MSG-STRUCTURE-OUTPUT=MSG(WRITE-MODE=*EXTEND)                     2.
/SHOW-VAR MSG                                                       3.
MSG(*LIST).MSG-TEXT = %  DMS0533 REQUESTED FILE NOT CATALOGED IN
   PUBSET '1SBZ'. COMMAND TERMINATED
MSG(*LIST).MSG-ID = DMS0533
MSG(*LIST).I0 = 1SBZ                                                4.
```

1. The S variable MSG is declared as a list variable that can be dynamically extended.
2. The output of the SHOW-FILE-ATTRIBUTES command is diverted to the S variable MSG by means of the EXECUTE-CMD command. Output to SYSOUT is suppressed.
3. The list variable MSG is output by means of SHOW-VAR MSG.
4. MIP creates the S variables MSG-TEXT, MSG-ID and I0 for this warranty message. The values of these S variables represent the first list element of the S variable MSG. If the EXECUTE-CMD command is executed again, each further message is added as a new list element because WRITE-MODE=*EXTEND is specified (see (2)).

16 Operator functions

The activities to be performed by an operator during the session originate from various functional areas. These are, for example:

- monitoring and control of system behavior
- monitoring and control of applications
- control of the data communication system
- operating peripheral devices

Operating and consoles

The workstations from which the system is operated are referred to as consoles.

When the data communication system is started, the system automatically opens the BCAM application $CONSOLE. User programs that connect to $CONSOLE can also detect operating functions. They are therefore also known as consoles.

The terms “physical console” and “virtual console” are used to distinguish the two types.

Operating can be distributed to several consoles, divided according to functional areas. These can also be virtual consoles as used, for instance, by the software products MAREN and ROBAR in order to distribute operating tasks for tapes.

The OPERATING privilege can be assigned to any user ID with the SECOS command SET-PRIVILEGE. Operating can also take place from a user task of this user ID.

Physical console

Physical consoles were in the past special hardware devices which were connected directly to the BS2000 server.

Today these devices are emulated in the SE Manager or in VM2000. A console screen is emulated on the SE Manager by the KVP (console distribution program), see the “Operation and Administration” manual [57].

Every physical console is assigned an unambiguous mnemonic device name (mnemonic, MN). The emulations in the SE Manager or in VM2000 enable multiple instances to be opened on a physical console.

The entry of commands and the exchange of messages with the BS2000 system can take place from physical consoles, also in operating phases when BCAM is not available in the BS2000 system addressed.

Precisely one of the physical consoles is the main console. It has special rights. By default this is the IPL console from which system initialization took place. The main console property can, however, also be transferred to other physical consoles.

Two operating modes can be set, depending on the system parameter NBCONOPI: if a logon with an operator ID for physical consoles, the so-called operator logon, is required (NBCONOPI=Y), the authorizations of the physical consoles are controlled with the help of operator roles.

If no operator logon for physical consoles is allowed (NBCONOPI=N), the routing codes are assigned via the parameter service (OPR parameter record) or using the ASR command.

An operator ID is any user ID with the OPERATING privilege.

> When NBCONOPI=Y, the operator works under his/her own user ID and has his/her own file space. For RUN files, the file name is supplemented with this user ID and no longer with TSOS.

If the operator logon is activated (NBCONOPI=Y), after “System ready” operating from a physical console is only possible after successful logon under an ID with the OPERATING privilege (with SET-LOGON-PARAMETERS).

Before logon, only the SHOW-PENDING-MSG command for displaying open queries is permitted. After successful logon, the operator still has no authorizations in the form of routing codes. These can be obtained with the REQUEST-OPERATOR-ROLE command. The current authorizations of the local ID are displayed with the SHOW-OPERATOR-ROLE command.

To log off from the system enter the EXIT-JOB command. This also implies the release of all routing codes (RELEASE-OPERATOR-ROLE OPERATOR-ROLE=*ALL).

The SYSOPR ID with the OPERATING and STD-PROCESSING privileges is always present in the system.

Virtual consoles/$CONSOLE applications

“Virtual consoles” are user programs which have established a connection to the system application $CONSOLE with the help of DCAM or some other network functions and communicate with the UCON task over this connection. They are therefore also called “$CONSOLE applications”. Depending on the type of logon with $CONSOLE, a distinction is made between “unauthorized” and “authorized” applications.

Virtual consoles can only be operated if BCAM is available in the BS2000 system.

Authorized virtual consoles (as with the MN of physical consoles) have a unique authorization name which can be used to identify them. A distinction is made here between statically generated authorization names and dynamically assigned authorization names.

Certain system operation tasks can be automated with authorized $CONSOLE applications. They are then referred to, for example, as “programmed operator” or “automatic operator”. They can, however, also enable interactive system operation over a network connection.

The $CONSOLE interface for virtual consoles is described in chapter "Automation of operator functions".

16.1 User task with the OPERATING privilege

System operation is possible from user tasks under an ID with the OPERATING privilege. The OPERATING privilege can be granted to any ID through SECOS. The system user ID SYSOPR is given the OPERATING and STD-PROCESSING privileges by default.

The data stream known to a console (so far only consoles) can be logically separated into a dialog stream in accordance with the active system operation and an event stream in accordance with the reactive system operation.

The **dialog stream** is the command-based portion of a data stream of a console. It processes the command dialog, i.e. the input of commands and output of command-based messages, and, if necessary, answers requests from command processing for additional information. The system-internal SCI interface (synchronous console interface) allows all operating commands to be input from a user task.

Command input in the dialog stream is synchronous, i.e. a command cannot be entered until the previous one has terminated. This is a significant difference from the command dialog on consoles where command input is asynchronous, i.e. commands can be entered irrespective of whether preceding commands have terminated. The synchronous function of the dialog stream allows the use of SDF-P for operating procedures in user tasks with the OPERATING privilege.

The **event stream** is the system-based, asynchronous portion of the data stream of a console. It basically handles unsolicited system messages (known to the consoles), to which the operator may have to respond. This functionality is integrated in a user task within the framework of the **event stream service** (ESS), which opens up new possibilities, even for users without the OPERATING privilege. An important feature of the event stream service is the ability to present messages of the system asynchronously, i.e. immediately after they have been generated on a data display terminal, within an interactive task.

A central system event-stream file (SESF) is created for this purpose, in which all system events are recorded by the system task SEST.
The name of the system event-stream file is $SYSAUDIT.SYSLOG.ESS.SYSTEM. It is created when the system is booted and is deleted again during shutdown.
The maximum size of the system event stream file can be set with the system parameter NBESSIZE. The default setting is 40000 PAM pages. The maximum retention period of the recorded events is 96 hours. They are deleted to reclaim space after this time.

System events are basically all unsolicited system messages distributed via routing codes and not linked to commands, which are also contained in the CONSLOG file. The ASSIGN-SYSEVENT command allows user tasks with the OPERATING privilege to assign themselves their own system event stream. For these, asynchronous messages directed to the user's own task and, depending on the setting in the ASSIGN-SYSEVENT command, parts of the user's own command dialog, are also recorded as system events in the system event-stream file.

The recorded events are displayed by a presentation function with mask-based display using FHS-TPR. The SHOW-SYSEVENT-LOG command starts the presentation function of the event stream. In user tasks with the OPERATING privilege, all system event streams present in the system can be displayed. Asynchronous messages directed at the user's own task and logged sections of the command dialog are always shown. The scope of the system event display depends on the so-called “viewer profile”, which must be set before the presentation function is called. This viewer profile is determined by the total number of routing codes set through the operator roles as well as by the settings for message suppression, message filtering and message ordering.

The presentation function provides a dynamic and a static mode. The dynamic mode allows immediate display of new, asynchronous events as they occur (as known from the console), but does not provide a dialog function. The static mode provides a mask-controlled user dialog for scrolling and positioning within the complete event stream, a search function, a function for answering open system questions and a help system. The presentation is available in either English or German, depending on how the message language is set for each particular task.

To prepare the output masks, the presentation function uses the subsystem FHS-TPR, which is usually available as of “System ready”. The necessary access to the FHS mask libraries SYSFHS.BS2CP.<version>.E or SYSFHS.BS2CP.<version>.D and SYSFHS.FHS-TPR.<version>.E or SYSFHS.FHS-TPR.<version>.D is made possible at system installation.

The SHOW-SYSEVENT-LOG-ATTRIBUTES command provides information on the characteristics and attributes of the accessible event streams.

Any user tasks can assign themselves their own user event streams which do not contain any operating functionality. For these, the events set with the ASSIGN-SYSEVENT command are recorded in a private user event-stream file (UESF). This file is created under the user's own ID and is called SYSLOG.ESS.USER.<log-id>.

User event streams are particularly suitable for recording and displaying asynchronous messages to the user (e.g. INFORM-JOBS or INFORM-ALL-JOBS) and for monitoring operations which hide behind a job command (e.g. ENTER-JOB). The presentation function is also started with the SHOW-SYSEVENT-LOG command. All existing user event streams of the user's own ID can be displayed.

In system operation from consoles, the operator is used to a mixed display of dialog stream and event stream. In system operation from a user task, this mixed form of display is not possible for technical reasons. A user-friendly workplace requires at least two interactive tasks with two terminals, or with one terminal emulation which supports multiwindowing. One interactive task is used for operating the dialog stream, the other for presenting the event stream. If further interactive tasks or presentation windows are available, several presentation functions with different viewer profiles according to different functional areas can be used in parallel. If only one interactive task is available, system operation is still possible from the user task, but in this case it is necessary to switch between the dialog stream and the event stream (starting and ending the presentation function as necessary).

16.1.1 Providing an ID for the operating function

If the operating functionality is to be provided from physical consoles (if the operator logon is activated), $CONSOLE applications with dynamic authorization names and user tasks, the preparations described below must be made. For the existing system user ID SYSOPR to be used, only the creation and assignment of operator roles is necessary.

Under the TSOS ID

- Set up an ID:
  `/ADD-USER USER-ID=OPEROPER, LOGON-PASSWORD=C'12345678', ACCOUNT=account`
- If the chargeable product SECOS is in use, for $CONSOLE applications with dynamic authorization names it may be possible to define extended access protection using the following command.

```
/MODIFY-LOGON-PROTECTION USER-ID=OPEROPER, ...,
                         OPERATOR-ACCESS-TERM= ...,
                         OPERATOR-ACCESS-PROG= ...
```

Under the SYSPRIV ID

- Assign the OPERATING privilege (not always necessary for $CONSOLE applications):
  `/SET-PRIVILEGE PRIVILEGE=OPERATING, USER-ID=OPEROPER`
- Set up operator roles:
  `/CREATE-OPERATOR-ROLE OPERATOR-ROLE=OPERROLE, ROUTING-CODES=<list>`
- Assign the operator roles to the ID:
  `/MODIFY-OPERATOR-ATTRIBUTES USER-ID=OPEROPER,ADD-OPERATOR-ROLE=OPERROLE`
- If desired, additional assignment of operator roles to the system operator's user ID:
  `/MODIFY-OPERATOR-ATTRIBUTES USER-ID=SYSOPR,ADD-OPERATOR-ROLE=OPERROLE`

Following these preparations and the entry of the SET-LOGON-PARAMETERS command with user ID, account number and password, it is possible for an operator, taking account of any special conditions, to perform operation from a physical console, a $CONSOLE application with dynamic authorization name or a user task. The operator requests the routing code via the REQUEST-OPERATOR-ROLE command.

16.1.2 Using the event stream service for operating from user tasks

A well-designed operator workplace is based on at least two user tasks. One user task (user task 1) is used for the input of commands and to receive the associated command results (dialog stream). The second user task (interactive task 2) is used for presentation of the asynchronous system messages (event stream).

Dialog stream (user task 1):

- Any commands protected with the OPERATING privilege may be entered.

  The authorization check is performed on the basis of the SDF syntax files (including any previously added group syntax file).

  For commands which are processed via the internal SCI interface of the UCON task in an operator task (environment), there is no authorization check based on the authorization codes in the command tables of the UCON task. Exception: commands protected with the authorization code “$” are also prohibited from within user tasks.
- Setting up a system event stream to receive asynchronous system messages:
  `/ASSIGN-SYSEVENT TO=*SYSTEM (LOG-ID=OPER),ADD-SYNCH-EVENTS=*NONE`

  ADD-SYNCH-EVENTS=*ALL can be used to also log the entire command dialog of user task 1 in the event stream OPER. The scope of the events (messages) to be logged is set using the command ASSIGN-SYSEVENT ..., ADD-SYNCH-EVENTS.

  The asynchronous messages sent directly to user task 1 are always logged. Depending on the setting (MODIFY-MSG-OPTIONS command), these are also output on the data display terminal. Output at the terminal can be suppressed with the following setting:
  `/MODIFY-MSG-OPTIONS OPERATOR-BROADCAST=*NO,OPERATOR-MSG=*NO,SYSTEM-MSG=*NO`
- The presentation of the event stream in the same task is possible but not advisable, because no commands can be entered during the presentation.
- The presentation of the event stream comes from the central system event-stream file. This contains all necessary messages (selectable in the presentation), within the limits set by the system parameter NBESSIZE, from the last hours of the session (96 hours maximum).

Presentation of the event stream (interactive task 2):

- Interactive task 2, also under an ID with the OPERATING privilege, is used only for the presentation of the event stream. The event stream assigned in user task 1 is accessible to interactive tasks from all IDs with the OPERATING privilege and is uniquely identifiable by its name (LOG-ID).

- For the presentation, the events to be displayed are selected, on the basis of the following criteria, from all the recorded events via the task-specific setting of the viewer profile. The individual criteria are listed in order of priority, i.e. as soon as one criterion is fulfilled, the ones that follow are no longer checked:
  1. Logged sections of the command dialog of user task 1 are always displayed.
  2. Asynchronous messages sent directly to user task 1 are displayed.
  3. System messages suppressed on the basis of their MSG-ID (SET-MSG-SUPPRESSION command) are not displayed, provided they are not queries.
  4. System messages ordered on the basis of their message ID (MODIFY-MSG-SUBSCRIPTION command) are displayed.
  5. System messages distributed via routing code are selected for display by means of the operator roles set (REQUEST- and RELEASE-OPERATOR-ROLE commands).
  6. In the case of messages distributed via routing code, the routing code-specific filter levels set (ADD- and REMOVE-CONSOLE-FILTER commands) are checked against the message weight.
  7. Event-stream-specific messages (e.g. successful recoveries or reorganizations) are always displayed, irrespective of the task-specific settings.
- The presentation function is started with the command
  `/SHOW-SYSEVENT-LOG LOG-ID=OPER(TYPE=*SYSTEM)`

  The presentation function appears with its main mask in static mode and offers input options via menu, command line and F keys. The current mask always shows the most recent events.

  The following functions are offered in static mode:
  - position and page within the event stream
  - search
  - respond to open system queries
  - switch to dynamic mode
  - provide a help system

  Dynamic mode offers the most important functionality for the system operator: the immediate output of events which are reported asynchronously as from the console. The only possible input is the K2 key to return to static mode.

If only the presentation of system events without “private sections” is desired, it is not necessary to set up a system event stream with the ASSIGN-SYSEVENT command. The direct call of the presentation function with
`/SHOW-SYSEVENT-LOG LOG-ID=*SYSTEM-MSG-ONLY`

shows the choice of system events according to the “viewer profile” of the operator.

For a detailed description of the presentation function, see the description of the SHOW-SYSEVENT-LOG command.

16.1.3 Using the event stream service for any user tasks

The functionality of the event-stream service can be used for any user task but is restricted by the specific operating functionality. It is a good idea to use at least two user tasks per workplace. One user task (user task 1) is used for the input of commands and to receive the associated command results. The second user task (interactive task 2) has to be an interactive task and is used for the presentation of asynchronous messages directed to the user task (e.g. BROADCAST), as well as of parts of the command dialog of user task 1, depending on the settings.

For every user event stream, the events are recorded in a separate user event-stream file in a separate batch task of the same ID. The name of the file is SYSLOG.ESS.USER.log-id. The suffix of the file name forms the name (LOG-ID) of the event stream. Access to the file is only possible via the presentation function. The lifetime of the batch task corresponds to the active phase of the event stream (from starting to termination or closing). The user event stream file is created the first time the event stream is started and deleted with final termination.

The name of the event stream (LOG-ID) is unique within the ID. Access to a user event stream is possible from all interactive tasks of the same ID. There is no special authorization for the TSOS ID.

Dialog stream (user task 1):

- Setting up a user event stream to receive asynchronous messages:
  `/ASSIGN-SYSEVENT TO=*USER (LOG-ID=MYES), ADD-SYNCH-EVENTS=*NONE`

  ADD-SYNCH-EVENTS=*ALL can be used to also log the entire command dialog of user task 1 in the event stream MYES. This means that the scope of the events to be logged (messages) is set by means of the ASSIGN-SYSEVENT command and the ADD-SYNCH-EVENTS operand.

  The asynchronous messages sent directly to user task 1 are always logged. Depending on the setting (MODIFY-MSG-OPTIONS command), these are also output on the data display terminal. The output on this terminal can be suppressed by the following setting:
  `/MODIFY-MSG-OPTIONS OPERATOR-BROADCAST=*NO.OPERATOR-MSG=*NO,SYSTEM-MSG=*NO`
- The presentation of the event stream in the same task is possible but not advisable, because no commands can be entered during the presentation.

Presentation of the event stream (interactive task 2):

- Interactive task 2, also under a task of the same ID, is used only for the presentation of the event stream. The event stream assigned in user task 1 is uniquely identifiable by its name (LOG-ID).
- All recorded events are always displayed, i.e. there is no “viewer profile” for user event streams.
- The presentation function is started with the command
  `/SHOW-SYSEVENT-LOG LOG-ID=MYES`

  The presentation function appears with its main mask in static mode and offers input options via menu, command line and F keys. The current mask always shows the most recent events.

  The following functions are offered in static mode:
  - position and page within the event stream
  - search
  - switch to dynamic mode
  - provide a help system

For a detailed description of the presentation function, see the description of the SHOW-SYSEVENT-LOG command.

16.2 Command input via console

Keyboards differ from server type to server type.

Command input

1. Press EINGEB (= input) key.
2. Enter a slash (/), followed by the command and the required operands.
3. Press EXT key (DUE1 or ENTER key)

Command formats

```
/[.cid] cmd[ operands]
```

.	Period
cid	command job ID The cid starts with a letter or one of the characters @, $, #, which may be followed by up to 7 additional characters (A-Z, 0-9, or @,$,#). In its format, it corresponds to the job name for user commands (see “Commands” manual [27]). The cid serves to identify the jobs associated with an operator command. The first three characters of the cid appear as the mid (message job ID, see “Formats of the messages”, "Format for receiving messages in dialog between system and user" in section "Message formats") in the command termination message issued by the system for all operator commands and in all command results.
cmd	Name of an operator command or special command.
''	Blank
operands	Command operands

The commands can be abbreviated as desired as long as they remain unambiguous.

Example

```
1. /.MINE CANCEL-JOB *TSN(1532)

2. /.YOURS SPECIAL OP1=<value1>,OP2=<value2>
```

The first command is an ordinary operator command and the second is a special command.

16.3 Messages

- Emergency messages
- Communication between system and operator
- Message formats
- Control of message delivery
- Communication between operators

16.3.1 Emergency messages

Critical system problems, such as memory shortfalls or hardware errors on system disks, may affect the BS2000 components which are responsible for the input and output of console messages. In situations in which the normal output paths are blocked, the system selects an alternative output path for important messages, the so-called “emergency path”.

Outputs via the emergency path are known as an emergency messages or emergency queries. They are initially output directly at the system’s main console via internal service interfaces.

Formats

%E.text	Emergency message
?E.text	Emergency query

After this - if possible - the messages from the regular message processing functions are transferred. See the next section.

Irrespective of the logical console output or display by means of the SHOW-PENDING-MSG command, it is only possible to answer emergency queries at the physical console at which the message was output.

Response format

E.text	Response to emergency query

16.3.2 Communication between system and operator

The system and its users have the option of sending messages to the consoles and receiving responses from them, where the “user” can also be an operator who is himself sitting at a console. By their nature, the contents and meaning of these messages are very diverse. To distinguish which messages to handle in which way, they are distinguished by types and attributes.

Types of message

To enable the recipient to tell as soon as possible what type and priority of message is involved, the system splits them up into types. All messages output at the console are assigned to one of these types and has a unique type identifier which is output with it.

The following types of message are defined:

% “Message”: Information message to be noted by the operator.

? “Query”: Message which requires a response.

. “Response”: Response by the operator to a query.

; “Instruction”: Message which the operator is asked to do something. From the internal point of view of the system, instructions are also questions, but differ in that the response cannot be given by the operator, but only by the sender of the query.

/ “Command”: Type identifier for the entry of a command.

+ “Command result”: Message from a command server to the sender of the command.

& “Additional command information request”: Message in which the sender of a command is asked to enter further information.

: “Additional command information”: Response by the issuer of the command to a additional command information request.

! “Command termination”: Message which the person who enters the command is informed of the end of runtime monitoring by the system. This is normally identical with the termination of the command, but there are exceptions. For more details, see the section "Command termination messages".

* “Rejection”: Message with which UCON rejects an errored entry. All possible rejections are set out in section "Return messages on physical and virtual consoles".

Message attributes

Certain other attributes of a message, not always apparent from the type, are also crucial for the system's message handling mechanisms and commands. The following properties are distinguished:

- “response” and “non-response” messages
- “command-linked” and “non-command-linked” messages
- “directed” and “non-directed” messages

Response and non-response messages

All messages of the following type are “response”:

- query
- request for additional command information
- instruction

All messages of all other types are “non-response”.

This message attribute is important, for instance, on output, where response messages are highlighted (on color screens, e.g. red instead of green) and can be redisplayed with the SHOW-PENDING-MSG command.

Command-linked and non-command-linked messages

All messages of the following types, including the command itself, are “command-linked”:

- command result
- request for additional command information
- additional command information
- command termination

All messages of all other types are “non-command linked”.

This message attribute is important when programming command servers for special operator commands (see "Special operator commands in authorized user programs").

Directed and non-directed messages

“Directed” messages are directed to a precisely defined recipient.

In contrast, “non-directed” messages have a routing code specified as the recipient. They are distributed by the system to all consoles that contain the specified routing code - this may be a large number or none at all.

Unlike the other message attributes, the message type here is not a unique criterion: responses (to the person making the query) and all command-linked messages are indeed directed, but the other types of message may be both directed and non-directed. This can be reliably identified only in the CONSLOG, where the recipient of each message is logged at the same time, and in authorized user programs that contain their messages in expanded form (see ''Authorized user programs with operator functions'').

Emergency messages are treated here in a special way. For technical reasons they are initially directed to the main console. Then, where still possible, they are distributed on a non-directed basis via the routing code *, but only to virtual consoles, not to physical ones.

This message attribute is important, e.g. in message filtering (ADD-/REMOVE-CONSOLE-FILTER commands). This mechanism affects only undirected messages with message type “%” (message).

Output format

All messages are provided with a uniform, 17-character prefix on output to a console. Exception: responses to queries from the operator have no time specification

Column 1	Message type identification: for possible value, see "Communication between system and operator"; Exception: only “?” and “Y” for emergency messages
Columns 2-5	Sender of the message; possible notations: • (mn): Sender is the physical console with the MN “mn” • name: Sender is the virtual console with the authorization name “name“ • tsn: Sender is the task with the TSN “tsn”
Column 6	Hyphen (minus sign)
Columns 7-9	Message job ID: three-digit, alphanumeric message identification
Column 10	Separator: If the system parameter SECSTART=N is specified, the separator is always a period. If SECSTART=Y is specified, the separator is a “#” if the message originates from the system itself, and is thus “trusted”. Other messages, e.g. output by the user with INFORM-OPERATOR, have a period as a separator.
Columns 11-16	Time in the format: hhmmss (in the case of responses, the response text starts in column 11).
Column 17	Blank
Columns 18ff	Text of message

Example

