1 Preface

1.1 Brief product description

System-managed storage is a data administration concept. It supports the distribution of tasks between systems support staff, users and the system, thus permitting highly efficient data administration. To make this possible, BS2000/OSD-BC provides extensive functionality in its pubset management and in its software configuration products, in particular HSMS. This functionality is described in detail in the associated manuals. The purpose of this manual is to clarify the global concept, the significance of the individual functions within the global concept, their interaction and interdependence, thus guiding systems support staff through the wide and varied world of SMS functionality.

1.2 Target group

This manual is intended for BS2000/OSD systems support staff.

1.3 Summary of contents

The manual consists of 11 chapters.

Chapter 1: Preface

describes the purpose of the manual, defines the target group and explains the structure of the manual.

Chapter 2: Basic principles of system-managed storage

presents the underlying idea on which system-managed storage is based.

Chapter 3: Overview of system-managed storage in BS2000

provides an overview of how the system-managed storage concept is implemented in BS2000.

Chapter 4: Characteristics of an SM pubset

describes the structure and characteristics of an SM pubset. SM pubsets are of crucial importance for the implementation of system-managed storage in BS2000.

Chapter 5: Setting up an SM pubset

describes how systems support staff can set up new SM pubsets.

Chapter 6: Generating an SM pubset from existing SF pubsets

describes how existing SF pubsets can be transformed into SM pubsets without any data loss. The chapter starts by dealing with the underlying problems which must be considered during SF to SM pubset migration. It then discusses general procedures and finally outlines the necessary steps for certain application scenarios.

Chapter 7: Information functions for SM pubsets

lists the information functions that are of relevance to SM pubsets.

Chapter 8: Modifying SM pubset characteristics

describes the options available for modifying the configuration and the characteristics of an existing SM pubset. The chapter pays special attention to the enlargement or reduction of the configuration of an SM pubset during active pubset operation. It also briefly describes the possibilities for reconfiguring SF pubsets.

Chapter 9: Managing file life cycles

describes the interfaces between the systems support staff and the user which are available for SM pubsets and which make it possible to perform the various file management operations which are necessary during a file's life cycle cheaply and efficiently (file creation, migration to background levels, recall to processing level, backups, recall). It also describes how systems support staff can implement their own storage utilization strategies and presents the options available for the automation of data administration.

It also shows how users who do not wish to use system mechanisms (e.g. users of physical allocation functions) can be integrated into SM pubsets and how SM pubsets therefore represent an attractive alternative to private disks.

Chapter 10: Pubset monitoring and pubset maintenance

outlines various tasks associated with pubset maintenance and indicates how the SM pubset functions available to systems support staff help them perform these tasks.

Chapter 11: Controlling resources in SM pubsets

summarizes the ways in which systems support staff can control and restrict the pubset resources demanded by individual users.

A list of related publications and an index follow at the back of the manual.

Related publications are referred to in short form throughout the manual. The full title of each appears in the list at the back of the manual.

1.4 Readme file

Functional changes and additions to the current BS2000/OSD version for this manual are described in the corresponding readme file.
You will find the readme file for BS2000/OSD V6.0 on your BS2000 system. Its name is SYSRME.BS2CP.150.E. To find out the user ID under which the readme file is located, please ask systems support. You can also obtain the full path name by using the following command:

```
/SHOW-INSTALLATION-PATH INSTALLATION-UNIT=BS2CP,LOGICAL-ID=SYSRME.E
```

You can view the readme file by means of the SHOW-FILE command or in an editor, or you can print it out on a standard printer by means of the following command:

```
/PRINT-DOCUMENT SYSRME.BS2CP.150.E,LINE-SPACING=*BY-EBCDIC-CONTROL
```

2 Basic principles of system-managed storage

System-managed storage (SMS) is a file management concept that provides for role distribution between systems support, the user and the system. In the case of large configurations of volumes, in particular, it provides the basis for the efficient administration and usage of storage resources. At the logical level, users use **file services** for this purpose. The physical implementation of these is largely concealed from them. The steps required to implement these services are carried out by systems support. This field of activity is known as **data administration**.The system provides systems support with extensive support when performing these tasks.