```
% 6C5-000.122342 % BLS0519 PROGRAM 'EDT' LOADED
```

16.3.3 Message formats

The following formats are described:

- Format for receiving messages in dialog between system and user
- Format for sending responses
- Format for command-linked responses
- Format for entering operator commands
- Format of command results and command termination messages

Format for receiving messages in dialog between system and user

```
{?|%|;}tsn-mid{.|#}hhmmss text
```

%	Message, response not required
?	Message requiring a response that can also be given by the operator
;	Message requiring a response that cannot be given by the operator
tsn	Task serial number of the task that caused the system message
-	Hyphen
mid	Message job ID, default value = 000 (A..Z, 0-9, @, #, $)
.	Period as separator. The sender is a user program.
#	Hash as separator. The sender is a system module. The use of period or “#” depends on the class 2 system parameter SECSTART.
hhmmss	Time (6 digits; hh = hour, mm = minute, ss = second)
text	Freely selectable message text

Example

```
%TSN1-000#142423 % NBR0970 OPERATOR TASK WITH TSN 'XAAA' CREATED FOR
                   CONSOLE 'K1'
?TSN9-000#142753 % NBR0820 SHOULD K1 BECOME MAIN OPERATOR TERMINAL ?
;TSN4-001#142823 % XYZ0123 SWITCH OFF DEVICE L1 URGENTLY
```

Messages of types “?” and “;” can be displayed again using the SHOW-PENDING-MSG command.
Messages of the type “;” can only be removed from the queue of unresolved queries by the sender of the message.
The operator has only indirect influence on this, in that he/she gives the response required by the sender (in the above example, switching off the device L1).

Format for sending responses

```
dest [-mid].[text]
```

dest	Destination of response This is the sender of the message to which a response is to be given.
-	Hyphen
mid	Message job ID (A..Z, 0-9, @, #, $); leading zeros may be omitted (mid must correspond to the message requiring a response)
.	Period, indicating a response
text	Arbitrary response text. Only certain texts are permissible for responses to system messages. You can obtain help on individual messages using /HELP-MSG-INFORMATION.

Example

```
TSN9-000.YES
```

```
TSN9.YES
```

Format for command-linked responses

```
dest [-mid]:[text]
```

dest	Destination of additional command information This is the sender of the message to which a response is to be given.
-	Hyphen
mid	Message job ID (A..Z, 0-9, @, #, $) (must correspond to the message to which a response is to be given)
:	Colon, indicating command status information
text	Arbitrary response text. Only certain texts are permissible for responses to system messages. You can obtain help on individual messages using /HELP-MSG-INFORMATION.

Example

```
TSN3-TUR:Y
```

Format for entering operator commands

```
/[.cid] cmd[ operands]
```

/	Slash. Character indicating a command
-	Period. Character indicating that a “cid” is specified
cid	command job ID The cid starts with a letter or one of the characters @, $, #, which may be followed by up to 7 additional characters (A-Z, 0-9, or @,$,#). In its format it corresponds to the job names for user commands (see “Commands” manual [27]). The cid serves to identify the jobs associated with an operator command. The first three characters of the cid appear as the mid (message job ID, see "Message formats") in all messages associated with the command (command results, requests for additional information and the additional information itself).
cmd	Describes the operation performed by an operator command
' '	Blank
operands	Command operands

Examples

```
/.MINE CANCEL-JOB *TSN(1532)
/SPECIAL-CMD OP1=<value1>,OP2=<value2>
/.EVAN16 SHOW-USER-ATTRIBUTES USER-IDENTIFICATION=RZ01
/.E SHOW-USER-ATTRIBUTES USER-IDENTIFICATION=RZ01
```

Unlike message job IDs, command job IDs may be no more than eight characters long. Command job IDs consisting of only one or two digits are filled to as many as three digits with leading zeros. Command job IDs must begin with a letter or the special characters $,#,@. The first three characters of the command job ID appear as the message job ID.

Format of command results and command termination messages

`{+|&|!}{tsn|an}-mid{.|#}hhmmss text`

+	Command result; no response required
&	Request for additional command information; the sender requires a response
!	Command termination; no response required
tsn	Task sequence number of the task in which the command is processed
on	Authorization name of the authorized user program performing command processing (special operator command)
-	Hyphen
mid	Message job ID, default value = 000 (A..Z, 0-9, @, #, $)
.	Period as a separator
#	Number character (hash) as a separator Whether the separator is a period or the “#” depends on the system parameter SECSTART.
hhmmss	Time of day (6 digits; hh = hour, mm = minute, ss = second).
text	Freely selectable message text

Examples

```
&TSN3-TUR.142623  %  EXC0422 PROCESSING OF /TURN TO BE CONTINUED? REPLY
                            (Y=YES; N=NO)
+TSN2-ASR.142523  %  NBR0825 THIS CONSOLE IS (K3)
!TSN2-ASR.142723  %  NBR0740 COMMAND COMPLETED 'ASR' (RESULT: SC2=00,
                            SC1=00, MC=CMD0001); DATE :<date>
```

16.3.4 Control of message delivery

The messages that can be sent to a console can be distinguished according to various criteria (see also section "Messages").

There are four mechanisms for controlling the output of messages:

- NOINF function
- Message suppression through filtering
- Requesting messages using the message code
- message suppression using the message code

> i None of the four mechanisms has any effect on console logging (CONSLOG function).
>
> With regard to message suppression functions, there is a general principle that messages requiring a response cannot be suppressed.

Command	**Meaning**
ADD-CONSOLE-FILTER	Suppress output of certain message groups or set console to the NOINF status
ASR	Set console to the NOINF status
MODIFY-MSG-SUBSCRIPTION	Order or cancel message subscription
REMOVE-CONSOLE-FILTER	Remove filter levels set with ADD-CONSOLE-FILTER
RESET-MSG-SUPPRESSION	Cancel suppression of message output
SET-MSG-SUPPRESSION	Suppress the output of specific individual messages to a console or an authorized user program
SHOW-CONSOLE-FILTER	Display filter level setting
SHOW-MSG-SUBSCRIPTION	Display message subscription and suppression of messages which were not ordered
SHOW-MSG-SUPPRESSION	Request information on message suppression
OPR statement	**Meaning**
SET-FILTER	Suppress output of certain message groups Set filter levels

Table 43: Interface overview for controlling message output

The recipients of a message are determined as follows:

- If it is a message directed to a specific console, this console is entered in the recipient list of the message.

- If it is a message assigned via a routing code (i.e. undirected), all consoles which have this routing code are entered in the recipient list.
- If it is an undirected message and not a query, and there are consoles which have requested the suppression of all undirected messages not explicitly ordered (with the command MODIFY-MSG-SUBSCRIPTION ..., DELIVER-OTHER-MSG=*NO or ASR NOINF), these consoles are removed from the recipient list.
- If it is an undirected message and there are consoles which have requested the suppression of messages of a relevant weight and routing code (ADD-CONSOLE-FILTER command), these consoles are removed from the recipient list.
- If it is an undirected message with a message code and is neither a question nor any other message marked as “unrequestable”, and if there are message orders (with the MODIFY-MSG-SUBSCRIPTION command) for the relevant message code (or parts of it), the consoles to which these message orders apply are entered in the recipient list.
- If the message has a message code and is not a question, and if there are consoles which have requested the suppression of messages with this message code (SET-MSG-SUPPRESSION command), these consoles are removed from the recipient list.

The suppression of message output using the message code has the highest priority. A console can therefore order a group of messages with the MODIFY-MSG-SUBSCRIPTION command (e.g. all messages with message class NBR) and can exclude individual message codes from this group (e.g. `NBR0740`) with the command SET-MSG-SUPPRESSION.

NOINF function

The ASR command can be used to place a console in the NOINF state. In this state, all messages which do not require a response (identified by % and to be distributed via a routing code) are suppressed.

The same effect can be achieved with the ADD-CONSOLE-FILTER command if all five filter levels are requested for all the routing codes for a console, or if the command MODIFY-MSG-SUBSCRIPTION DELIVER-OTHER-MSG=*NO is entered.

Message suppression through filtering

The ADD-CONSOLE-FILTER command or the SET-FILTER parameter service statement enables the operator to suppress the output of specific message groups by setting filter levels.

These are messages which do not require a response (identified by % and to be distributed via a routing code) and which have a weight code (i.e. they originate from a message file and were produced via the MSG7 or MSG7X macro).

Five filter levels can be specified. These are assigned to the “weight codes” of the messages as follows:

Weight Code	Filter level	Meaning
up to 19	1	Available to customers
20 - 39	2	Pure informational messages
40 - 59	3	Relatively important information messages
60 - 79	4	Important information messages

80 - 99	5	Very important information messages

For example, if filter level 2 is activated, all messages with weight codes from 20 through 39 are suppressed.

The filter level setting can be displayed with the SHOW-CONSOLE-FILTER command. Filter levels set with the ADD-CONSOLE-FILTER command can be removed again with the REMOVE-CONSOLE-FILTER command; filter levels set with the SET-FILTER parameter service instruction cannot be canceled.

Which weight codes are assigned to the individual messages is described in the message files. Message files can be viewed using the MSGMAKER utility routine. Systems support can change the weight code of a message with the aid of MSGMAKER.

Requesting messages using the message code

It is possible for consoles to explicitly request messages using their message codes. Only these ordered messages are then output to the relevant console.

Messages are ordered via the MODIFY-MSG-SUBSCRIPTION command, in which either complete or abbreviated message codes can be specified.
The ADD-MSG-ID operand adds messages to the existing set of message requests, and the REMOVE-MSG-ID operand removes message orders from this set.
Another operand determines whether all messages to the console which have not been explicitly requested are to be suppressed. With DELIVER-OTHER-MSG=*NO, only the messages ordered with ADD-MSG-ID as well as questions and messages directed explicitly to this console are delivered (equivalent to the ASR NOINF function).

Message orders do not influence the delivery of the relevant messages to other consoles.

Commands for ordering messages can be entered on all types of console, i.e. on both physical and virtual consoles ($CONSOLE applications) and in interactive tasks with the OPERATING privilege.
On physical and virtual consoles, the commands influence message output on the console from which they were entered.
In interactive tasks with the OPERATING privilege, the commands affect a system event stream (see also "User task with the OPERATING privilege").

Message suppression using the message code

To suppress the output of specific messages to a console or an authorized user program the operator can use the SET-MSG-SUPPRESSION command. Message output of up to 1000 different message numbers can be suppressed.

The message must be associated with a number, i.e. it must originate from a message file and have been produced via the MSG7 or MSG7X macro. All message types that do not require a response (identified by %, + and !) can be suppressed.

Output of messages with a specified message number can be activated again by means of the RESET-MSG-SUPPRESSION command, and the SHOW-MSG-SUPPRESSION command can be used to list the arrangements in force.
This fine filtering which can be achieved by specifying the message numbers may be performed at all consoles and by all authorized user programs; however, arrangements for other consoles or user programs can only be made from the main console.

The message flow can be reduced at system initialization by specifying corresponding values for the OPR parameter within the BS2000 startup parameter file (see section "Configuration and suppressing the output of messages at consoles (OPR)").

16.3.5 Communication between operators

The operators can exchange messages via consoles.

A message can be sent from a given console:

- to a specific console (using the mnemonic device name)
- to consoles to which a particular functional area has been assigned (with the routing code; see section "Functional areas and their allocation to consoles").

Any number of blanks is permitted between two syntactical units. When sending a message blanks are permitted, for example, between the syntactical units “an” and “-mid” and “%”. See also the relevant examples.

The following formats are described:

- Format for sending operator messages
- Format for receiving operator messages
- Format for sending operator responses
- Format for receiving operator responses

Format for sending operator messages

```
{<rc | an | (mn)}[-mid] {% | ?}text
```

<	Left angle bracket for opening the routing code
rc	Routing code (1 character) The message goes to all assigned consoles and authorized user programs. The code * (asterisk) means That the message *always* goes to the main console too.
on	Authorization name of an authorized user program (4 characters). The message goes to the authorized user program which is linked to this authorization name.
mn	Mnemonic device name of a console (2 characters). The message goes to the designated console.
-	Hyphen (minus sign)
mid	Message job ID (A..Z, 0-9, @, #, $) The last three characters are taken over as mid for the output. If less than three characters are specified, the entry is padded with zeros from the left.
%	Percent character; identifies the message as being purely an information message (no response required)
?	Question mark; identifies the message as a question which requires a response from the recipient
text	Any text

Format for receiving operator messages

```
{% | ?}{an | (mn)}-mid.hhmmss text
```

%	Percent character; the message requires no response.
?	Question mark; the sender requires a response.
on	Authorization name of the authorized user program from which the message originates (4 characters).
mn	Mnemonic device name of the console from which the message originates.
-	Hyphen
mid	Message job ID, default value = 000 (A..Z, 0-9, @, #, $)
.	Period as a separator
hhmmss	Time (6 digits; hh = hour, mm = minute, ss = second)
text	Freely selectable message text

Examples

```
Input of  (K3): <A % THIS IS A MESSAGE WITH NO MID
Output to *)   : %(K3)-000.142423 THIS IS A MESSAGE WITH NO MID

Input of (K4): <*-AKZ% THIS IS A MESSAGE WITH AN MID
Output to **)  : %(K4)-AKZ.142523 THIS IS A MESSAGE WITH AN MID

Input of (K4): <*-123456789 % THIS IS A MESSAGE WITH A LONG MID
Output to **)  : %(K4)-789.142623 THIS IS A MESSAGE WITH A LONG MID

Input of (K4): <*-@#$ % THIS IS A MESSAGE WITH SPECIAL CHARACTERS IN THE MID
Output to **)  : %(K4)-@#$.142623 THIS IS A MESSAGE WITH SPECIAL CHARACTERS
IN THE MJID

Input of (K4): <*-1 % THIS IS A MESSAGE WITH A SHORT MID
Output to **)  : %(K4)-001.142623 THIS IS A MESSAGE WITH A SHORT MID
```

*) The message is issued to all owners of routing code A

**) The message is issued to all owners of routing code *

> For reasons of compatibility, the last three characters of the message ID are used for output. Message IDs consisting of one or two digits are prefixed with leading zeros. Message job IDs may also be more than eight characters long. The first character may also be a numeric character or one of the characters @, $, or #.
> These conventions for message job IDs are not the same as those for command job IDs (see section “Format for entering operator commands” in chapter "Message formats").

Format for sending operator responses

```
dest [-mid].[text]
```

dest	Destination of response/additional command information This is the sender of the message to which a response is to be given.
-	Hyphen
mid	Message job ID (A..Z, 0-9, @, #, $); leading zeros may be omitted (mid must correspond to the message requiring a response)
.	Period indicating response to questions
text	Arbitrary response text. Only certain texts are permissible for responses to system messages. You can obtain help on individual messages using /HELP-MSG-INFORMATION.

Format for receiving operator responses

```
.{rc | (mn)}[-mid].[text]
```

.	Period as the message type identifier for an operator response
rc	Authorization name of the authorized user program delivering the response (4 characters)
mn	Mnemonic device name of the console delivering the response (2 characters).
-	Hyphen
mid	Message job ID (A..Z, 0-9, @, #, $); filled to 3 characters with leading zeros. default value = 000
.	Period as a separator
text	Arbitrary response text.

Example