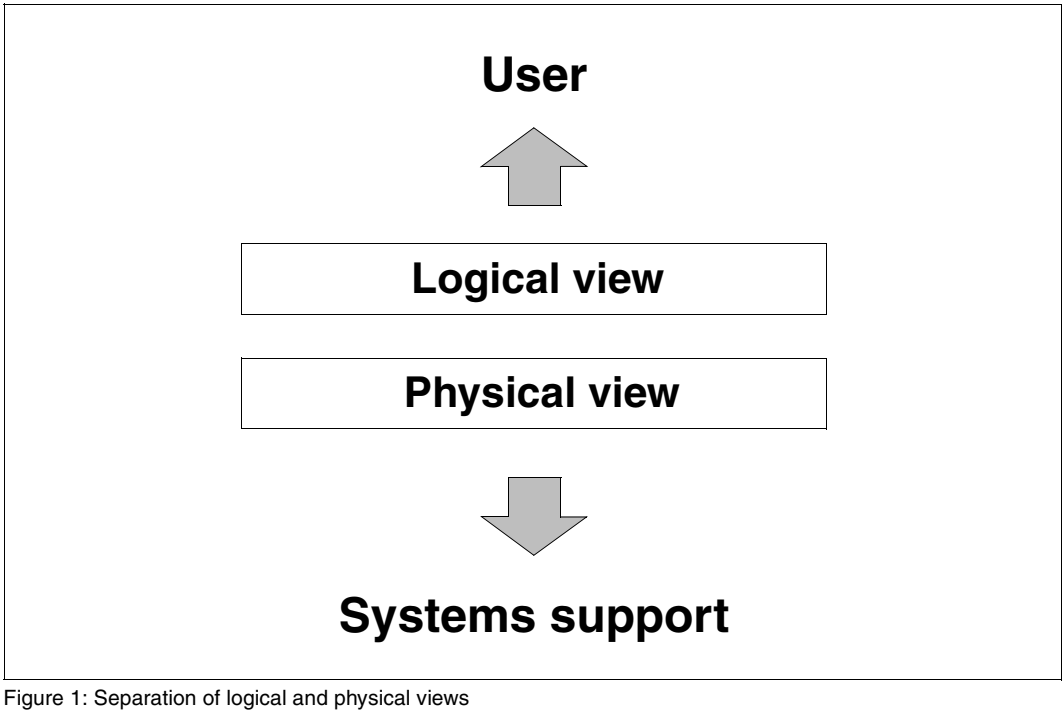


Figure 1: Separation of logical and physical views

All file administration operations are affected by the separation of logical and physical views:

- creating a file on the medium which is best suited to the files' processing requirements
- relocating the file to a more suitable processing medium if the data processing requirements for the file change or if new media are used
- the migration of a file to a low-cost medium in inactive phases during which the file is not accessed (Migrate) and its retrieval to the reading level when it is required for processing again (Recall)
- the creation and management of backups (Backup) and the restoration of backups (Restore)
- the removal of files and backup copies which are no longer required.

The various file management operations are illustrated in figure 2.