```
Sender Input: (C2)-1? TO BE OR NOT TO BE ?
Recipient Output: (C1)-001.092312 TO BE OR NOT TO BE ?
               Input: (C1)-1. TO BE - THAT IS THE ANSWER
Sender Output: (C2)-001. TO BE - THAT IS THE ANSWER
```

16.4 Use of multiple consoles

This section describes system operation from physical and virtual consoles. System operation is also possible from user tasks with the OPERATING privilege.

BS2000 offers the capability of using multiple consoles and assigning them areas of functional responsibility as required by Server Unit operation (see section "Main consoles and subsidiary consoles"). Up to 24 physical and 192 virtual consoles can be operated.

Moreover, BS2000 allows data center staff to employ user terminals for system operation during the session. The software product OMNIS (including OMNIS-PROP) can be used for this purpose, or automation programs can be written (see also section "Software products OMNIS and PROP-XT").

A protection mechanism using passwords prevents the misuse of this function by users without the necessary authorization.

Use of this function would be of advantage, for example, if system administration - by way of exception - wished to intervene in the operation of the system, parts of system operation are to be automated or similar jobs for several computers are to be processed by one operator.

Assignment of functional areas to consoles

The functional areas can be allocated to the consoles in such a way that

- each functional area is uniquely assigned to one console; this means a strict division of work according to functional areas, where it is quite possible for certain consoles to have several functional areas assigned to them
- individual functional areas can be handled on more than one console; this means an indistinct division of work according to functional areas, and should be employed in special cases only.

An overview of the functional areas is contained in section "Functional areas and their allocation to consoles".

Routing codes are assigned (allocated to functional areas) as follows:

- for virtual consoles with generated authorization names via the parameter service or the ASR command
- for virtual consoles with dynamic authorization names via operator roles
- for physical consoles via one of the above two methods, depending on the system parameter NBCONOPI

Effects of the assignments in the session

The assignments of functional areas to consoles manifest themselves during operation in the fact that messages pertaining to a particular functional area are displayed only on the assigned consoles. Conversely, commands pertaining to this functional area can be entered on the assigned consoles.
If the routing code required in order to use a particular command is not assigned to any console, the command involved can nevertheless be entered from the console with the “Main console” function, provided NBCONOPI=N is set. If NBCONOPI=Y is set, even the main console is only allowed those commands for which it has explicit authorization. To prevent a command from being entered from any terminal in the system, it must be protected by the routing code $ in the parameter service.

If virtual consoles (authorized user programs) are used, then the above statements apply to physical and virtual consoles.

Communication between consoles

The operators are able to communicate with each other by exchanging messages via the consoles.

From a given console, a message can be sent to the following destinations:

- another console (with the mnemonic device name)
- to consoles to which a particular functional area has been assigned (with the routing code; see section "Functional areas and their allocation to consoles").

The formats of the messages are described in section "Message formats".

Command	Meaning
REQUEST-MAIN-CONSOLE-FUNCTION	Request “Main console” attribute
SHOW-CONSOLE-STATUS	Display configuration and status of consoles

Table 44: Overview of commands for using multiple consoles

16.4.1 Main consoles and subsidiary consoles

A distinction is made between the main console and – if available – subsidiary consoles.

The main console is initially always the one from which the system is loaded or the one which is recognized as the first operable console.

The various console windows which are opened on the SE Manager are only different views of the same console (which can be recognized from the 2-character console mnemonic in the status line).

Main and subterminals do not differ technically. BS2000 differentiates them only according to their function.

Main console

This function is assigned to just one console at all times. If it fails, the function is transferred automatically to one of the remaining available consoles by the system. The operator can also transfer the function to another specific console using the REQUEST-MAIN-CONSOLE-FUNCTIONS command. The console which is currently the main console has the following properties:

- It receives the routing code * – even when it was not explicitly assigned to it.
- It receives all “emergency messages”.
- All queries and requests for additional command information, including emergency queries, can be answered from it (exception: queries connected with SCI commands and “;” queries).

With NBCONOPI=N

- It is authorized to execute all commands whose routing code is not assigned to a console.
- Subsidiary consoles can be assigned and other consoles can be added or removed by command from this console.
- The assignment of functional areas to other consoles can be changed from this console at any time.
- It normally receives all the messages which cannot be assigned to any other console.

Additional with NBCONOPI=Y

- The assignment of functional areas to virtual consoles with generated authorization names can be modified with the ASR command from this console.
- It receives all queries which could not be output at their desired output location.

As with all consoles, the capabilities of the main console depend on which functional areas are assigned to it.

> Up to the end of the loading process, the main console function is fulfilled by the device from which the system was loaded. The main console function can only be assigned to a physical console, not to a virtual console (an authorized user program). Only the right to respond to open queries can be associated with the routing code * (asterisk) by means of the system parameter NBREPLY.

Subsidiary console

All consoles which do not have the main console function are regarded as subsidiary consoles by the system. The capabilities of a subsidiary console depend on the functional areas that are assigned to it.

Switching the main console function

The system switches over the main console function in the following situations:

- When requested by an operator.
  The switchover must be requested from the console which is to receive the main console function (with the REQUEST-MAIN-CONSOLE-FUNCTIONS command). The system then calls for a confirmation from the current main console. If this is received, the switchover is carried out. Otherwise, the system rejects the request.
- Upon failure of the current main console.
  In this case the system switches over automatically to another console; if NBCONOPI=N is specified, the standby console predefined in the parameter service is selected (see section "DEFINE-CONSOLE statement"). This selection can be influenced by the operator by means of the CONSOLE command. If NBCONOPI=Y is specified, the new main console is selected via an internal algorithm.

16.4.2 Standby consoles

A standby console can be defined for each console if NBCONOPI=N is specified. Upon failure of a console, the standby console assigned to it is activated automatically. If standby consoles are not defined in the parameter service, they are selected by the system by means of an internal algorithm.
The operator can change the assignments or switch over to standby consoles (see the CONSOLE command).

If NBCONOPI=Y is specified, the “Standby console” function is not available.

16.4.3 Functional areas and their allocation to consoles

Functional areas in BS2000 operation

The following table provides an overview of the functional areas in BS2000 operation. The table also includes the commands (in alphabetical order) which belong to the individual functional areas. This is a standard allocation of commands, which can be changed at system initialization (separately for each command; see section "SET-CMD-CODE statement").

For some commands, the default assignment to a functional area and therefore also the assignment of an authorization code depends on the system parameter NBCONOPI. Some commands which are protected by default with the authorization code $ with NBCONOPI=N, receive a different default authorization code if the operator logon is activated (NBCONOPI=Y). These are indicated in the table.

Every functional area is provided with an authorization code (AC), also known as a routing code (RC), which must be specified in each case when the assignment is made. The meaning of this authorization code is described on "Notes on the individual functional areas".
Information and the assignment of functional areas to consoles can be found under "Assignment of functional areas to consoles".

An assignment of “operator commands to routing codes” is provided in the “Commands” manual [27].

> ISP commands may be entered for compatibility reasons. A detailed description of these commands is only to be found only in the “Operator Commands (ISP format)” manual [35].

As a rule, the following table applies only in conjunction with NBCONOPI=N. Exceptions are indicated separately.

AC/RC	Task area	Standard for command
A	System administration	CHANGE-SERSLOG-FILE, SHOW-SERSLOG-STATUS, START-/STOP-SERSLOG
C	Operation of the data communication system	All BCAM commands, e.g.: DCSTART, BCEND
	File Transfer operation and control	openFT commands, see manual “openFT” [23], e.g. ADD-FT-PARTNER, START-/STOP-FT
D	Operation of disks	CHANGE-DISK-MOUNT, CHECK-DISK-MOUNT, DRV commands, see the “DRV” manual [17], e.g. SHOW-DRV-STATUS, START-/STOP-DRV-DUAL-MODE, SET-DISK-DEFAULTS, SET-DISK-PARAMETER, UNLOCK-DISK

E	General tasks and responsibilities	CANCEL-RUN-PROCESS, INFORM-ALL-JOBS, INFORM-JOB,
	With **NBCONOPI=Y**, some commands which otherwise have the AC $ are given the new AC E.	MODIFY-MSG-FILE-ASSIGNMENT, MODIFY-MSG-SUBSCRIPTION, PROTECT-FITC-APPLICATION, REQUEST-MAIN-CONSOLE-FUNCTIONS, SHOW-CJC-STATUS, SHOW-DEVICE-CONFIGURATION, SHOW-DEVICE-DEPOT, SHOW-DEVICE-STATUS, SHOW-DISK-DEFAULTS, SHOW-DISK-STATUS, SHOW-JOB-STATUS, SHOW-MASTER-CATALOG-ENTRY, SHOW-MOUNT-PARAMETER, SHOW-PUBSET-ATTRIBUTES, SHOW-RESOURCE-ALLOCATION, SHOW-RESOURCE-REQUESTS, SHOW-SHARED-PUBSET, SHOW-SYSTEM-STATUS, SHOW-TAPE-STATUS, SHOW-USER-STATUS SHOW-XCS-PUBSET, START-/STOP-DIALOG-APPLICATION
W	Device management	ADD-IO-UNIT, ATTACH-DEVICE, DETACH-DEVICE, INCLUDE-DEVICE-CONNECTION, MODIFY-IO-UNIT, MODIFY-MOUNT-PARAMETER, MOUNT-NET-STORAGE, REMOVE-DEVICE-CONNECTION, REMOVE-IO-UNIT, SHOW-NET-STORAGE, START-/STOP-CONFIGURATION-UPDATE, UMOUNT-NET-STORAGE, UNLOCK-DEVICE
H	Hardware maintenance	
J	Job control	HOLD-JOB, HOLD-JOB-CLASS, HOLD-JOB-STREAM, HOLD-TASK,

	With **NBCONOPI=Y**, some commands which otherwise have the AC $ are given the new AC J.	MODIFY-JOB, MODIFY-JOB-CLASS, MODIFY-JOB-STREAM, MODIFY-JV, MODIFY-RESOURCE-COLLECTION, RESUME-JOB, RESUME-JOB-CLASS, RESUME-JOB-STREAM, RESUME-TASK, SHOW-JV, START-/STOP-JOB-STREAM, START-/STOP-RESOURCE-COLLECTION
K	OPR command management	CONNECT-CMD-SERVER
N	Monitoring of remote SPOOL	REDIRECT-PRINT-JOB
O	Operation of printers	HOLD-PRINT-JOB (alias HOLD-SPOOLOUT), RESUME-PRINT-JOB (alias RESUME-SPOOLOUT)
P	Control of tasks	CANCEL-JOB, CHANGE-TASK-CPU-LIMIT, CHANGE-TASK-PRIORITY, ENTER-JOB, FORCE-JOB-CANCEL, INFORM-PROGRAM, SEND-MSG

R	Monitoring and control of system behavior	ACTIVATE-SNAPSHOT, CANCEL-PUBSET-EXPORT/-IMPORT, CHANGE-CONSLOG-FILE, CREATE-/DELETE-PAGING-FILE, DEACTIVATE-SNAPSHOT, EXPORT-PUBSET, EXTEND-PAGING-AREA, FORCE-PUBSET-EXPORT, HOLD-SUBSYSTEM, IMPORT-PUBSET, MODIFY-MEMORY-PARAMETERS, MODIFY-MSCF-CONNECTION, MODIFY-MSCF-ENVIRONMENT, MODIFY-NSM-ENVIRONMENT, MODIFY-PAGING-AREA-ATTRIBUTES, MODIFY-PCS-OPTION, MODIFY-SYSTEM-BIAS, MODIFY-TASK-CATEGORIES, REDUCE-PAGING-AREA, RELEASE-CLUSTER-RECOVERY-LOCK, RESUME-SUBSYSTEM, SET-DSSM-OPTIONS, SET-MSCF-ENVIRONMENT, SET-RESTART-OPTIONS, SET-XCS-PUBSET, SHOW-MEMORY-CONFIGURATION, SHOW-PAGING-CONFIGURATION, SHOW-PCS-OPTION, SHOW-RESTART-OPTIONS, SHOW-SNAPSHOT-STATUS, SHOW-SUBSYSTEM-STATUS, SHOW-TRACE-STATUS, SHUTDOWN, START-/STOP-MSCF-CONNECTION, START-/STOP-SUBSYSTEM, START-/STOP-TRACE
S	Control of spoolout operation Loading SPOOL	MODIFY-PRINTER-OUTPUT-STATUS, MODIFY-TAPE-OUTPUT-STATUS, SHOW-ACTIVE-SPOOL-DEVICES, SHOW-PRINT-JOB-STATUS, START-/STOP-PRINTER-OUTPUT, START-/STOP-TAPE-OUTPUT, START-/STOP-TAPE-REPLAY

T	Operation of magnetic tape devices	ADD-DEVICE-DEPOT, CHANGE-TAPE-MOUNT, CHECK-TAPE-MOUNT, REMOVE-DEVICE-DEPOT, UNLOAD-TAPE
U	File management	
W,X,Y, Z	Freely selectable	
B,F,I,L, M, Q,V, 0.. 8	Reserved	
9	VM2000 administration	All VM2000 commands
#	POSIX	

@	Special meaning (no authorization required)	ADD-CONSOLE-FILTER, DISCONNECT-CMD-SERVER, GRANT-PROP-CONNECTION, HELP-MSG-INFORMATION, MODIFY-CONSOLE-OPTIONS, MODIFY-USER-PROTECTION, RELEASE-OPERATOR-ROLE, REMOVE-CONSOLE-FILTER, REQUEST-OPERATOR-ROLE, RESET-MSG-BUFFER, RESET-MSG-SUPPRESSION, SET-MSG-SUPPRESSION, SHOW-CMD-ATTRIBUTES, SHOW-CONSLOG-ATTRIBUTES, SHOW-CONSOLE-FILTER, SHOW-CONSOLE-OPTIONS, SHOW-CONSOLE-STATUS, SHOW-FILE-ATTRIBUTES, SHOW-IOCF, SHOW-MSCF-CONFIGURATION, SHOW-MSG-SUBSCRIPTION, SHOW-MSG-SUPPRESSION, SHOW-NSM-CONFIGURATION, SHOW-OPERATOR-ATTRIBUTES, SHOW-OPERATOR-ROLE, SHOW-PENDING-MSG, SHOW-PUBSET-CACHE-ATTRIBUTES, SHOW-PUBSET-DEFINITION-FILE, SHOW-PUBSET-IMPORT-EXPORT, SHOW-PUBSET-OCCUPATION SHOW-PUBSET-PROCESSING, SHOW-PUBSET-RESTRICTION, SHOW-PUBSET-SPACE-ALLOCATION, SHOW-PUBSET-SPACE-DEFAULTS, SHOW-SDF-PARAMETERS, SHOW-SPACE-SATURATION-LEVELS, SHOW-SYSTEM-INFORMATION, SHOW-SYSTEM-PARAMETERS, SHOW-XCS-OCCUPATION
	Additional with **NBCONOPI=Y**	EXIT-JOB, SET-LOGON-PARAMETERS
*	Main console	CONSOLE (in conjunction with NBCONOPI=Y, this command is no longer allowed) MODIFY-SDF-PARAMETERS

$	Special meaning (command subject to general lock)	ADD-MASTER-CATALOG-ENTRY, ADD-USER, LOCK-USER, MODIFY-MASTER-CATALOG-ENTRY, MODIFY-SPACE-SATURATION-LEVELS, MODIFY-USER-ATTRIBUTES, MODIFY-USER-PUBSET-ATTRIBUTES, PRINT-DOCUMENT (PRINT-FILE), REMOVE-MASTER-CATALOG-ENTRY, REMOVE-USER, SHOW-FILE-TRANSFER, SHOW-PRINTER-POOLS, SHOW-SPOOL-CHARACTER-SETS SHOW-SPOOL-DEVICES SHOW-SPOOL-FORMS SHOW-SPOOL-PARAMETERS SHOW-USER-ATTRIBUTES, UNLOCK-USER WRITE-SPOOL-TAPE
	In conjunction with **NBCONOPI=Y**, the AC changes from $ to E	ADD-/SET-FILE-LINK, ADD-PASSWORD, COPY-FILE, CREATE-FILE, CREATE-FILE-GENERATION, CREATE-FILE-GROUP, CREATE-TAPE-SET, DELETE-FILE, DELETE-FILE-GENERATION, DELETE-FILE-GROUP, DELETE-SYSTEM-FILE, DELETE-TAPE-SET, EXPORT-FILE, EXTEND-TAPE-SET, IMPORT-FILE, MODIFY-FILE-ATTRIBUTES, MODIFY-FILE-GENERATION-SUPPORT, MODIFY-FILE-GROUP-ATTRIBUTES, REMOVE-PASSWORD
	In conjunction with **NBCONOPI=Y**, the AC changes from $ to J	CREATE-JV, DELETE-JV, MODIFY-JV-ATTRIBUTES, REMOVE-JV-LINK, SET-JV-LINK, SHOW-JV-ATTRIBUTES, SHOW-JV-LINK

Table 45: authorization code (AC/RC) - functional areas - commands

Notes on the individual functional areas

A System administration

Comprises activation/deactivation of the software error logging function and changing the SERSLOG file.

C Operation of the data communication system

Includes the following activities, for example:

- starting the Data Communication System
- defining and activating communication partners
- activating lines
- defining and activating routes to communication partners

Every assigned console also receives messages which indicate errors in the Data Communication System.

D Operation of disks

Includes the mounting of volumes (disks). Every assigned console also receives messages which indicate inconsistencies in the volume serial numbers, or other errors.

E General tasks and responsibilitiesIncludes no precisely defined activities; the associated commands should rather be available to every console.

W Device management

Includes the following activities, for example:

- detaching a device from the system
- attaching a device to the system
- dynamic I/O configuration change
- connecting and disconnecting Net-Storage

Every assigned console also receives messages which indicate inconsistencies.

H Hardware maintenance

Every assigned console receives messages which indicate errors in the server and which are evaluated by the maintenance personnel.

J Job control

Every assigned console receives messages which indicate the start of or any particularities or errors in the processing of jobs, so that the operator can monitor job execution.

K OPR command management

Authorized user programs can define operator commands and delete the commands they have defined. They can also assume and relinquish responsibility for processing existing operator commands.

N Monitoring of remote SPOOL

O Operation of printers

P Control of tasks

Includes the following activities, for example:

- determining priorities for individual jobs
- starting batch jobs
- terminating user jobs

R Monitoring and control of system behavior

Includes the following activities, for example:

- changing the category-specific specifications for the number of jobs and I/O priority
- ending the session

Every assigned console also receives messages which indicate system errors or server faults or which indicate that users have requested more space on public volumes than they are permitted.

S Control of spoolout operation

Includes the following activities, for example: assigning output devices to spoolout jobs.

Every assigned console also receives messages which indicate inconsistencies.

T Operation of magnetic tape devices

Includes the mounting/changing of volumes. Every assigned console also receives

messages which indicate errors.

U File management

W,X,Y,Z

Freely selectable. These functional areas are freely available to the user for specific purposes (e.g. for special commands).

B,F,I,L,M,Q,V,1,2,3,4,5,6,7,8

Reserved for future system expansion.

These functional areas have not yet been defined but may be defined in future versions.

9 Reserved for VM2000 management

POSIX

Authorization codes with special meanings

0 Messages with this authorization code cannot be requested. Messages without a response are simply stored in the CONSLOG file. Messages with a response (queries) are delivered to the main console.

* Main console
This functional area is also always implicitly assigned to the current main console. Whether a command with

this routing code was actually entered at the main console is only checked in the case of the CONSOLE command.

In addition, every assigned console receives messages which provide information on the status of accounting files.

@ Messages with this authorization code are requestable (MODIFY-MSG-SUBSCRIPTION command). Messages without a response are simply stored in the CONSLOG file. Messages with a response (queries) are delivered to the main console. Commands with this authorization code are unprotected; any operator is authorized to enter them.
So in practice there is no sense in assigning code @ to consoles. It does, however, make sense to assign this code to messages or commands.

$ No user is allowed to use a command protected by code $. Although authorization code $ may be assigned to any console or authorized user program, it does not authorize them to issue a command protected by $. Code $ has no special meaning as a message distribution code.

Assignment of functional areas to consoles

The following methods are available for assigning functional areas to consoles:

NBCONOPI=N

- At system startup

  Functional areas can be assigned to physical or virtual consoles (but not to authorized user programs) via the STARTUP parameter file during system startup. This is done using the SET-CODE statement (SET-CODE CODE=rc,CONSOLE=mn; see "SET-CODE statement").

- During the session

  The ASR command can be used on the relevant main console to create any desired assignment for physical consoles and for virtual consoles with generated authorization names.
  Depending on the system parameters ASRSW1 and ASRSW2 the ASR command can also be used from another physical console or from a virtual console with a generated authorization name (so long as the ASR command is permitted there) to set the assignment
  ASR commands to change the allocation of functional areas are rejected if they are entered by authorized user programs with dynamic authorization names or in reference to these. Authorized user programs with dynamic authorization names assign themselves to their functional areas with the REQUEST-OPERATOR-ROLE command.

  The “Main console” function can only be switched over if the previous main console agrees.

  If the current main console fails, the system automatically switches the “main console” function to another console, selecting the standby console predefined in the parameter service (see section "DEFINE-CONSOLE statement"). This selection can be influenced by the operator by means of the CONSOLE command.

NBCONOPI=Y

The routing codes assigned to the IPL console in the startup parameter file by means of the SET-CODE statement are called “implicit routing codes”. Their effect is controlled by the system parameter NBIMPRCA. When NBIMPRCA=Y is specified, they are effective throughout the whole session, but when NBIMPRCA=N is specified, they only apply until “System ready” and after system termination (successful shutdown).

Implicit routing codes basically work only like routing codes (for distributing messages) and not like authorization codes (for input of commands).

- At system startup and after system termination

  The main console receives all messages of the routing codes which were defined as implicit routing codes.

- During the session (until shutdown)

  When NBIMPRCA=Y is specified, the main console receives all messages of the implicit routing codes, provided these are not explicitly assigned to any other console.

  A console can be:

  - operable / inoperable
  - active / inactive
  - authorized / unauthorized

  A console is inoperable when it is managed system-internally as “defective”. This also applies to non-existent devices or devices in the DETACHED state. All other consoles are operable.
  An operable console is activated by the SET-LOGON-PARAMETERS command and deactivated by the EXIT-JOB command.
  An active console receives various authorization codes through the REQUEST-OPERATOR-ROLE command, which represent its authorization scope. Active consoles are always authorized for the input of @ commands.
  Inactive consoles only have the following commands at their disposal:
  SET-LOGON-PARAMETERS and SHOW-PENDING-MSG.

  The first console on which the SET-LOGON-PARAMETERS command was successfully entered receives the “Main console” attribute. It is now active but is only authorized for the input of @ commands.

  The “Main console” function can only be switched over if the previous main console agrees.

  If the current main console fails, the system automatically switches the “Main console” function over to a different console; it searches first in the set of active consoles and then in the set of operable consoles. An EXIT-JOB command from the main console also initiates this search.

  With ASR authorization, the main console can create any required assignment of routing codes to user programs with generated authorization names.

Notes

Assigning a console to the functional area “@” has no real effect.

It is also possible to have a routing/authorization code (e.g. A, C, R, U) that is not assigned to a console. When routing codes are not assigned to a console, the following occurs:

- Messages requiring no response are not output at any console. They are only written to the logging file (this is always the case with the @ code).
- Messages requiring a response from the operator are output at the main console. Even messages routed via the @ code are output at the main console if they require an operator response.
- With NBCONOPI=N, commands bearing authorization codes of this type that are not assigned to any console are accepted by the system if issued from the main console.
  Exception: commands with the code @ are always accepted.

16.5 Return messages

- Command termination messages
- Return messages on physical consoles
- Return messages on physical and virtual consoles

16.5.1 Command termination messages

Command termination messages are identified by the message type “!” as the first character of the output. The following command termination messages may be issued:

`NBR0740` `COMMAND COMPLETED 'ASTOP' ; RESULT: (SC2=00, SC1=00, MC=CMD0001); DATE:<date>`
appears if no errors occurred during command execution.

`NBR0741` `COMMAND SUCCESS UNCERTAIN 'SHOW-CONSLOG'; RESULT: (SC2=02, SC1=00, MC=NBR0741)`
appears if an error resulting in termination of the operator task occurred during command execution.

`NBR0741` `COMMAND SUCCESS UNCERTAIN 'SPECIAL'; RESULT: (SC2=02, SC1=64, MC=NBR0741)`
appears if the connection to the DCAM application $CONSOLE was released during command execution in an authorized user program.

`NBR0742` `COMMAND NOT ALLOWED 'ASR'; RESULT: (SC2=00, SC1=64, MC=CMD0216)`
appears if the console from which the command was issued has no authorization for this command.

`NBR0743` `COMMAND NOT AVAILABLE 'SPECIAL'; RESULT: (SC2=00, SC1=130, MC=CMD0200)`
appears if the routine processing the command is not in the system or if there is no connection between the DCAM application $CONSOLE and the authorized user program which is supposed to execute the command.
Example: `/.FSTAT DADM xxx`
If TDADM is not loaded, this command is rejected with message `NBR0743`.

`NBR0744` `COMMAND NOT FOUND 'ADR'; RESULT: (SC2=00, SC1=64, MC=NBR0744)`
appears if the command entered is not an operator command. Example:
`/.A1234567 ADR HELP`

`NBR0745` `COMMAND ID INVALID '.1WRONGID'; RESULT: (SC2=00, SC1=01, MC=NBR0745)`
appears if a command job ID contains an invalid character. Example:
`/.1WRONGID ASR HELP`

`NBR0746` `COMMAND CANCELLED 'RUN'; RESULT: (SC2=02, SC1=00, MC=NBR0746)`
appears if a RUN command was canceled with the CANCEL-RUN-PROCESS command or if the connection to the console was cleared down and, as a result, the associated operator task was terminated before it could process the command.

`NBR0747` `COMMAND ID AMBIGUOUS 'ABC'; RESULT: (SC2=00, SC1=64, MC=NBR0747)`
appears if the command job ID with which the command was entered is not unique. The command job ID must only be unique in the case of commands which can request additional command information and which are processed in an authorized user program.

`NBR0748` `SYSTEM SHUTDOWN IN PROGRESS. COMMAND IGNORED 'ASR'; RESULT: (SC2=02, SC1=00, MC=CMD0198)`
appears if commands are issued during system termination (with the exception of SHOW-PENDING-MSG).

`NBR0749 COMMAND FROM /RUN FILE CANCELLED: 'RUN'; RESULT: (SC2=02, SC1=00, MC=NBR0749)`

appears if a command has been read from the RUN file but cannot be executed due to a previous CANCEL-RUN-PROCESS command.

16.5.2 Return messages on physical consoles

`NBC0050 CONSOLE ERROR - TRY AGAIN`
appears if an error occurred during reading.

`NBC0051 TIMEOUT - INPUT IGNORED`
appears if the previous input operation was aborted by the software because of an error.

`NBC0052 INPUT NULL - IGNORED`
appears when the operator has entered only `END OF TEXT`.

`NBC0053 RECIPIENT NOT AVAILABLE - INPUT IGNORED`
appears if an entry cannot be forwarded to the recipient because the recipient is not available.

`NBC0054 UCON NOT ACTIVE - INPUT IGNORED`
appears if entries not beginning with `P.` were made between the time when IPL was terminated and UCON was started.

`NBC0055 INVALID PREFIX ENTERED - NO SUCH ANSWER EXPECTED`
appears if an entry beginning with `P.` or `E.` was made although no query with one of these prefixes is open.

`NBC0058 UCON BUSY - INPUT IGNORED`
appears if an entry could not be passed as a result of UCON problems.

`NBC0059 NO PF1-, PF2- OR ENTER-KEY USED - INPUT IGNORED`
appears if an entry was not terminated with the PF1, PF2 or ENTER key.

`NBC0070 CIO TESTS THIS CONSOLE`
appears after the system has recovered from a situation in which no operable console was available. If all consoles are defective, they are tested cyclically with this message. The first operational console found outputs this message.

16.5.3 Return messages on physical and virtual consoles

`REJ1` `TEXT NOT RECOGNIZED`
appears under the following conditions:

- if no recipient is specified for the message,
- if the recipient is specified with more than 4 characters,
- if the recipient is not followed by one of the characters “/ % ? . : -” and no blank, or if the job ID specified after the recipient is prefixed with “-” but none of the characters “/ % ? . :” is contained in the string following the hyphen,
- if only a “/” or a “/” and a command job ID (cid) are specified –

`REJ5` `DESTINATION NOT FOUND`
appears if a message contains an invalid mnemonic device name for a physical console, or an invalid authorization name.

`REJ6` `NO MATCH FOR REPLY`
appears if the appropriate query for the response sent cannot be found.
Rejection also occurs if a period is entered in response to a request for additional command information or a colon is entered in response to a query.
Neither physical nor virtual consoles can respond to messages of the message type “;” (they are withdrawn by the system when an expected event occurs).

`REJ7` `USER NOT ALLOWED TO REPLY`
appears if an unauthorized user attempts to give a response to a question.

`REJ8` `CONSOLE OFF - INPUT IGNORED`
appears if the console is in the software state “SWITCHED OFF” (see the CONSOLE command in the “Commands” manual [27]).

`REJ9` `CONSOLE mn INOPERABLE - INPUT IGNORED`
appears when the console is marked as defective, i.e. when the hardcopy device is faulty and the operand FORCED-HARDCOPY=YES is set (see the MODIFY-CONSOLE-OPTIONS command in the “Commands” manual [27]).

`REJ10` `QUESTION IDENTIFICATION ALREADY EXISTING`
appears if a query is entered at a console and its ID has already been used for another query entered at the same console and for which no response has yet been received.

`REJ11` `MESSAGE CANNOT BE MATCHED TO AN ORDER`
appears if an authorized user program, in its capacity as command processor, refers to an incorrect command environment.

`REJ12` `NBMHE/NBMAP ERROR IN PROTOCOL`
appears if a $CONSOLE application connected via V02 sends a message with a syntactically incorrect NBMHE or NBMAP.

> In the case of messages `REJ1` and `REJ5` through `REJ7` the unidentified text or the rejected message /query/response is also specified.

The following messages have been replaced by command termination messages:

`REJ2` `COMMAND NOT ALLOWED`
replaced by the command termination message `NBR0742.`

`NBR0742` `COMMAND NOT ALLOWED '...'; (RESULT: SC2=00, SC1=64, MC=CMD0216)`
appears if the console from which the command was issued is not authorized to issue that command.

`REJ3` `COMMAND NOT AVAILABLE`
replaced by the command termination message `NBR0743.`

`NBR0743` `COMMAND NOT AVAILABLE '...'; (RESULT: SC2=00, SC1=130, MC=CMD0200)`
appears if the routine processing the command is not in the system or if there is no connection between the DCAM application $CONSOLE and the authorized user program which is supposed to execute the command.

`REJ4` `COMMAND NOT FOUND`
replaced by the command termination message `NBR0744.`

`NBR0744` `COMMAND NOT FOUND '...'; (RESULT: SC2=00, SC1=64, MC=NBR0744)`
appears if the command entered is not an operator command.

16.6 Replacement of the STATUS MSG and ASR commands

The STATUS MSG and ASR commands have already been replaced in BS2000/OSD-BC V3.0 by the following commands. They continue to be supported, however, for reasons of compatibility.
The ASR command is now only available to the operator with a limited functional scope and is no longer provided automatically on the main console. It can now assign authorization codess (routing codes) only to applications with a generated authorization name.

Command STATUS	New command	BS
MSG	SHOW-PENDING-MSG	@
MSG, ALL	SHOW-PENDING-MSG DESTINATION=*ANY	@

Command ASR	New command	BS
ADD, CODE=...	REQUEST-OPERATOR-ROLE	@
ADD, CODE=..., FILTER=....	ADD-CONSOLE-FILTER FILTER=...,ROUTING-CODE=...	@
ADD, CODE=..., FILTER=ALL	ADD-CONSOLE-FILTER FILTER=(1,2,3,4),- ROUTING-CODE=...	@
ADD, CONSOLE=...	No replacement command	--
DELETE, CODE=...	RELEASE-OPERATOR-ROLE	@
DELETE, CODE=..., FILTER=...	REMOVE-CONSOLE-FILTER FILTER=...,- ROUTING-CODE=...	@
DELETE, CODE=..., FILTER=ALL	REMOVE-CONSOLE-FILTER FILTER=(1,2,3,4),- ROUTING-CODE=...	@
DELETE, CONSOLE=...	No replacement command	--
DESTINATION	SHOW-SYSTEM-PARAMETERS PARAM=MSGDEST	R
HELP	SHOW-CONSOLE-STATUS CONSOLE=*OWN	@
HELP, CODE=...	SHOW-CONSOLE-STATUS CONSOLE=*ALL [1]	@

HELP, CODE=..., FILTER=...	SHOW-CONSOLE-FILTER SELECT=*FILTER(...) or SHOW-CONSOLE-FILTER SELECT=*ROUTING-CODE(...)	@
HELP, COMMAND=....	SHOW-CMD-ATTRIBUTES	@
HELP, CONSOLE=...	SHOW-CONSOLE-STATUS CONSOLE=...	@
HELP, CONSOLE=ALL	SHOW-CONSOLE-STATUS CONSOLE=*ALL	@
HELP, CONSOLE=..., FILTER=...	SHOW-CONSOLE-FILTER SELECT=*FILTER(...) [2]	@
INF	REMOVE-CONSOLE-FILTER FILTER=*ALL,- ROUTING-CODE=*ALL or MODIFY-MSG-SUBSCRIPTION DELIVER-OTHER-MSG=*YES	@ E
MAIN	REQUEST-MAIN-CONSOLE-FUNCTIONS	E
NOINF	ADD-CONSOLE-FILTER FILTER=ALL,- ROUTING-CODE=*ALL or MODIFY-MSG-SUBSCRIPTION DELIVER-OTHER-MSG=- *NO	@ E
PRIMARY	No replacement command	--
SUPPRESS	RESET-MSG-BUFFER SENDER=...	@

[1] Not directly mapped but the information is contained in the specified command

[2] Only in relation to the console via which the entry is made

17 Automation of operator functions

- Authorized user programs with operator functions
  - Connections with generated authorization names
  - Connections with dynamic authorization names
  - Exchange of messages
  - Message formats
- Special operator commands in authorized user programs
  - Command definition
  - Message formats
- Software products OMNIS and PROP-XT
- Command files for the operator
  - Executing and aborting a command file
  - Structure of command files
- System administration functions performed by the operator

17.1 Authorized user programs with operator functions

With the $CONSOLE interface, BS2000 offers the option of freeing the operator from functions that can be executed under program control, such as:

- regular interrogation of certain system load parameters (e.g. number of jobs by different criteria, main memory utilization); and initiation of appropriate measures if required
- regular checks for the existence of unanswered messages; broadcasting of messages to the consoles
- Device management

Such activities can be performed by user programs which, as BCAM-DCAM applications, have identified themselves to the operating system as authorized $CONSOLE applications and which function as virtual consoles. These programs, which function as consoles, are generally referred to as **authorized user programs**. They are authorized to execute operator functions according to their authorization profile (routing code set).

An unauthorized user program can only send messages to the system and receive any replies. These are BCAM /DCAM applications which, although they connect to $CONSOLE, have not requested special (authorized) connection setup.

Authorized programs can execute all conventional operator activities.
In addition, an authorized program can occur as a command execution instance for special operator commands (defined with parameter statement ADD-CMD-ENTRY or the CONNECT-CMD-SERVER command); see "Special operator commands in authorized user programs".

When logging on your own operator command with CONNECT-CMD-SERVER, you can specify COMPLETION-CONTROL=*YES. Note, however, that - unlike with consoles - command input in a user task is synchronous.

The input is therefore only possible when the previous command has finished executing. Consequently, the command should either have a short processing time or be logged on with COMPLETION-CONTROL=*NO.

> The manufacturer offers products based on this interface with their own user interfaces, e.g. OMNIS and PROP-XT (see "Software products OMNIS and PROP-XT"), MAREN (see “MAREN” [31] and ROBAR (see “ROBAR” [41]).
>
> In the operating mode NBCONOPI=Y, the operator task for an authorized user program with a dynamic authorization name has the privileges of its user and the OPERATING privilege.

System operation can be split into functional areas for authorized user programs in the same way as for physical consoles.
Every user program which wishes to exercise operator functions must make a “connection” to the central system operation task (under the application name $CONSOLE). If the operator functions to be exercised require an authorization (authorization/routing code), the user program must also prove its right to make this connection.

The link between a user program and an **authorization name** gives this program a feature which is comparable with the mnemonic device name of a physical console.

For authorized user programs which connect to the system using this authorization name (generated authorization name), an entry in the authorization name table must be created in the OPR parameter service with SET-CODE or ADD-CMD-ENTRY. An authorization name consists of 4 alphanumeric characters (values: A-Z, 0-9, or @, $, #), where the first character must not be a digit or “#”. “@” should not be used as the first character. The system always

creates 512 authorization names. A maximum of 384 authorization names (generated authorization names) can be specified in the parameter service. The remaining authorization names are assigned by the system (dynamic authorization names, @001 to @512, unless already assigned).

Authorized user programs can thus be assigned names that have the same lifetime as the mnemonic device names of physical consoles.

Authorization names have the following characteristics:

- The lifetime of authorization names equals the lifetime of the system and hence the lifetime of the device definitions for the physical consoles.
- As soon as a user program has been linked to an authorization name it is regarded as an authorized user program. This linkage must be initiated by the program and remains valid until it is canceled by the user program, until the program is terminated or until UCON severs the connection.
- The assignment of functional areas for authorized user programs can be made either via authorization names or via operator roles, depending on the type of connection.
- After a connection has been accepted by the system, an authorized user program can be identified as the sender or recipient of messages only by the authorization name.

A command processing function may be assigned to an authorization name. If an authorized user program is to execute commands, the system must be informed of the commands for which it functions as the command processor (command server for special commands). For authorized applications with authorization names which were generated, the command name/authorization name, assignment is made in the OPR parameter service with the ADD-CMD-ENTRY record.
In addition, the CONNECT-CMD-SERVER command is available to **all** authorized user programs for linking authorized applications with operator commands. The DISCONNECT-CMD-SERVER command can be used to remove the link (see the “Commands” manual [27]). By default, both commands are protected by the authorization code K and are only permitted for authorized user programs.

Systems support creates an entry in the user catalog for each authorized user program which is permitted to make a connection to the system. In doing so, passwords should be specified to protect against the unauthorized establishment of a connection to the $CONSOLE (UCON) application. Entries are made in the user catalog by an ADD-USER command; if an application has a generated authorization name, this is used as the user ID, if it has a dynamic authorization name, the OPERID is used as the user ID.

Example of an entry in the user catalog

```
/ADD-USER USER-ID=RUDI,PROTECTION-ATTRIBUTES=(-
LOGON-PASSWORD=C'FOX#HOLE'),ACCOUNT-ATTRIBUTES=(ACCOUNT=K0815)
```

If the user program has a generated authorization name, this is used as the user ID.

```
/ADD-USER USER-ID=ISOLDE,PROTECTION-ATTRIBUTES=(-
LOGON-PASSWORD=C'MUSTARD!'),ACCOUNT-ATTRIBUTES=(ACCOUNT=K0815)
```

If the user program has a dynamic authorization name, the OPERID is used as the user ID.

A hexadecimal password is not permitted.

An authorized user program can use any operator command; the only exceptions are the commands subject to restrictions with regard to their input location (e.g. REQUEST-MAIN-CONSOLE-FUNCTIONS). An authorized user program cannot be a replacement or main console.

If the assigned operator role includes authorization for the CONNECT-CMD-SERVER command - by default this has the authorization code K - then the authorized user program may use this command to indicate that it is responsible for processing operator commands. The system will then route those operator commands, for the execution of which the authorized user program has taken responsibility, without subjecting them to any syntax analysis.

CAUTION!
Owners of authorization code K may inadvertently issue important system operation commands and, for instance, disable them. Systems support should therefore grant this authorization only to selected user processes.

Structure of the authorization name table

A maximum of 384 authorization names are created in the startup parameter service with the OPR parameter set (generated authorization names); to these, the system will add any unique authorization names for authorized user programs which establish a connection using an operator identification, until the maximum number of 512 entries has been reached.

The command CREATE-OPERATOR-ROLE allows systems support to define a list of authorization codes which comprise an operator role, and to declare it to a specified pubset. The command MODIFY-OPERATOR-ATTRIBUTES is used to create or modify the assignment of operator roles to an operator identification.

A connection as an authorized user program can be made in two different ways:

- as a connection with an authorization name which was generated (old-style user program); see "Connections with generated authorization names"
- as a connection with a dynamic authorization name; see "Connections with dynamic authorization names"

Command	Meaning
ADD-USER	Make entries in the user catalog
CREATE-OPERATOR-ROLE	Define one or more operator roles
DELETE-OPERATOR-ROLE	Delete the definition of an operator role
EXIT-JOB	Exit the $CONSOLE application, including connection release
MODIFY-OPERATOR-ATTRIBUTES	Create or modify assignment of operator roles to an operator identification
RELEASE-OPERATOR-ROLE	Request release of a specific work area
REQUEST-OPERATOR-ROLE	Request assignment of one or more operator roles
SHOW-OPERATOR-ROLE	Display information on operator roles
Macro	**Meaning**
NBMAP	Write message trailer (see "NBMAP macro" in section "Message formats")
NBMHE	Write the format of the message header (see "NBMHE macro" in section "Message formats")

Table 46: Interface overview for “Authorized user programs with operator functions”

17.1.1 Connections with generated authorization names

A static authorization name can be generated in the startup parameter service, OPR parameter set, using the SET-CODE or ADD-CMD-ENTRY statements, see section "Configuration and suppressing the output of messages at consoles (OPR)". The SET-CODE statement defines the assignment of routing codes to a $CONSOLE application with this authorization name. The assignment can be changed with the ASR commands.

The connection to an authorization name thus created is realized by specifying the following parameters in the connection message:

```
[NAME=]<name 4..4>
[,[PASSWORD=]C'password 1..8']
[,[PROTVERS=]Vn]
[,[DISCON=]{YES/NO}]
```

NAME=<name 4..4>
The specified name must correspond to one of the authorization names set up in the setup parameter service with the SET-CODE or ADD-CMD-ENTRY statements.
In addition, an identically named user ID must exist in the user catalog of the home pubset. The ID should be password protected. It may also be locked (LOCK-USER command). It does not require SECOS access authorizations. Access takes place in the UCON access class.

PASSWORD=<password 1..8>
A password protects the access to $CONSOLE with this authorization name. The password entered for the authorization name in the user catalog must be specified.
A hexadecimal password is not permitted. The password may be a maximum of 8 characters in length.

PROTVERS und DISCON
See the description on "Format of the connection message for dynamic authorization names" in section "Connections with dynamic authorization names".

17.1.2 Connections with dynamic authorization names

Connections which have a generated authorization name no longer meet the strict security requirements of a system, because it is only possible to assign functional areas (i.e. a collection of authorization codes) to a device or to an authorized user program, and not to an individual person.

Such personalization is achieved by using connections with dynamic authorization names, because when a connection is made it is necessary to specify an operator ID (this corresponds to a user ID). Immediately after the “connection”, the user program has no authorizations, and can therefore only use operator functions which require no authorization, and commands which have the authorization code @. Not until the REQUEST-OPERATOR-ROLE command is issued (authorization code @) can authorizations be applied for, and their assignment is subject to an OPERID-related check. An operator role corresponds to a functional area, and is a set of authorization codes compiled by systems support, in which any number and combination from a total of 40 authorizations may be specified. An authorized user program may apply for several operator roles. The logical sum of the operator roles then produces the authorization profile of an authorized user program.

The definition of operator roles by systems support makes it possible to allocate operator functions to **individual persons** and to assign them to authorized user programs.

Setting up the connection for a dynamic authorization name

The connection for an authorized application with a dynamic authorization name is always set up in several steps. Most are optional, depending on the specifications made in the connection message and how the used operator ID is protected.

$CONSOLE (system task UCON) cannot check whether $CONSOLE or @CONSOLE, specified in the incoming connection request, corresponds to the reality ($CONSOLE for the data display terminal as the terminal user and @CONSOLE for a program as the terminal user) since these connection requests are always submitted by a mediator program (e.g. OMNIS).
Systems support should ensure that only mediator programs which can check that the input is correct (first character either “$” or “@”) are used. It can restrict access to BS2000 for selected IDs exclusively to the data display terminal (see the SET-LOGON-PROTECTION command).

- Step 1: request connection

  The authorized user program initially sends a connection request to $CONSOLE on the desired system with the following structure (x=$ or x=@, depending on the type of program connection, see also section "Format of the connection message for dynamic authorization names"):
  `NAME=xCONSOLE[,OPERID=<name>][,PASSWORD=<password>][,PROTVERS=V<integer>]`

  The connection is accepted **provisionally** if the following criteria are satisfied:

  - not all of the dynamic authorization names are in use
  - the logon is permitted by the system parameter NBBAPRIV

  The provisional acceptance is accompanied by a text, from which the protocol version (PROTVERS) accepted by $CONSOLE (UCON), and the BS2000 version are to be taken. If the protocol version requested is greater than the highest version which is supported by $CONSOLE, the accompanying text contains the maximum value that UCON can accept.

Structure of the accompanying text:

00 - 18 DC	C'CONNECTION ACCEPTED'	19 bytes
19 DC	C' ' (Blank)	1 byte
20 - 21 DC	C'nn' (Protocol version number)	2 bytes
22 DC	C',' (Comma)	1 byte
23 - 26 DC	C'nn.n' (BS2000 version number)	4 bytes

For BS2000 OSD/BC V11.0 the BS2000 version number output is C'20.0'.

Irrespective of the version of the protocol which was requested in the connection request, from this point on until the connection is finally accepted or rejected, all messages in both directions will be given a header. The protocol must be strictly adhered to; any unexpected inputs during the logging on dialog will lead to rejection of the connection attempt.

- Step 2: OPR-LOGON-REQUEST

  If the OPERID operand has not been specified, it will be requested by the system in a dialog with the `"PLEASE LOGON"` message.
  The authorized user program must now respond with the following message (the ISP format `LOGON <name>,,<password>` is permissible):
  `SET-LOGON-PARAMETERS USER-ID=<name>[,PASSWORD=<password>]`

  The password specified in SET-LOGON-PARAMETERS is used for subsequent checks. Any password which was specified in the connection message will always be ignored if a retrospective SET-LOGON-PARAMETERS command was requested.

- Step 3: protection criteria check

  Using the information received up to this point, the protection criteria stored by the system in the user catalog for the operator ID, is now requested.

  The connection is shut down if the authorization data check shows that a connection in the specified mode (OPERATOR-ACCESS-TERMINAL or OPERATOR-ACCESS-PROGRAM in the user catalog) is not allowed for this operator ID.

  Step 4 is skipped if the authorization data check shows that the connection is not protected by a password.

- Step 4: request for password

  If the password has not been input, it is requested by the system with `"PLEASE ENTER PASSWORD"`.
  The authorized user program must now respond with the password. The input is hidden under OMNIS.

- Step 5: check of access authorization

  After the authorized user program has passed all necessary data to the system, it must still await confirmation of the connection to the system before it can start normal message communications, in conformity with the version of the protocol specified in the connection request.

  The system checks the connection data and makes a **final** acceptance of the connection if the specified password is correct or no password is required and OPERID and PASSWORD in the user catalog entry agree (if the OPERID has to be authenticated against a password).

  The confirmation contains the following text to be output on the data display terminal:
  `CONNECTION REQUEST ACCEPTED, APPLICATION NAME = @xxx`

Initially, the connected authorized user program can only enter a few commands which do not merit protection (code @). In order to execute operator functions, the program must now request one or more operator roles with a REQUEST-OPERATOR-ROLE command.

Format of the connection message for dynamic authorization names

```
[NAME=]xCONSOLE [,[OPERID=]name]
[,[PASSWORD=]C'password 1..8']
[,[PROTVERS=]Vn]
[,[DISCON=]{YES/NO}]
```

The connection request can also be made without a password.

xCONSOLE
Identifies the type of program connection in a secure system (see the SET-LOGON-PROTECTION command in the “SECOS” manual “Access Control”[46]). For **x** either “$” or “@” can be used:

$CONSOLE
Indicates that the authorized user program is operating interactively with a person. The access authorization is checked against the definitions set with the OPERATOR-ACCESS-TERM operand of SET/MODIFY-LOGON-PROTECTION.

@CONSOLE
Indicates that the authorized user program is operating in batch mode. The access authorization is checked against the definitions set with the OPERATOR-ACCESS-PROGRAM, operand of SET/MODIFY-LOGON-PROTECTION.

OPERID=<name>
An operator identification name. An operator identification is a user ID for an operator. Initially, there is no functional difference between this and a conventional user ID. Only when a SET-LOGON-PROTECTION command has been used to assign operator authorizations to a user ID does it become meaningful to refer to an OPERATOR Identification.

PASSWORD=C'<password 1..8>'
As for any user ID, a password can be specified for an operator ID. The password entered for the operator ID in the user catalog must be specified.

A hexadecimal password is not permitted. The password may be a maximum of 8 characters in length.

PROTVERS=Vn
Designates the version number of the interface between the application and $CONSOLE (n = 00, 01, 02). Each version number corresponds to a unique communications protocol, which is supported and observed by $CONSOLE.

The default value is 00. In this case, messages transmitted in either direction between the system and authorized user programs have no headers.

Under version 01, the messages received from the system by an authorized user program include a header. For messages from the message file, output by the MSG7 or MSG7X macro, the transmitted message has a trailer appended, containing a “data-oriented” mapping of the message, for a programmable evaluation. The data format of the header is defined by the NBMHE macro, that of the trailer by the NBMAP macro (see "NBMHE macro" in section "Message formats" and "NBMAP macro" in section "Message formats"). Messages transmitted from an authorized user program to the system are, as for protocol version 00, sent without headers.

Under version 02, messages transmitted in either direction between the system and authorized user programs have headers. This enables authorized user programs to identify messages as the results of commands or supplementary information, and the system to report the end of command processing.

If the PROTVERS operand is missing from the connection message, the default value 00 is set. If too high a value is specified in PROTVERS, the highest version operated by the system is entered in the text accompanying the connection acceptance. The version number of the interface requested can therefore be set. If the authorized user program does not match the version number set, it can clear the connection.

DISCON
Defines how the system reacts if the authorized user program has such a backlog of messages to receive that the time defined by the system parameter NBRCSCK[N] is exceeded.

DISCON=YES
Default: the system clears the connection to the authorized user program.

DISCON=NO
The authorized user program is not disconnected. Instead of this, all messages waiting to be sent to the authorized user program are deleted and replaced with a single `NBR0601` message.