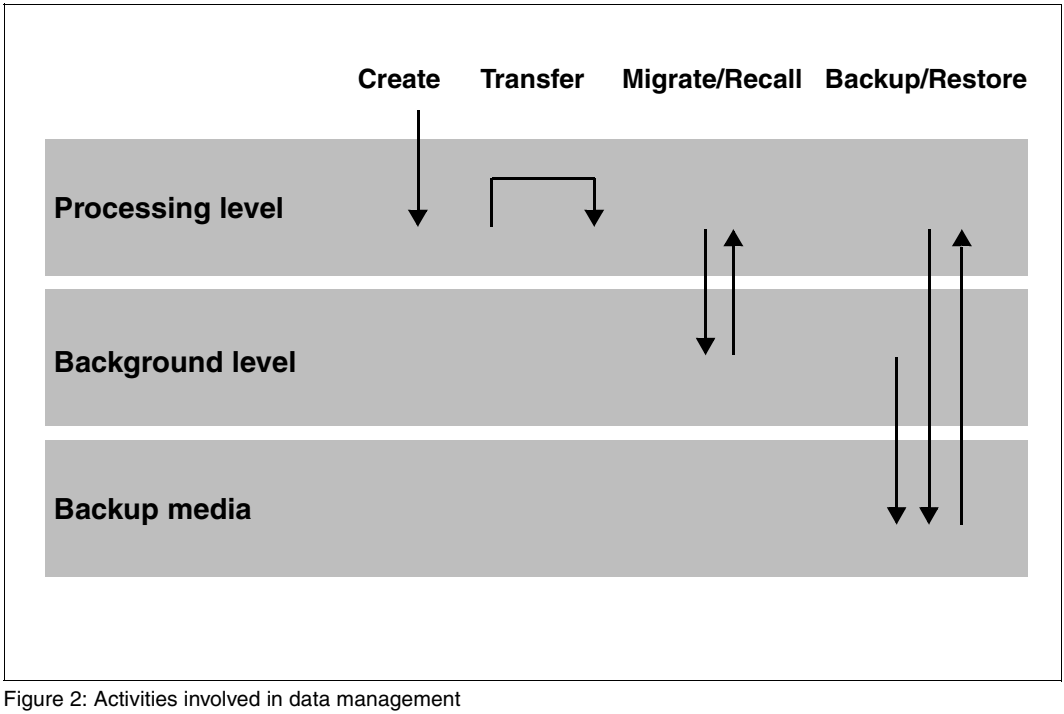


Figure 2: Activities involved in data management

The file services which relate to a file's processing attributes and which are associated with the location of the file within the processing level are referred to as **storage services**. They permit users, for example, to specify the performance requirements for a given file. The file services which allow users to migrate files to background levels, create backups and remove files and backup copies which are no longer required are known as **HSMS management services**.
The distribution of tasks between systems support staff, user and system which system-managed storage makes possible provides the basis for efficient, low-cost file management and the associated economic benefits:

- Simplified user interface
  Since users specify their own requirements regarding file management at the logical level but do not need to worry about their physical implementation they are free to concentrate on their real work. There is no need for users to learn about areas which are generally foreign to them, for example knowledge of the attributes of different storage media in order to specify the most suitable location for a file. This also helps attenuate the problem that users are unable to optimize data administration and ensure the efficient utilization of resources because of insufficient knowledge.

- Efficient data administration
  Data administration is (largely) performed by specialized systems support staff. In performing their tasks, systems support staff can attempt to find optimized solutions to specific user requirements while taking global considerations into account. This permits both efficient organization of data administration and efficient use of the available volumes. A precondition for centralized data administration is that the scope for user intervention can be restricted. This is made possible by the logical interface which allows users to formulate their wishes while screening them from data administration operations.

- Flexible adaptation of configurations
  Since the user is screened from the requirements of data administration, it is possible to modify volume configuration, data distribution across volumes and data management procedures without users having to perform any modifications themselves. Since users do not know the precise location of files and since this location is not required in any command procedures, it is possible to relocate files to other volumes without invalidating any command procedures. This forms a basis on which the data distribution across the various volumes can be dynamically modified. In conjunction with the possibility, provided for in SMS, to modify volume configuration during device operation, this provides a flexible way of adapting data distribution to changing requirements and workloads. This also makes it possible to replace existing storage media with more modern storage media which can be used in conjunction with existing files without users having to perform any adaptations. Because there are no problems associated with file transfer, the ongoing reorganization of media during system operation is also possible.

- System support for systems support staff
  Data administration requirements in complex configurations can be extremely high. In order to limit the work required, it is necessary to automate as many operations as possible and especially the routine activities performed by systems support. The system can support systems support in a number of ways:

  a) through mechanisms which are integrated into the system and which can be adapted to specific application environments when suitably parameterized

  b) through interfaces which make it possible to develop automatically executable procedures or programs for data administration.

The figure below illustrates how the interaction of the above-mentioned factors can result in economic benefits.