```
NBR0601 SOME MESSAGES DISCARDED SINCE THE DESTINATION APPLICATION DID NOT
        RECEIVE THEM JUST IN TIME
```

Connection cleardown

To terminate the $CONSOLE application and release the connection, the EXIT-JOB command (in conjunction with NBCONOPI=Y) is offered. This also implicitly implements the function of RELEASE-OPERATOR-ROLE OPERATOR-ROLE=*ALL and thus the release of all authorization codes.

17.1.3 Exchange of messages

Every authorized user program exchanges its messages via the DCAM applications. DCAM offers interfaces for the programming languages COBOL and Assembler:

- If the problem to be solved makes no special demands on message transmission, COBOL can be used.
- If, however, asynchronous processing of incoming messages is required, the programmer has to use Assembler (with macro calls for event-driven processing). For details refer to the manuals “DCAM Program Interfaces” [13], “DCAM COBOL Calls” [11], “DCAM Macros” [12] and “Executive Macros” [30].

Schema of an authorized user program

The following diagram gives an overview of how an authorized user program exchanges messages with the system.

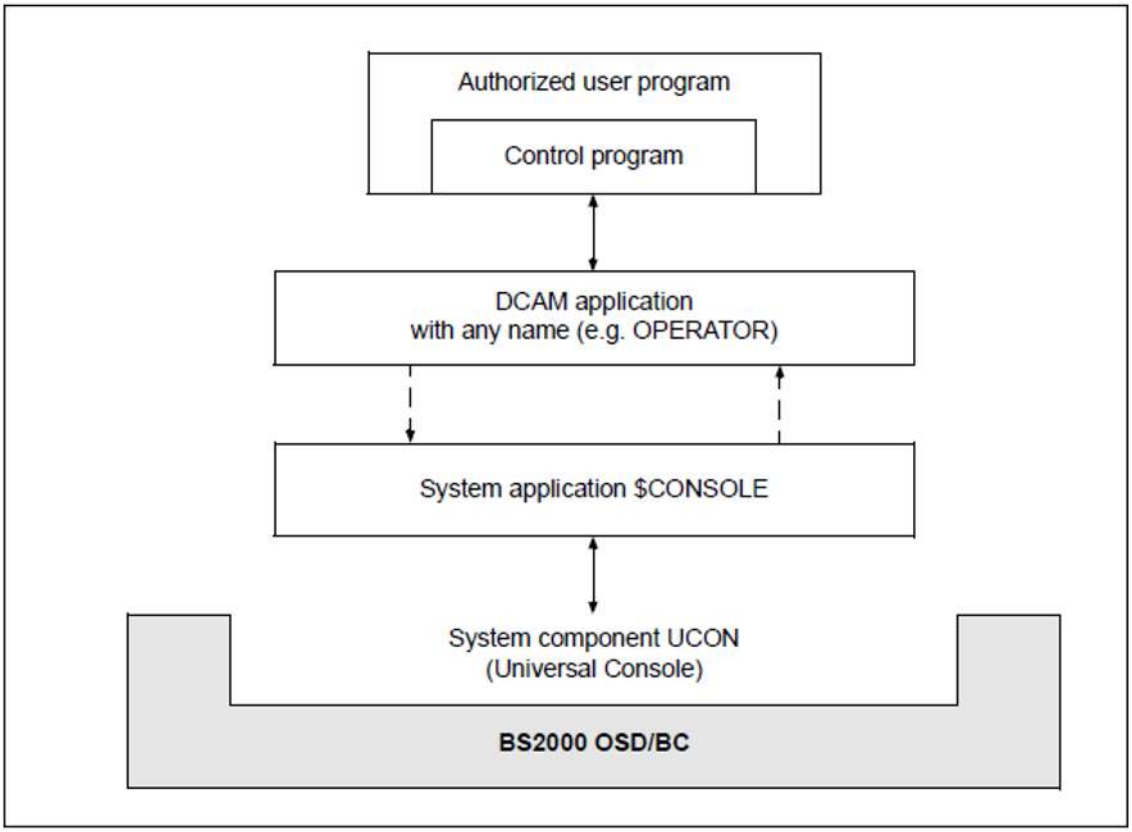


Figure 26: Message exchange between BS2000 and an authorized user program

As every authorized user program exchanges its messages with the system via DCAM applications, such programs have two basic structures for such programs: synchronous and asynchronous.

- Synchronous processing

  The following figure shows the schema of a program that processes incoming messages synchronously. The contents of the loop in this figure may be arranged differently, according to the functional requirements.

  The applicable COBOL calls or macro calls appear in parentheses.

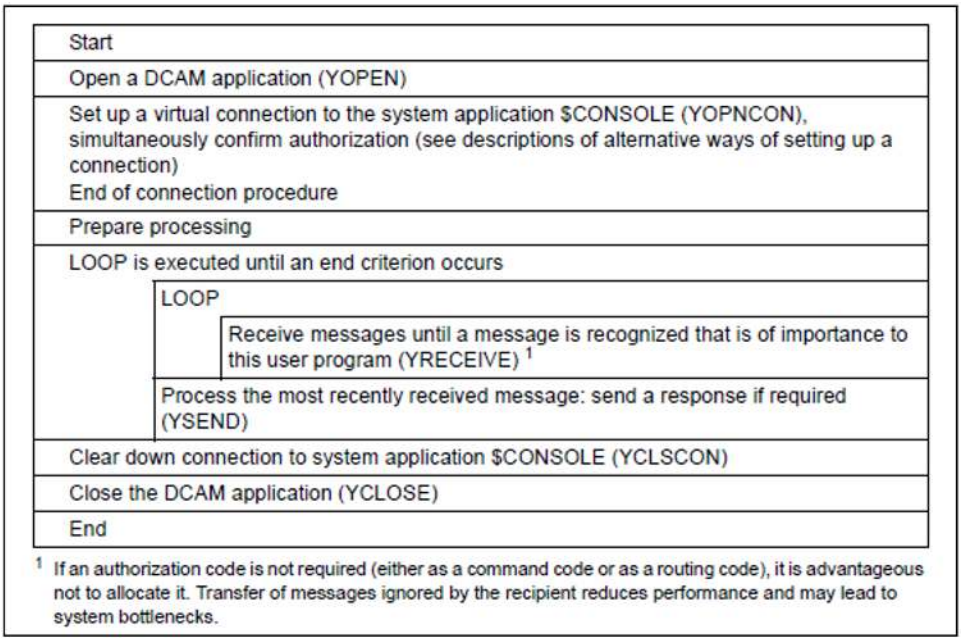


Figure 27: Schema of an authorized user program (process messages synchronously)

- Asynchronous processing

  The schema of a program that processes incoming messages asynchronously differs from the above mainly in the contents of the loop and a minor change in the “Prepare processing” step: event items must be created and contingency definitions communicated to the system; see macro calls ENAEI and ENACO in the “Executive Macros” manual [30]).
  Here, the loop contains only the message request; when the message arrives, it may be processed in a contingency process, including transmission of a response. Further activities not belonging to message processing may be included in the loop. Asynchronous processing of messages is therefore recommended for situations where additional coordination operations are required or several communication partners have to be serviced.

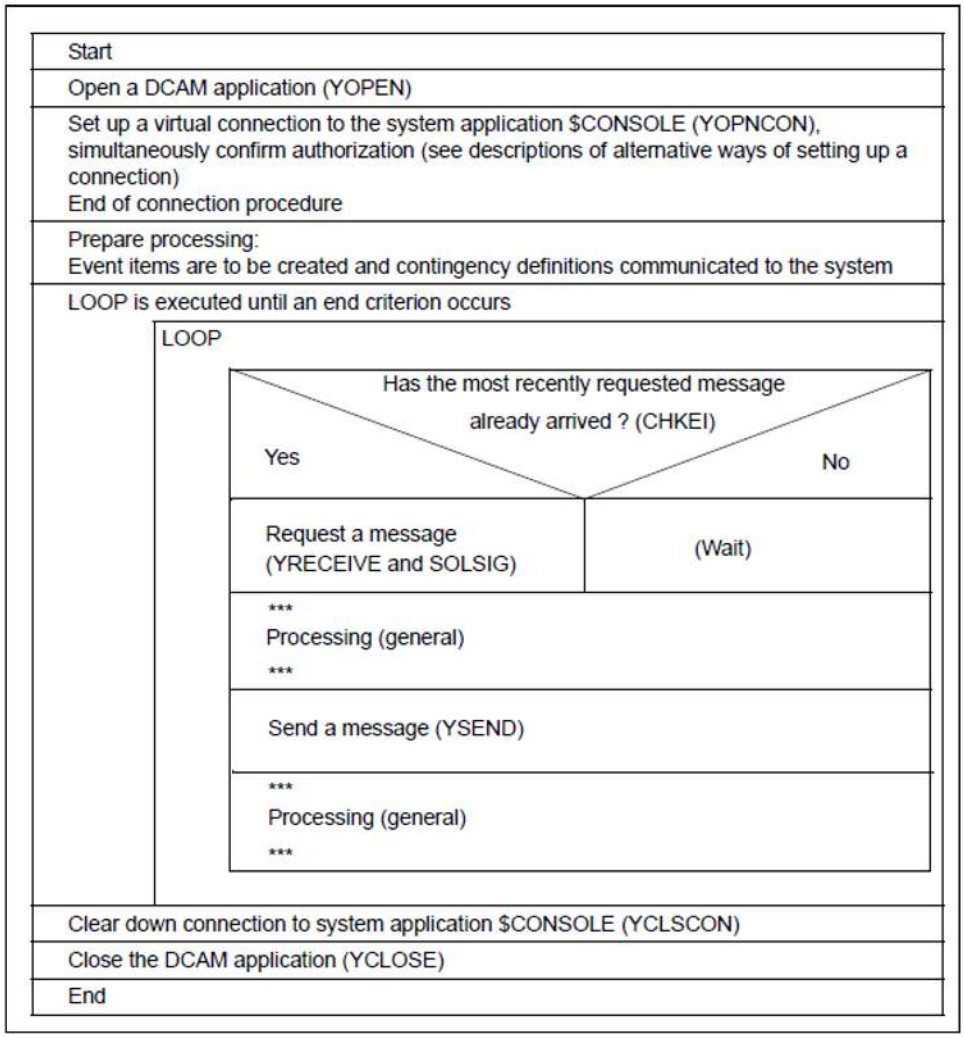


Figure 28: Schema of an authorized user program (processing messages asynchronously)

Monitoring message acceptance

The UCON tasks checks on a cyclical basis (every minute) whether authorized user programs accept the messages sent to them. If it is found that the next message pending for output has been waiting longer than a minute for receipt by the authorized user program, this fact is reported to the console with message `NBR0600`.

If it is also found that the next message has been pending longer than the maximum waiting time laid down by the system parameter NBRCSCK or NBRCSCKN, the system reacts in accordance with the DISCON specification in the connection message (see "DISCON" in section "Connections with dynamic authorization names"):

DISCON=YES: The connection of the relevant authorized user program to the UCON task ($CONSOLE) is aborted.

DISCON=NO: All messages waiting to be output to the authorized user program are deleted and replaced by a single `NBR0601` message.

17.1.4 Message formats

Message formats and logging versions

A message sent by an authorized user program must not exceed 31 Kbytes. Longer messages are rejected with REJ1. Messages to be received by the user program thus also have a maximum size of 31 Kb plus the length of the possibly added header (see the NBMHE macro on "NBMHE macro"), since messages originating from other senders are in any case much smaller (for instance, the maximum length of a message output with MSG7X is 4 Kb).

The text of these messages always has the same format as for input and output at physical consoles (see previous chapter). Additional formats arise only if the authorized user program is acting as a server of a special operator command. These enhancements are described from "Special operator commands in authorized user programs" onwards.
However, pure text format is unsuitable for automatic evaluation of the messages received, since each individual piece of information, e.g. the value of a particular insert in a message, would first have be searched for in the text. The format of output texts is not, however, a guaranteed interface that will not alter when a version is changed.
The system thus makes it possible to exchange formatted messages with authorized user programs, which contain additional information other than text. This is controlled by the logging version that requests the authorized user program which establishes the connection to $CONSOLE. This procedure is different for connections under generated and dynamic authorization names, and is described in the corresponding sections on "Connections with generated authorization names". However, the meaning of the selected logging version is always the same.

Logging version 0

This represents the same declaration that the exchange of messages between the system and the authorized user program is restricted to message text in both directions. Additional information is not exchanged.

Logging version 1

If the authorized user program with the $CONSOLE interface declares logging version 1, the system still expects only text in known formats to be entered. When sending messages by the authorized user program there is thus no difference from logging version 0.However, the authorized user program receives messages from the system in expanded form, comprising a header, the text itself, as it would have been transmitted with logging version 0, and, if the message was one output with MSG7[X], a filler byte followed by a data-oriented description (“mapping format”) of the structure of the message.
A DSECT for describing the message header can be generated with the NBMHE macro, described below.
For the structure and contents of the data-oriented message description, see the description of the NBMAP macro on "NBMAP macro".

Logging version 2

If the authorized user program declares logging version 2 with the $CONSOLE interface, it receives all messages from the system in expanded form. As regards the receipt of messages by the authorized user program, there is thus no difference to logging version 1.Messages sent from the authorized user program to the system must, however, have the same expanded form; it is not sufficient merely to transmit the straightforward text.Generally, creating the message header for the authorized user program merely makes extra work (and sources of error), but entails no particular benefit. The only time it is necessary to user logging version 2 is if the authorized user program is to be used as a server for a manageable special operator command. The special features associated with this are described on "Special operator commands in authorized user programs".

NBMHE macro

The NBMHE macro defines the format of the message header. The Assembler interface has the following call format:

Macro	Operands
NBMHE	MF=D/C, PREFIX=prefix, MACID=macid, VER=1/2/3

Users are not permitted to specify VER=1!

The header has the following structure:

Parameter	V	DS	AT	LH	L2	L1	D	AK	SP	BC	RC2	RC1	FL	UID
Apples for version	123	123	123	123	123	123	123	23	23	23	23	23	3	3

where:

V	Version number of the NBMHE macro which generates this structure; 1 byte
DS	Dialog status byte; 1 byte X'80' Logon dialog is running X'40' Connection provisionally accepted X'10' Final acceptance of the logon X'08' Logon rejected X'04' Operator logon requested X'02' Password requested
AT	Job type; 1 byte Values for macro V01: %?/.*; Additional values as of macro V02: +!&:
LH	Length of the header; 2 bytes
L2	Displacement to the data-oriented section; 2 bytes (L2=0000,if there is no data-oriented section (described with NBMAP))
L1	Length of the text-oriented section; 2 bytes
D	Message destination; 4 bytes

Notes on individual fields

Most NBMHE fields are relevant only in particular contexts, as shown in the table. If the particular circumstance is not given, this means that when messages are received by the system, the field is uniformly undefined, and when messages are sent by the system, the contents of the field are not evaluated by the system, and so can be chosen at random.

For reasons of clarity, the logon dialog of the authorized user program dependent on the type of authorization name used is described separately, see "Connections with generated authorization names". The table relates to the

exchange of messages - separate from the type of authorization name - after a connection has been successfully established (dialog status byte = 0).

V Always relevant
The version number is determined by the VER parameter of the NBMHE macro.
The system always uses the latest version for its messages. The user program is not, however, obliged to use the same version when sending messages; with the exception of version 1, the system accepts all lower versions as compatible and also accepts possible higher versions with the restriction that newly added fields cannot of course be taken into account.

DS Always relevant
The dialog status byte shows which phase of the logon dialog the authorized user program has reached. Values other than zero can occur only during the logon dialog. After the connection has been successfully established, the byte is always 0.

AT Always relevant
Indicates the type of message (see "Communication between operators").

LH Length of the header and simultaneous - since the text of the message follows directly on from the header - the displacement from the start of the message to the text. The field must correspond to the specified NBMHE version (i.e. 25 bytes for version 2, 34 bytes for version 3). To make sure it is also possible to work with future versions, the length must nevertheless be taken from the length field.

L2 (DLL2) Always relevant
Displacement from the start of the message to the data-oriented section (NBMAP).
If no data-oriented section is present, the displacement is 0.
Important: if a data-oriented section is present, it must not immediately follow the text of the message, but must be separated from it by at least one filler byte. L2 is thus inevitably larger than LH+L1.

L1 (DLL1) Always relevant

D Only relevant for messages received

AK Only relevant when receiving commands (AT = “ / ”) and additional command information (AT = “ : ”) and when sending messages as part of command handling (for details, see the relevant section on "Special operator commands in authorized user programs").

SP Only relevant when sending messages
This byte must always be set to zero. Other values are reserved for system applications.

BC Only relevant if SP is not equal to zero (i.e. exclusively for system applications).

RC2 Only relevant for “command termination” messages (AT = “ ! ”).

RC1 Only relevant for “command termination” messages (AT = “ ! ”).

FL Only relevant when receiving commands (AT = “ / ”).
The term “SCI task” here relates to a user task with the privilege needed to enter the command (in the case of self-defined special operator commands this need not be the OPERATING privilege).

UID Only relevant when receiving commands (AT = “ / ”).

The version number of the header should not be confused with the logging version number of the connection.

The field LH is at displacement X'03' in the header in all versions. It shows the displacement to the start of the following text, as this starts immediately after NBMHE.

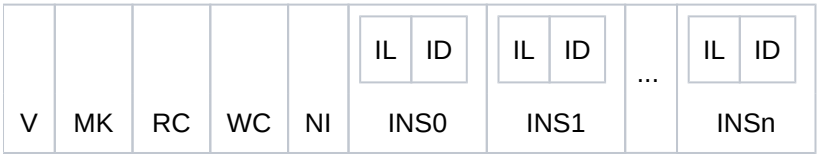
NBMAP macro

The **NBMAP** macro defines the message trailer. The Assembler interface has the following call format:

Macro	Operands

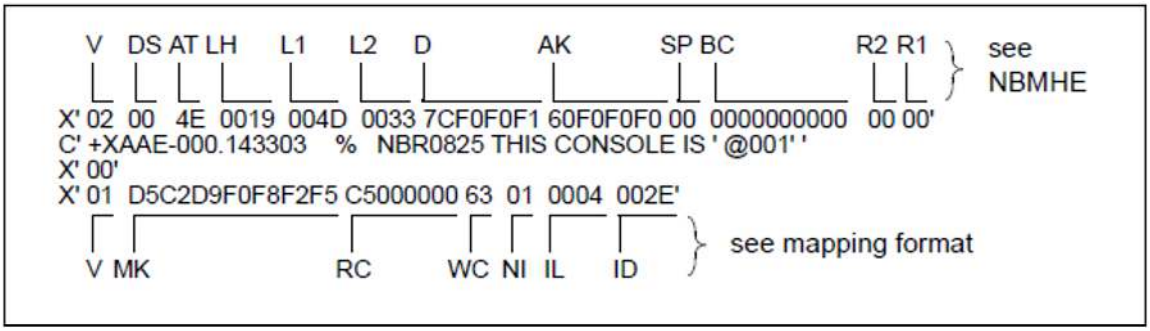
NBMAP	`MF=D/C/L,PREFIX=chars 1, MACID=macid 1..3, VERSION=1`
	`,MSGKEY=chars 7..7,MSGRC=chars 1..1`
	`,MSGWEIGHT=msgweight,MSGINSNUM=integer 0..15`
	`,INSERTS=((length1,distance1),...,(length,distancex))`

The structure is as follows:



V	Version number of the NBMAP macro which generates this structure; 1 byte
MC	Message code; 7 bytes
RC	Routing code; 4 bytes (left-justified, remaining positions filled with 0)
WC	Weight of the message; 1 byte
NI	Number of inserts; 1 byte
IL	Length of current insert; 2 bytes
ID	Displacement of current insert from start of message text; 2 bytes
INS0 .... INSn	0...n message inserts

Example using the message header V02



17.2 Special operator commands in authorized user programs

A special command is a privileged operator command whose format and function are defined by systems support. Any special command received is passed by the system to an authorized user program which interprets the command and performs the desired actions. The processing of special commands is thus a special application of authorized user programs.

The following applies:

- An operation code (= name) must be defined for each special command. This operation code must not coincide with the operation codes of ordinary operator commands. For example, the operation code “CANCEL” is not permitted as an operation code for a special command.
- Every special command has to be assigned to a functional area, i.e. the command is given a routing code. If a special command cannot be linked to any of the existing functional areas, it must be assigned one of the private areas (routing codes W, X, Y, Z).
- In order to enable a special command to be input from a user task with the OPERATING privilege, it must be defined in an SDF syntax file with SDF-A. This syntax file must precede the user task. ADD-CMD ..., IMPLEMENTOR=APPLICATION defines that the command processing is executed in an authorized user program.

From the point of view of the user, an important factor for the behavior of special operator commands is whether the commands have the manageability attribute. A distinction is made between manageable and non-manageable special operator commands.

Manageable special commands

These are managed by the system in just the same way as normal operator commands. It is possible to exchange command-linked messages. The command termination message is - except in error situations - prompted by the command execution $CONSOLE application and includes an evaluable return code.

Non-manageable special commands

These are regarded by the system as “terminated” as soon as they have successfully been sent (with DCAM) to the command execution $CONSOLE application. It is not otherwise possible to exchange command-linked messages. If the operator uses a nonadministerable special command, he will receive the following message

```
NBR0740 COMMAND TERMINATED '<cmd-name>':
        RESULT: (SC=00, SC1=00,MC=NBR0768); DATE:<date>
```

However, this only means that the command was passed to the relevant server. The final result of the command is not included in a termination command and command job IDs are not found as message job IDs in the command results.

Command	Meaning
CONNECT-CMD-SERVER	Create link between authorized applications and operator commands
DISCONNECT-CMD-SERVER	Delete link between authorized application and operator command
SHOW-CMD-ATTRIBUTES	Output information on operator commands

Table 47: Command overview of special operator commands

17.2.1 Command definition

All entries made at a console are initially routed to UCON, the central console task of the system. If it is a command being entered, UCON must first carry out the authorization check linked to the routing code of the command. Only when this is done may the command be passed to the relevant command server.
As the routing code of a command (and in the case of special operator commands also the command server) is not however part of the SDF syntax file, UCON must keep a separate command table, in which the required information is held. Commands not contained in the UCON command table cannot be executed at consoles. Special operator commands must therefor be the first entries in these tables.

Static entries

One option for entering special operator commands into the UCON command table is the parameter service instruction ADD-CMD-ENTRY (see "ADD-CMD-ENTRY statement"). The entries generated in this way are static, i. e. are retained for the entire duration of the session. This means that only a $CONSOLE application with a (statically) generated authorization name can be specified as the relevant command server (if the specified authorization name is not yet known, it is automatically generated. In addition, there are no facilities here for declaring the command as manageable.

If a special operator command is to be executed in manageable form or is to be executed by a $CONSOLE application with dynamic authorization names, the entry in the command table must be made dynamically with the aid of the CONNECT-CMD-SERVER command.

Dynamic entries

A $CONSOLE application that is to act as a server for a special operator command can log itself on as a command server dynamically in the case of UCON with the CONNECT-CMD-SERVER command. UCON then automatically creates a corresponding entry in the command table.
It should be noted that up to four entries in the command table are permitted for each command. CONNECT-CMD-SERVER can also be executed for commands which have already been entered (also system commands), in which case not all attributes of the command can be changed; see command description. The new entry then “overlays” the old one, and the commands are henceforth delivered to the $CONSOLE application that possesses the latest entry. Which entries exist can be displayed using the SHOW-CMD-ATTRIBUTES command.
A dynamic entry in the UCON command table is automatically removed if the connection between $CONSOLE and the relevant command server is cleared. The application can also delete it explicitly by means of the DISCONNECT-CMD-SERVER command.

If the entry removed was the only one for this command, the command is no longer recognized. If there are other entries, the latest of the entries still existing becomes (or remains) effective; the commands are delivered to the owner of this entry.

Definition in the syntax file

SDF-A permits the creation of syntax file entries for special operator commands (statement ADD-CMD <name>, IMPLEMENTOR=*APPLICATION). Such a syntax file entry is necessary if the command contains passwords that are not to be logged in the CONSLOG. Such commands are entered into the UCON command table with PASSWORD-POSSIBLE=*YES. The CLOG task then calls SDF before logging to CONSLOG and removes all operands identified as “secret” (SDF-A: ADD-OPERAND <name>, SECRET-PROMPT=*YES). If there is no syntax entry, the command is logged unchanged in plaintext.
In addition, special operator commands for which there is a syntax file entry can be used not only at consoles, but in all user tasks with the appropriate authorization in accordance with the command privileges entered in the syntax file. An authorization check on the basis of the routing code is not made on entry in a user task.

If a special operator command is to be made available to a particular user group by entry into the system syntax file, the following point must be noted:
The syntax file entry does not replace the entry in the UCON command table. Commands which are entered in the system syntax file but not in the UCON command table are rejected with message NBR0744. The same procedure applies for additional alias names of a known command which are, however, unknown to UCON.

The OPERATING privilege can be assigned to any user IDs. This allows them to use almost all so-called operator commands.

17.2.2 Message formats

The following formats are described:

- Format of the commands for receiving operator commands
- Format for sending command results and additional command information
- Format for sending end-of-command messages
- Format for receiving additional command information

Format of the commands for receiving operator commands

```
[header]/{tsn | an | (mn)}:  cmd[ operands]
```

header	For applications connected to $CONSOLE with protocol version 01 or 02, the format contains a message header n characters in length which is defined by means of the NBMHE macro.
/	Slash character indicating type of message.
tsn	Task sequence number of the user task by which the special operator command was entered.
on	Authorization name of the authorized user program, by which the command was entered (4 characters).
mn	Mnemonic device name of the physical console at which the command was entered (2 characters).
:	Colon as a separator
' '	Blank
cmd	Name of a command.
operands	Command operands

> The task identification that an authorized user program is given in the header on receipt of the command is a unique task identification assigned by the system. It is not identical to the task number assigned when the command is entered.
> The internal job identification is transparent for the authorized user program. The system manages the command on the basis of this task identification and ensures that the command results sent in connection with the command, the additional requests and the command termination message are always provided with the task identification entered by the person issuing the command and reach the person issuing the command.
> A precondition is that the internal task identification and the task type (command result (+), additional command information request (&) or command termination (!)) are supplied correctly in the header.

Examples

1. without header

   ```
   /OPR1: SPECI OP1=VALUE1,OP2=VALUE2
   ```

2. with header

   ```
   <header>/OPR1: SPECI OP1=VALUE1,OP2=VALUE2
   ```

For the header data format, see the NBMHE macro ("NBMHE macro" in section "Message formats").

Format for sending command results and additional command information

```
<header> text [<mapping-format>]
```

header The corresponding task type must be identified in the header and the task identification field supplied with the task identification taken from the header of the command. The system ensures that the text appears at the place the command was entered, with the task identification contained in the command.

text Free-format text or message text in MSG7X format

mapping-format

If a $-CONSOLE application sends messages of type “+” or “&”, specifies a mapping format (see the NBMAP macro, "Message formats") and the sender of the command was a user tasks, the system ignores the specified message text for message output.

However, the specified message text is included in the CONSLOG. It should therefore be in MSG7X format. Furthermore, internal checks (e.g. the message number at offset 4 in the message text) are performed which can generally only be successful when MSG7X format is used.

The sender of the command receives a message containing the outputs created by the MSG7X macro with the parameters specified in the mapping format. The language valid for the environment of the sender of the command is used automatically. The purpose of this MSG7X macro is thus to output the message text in the language required.

Format for sending end-of-command messages

```
header maincode
```

header The job type “!” must be entered in the heading and the fields RC1 and RC2 supplied with the subcode 1 or subcode 2 of the command result.

The job identification field must also be supplied with the job identification taken from the header of the command. The system ensures that the message `NBR0740` is output to the place where the command is received.

```
NBR0740 COMMAND COMPLETED 'xxxx'; (RESULT:....); DATE:.....
```

The message also contains the job identifier entered when the command was entered and is identified with an exclamation mark.

maincode 7-character maincode of the command result

> Once a command termination indicator has been sent, it is no longer possible for an authorized user program to use the internal job identification. Jobs for commands whose termination has already been indicated or jobs with an incorrect job identification are rejected by the system with message `REJ11`.

Format for receiving additional command information

```
header :{tsn | an | (mn)}[-mid]:[text]
```

header	Header of length n characters, described by the NBMHE macro
:	Message type ID (colon)
on	Authorization name of the authorized user program delivering the response (4 characters)
mn	Mnemonic device name of the physical console delivering the response (2 characters).
tsn	Task serial number of the responding user task.
-	Hyphen
mid	Message job ID (A..Z, 0-9, @, #, $); filled to 3 characters with leading zeros. default value = 000
:	Colon as a separator
text	Arbitrary response text.

17.3 Software products OMNIS and PROP-XT

The software product **OMNIS** is a control system for central operation of multiple applications in BS2000.

In BCAM, a terminal can, as a rule, maintain only one connection to one communication partner at any one time. OMNIS is a program that allows this restriction to be overcome.

OMNIS lets you set up connections from one terminal to a number of partners in a server network concurrently. Using OMNIS you can also maintain multiple connections to one partner. OMNIS further allows you to use 9750 terminals for applications that are not normally supported by terminals, such as UCON.

OMNIS is described in detail in the “OMNIS/OMNIS-MENU” manual [33].

PROP-XT is a product used for server control in the field of data center automation. It permits programmed operation by the user with the user-friendly language resources of SDF-P-generated administration procedures.

PROP-XT is described in detail in the “PROP-XT” manual [40].

17.4 Command files for the operator

Command files on system startup

After system initialization the operator usually has to make preparations for processing in interactive mode. This includes startup of the data communication system and initiation of the inquiry and transaction applications, among other things.

> The data communication system may be started even before “System ready”. For this purpose, the appropriate BCAM commands can be entered in the startup parameter file (BCAM section). See also the “BCAM” manual [4].

The commands required for preparing for interactive operation may be stored in a command file.
If the name of this command file was already defined in the startup parameter service (system parameter CMDFILE), the system automatically processes this file immediately after system startup.
The operator can either have a different command file executed by modifying the system parameter CMDFILE by means of the parameter service or during dialog startup or enter commands manually (see also the system parameter NBRUNUID).

Command files for recurrent command sequences

Command sequences which the operator has to enter repeatedly on different occasions – for example in the case of frequently recurring jobs – can be stored in command files.

The operator uses a RUN command to start execution of a command file. The RUN command may be issued either from the physical console or by an authorized user program. In either case it is necessary that those functional areas referenced by the commands in the command file be assigned to the calling physica operatorl terminal or to the authorized user program. In addition to the command files, systems support may create ENTER jobs that are started by the operator from the console. As command sequences in ENTER jobs contain user commands and do not require any special operator action following startup, these command sequences are skipped in the description below.

The provision of operator functionality in any user tasks with the OPERATING privilege means that any procedures (with and without SDF-P) can be provided for the operator. These command files can contain a random mixture of operator and user commands. Starting the procedures with CALL-PROCEDURE and ENTER-PROCEDURE, however, is only possible from user tasks and not from physical consoles or authorized applications. The functions are the same as the procedures in user tasks and are therefore not described here.

The following thus refers exclusively to operator command sequences.

Command	Meaning
AGOGO	Continue a command file
ASTOP	Stop processing the command file and define the continuation condition
CANCEL-RUN-PROCESS	Aborting a command file
ENTER-JOB	Start an ENTER job
RUN	Start a command file
SHOW-PENDING-MSG	Repeat the display of system messages

Table 48: Overview of commands for command files

17.4.1 Executing and aborting a command file

Execution of a command file

A command file can be called from any physical console and from any authorized user program, provided it has authorization code “E” (see section "Authorized user programs with operator functions"). However, all the functional areas to which the commands in the command file refer must have been assigned to the calling console or user program (via the authorization code).

The system processes no more than one command file at a given time. If a further RUN command is received during processing of a command file, the system delays execution of this command until processing of the current command file has been completed.

The system behaves as follows:

- if there are a number of RUN commands within a command sequence, they are processed in the order in which they appear in the command sequence.
- if a RUN procedure is stopped by a CANCEL-RUN-PROCESS command, none of the RUN commands held in the RUN file will be executed after that.

When execution of a command file is started, message `NBR1000` indicates this. The caller of a command file receives only messages that are generated on execution of the various commands; the commands themselves are normally not displayed to the caller. If the issuer of a command wishes to record the command from the RUN file on the console, this can be arranged by means of the system parameter NBRUNSP=Y. Irrespective of the system parameter NBRUNSP, commands from RUN files are recorded in the logging file (CONSLOG).

If a command file contains special commands, it will only execute correctly if the authorized user program that is to receive and execute the special commands has already been started.

Therefore the execution of any command file containing special commands must be coordinated with the start of the authorized user program. To this end, the following actions have to be initiated from the RUN procedure:

1. Start authorized user programs for handling of special commands (using ENTER-JOB commands)
2. Halt processing of command file and define continuation criterion (using the ASTOP command)

In order to take advantage of the startup interval required by the tasks it is advisable to place the ENTER-JOB commands at the beginning of the command file and the ASTOP command directly before the first special command.

Processing of the command file continues if one of the following conditions is satisfied:

- as many AGOGO commands have arrived as are specified in the criterion of the ASTOP command or
- the waiting time has elapsed and the required number of AGOGO commands have not arrived.

The time interval for which command processing is interrupted by an ASTOP command can be specified using the system parameter NBRUNWT. This can have a value from 10 to 255 seconds, and by default is 180 seconds.

Example for a special command

> Systems support has provided the special command SPECI for the system which is its responsibility; this command is to be used in the command file following system startup.

This file must contain the following commands (among others):

```
/ENTER-JOB SPEC.CMD-PROC
  :
/ASTOP
/SPECI
   :
```

The SPEC.CMD-PROC file must contain a suitable command sequence for starting the user program which was created for processing the SPECI command. In addition, the file should contain the AGOGO command which, in conjunction with the ASTOP command of the RUN procedure, ensures the synchronization.

> Some commands, such as SHOW-PENDING-MSG, are processed by the system with a higher priority than others. If such commands are specified in a command file, they can “overtake” other commands which are in front of them in the command file.

The interplay between command file processing and the actions of a single authorized user program is shown in the following diagram.

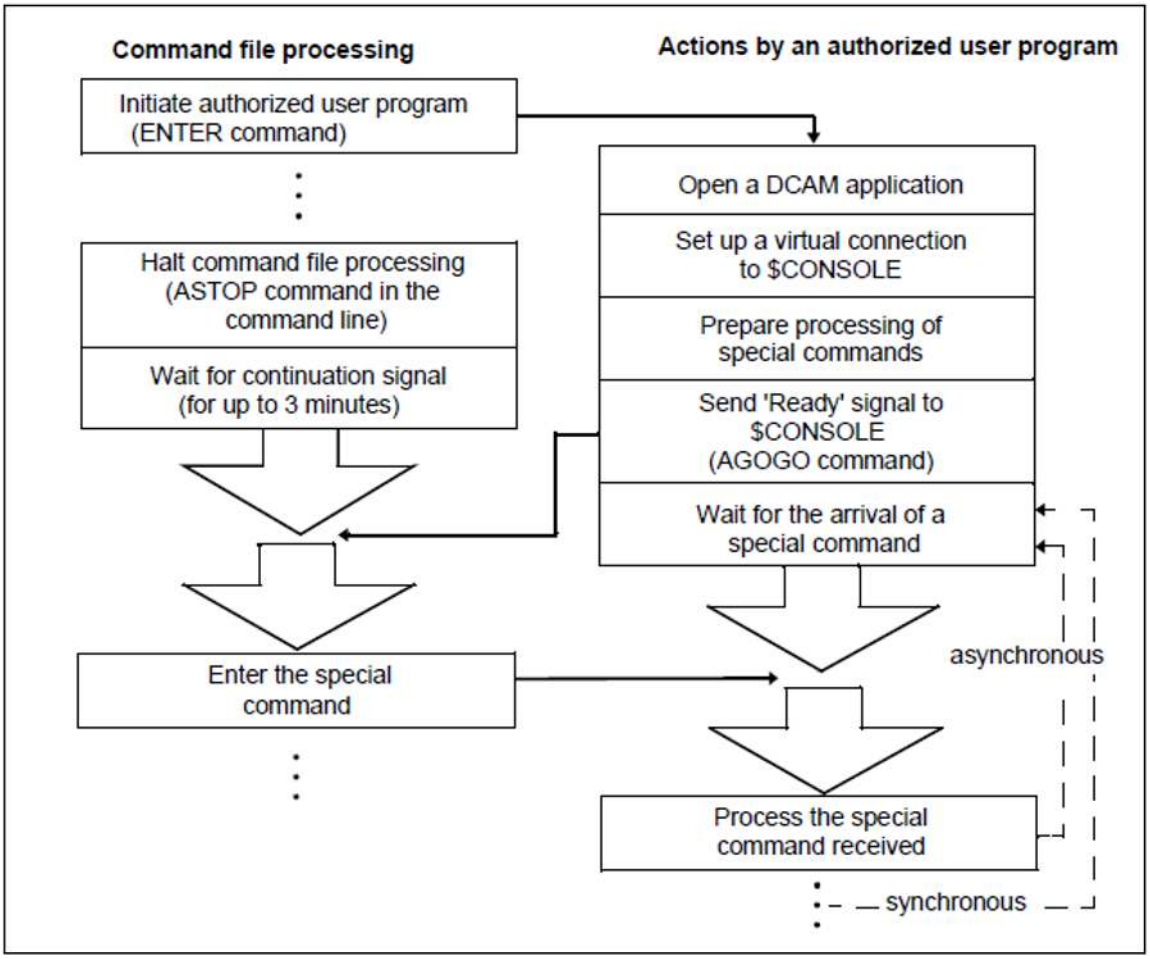


Figure 29: Processing a command file

Aborting a command file

The CANCEL-RUN-PROCESS command aborts processing of a command file that has been started. To execute this command, the operator must specify a run ID that is assigned to each command file after it is started by the RUN command. A command file that is started recursively from another RUN sequence has the same run ID as the command file that generated it. Aborting a command file with the CANCEL-RUN-PROCESS command automatically aborts all command files generated by this command file. This command does not necessarily have to be entered at the same console as the RUN command. However, it does require the authorization code E.

17.4.2 Structure of command files

A command file started by the operator from the console is a SAM or ISAM file or a library element (of type J) on public volumes. The data records may contain any operator commands and, with certain restrictions, privileged and user commands. The RUN command sequence must not contain a SET-LOGON-PARAMETERS or EXIT-JOB command. The command file records are of either fixed or variable length, the maximum record length being 201 bytes. Continuation lines are not supported.
Systems support can protect the command file against unauthorized execution by the operator by assigning an execution password. The command file is then processed by the system only if the operator enters the correct password in the RUN command. The password is not logged in the CONSLOG file.

RUN command sequences may consist of any number of commands. At the latest after 30 commands, however, systems support should insert the ASTOP command in order to ensure structured execution of the commands. It is also a good idea to interrupt the sequence whenever the execution of a command necessitates the complete processing of the previous one or the invocation of dependent functions.
For example, if 10 commands are contained in the RUN command file, these commands are read and processed in succession. If a SHOW-PENDING-MSG command occurs in the command sequence, the procedure after this command should be interrupted by means of an ASTOP command. The operator may then answer any outstanding queries. Processing of the command sequence is resumed after the waiting time has expired or an AGOGO.

One advantage of splitting up a RUN file with ASTOP commands is that the command sequence does not occupy class 4 memory unnecessarily during execution.
An extensive command file created in this way is read by the system into class 4 memory section by section, i.e. from one ASTOP command to the next; each file section read in is executed before a new one is read.

If DSSM commands are part of the command file, systems support should note the following:
DSSM commands are processed by a separate task. Commands which cannot be issued until the relevant subsystem is loaded must wait for the DSSM commands to be processed first. The ASTOP command must therefore be inserted in the command file at this point.

The CMDFILE command file (library members are not permissible) may contain a sequence of operator commands that execute invariably on every system startup. For example, this file contains calls for the following command sequence:

DCSOF	Command file (Start Option File) for startup of the data communication system (DCS)

Example of a CMDFILE

```
/MODIFY-JOB-CLASS CLASS-NAME=JCDIALOG,C-L=0                                  1.
/MODIFY-JOB-CLASS CLASS-NAME=TSOSDIA,C-L=0                                   2.
/MODIFY-JOB-CLASS CLASS-NAME=JCBATCH,C-L=40                                  3.
/SET-MSG-SUPPRESSION MSG-ID=(BLS0980,JMS0154),CONSOLE=C0                     4.
/DCSTART DCSOF=SOF.ACT.DCM8                                                  5.
/ASTOP                                                                       6.
/MOD-TASK-CAT CATEG-NAME=DIALOG,WEIGHT-CODE=400,MIN-ACT=35,MAX-ACT=50        7.
/MOD-TASK-CAT CATEG-NAME=BATCH,WEIGHT-CODE=10,MIN-ACT=5,MAX-ACT=20           8.
/MOD-TASK-CAT CATEG-NAME=TP,WEIGHT-CODE=500,MIN-ACT=20,MAX-ACT=40            9.
/IMPORT-PUBSET PUBSET=L                                                     10.
/ATTACH-DEVICE (X3,X4,X5,S1,S3,K7,X0,Y3)                                    11.
/SET-DISK-P V=(PRI009,PRI010,PRI037,PRI038,SESAMA),USER=SHARE,ASS=OPER      12.
/SET-DISK-P V=(WORK11,SLED),USER=SHARE,ASS=OPER                              13.
/START-SUBSYS SUBSYS-NAME=MSCF,SUBSYS-PARAM='FILE=$TSOS.RFA.MSCFSTART'      14.
/START-SUBSYS SUBSYS-NAME=SPOOL,SUBSYS-PARAM='RSO=STD'                      15.
/ASTOP                                                                      16.
/MODIFY-JOB-CLASS CLASS-NAME=JCDIALOG,CLASS-LIMIT=20                        17.
/MODIFY-JOB-CLASS CLASS-NAME=TSOSDIA,CLASS-LIMIT=4                          18.
```

1. Do not permit any jobs for job class JCDIALOG.
2. Do not permit any jobs for job class TSOSDIA.
3. Do not permit more than 40 jobs for job class JCBATCH.
4. Messages `BLS0980` and `JMS0154` will be suppressed at console C0.
5. Load the Data Communication System.
6. Interrupt the command file until the Data Communication System has been loaded.
7. Define the number and weight of tasks for task category DIALOG.
8. Define the number and weight of tasks for task category BATCH.
9. Define the number and weight of tasks for task category TP.
10. Import pubset L.
11. Attach devices to the system.
12. Set values for disk monitoring.
13. Set values for disk monitoring.
14. Activate MSCF communication.
15. Load the subsystem SPOOL with automatic loading of RSO.
16. Interrupt the command file until SPOOL has been loaded.
17. Do not permit more than 20 jobs for job class JCDIALOG.
18. Do not permit more than 4 jobs for job class TSOSDIA.

The system parameter NBCONOPI=Y can be used to request a system in which a logon is also necessary for physical consoles. In such systems therefore, entry of, say, a RUN command at a physical console is only possible after a successful SET-LOGON-PARAMETERS and a corresponding REQUEST-OPERATOR-ROLE.

If, when the system parameter NBCONOPI=Y is used, the file name of the RUN file is specified without a user ID, the file is sought first under the ID of the operator, then under the ID specified in the system parameter NBRUNUID, and finally under the TSOS user ID. Because the OPERATING privilege no longer requires the TSOS ID, the RUN file may have to be made “shareable”.

When the system parameter NBCONOPI=Y is used, it is not the case that every operator can enter the AGOGO or CANCEL-RUN-PROCESS command for every RUN sequence, even if he/she has the necessary authorization code.
Both commands are only permissible when at least one of the following conditions is fulfilled:

- the user ID of the caller matches the ID under which the RUN command was issued
- the user ID of the caller is TSOS
- the RUN command was entered by an authorized user program with a generated authorization name
- the caller of the RUN command has logged off from the system.

At system startup, the CMDFILE is processed, being treated like a RUN command issued on this file from the main console. It is not necessary to enter the SET-LOGON-PARAMETERS or REQUEST-OPERATOR-ROLE command beforehand, because, for this CMDFILE and only at startup, the authorization check is performed using authorization codes as though the main console had been assigned all authorization codes. All RUN commands called from the CMDFILE also enjoy this privilege.

The RUN command and the commands that are logically linked to it (e.g. AGOGO and CANCEL-RUN-PROCESS) are not allowed for a user task, even with the OPERATING privilege. Here, SDF-P procedures can be used, in which commands of the OPERATING privilege can also be used.

17.5 System administration functions performed by the operator

The SHOW-CMD-ATTRIBUTES command informs the operator at the consoles of all commands known at this time. Commands which have the code $, however, are prohibited (entry is rejected).

The set of $ commands also contains commands which do not offer any systems support functions. If some of these commands are to be allowed on consoles, they must be assigned to a different authorization code via the parameter service at system initialization (parameter record OPR, see "Configuration and suppressing the output of messages at consoles (OPR)").

For this purpose, a SET-CMD-CODE statement is required for every command. The authorization (routing) code must be defined by systems support.

Example

The command with the name SHOW-FILE-ATTRIBUTES is to be assigned to the functional area “general tasks” (routing code E). Consoles K1 and K2 are to be allocated “general functions and responsibilities”.

```
...
SET-CMD-CODE AUTHORIZATION-CODE=E,CMD-NAME=SHOW-FILE-ATTRIBUTES
SET-CODE CODE=E,CONSOLE=(K1,K2)
...
```

The manufacturer supplies the following commands with the authorization code $ by default:

```
ADD-FILE-LINK
ADD-MASTER-CATALOG-ENTRY
ADD-PASSWORD
ADD-USER
COPY-FILE
CREATE-FILE
CREATE-FILE-GENERATION
CREATE-FILE-GROUP
CREATE-JV
CREATE-TAPE-SET
DELETE-FILE
DELETE-FILE-GENERATION
DELETE-FILE-GROUP
DELETE-JV
DELETE-SYSTEM-FILE
DELETE-TAPE-SET
EXPORT-FILE
EXTEND-TAPE-SET
IMPORT-FILE
LOCK-USER
MODIFY-FILE-ATTRIBUTES
MODIFY-FILE-GENERATION-SUPPORT
MODIFY-FILE-GROUP-ATTRIBUTES
MODIFY-JV-ATTRIBUTES
MODIFY-MASTER-CATALOG-ENTRY
MODIFY-SPACE-SATURATION-LEVELS
MODIFY-USER-ATTRIBUTES
MODIFY-USER-PUBSET-ATTRIBUTES
PRINT-DOCUMENT
REMOVE-JV-LINK
REMOVE-MASTER-CATALOG-ENTRY
```

```
REMOVE-PASSWORD
REMOVE-USER
SET-JV-LINK
SHOW-FILE-TRANSFER
SHOW-JV-ATTRIBUTES
SHOW-JV-LINK
SHOW-SPOOL-CHARACTER-SETS
SHOW-SPOOL-DEVICES
SHOW-SPOOL-FORMS
SHOW-SPOOL-PARAMETERS
SHOW-USER-ATTRIBUTES

UNLOCK-USER

WRITE-SPOOL-TAPE
```

The SHOW-FILE-ATTRIBUTES and SHOW-SDF-PARAMETERS (preset routing code @) and MODIFY-SDF-PARAMETERS (preset routing code *) commands, on the other hand, are available to the operator by default. They are needed in the event of an emergency to permit the syntax file configuration to be reestablished to such an extent that it is posible to log on again in interactive mode.

If this is not required, the commands can be forbidden by assigning the routing code $.

18 System time administration

There are different clocks and thus different times on an SE server:

- The BS2000 system time based on the TODR (Time of Day Register);
  On SU /390 the TODR is an autonomous clock.
  On SU x86 the TODR is emulated by X2000.
- The time of the service processor (SVP time), which can be determined by BS2000 using the “Store Real Clock” command.
  On SU /390 the SVP clock is an autonomous clock, which is synchronized by the Management Unit with its own time.
  On SU x86 it is emulated by X2000 and supplied with the carrier system time.
- The time of the Management Unit.
  The Management Unit can be synchronized with external timers.
- The time of the carrier system on SU x86, which is synchronized with the time of the Management Unit via NTP. This is emulated by X2000 as the SVP time.

Furthermore, BS2000 can be integrated in a computer network in which the time of the participating systems is synchronized.

This chapter explains these interactions. It provides an overview of the administration of the system time in BS2000 and describes the initialization and synchronization of the system time as well as the summer time/winter time changeover and special cases.

For information on determining the time on system initialization see "Determining the time on system initialization".

18.1 System time

The legal local time is used as the system time for a BS2000 session. All time stamps with privileged status (TPR /SIH/MER) are based on this time. It is supplied by the time information services (GTIME, GDATE, GETOD) to user programs (TU).

Certain date specifications (especially the time stamps in file catalogs) are made in UTC time. Some low-level, performance-critical components use the TODR value which is ascertained with STCK (STore ClocK).

The current system time is calculated as follows:

- the current STCK value is determined
- the TODR correction value, which is calculated on the basis of time corrections from concluded synchronization jobs and the initial IPL correction value, is subtracted
- if a synchronization job is currently active, it is also taken into account

As a result, a time value calculated by means of STCK does not correspond to the time reported by the official information services and does not therefore represent the current time.

The command SHOW-SYSTEM-INFORMATION INFORMATION=*SYSTEM-TIME-PARAMETER can be used to obtain information on the time setting parameters.

18.1.1 TODR as hardware clock

The Time of Day Register (TODR) is a 64-bit register with the following properties:

- It is increased by the value 1 every microsecond in the range [bit 63, bit 12] and therefore functions as a clock.
- It can be modified by the privileged Set Clock (SCK) instruction.
- It can be read by the non-privileged Store Clock (STCK) instruction.

The width of the clock in the TODR also limits the size of the time span which can be displayed. BS2000 extends the time display in the TODR compatibly using the so-called epoch, see the section "TODR epoch".

The TODR is initialized during the first BS2000 IPL. With the default epoch 00, this has the same meaning as the value ”number of microseconds since 1900-01-01 00:00:00”. For other epoch designations, the TODR must be interpreted in accordance with the epoch in order to obtain a correct time value, see the section "TODR epoch".

On SU /390 Server Units, this initialization is repeated during each additional IPL.
On SU x86 Server Units, this initialization takes place only once. If a re-initialization of the TODR is necessary on SU x86, the Server Unit must be restarted in native mode, while under VM2000 the VM must be recreated (for example with /DELETE-VM and /CREATE-VM).

Virtual TODR on SU /390 under VM2000

The VM2000 firmware enables a separate virtual TODR to be maintained for each VM. The hardware TODR of the SU /390 is available only to the Hypervisor. All TODRs are clocked identically. The hardware context of a VM includes a VM-specific correction value of the virtual TODR relative to the hardware TODR.

On SU /390 the Hypervisor initializes the hardware TODR with the value “monitor time – 24 hours”.

Emulated TODR on SU x86

On SU x86, Set Clock and Store Clock are emulated by the firmware. The emulated TODR is clocked independently of the carrier system time. All VM2000 guest systems have their own virtual TODR.

18.1.2 TODR epoch

BS2000 extends the TODR by a TODR epoch. This enables a system time to be displayed different from the past default range (from 1900-01-01 00:00:00.000000 to 2042-09-17 23:53:47.370495) which corresponds to the default epoch 00.

The TODR epoch for the current session is set in the startup parameter service, GTIME parameter record, EPOCH parameter, see the section "System time control (GTIME)". It cannot be changed during the session.

The epoch designation is automatically taken into account in the user macros CTIME (Time stamp calculations) and GTIME (Get date and time), see the “Executive Macros” manual [30]. The value of the current epoch designation can be obtained via the user macro GTIME and also the SHOW-SYSTEM-INFORMATION command.

> **!** To ensure that the functions GTIME and CTIME correctly interpret the TODR, even after its overflow in the year 2042, the default epoch will be changed from 00 to 08 well in advance. This will allow the representation of a date range from 1971 to 2114 (see table below).
>
> For applications which use timestamps in TODR format, this may lead to incompatibilities and they must be adapted before changing the epoch. Therefore it is strongly recommended that applications do not store timestamps in TODR format to allow trouble-free changes of the TODR epoch.

Interpretation of TODR Time Stamps with TODR Epochs 00 and 08

TODR Time Stamp (hex)	Date and Time Interpretation	
	EPOCH=00	EPOCH=08
00000000 00000000	1900-01-01 00:00:00.000000	2042-09-17 23:53:47.370496
7FFFFFFF FFFFF000	1971-05-11 11:56:53.685247	2114-01-26 11:50:41.055743
80000000 00000000	1971-05-11 11:56:53.685248	1971-05-11 11:56:53.685248
FFFFFFFF FFFFF000	2042-09-17 23:53:47.370495	2042-09-17 23:53:47.370495

Table 49: Comparison of the current default epoch 00 and the future default epoch 08

TODX Time Stamp Format

As an alternative to the TODR format, the TODX time stamp format is also available in BS2000. TODX consists of a double word and contains the number of microseconds since 1900-01-01 00:00:00.000000. In contrast to the TODR format, the TODX format covers a much wider time span (from the year 1900 to the year 4317).

The TODX format is used/supplied by the user macros CTIME (Time stamp calculations) and GTIME (Get date and time), see the “Executive Macros” manual [30]. It is useful for epoch-independent representation of a time stamp and for calculating with time stamps from different epochs.

With the current default epoch 00, a TODX time stamp corresponded to a TODR time stamp whose content has been moved 12 bits to the right.

Programs which themselves calculate using TODR values can use the algorithms described in the section "Converting TODR time stamps to TODX time stamps".

Converting TODR time stamps to TODX time stamps

The algorithms below use a TODR value and the associated value of the epoch designation to calculate the number of microseconds since 1900-01-01 00:00:00.000000, i.e. the TODX value.

The variables and operators used are:

```
TODX   :  TODX value, i.e. number of microseconds since 1900-01-01 00:00:00.000000
TODR   :  TODR value
EPD    :  value of the epoch designation (1 byte)
EPC    :  epoch counter (first hexadecimal digit of EPD)
EPO    :  epoch offset (second hexadecimal digit of EPD)
x >> n :  Right-shift value x by n bits
y & y  :  AND operation (bit by bit)
```

Algorithm for the current default epoch designation 00:

```
TODX := TODR >> 12;
```

Algorithm for the future default epoch designation 08:

```
TODX := TODR >> 12;
if ((TODX >> 48) < 8)
  TODX := TODX + x'0010000000000000';
```

Common algorithm for any epoch designation:

```
TODX := TODR >> 12;
EPC  := EPD >> 4;
EPO  := EPD & x'0F';
if ((TODX >> 48) < EPO)
  EPC := EPC + 1;
TODX := TODX + EPC * x'0010000000000000';
```

Ranges of other TODR Epochs

EPOCH	Date and Time	
	from	to
00	1900-01-01 00:00:00.000000	2042-09-17 23:53:47.370495
01	1908-12-02 19:29:36.710656	2051-08-19 19:23:24.081151
02	1917-11-03 14:59:13.421312	2060-07-20 14:53:00.791807
03	1926-10-05 10:28:50.131968	2069-06-21 10:22:37.502463
04	1935-09-06 05:58:26.842624	2078-05-23 05:52:14.213119
05	1944-08-07 01:28:03.553280	2087-04-24 01:21:50.923775
06	1953-07-08 20:57:40.263936	2096-03-24 20:51:27.634431
07	1962-06-09 16:27:16.974592	2105-02-24 16:21:04.345087
08	1971-05-11 11:56:53.685248	2114-01-26 11:50:41.055743
09	1980-04-11 07:26:30.395904	2122-12-28 07:20:17.766399
0A	1989-03-13 02:56:07.106560	2131-11-29 02:49:54.477055
0B	1998-02-11 22.25:43.817216	2140-10-29 22:19:31.187711
0C	2007-01-13 17:55:20.527872	2149-09-30 17:49:07.898367
0D	2015-12-15 13:24:57.238528	2158-09-01 13:18:44.609023
0E	2024-11-15 08:54:33.949184	2167-08-03 08:48:21.319679
0F	2033-10-17 04:24:10.659840	2176-07-04 04:17:58.030335

18.1.3 TODR correction values

Time management TODR correction values have the followings aims:

1. The system time can be adapted to external timers or to a network time without modifying the TODR; this makes it possible to put the system time back.
2. It shifts the system time without affecting the TODR on summer time/winter time changeovers

It is particularly important not to put back the TODR while the system is running, since many algorithms in the operating system and application software are based on the unbroken ascending incrementation of the TODR.

Whenever a system time synchronization job is concluded, the currently valid TODR correction value is increased by this job value if the time is put back or decreased by this value if the time is put forward. Declared real time timers (STXIT or system-internal) are adapted accordingly. When the current system time is provided in a GTIME, GDATE or GETOD, the current TODR correction value is taken into account.

18.1.4 Synchronization of the system time with external timers or in a network

BS2000 provides a privileged and a POSIX “ADJUST TIME” interface for the adaptation of the system time (not the hardware TODR!) to external clocks. This interface is used for:

- synchronization in an NTP network
- synchronization in an XCS network
- Synchronization with the SVP time (SU /390)
- Synchronization with the carrier system time (SU x86)

This type of synchronization job causes the system time to be clocked more slowly or more quickly for a period which is four times as long as the required adaptation value. Here, it should also be noted that two successive GTIME calls for a task continue to return time values that increase in unbroken sequence.

During the adaptation phase, the system time that is being calculated is also corrected by the synchronization calculation that has already been performed:

$(-t - t_a) / (t_e - t_a) * d$

where:

t current time

t_a time of start of synchronization

t_e time of end of synchronization

d adaptation value

Only when synchronization is terminated is the TODR correction value corrected by the required adaptation value.

There is a defined sequence of priorities, which corresponds to the list below, between the individual instances which may request synchronization jobs. If two or more instances are active, then only jobs requested by the instance with the highest priority are performed.

The priorities have been defined in the following ascending order (1. = highest priority, 2. = second-highest priority, etc.):

1. BS2000 is a participant in an NTP network which has been configured with a trustworthy external timer (e.g. NTP server whose stratum value is not greater than 4).
2. The Management Unit, which synchronizes the SVP/carrier system time of the Server Units, is connected to a trustworthy external timer (NTP server whose stratum value is not greater than 4).
3. BS2000 is a participant in an XCS network.
4. BS2000 is a participant in an NTP network which has been configured without an external timer.
5. The Management Unit, which synchronizes the SVP/carrier system time of the Server Units, is either not connected to an external timer, or it is connected to an "untrusted" timer (stratum value of the NTP server is greater than 4).

i When time is synchronized on the basis of the SVP clock (items 2 and 5), it must be borne in mind that the SVP clock only has a resolution in the seconds range. As a result, deviations of ± 1 second can occur

between the SVP clock and the BS2000 time, and on SU /390 also between the SVP clock and the time of the Management Unit. In total this results in a maximum deviation of ± 2 seconds from the time source for the BS2000 system time on SU /390, and of ± 1 second on SU x86.

In an NTP or XCS network (items 1, 3 and 4) the transfer and synchronization of the time takes place with a considerably higher resolution. The deviation of the BS2000 system time from the authoritative time source is in the millisecond range.

18.1.5 GET-TIME subsystem

BS2000 provides a variety of interfaces for determining the current system time: GTIME, GDATE, GETOD. Runtime routines coded in high-level languages with analog semantics make use of the interfaces accordingly.

GTIME provides the system time (local time and UTC time) in binary, ISO4 and TODR/TODX format. It should be noted that the system time in TODR format is different from any simultaneously determined STCK value because of the TODR correction values.

The server for the long-term, guaranteed GTIME interface is implemented as a BS2000 OSD/BC component and as a nonprivileged TU GET-TIME subsystem in order to provide SVC-free, high-performance processing in the event of calls from TU. For the provision of the system time in TODR format, the GTIME call from TU has a path length of less than 200 commands. The use of TASKDATE (see "TASKDATE: testing in simulated time") or the determination of the XCS time (see "Synchronization in an XCS network") results in processing in the TPR. The subsystem still contains the server for CTIME, the macro interface for arithmetic processing with time stamps and for the conversion of time stamps in terms of format and time base. The basic data for the two services (TODR correction values, data relating to an active synchronization operation as well as summer time/winter time changeover points) is stored in the system address space for the operating system module and in TU write-protected form in the subsystem.

The GET-TIME subsystem is mandatory, is started automatically before “System Ready” and is present in two versions controlled via the explicit selection of one of the four supplied subsystem catalogs:

- loaded below the 16-Mb boundary (default) if XS-incompatible programs are used in the system
- loaded above the 16-Mb boundary (to reduce the load on this lower address space) if only XS-compatible programs use the subsystem

In addition to the GET-TIME subsystem, the GET-TIMX subsystem which is used only within the operating system also exists on SU x86.

GET-TIME startup parameters

The startup parameter file contains a section for GET-TIME (GTIME parameter record). This describes the following information for one or more time zones:

- information about the time zone (i.e. difference from UTC time)
- summer time information (difference between summer time and winter time, elapsed and future time changeover points)

This information is needed for

- time conversions from local time / UTC time (and vice versa) for any time stamps thanks to the CTIME function. The more time changeover points there are available, the greater is the time range in which CTIME can perform precise conversion of local time / UTC.
- identification of summer time / winter time on startup
- conversion of the system time on the next summer time / winter time change

On SU /390 the time zone of the server location should be set by means of an SVP frame. On SU x86 this information is known in the carrier system. The startup routine ascertains this date and uses it to select the corresponding time zone parameter section with priority.

For more information on the GTIME startup parameter, and in particular on the scope of the provided time changeover points, see section “System time control (GTIME)”.

18.2 SVP time

BS2000 considers SVP to possess its own clock, which can be polled by the operating system via the privileged Store Real Clock interface. The SVP clock must always run in local time. Store Real Clock provides the SVP time with a resolution in seconds. The supplementary information provided by Store Real Clock depends on the architecture of the Server Unit:

- Time zone
- Display: synchronized with external timer

Server Unit	Supplementary information
SU /390	Time zone, Display: • Synchronization by the MU active / failed • MU synchronized with external timer
SU x86	Time zone, Display: • MU synchronized with external timer

Depending on the architecture of the Server Unit, this clock may be present as a hardware clock or may be emulated by X2000, but in both cases it runs independently of the TODR.

Server Unit	SVP clock integration mode
SU /390	Autonomous SVP clock exists, synchronized by the MU. The MU can be synchronized with an external time or with other systems by means of NTP.
SU x86	SVP clock is emulated by X2000 with the help of the carrier system time. The carrier system time is synchronized by the Management Unit.

Notes for use under VM2000

On SU /390 the “Store Real Clock” of a guest system is emulated by the Hypervisor under VM2000. The monitor system’s system time is generally used to achieve this. The following exceptions exist which result in a Store Real Clock command being issued:

- The monitor system was started using a time entered explicitly by the operator.
- The monitor system is part of an XCS network.
- The monitor system is part of an NTP network which has not be configured with an external timer.
- The monitor system cannot be reached at the time the command is executed (e.g. when the monitor system is restarted).

On SU x86 the “Store Real Clock” is also emulated under VM2000 like a native of X2000.

18.3 Initializing the system time

The system time is initialized during IPL by the TODR being loaded.

Time sources for system time initialization

The SVP time determined by the “Store Real Clock” interface is used as the time source for initialization.

The following special case also exists: The system is started in DIALOG mode with the UNLOCK option and the operator makes an explicit time specification.

The employed time source is logged as an insert in startup message `NSI1163`.

The setting for summer time/winter time is initialized by means of the SVP information. If the information is not available in SVP or if the time has been specified manually (by means of the UNLOCK option), it is not possible to unambiguously recognize a duplicated hour on summer time/winter time changeover. In this case, startup assumes winter time.

For information on initializing the system time, see also section "Determining the timesystem initialization".

18.4 Synchronizing the system time

- Synchronization with the SVP clock on SU /390
- Synchronization with the carrier system time on SU x86
- synchronization in an NTP network
- Synchronization in an XCS network
- General recommendation for the XCS network
- Synchronization outputs in /SHOW-SYSTEM-INFORMATION

18.4.1 Synchronization with the SVP clock on SU /390

The Management Unit synchronizes the SVP clock to its own time.

If no other synchronization instance is present in BS2000, on SU /390 the system time is synchronized with the SVP clock, and thus implicitly with the time of the Management Unit. This is achieved by performing a periodic time comparison in BS2000 between the SVP time and the BS2000 system time, and the BS2000 system time is then adapted as necessary.

> The corresponding information can be found in the SYSTEM-TIME-PARAMETER section of the SHOW-SYSTEM-INFORMATION command's output: `SYNCHRONIZATION=SKP-X`.

The MU can be connected to an external timer. The MU shows the BS2000 that an external timer exists if the NTP time has a *stratum* <= 4.

> The corresponding information can be found in the SYSTEM-TIME-PARAMETER section of the SHOW-SYSTEM-INFORMATION command's output: `SYNCHRONIZATION=SERVER-CONN-EXT-REF`.

In addition, in BS2000 you can configure an NTP network with an external timer which then constitutes the highest synchronization instance (see section "synchronization in an NTP network".

18.4.2 Synchronization with the carrier system time on SU x86

On SU x86 the system time is automatically synchronized with the SVP clock, in other words with the time of the carrier system, if no other synchronization instance is present in BS2000. The carrier system receives the time of the Management Unit using NTP.

> The corresponding information can be found in the SYSTEM-TIME-PARAMETER section of the SHOW-SYSTEM-INFORMATION command's output: `SYNCHRONIZATION=X2000`.

The MU can be connected to an external timer. These external timers are visible in the BS2000 time only if their quality is retained through to the carrier system (i.e. the *stratum* of the NTP server in the carrier system is <= 4).

> The corresponding information can be found in the SYSTEM-TIME-PARAMETER section of the SHOW-SYSTEM-INFORMATION command's output: `SYNCHRONIZATION=SERVER-CONN-EXT-REF`.

18.4.3 synchronization in an NTP network

The software system NTP (Network Time Protocol) ensures that the systems connected to a network of this type use a common time. BS2000 can be a participant in such a network via POSIX.
Each network participant queries the time of selected systems, taking into account transmission times and calculates a network time with which it synchronizes the system time of its own host. To do this, NTP uses the POSIX interface version of ADJUST-TIME. Servers with external timers can be declared as so-called “time servers” which do not adapt their system times to a network time, but vice versa: their time is used by the other network participants as the exclusive source for determining the network time.

The two following NTP versions in terms of prioritization can be used as the synchronization instance for the system time in BS2000. They are distinguished by means of the NTP time quality *stratum*.

- STD usage
  If the NTP time has a *stratum* > 4, the BS2000 time management assumes that the NTP server does not have a reliable time source and assigns it a correspondingly low priority. If a higher-priority time source (timer connected to the Management Unit or XCS network, see section "Synchronization of the system time with external timers or in a network") exists, the BS2000 time management does not permit any synchronization requests from NTP. Only if no higher-priority time source exists is the BS2000 system time synchronized with the NTP time.

  > The corresponding information can be found in the SYSTEM-TIME-PARAMETER section of the SHOW-SYSTEM-INFORMATION command’s output: `SYNCHRONIZATION=DCE/NTP`.

- NTP network with a connection to an external timer
  If the NTP time has a *stratum* <= 4, the BS2000 time management assumes that the NTP server has a reliable time source and assigns it highest priority. The BS2000 system time is synchronized with the NTP time. The precision of the NTP time lies in the millisecond range.

  > The corresponding information can be found in the SYSTEM-TIME-PARAMETER section of the SHOW-SYSTEM-INFORMATION command’s output: `SYNCHRONIZATION=BS2-CONN-EXT-REF`.

Summer time/winter time conversions are performed for the system time which is controlled by NTP. NTP can continue running after such a time conversion.

Notes for use under VM2000

A VM2000 guest system (including the VM2000 monitor system) can participate in a NTP network independently of the remaining VMs of a server running under VM2000. If the monitor system is a participant in such a network, on SU /390 the emulation of the SVP time (“Store Real Clock”) supplied to the guest systems depends on the NTP variant:

- In the case of NTP with a connection to an external timer, the guest systems are provided with the monitor time. Participation of the monitor VM in such a network also ensures that the network time is supplied for all guest systems, but with a precisions of ± 1 second. If equally great precision is required, the guest system must function as an explicit participant in the network.
- In the case of an NTP standard configuration in the monitor system, the guest systems are supplied with the real SVP time. Thus, if required, all guest systems must be participants in the network.

On SU x86 Dom0 with Linux/X2000 participates in the NTP network with a connection to the MU. This guarantees the NTP network time for all guest systems, including the monitor system, but with a precision of ± 1 second. If greater precision is required, the system must act as an explicit network participant.

18.4.4 Synchronization in an XCS network

An XCS network is a variant of the HIPLEX-MSCF functionality which is available in BS2000. In particular, it permits the use of distributed applications and comprises the following functionality:

- automatic failure detection and reconfiguration
- lock management for distributed applications (DLM)
- uniform time throughout the network

The synchronization of the system time in the network is performed by the XCS TIME subsystem, which is initialized on XCS startup and is based on the XCS time, which represents a second time alongside the system time. The XCS time is supplied via GTIME (with SVC however) in TODR format and is based on UTC.

XCS time

Alongside a lock function, distributed applications often require a network-wide, monotonic time, e.g. to permit the correct chronological compilation of locally generated logging files. XCS time guarantees this monotonic time in the event of two successive queries to any systems, provided that the time is determined under the same DLM lock:

System A	`GET-LOCK (Lock-Name=X1,exclusive)` for data block x
	`GTIME(XCS-TIME)`
	`RELEASE-LOCK(Lock-Name=X1)`
System B:	`GET-LOCK (Lock-Name=X1,exclusive)` for the same data block x
	`GTIME(XCS-TIME)`
	`RELEASE-LOCK(Lock-Name=X1)`

The two GTIME calls provide genuine monotonic time values in which the logging information for manipulations performed to data block x may be chronologically different.

Each system maintains its own XCS time as follows:

1. In the case of exclusive lock allocation by system A, which is the administrator of a DLM lock, to a second system B, the maximum from the XCS time and system time, incremented by the value “one microsecond”, is sent to the lock requester together with an acknowledgment of allocation.
2. System B receives the lock allocation from system A

   The maximum of the time on system A supplied with the lock allocation and the XCS time on system B, together with its current system time, forms the new current XCS time for system B.
3. GTIME call for the reception of the XCS time by an application

   The maximum, incremented by the value “one microsecond”, of the current system time and the XCS time becomes the current XCS time and is made available to the caller.

This procedure keeps the XCS time of all the participating systems close to the system time of the system with the most advanced time.

> The corresponding information can be found in the SYSTEM-TIME-PARAMETER section of the SHOW-SYSTEM-INFORMATION command’s output: `SYNCHRONIZATION=XCS` .

To keep the system times as close together as possible (e.g. to keep the chronology of decentralized system logging data as correct as possible), the XCS-TIME subsystem synchronizes the system time with the XCS time of the system in every participating system. The system time and the XCS time of the individual servers are therefore only approximately equal; no maximum divergence can be guaranteed.

No server can be accepted into a network until its system time had been adapted to that of the partners which are already active. If the time difference on entry is greater than 15 minutes, entry is denied.

External timers and XCS networks

An XCS network can also be synchronized with an external timer.

> **i** You are recommended to connect an external time either to all participants of the XCS network or only to one of them.
>
> *Explanation*
> If not all participants of the network are connected to this timer, the connected participants monitor the times of the participants who are not connected and, when required, send time adjustment requests to them.
> On the recipient side, adjustment requests from different senders can appear contradictory owing to different runtimes. This can lead to increased CPU time requirements and to time deviations.

Notes for use under VM2000

A VM2000 guest system (including the monitor system) can participate in an XCS network independently of the remaining VMs of the server. If the monitor system is a participant in such a network, the emulation of the SVP time (Store Real Clock) supplied to the guest systems is based on the actual SVP time and not on the NTP-synchronized system time of the monitor system.

18.4.5 General recommendation for the XCS network

The MUs of all servers on which systems involved in an XCS network run (witho r without external timers) should all be included in an NTP network.

18.4.6 Synchronization outputs in /SHOW-SYSTEM-INFORMATION

The table below shows the possible outputs, in ascending order of priority, in the SYSTEM-TIME-PARAMETER section of the SHOW-SYSTEM-INFORMATION command for the value of SYNCHRONIZATION on both server types. If more than one timer is present, the highest priority amongst them determines the value of SYNCHRONIZATION.

SYNCHRONIZATION =	**SU /390**	**SU x86**
*NONE	In the event of an error: SVP clock currently not available or difference between system time and timer too great to permit synchronization	In the event of an error: SVP clock currently not available or difference between system time and timer too great to permit synchronization
X2000	Not applicable	Default value (without external synchronization)
SKP-X	Default value (without external synchronization)	Not applicable
DCE/NTP	Synchronization by NTP network (in BS2000)	Synchronization by NTP network (in BS2000)
XCS	Synchronization by XCS network	Synchronization by XCS network
SERVER-CONN-EXT-REF	External timer via the MU or (for VM2000 guest systems) connected in the monitor system	External timer via the MU or (for VM2000 guest systems) connected in the monitor system
BS2-CONN-EXT-REF	External timer connected via NTP (BS2000)	External timer connected via NTP (BS2000)

18.5 Interrupt-free summer time/winter time changeover

It is possible to perform the time change without interrupting to the system, i.e. the system continues to operate across the time change.

The system time is adapted accordingly on both time change dates: i.e. in the Central European Time zone, the system time is advanced by 1 hour in the spring (usually during the last weekend of March) and put back by 1 hour in the autumn (during the last weekend of October).

This time change involves a single step and is therefore different from the other time synchronization operations: the TODR is not modified and continues to run in monotonic mode. Instead, the TODR correction value is manipulated in order to yield the adjusted system time. This conversion is triggered when the system task TIME sets a timer with the next conversion date on startup (and following every changeover).

Changeover dates are declared in the GTIME section of the startup parameter file. If the future changeover dates are missing in this file, a corresponding warning is output, unless the SINGLE keyword is specified after the last changeover date. Time reset points can also be managed during ongoing operation using the ADD-/ MODIFY-/REMOVE-/SHOW-CHANGE-DATE commands. However, changes to time reset points by means of commands must be entered manually in the parameter file if they are to apply in the next session.

For regions currently using summer/winter time changeover, but are planning to move to a single season time in future, the keyword SINGLE should be specified after the last changeover date in the system parameter file.

If the application software has not yet been fully converted to GTIME by means of STCK time determination, the system time changeover can be prevented by specifying DIFF=0:00 in the parameter service. The system must then be restarted.

Notes

- The CTIME interface also issues warnings for input or output time stamps in the duplicated or skipped hour. The system time supplied by GTIME (local time) is shifted accordingly at changeover time. Applications that work with or compare time stamps must be able to process negative shifts in the local time. A procedural change to UTC time or the maintenance of a summer time / winter time indicator, which is also provided in GTIME/CTIME, is recommended. The current time should not be determined by directly accessing the TODR.
- If user programs continue to use the STCK command and their own conversion algorithms for time stamps instead of using GTIME, then they are still dependent on the local time in the TODR. In this case, the system must be reloaded as previously on time changeover.
- If all the user programs have already been converted to GTIME for time polling or if they only calculate STCK values to determine differences, then interrupt-free time changeover can be used.

Job management behavior on changeover

- Changeover from winter time to summer time
  - Scheduled jobs

    As a result of the conversion of the start time to UTC and the subsequent back-conversion, the start time is delayed by one hour.

    Example: ENTER <file>,START=AT(02:30) results in the job being started at 03:30 (LTI). This start time is inherited by successor jobs if the job is the first of multiple repeat jobs.
  - Repeat jobs with the period DAILY/WEEKLY

    Any job repetitions which have to start in this time period are omitted. Instead, the next successor is used and the next start takes place 1 day or 1 week later.

- Repeat jobs with hour/minute periods

  The start times of job repetitions are calculated simply by adding the minutes of the period to the last start time (in JMS time, UTC); therefore this period is simply skipped.

  Example start times:... 01:45 , 01:55 , 03:05 , 03:15 ... (LTI)

- Changeover from summer to winter time
  - Scheduled jobs

    The internal conversion of a start time of a scheduled job in UTC assumes winter time; the start is therefore correspondingly performed in the second hour of the period.

  - Repeat jobs with the period DAILY/WEEKLY

    Job repetitions which have to start during this period are handled in the same way i.e. they start in the second hour.

  - Repeat jobs with hour/minute periods

    The start times of job repetitions are calculated by simply adding the minutes of the period to the last start time (in JMS time, i.e. UTC); therefore they may run in both of the two hours.

Notes on AVAS behavior on time changeovers

- Changeover from winter time to summer time

  Nets planned for the period 2.00 - 3.00 are suddenly started in accordance with the delay settings (NET-DELAY-SOLUTION) and run normally. Shortly after 3.00 am there is therefore increased AVAS activity since all the nets due to be started in the lost hour are immediately processed.

- Changeover from summer to winter time

  The nets which have already been started (since 2.00 am) continue to run normally since they are started on the basis of the predecessor/successor principle independently of the time. When the time is put back, no nets are started until 3.00 am. Time stamp-related problems may affect sorting in the journal file since the start times of successive jobs may lie before those of the predecessors.

18.6 System start with special system time

System time can be initialized with any value. To do this, the system start must be performed in DIALOG mode with the IPL option UNLOCK. By responding to the output messages the operator can specify any time. The SVP time remains unaffected by this.

Under VM2000, it is possible to run any VM with a fictitious system time and correspondingly offset UTC time. However, this guest system must always be started in DIALOG mode with the IPL option UNLOCK.

No synchronization is then performed with any external timer which may be present when the manually set time is more than 15 minutes different from the SVP time. The system time is clocked with the hardware TODR's quality. Since the modified system time is also used for time stamps and the system catalog, then, for example, if the system time has been put forward and the pubset is then reused at the ACTUAL time, file access may be rejected because the EXPIRATION date has not yet arrived. Application-specific data with time stamps may also be unpredictably inconsistent after such dual manipulations and may result in undefined program behavior (in particular in the case of databases).

The permanent operation of a system with a foreign time zone represents a special case. Conversion to another time zone in the monitor system or in the native system works only if the new time zone is set in the SVP resp. carrier system and startup is then performed with IMPL.
Otherwise (i.e. in particular in the case of a foreign time zone for a guest system) the system may not use the time received from the SVP at startup via Store Real Clock as the system time. The correct time for the required time zone can only be obtained by being input manually. To permit this, every startup of this system must be implemented as a dialog startup with the UNLOCK option.
The following must also be taken into account:

- If the GTIME parameter file only contains information on the required time zone, the CTIME macro cannot carry out any conversions for foreign time zones during ongoing operation.
- If the GTIME parameter file contains information on multiple time zones, it is not possible to recognize which of these is the correct one after the system time has been entered manually. The time initialization routine then engages in a message dialog with the operator which enables a zone to be entered but not a difference between summer time/winter time or a changeover points for this zone.
  The zone entered is stored in the SVL , but effective operation is not possible with such a system. Only at the next startup – naturally in DIALOG UNLOCK mode – is the zone stored in the SVL used in place of the SVP zone, thus permitting evaluation of the converted parameter file.

18.7 TASKDATE: testing in simulated time

The TASKDATE tool was developed to test applications that are sensitive to the date and time.

It permits a task with nonprivileged status (TU) to run in offset time. When this is done, all time queries from TU (GTIME, GDATE, GETOD as well as high-level language time services - not STCK) are responded to by the system with an offset time which has previously been defined by means of the SET-TASK-CLOCK command. The simulated time can optionally be inherited by successor tasks (ENTER). The system, task catalog entries and job start times continue to be based on the system time. The tool is implemented as a DSSM subsystem as well as an extension to the BS2000 time management function, JMS, SDF P, JV and openUTM. The tool can be started with system running using a supplied start procedure.

19 Appendix

The appendix contains the following lists and tables:

- Character set for I/O operations via the console
- Overview of test privileges

Special lists which are no longer included in this Appendix are contained elsewhere in this manual or in other manuals:

- Class 2 system parameters (in short: system parameters), see the description of the SHOW-SYSTEM-PARAMETERS command in the “Commands” manual [27]
- Device type table, volume type table, organization of the disk storages (see the “System Installation” manual [55])
- Overview of commands with the associated privileges, provided they are part of BS2000 OSD/BC and are described in the “Commands” manual, see the “Commands” manual [27]
- Overview of operator commands and their routing codes, see section "Functional areas and their allocation to consoles"

19.1 Character set for I/O operations via the console

All characters which can be **entered** via a particular console are supported when entered at that console. They are converted internally into EDCDIC.DF.04 by the console's character code.

For console **outputs**, the character code is converted into a character code which can be represented on the console being used. All non-printable characters, except X'15', are converted into blanks X'40'.

19.2 Overview of test privileges

The test privileges control the executability of software and hardware diagnostic activities in BS2000. The main user is the debugger AID, see the “AID” manual [2].

The test privileges are divided into read and write privileges. A user’s write privilege may not be greater than his/her read privilege. Read and write privileges are defined hierarchically from 1 (low) through 9 (high). A privilege with the value n implicitly includes the access rights of the lower privilege levels 1 through n-1.

The maximum test privileges for a user ID are defined using the privileged commands ADD-USER and MODIFY-USER ATTRIBUTES. System-wide maximum values for the read and write privileges can be defined using the system parameters RDTESTPR and WRTESTPR. The task-specific setting is defined using the MODIFY-TEST-OPTIONS command.

Information on the current test privileges is supplied by the SHOW-TEST-OPTIONS command.

The tables below provide an overview of the test privileges for AID and for other software diagnostic products.

19.2.1 Test privileges for AID

If data is loaded (dynamically) from a read-protected file and if the read password was not specified, an “execute only” situation exists. Testing with AID is not possible in an “execute-only” situation, regardless of the test privileges set.

Value	Characteristics
1	• Access to the pages in the user’s own user address space[1)] and in the system address space[2)] which are not secret pages[3)] are readable with the access rights of the normal user • Access to the user PCBs of the user’s own task
2	• Also access to the user TCB of the user’s own task
3	• Also access to the secret pages in the user’s own user address space which are readable with the access rights of the normal user
4	Not assigned
5	Not assigned
6	• Also access to the pages in the user’s own user address space which are not secret pages which are readableonly with the access rights of the operating system • Also access to the system PCBs of the user’s own task and to the XVT
7	Not assigned
8	• Also access to all pages in the system address space, all secret pages and all pages of other tasks • Also access to the user PCBs, system PCBs and TCBs of other tasks
9	Not assigned

Table 51: Read test privileges for AID read accesses

Value	Characteristics
1	• Access to the pages in the user’s own user address space[1)] which are not secret pages[3)] and which can be written with the access rights of the normal user • Also access to the user’s local task pages in the user’s own program space which are not secret pages and are “read only”[4)] with the access rights of the normal user • Access to the user PCBs of the user’s own task

2	Not assigned

3	• Also access to the secret pages in the user’s own user address space which can be written with the access rights of the normal user • Also access to the user’s local task secret pages in the user’s own program space which are “read only” with the access rights of the normal user
4	Not assigned
5	Not assigned
6	• Also access to the pages in the user’s own user address space which are not secret pages and can be written only with the access rights of the operating system
7	Not assigned
8	• Also access to all pages in the system address space[2)], all secret pages and all pages of other tasks • Also access to the XVT and to the user PCBs, system PCBs and TCBs of all tasks
9	Not assigned

Table 52: Write test privileges for AID write accesses

Key

[1)] Program space and data spaces of the user
[2)] System space and data spaces of the system
[3)] Pages protected against diagnostic accesses
[4)] The page attribute “read only” is ignored by AID in this case.

Note on OWN-UID-DEBUGGING

The “static AID test with low test privilege” can be permitted for other taks which run under the user’s own user ID by means of the OWN-UID-DEBUGGING operand in the MODIFY-TEST-OPTIONS command. If this is the case, the same privileges as for accessing your own user address space are sufficient for the read accesses to this task’s user address space.

A write privilege of at least 2 is required for write accesses to this task’s user address space; in addition, the same test privileges apply as for write accesses to the user address space of the user’s own task. In the case of write accesses to these tasks’ memory pool pages, write privilege 8 is always required.

19.2.2 Test privileges for other software diagnostic products

Value	Product	Characteristics
3	CDUMP	A system dump may be taken under a normal user ID.
8	ANITA	The active system may be diagnosed with DAMP and other user programs which use ANITA.

Table 53: Read test privileges for software diagnostic products

20 Related publications

You will find the manuals on the internet at *http://bs2manuals.ts.fujitsu.com*. You can order printed versions of manuals which are displayed with the order number.

[1] **BS2000**
Accounting Records
User Guide

[2] **AID**
Advanced Interactive Debugger
Core Manual

[3] **ARCHIVE**
User Guide

[4] **openNet Server**
BCAM
User Guide

[5] **BLSSERV**
Dynamic Binder Loader / Starter in BS2000
User Guide

[6] **BINDER**
Binder in BS2000
User Guide

[7] **CALENDAR**
Kalender erstellen und bearbeiten
User Guide

[8] **ETERNUS CS**
Operating and Administering the ETERNUS CS
User Guide

[9] **CRYPT**
Security with Cryptography
User Guide

[10] **DAB** (BS2000)
Disk Access Buffer
User Guide

[11] **DCAM**
COBOL Calls
User Guide

[12] **DCAM**
Macros
User Guide

[13] **DCAM**
Program Interfaces
Reference Manual

[14] **BS2000 OS DX**
Diagnostics Handbook
User Guide

[15] **BS2000 OS DX**
Utility Routines
User Guide

[16] **Distributed Print Services**
Printing in Computer Networks
User Guide

[17] **DRV** (BS2000)
Dual Recording by Volume
User Guide

[18] **DSSM/SSCM**
Subsystem Management in BS2000
User Guide

[19] **BS2000 OS DX**
Introductory Guide to DMS
User Guide

[20] **BS2000 OS DX**
DMS Macros
User Guide

[21] **ELSA**
Error Logging System Analysis
User Guide

[22] **FDDRL**
User Guide

[23] **openFT (BS2000)**
Managed File Transfer in the Open World
User Guide

[24] **HSMS** (BS2000)
Hierarchical Storage Management System
User Guide

[25] **IMON**
Installation Monitor
User Guide

[26]

JV
Job Variables
User Guide

[27] **BS2000 OS DX**
Commands
User Guide

[28] **BS2000**
System-Managed Storage
User Guide

[29] **LMS**
SDF Format
User Guide

[30] **BS2000 OS DX**
Executive Macros
User Guide

[31] **MAREN**
Tape Management in BS2000
User Guide

[32] **HIPLEX MSCF**
BS2000 Processor Networks
User Guide

[33] **OMNIS/OMNIS-MENU**
Functions and Commands
User Guide

[34] **OMNIS/OMNIS-MENU**
Administration and Programming
User Guide

[35] **BS2000**
Operator Commands (ISP Format)
User Guide

[36] **PCS**
Performance Control Subsystem
User Guide

[37] **BS2000**
Performance Handbook
User Guide

[38] **POSIX** (BS2000)
Commands
User Guide

[39] **POSIX**
POSIX Basics for Users and System Administrators

User Guide

[40] **PROP-XT**
Programmed Operating with SDF-P
Product Manual

[41] **ROBAR**
Controlling MTC Archive Systems
User Guide

[42] **PRM**
User Guide

[43] **SDF**
SDF Dialog Interface
User Guide

[44] **SDF-A**
User Guide

[45] **SDF-P**
Programming in the Command Language
User Guide

[46] **SECOS**
Security Control System - Access Control
User Guide

[47] **SECOS**
Security Control System - Audit
User Guide

[48] **SHC-OSD**
Storage Management for BS2000
User Guide

[49] **openSM2**
Software Monitor
User Guide

[50] **SM2-PA**
SM2 Program Analyzer
User Guide

[51] **SNS**
SPOOL Notification Service
User Guide

[52] **SPACEOPT**
Disk Optimization and Reorganization
User Guide

[53] **SPOOL**
User Guide

[54] **SPSERVE**
User Guide

[55] **BS2000 OS DX**
System Installation
User Guide

[56] **BS2000 OS DX**
System Exits
User Guide

[57] **FUJITSU Server BS2000 SE Series**
Operation and Administration
User Guide

[58] **Unicode in BS2000**
Unicode Introduction

[59] **openUTM** (BS2000, UNIX, Windows)
Generating Applications
User Guide

[60] **VM2000**
Virtual Machine System
User Guide