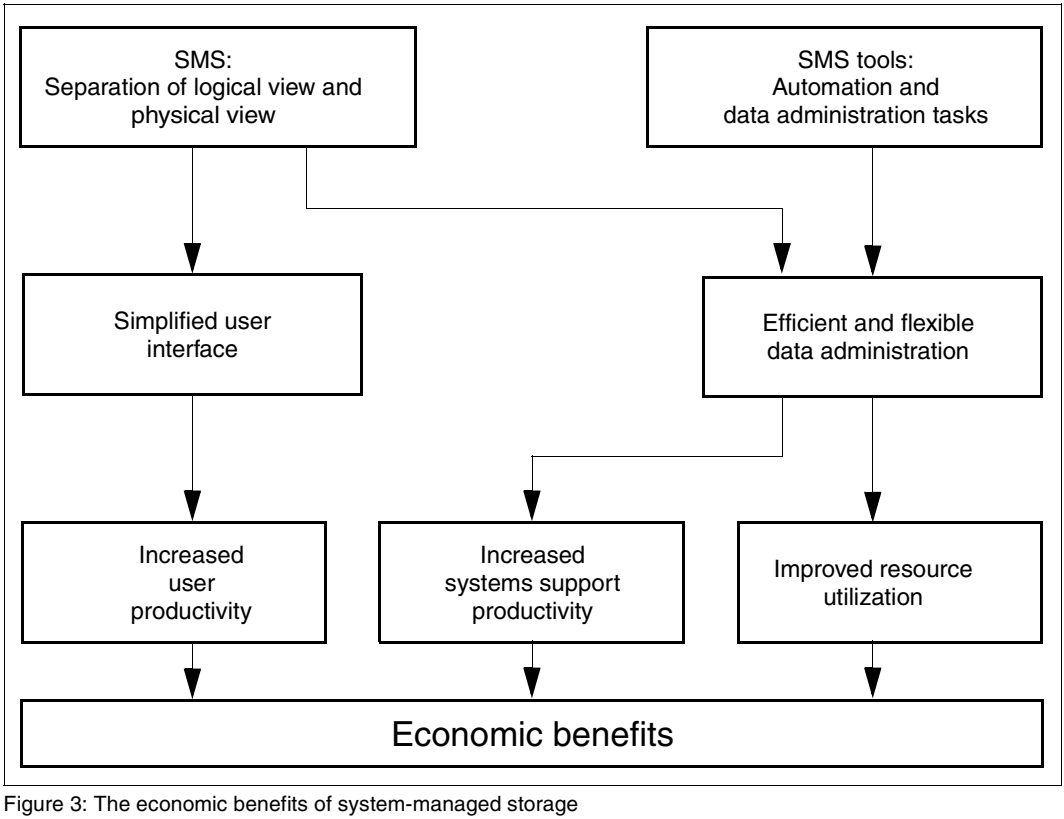


Figure 3: The economic benefits of system-managed storage

3 Overview of system-managed storage in BS2000

3.1 Data administration and pubset management

In BS2000, systems support staff provide pubsets together with the allocated background levels and backup configurations as an environment within which users can create, process and save files. This means that data administration as a system support task in BS2000, both locally and in shared pubset networks, is largely equivalent to **pubset management**. This covers the following areas:

- generation of pubsets and their allocated background levels and backup configurations
- taking pubsets into and out of service
- allocation of pubset resources to different users
- user support in file management in the different phases of file life cycles (file creation, backup, recall, migration to background levels,...)
- prevention, recognition and elimination of critical pubset statuses
- adaptation of pubset configuration to changing requirements

Pubset management as a system support task in BS2000 affects the system administrator, operators and the HSMS administrator. In general we do not differentiate between these roles in this manual.

3.2 Pubset concept and SMS

The pubset concept provides an ideal framework for the implementation of the idea behind SMS in BS2000. The systems support operative can set up two different types of pubset: **single-feature pubsets (SF pubsets)** and **system-managed pubsets (SM pubsets)**. SM pubsets are an enhanced version of SF pubsets. They have special properties that make them particularly suitable for system-managed storage.

However, SF pubsets also have many of the features required for SMS. For example, users do not have to concern themselves about the storage location of a file in a pubset. The physical structure of the individual volumes remains hidden from them. As far as attributes such as performance, availability and format are concerned, however, SF pubsets are homogeneous units. Users themselves must ensure that files are created on SF pubsets that meet the requirements of these files best.

The following additional features of SM pubsets allow the idea behind SMS to be developed further:

- Unlike SF pubsets SM pubsets make it possible to combine resources with different attributes and are therefore suitable for the storage of files with very different requirements. Even if the requirements for a file located in an SM pubset change during the file's lifetime, these requirements can be met by moving the file within the pubset. It is therefore not necessary to remove the file from the SM pubset. As long as the file moves within the same SM pubset it is addressed in precisely the same way (same catalog ID).
- SM pubsets provide special options which allow users to formulate their requirements at the logical level without needing to know the internal structure of the SM pubset.
- The volume configuration of an SM pubset can be enlarged or reduced with the pubset online. This means that pubset storage capacity can be easily adapted to fluctuating requirements. It is also possible to replace defective or obsolete volumes with the pubset online. The modifications to the pubset configuration can be implemented by redistributing user files within the SM pubset. Although SF pubsets can also be increased or reduced in size dynamically, SM pubsets have the advantage for systems support that they lend themselves better to being reduced in size, in particular. With SF pubsets, on the other hand, the conditions that have to be fulfilled before they can be reduced in size are often very difficult to bring about, and this imposes considerable constraints on the flexible use of pubset resources.
- When compared to SF pubsets, SM pubsets are better suited to automating data administration activities.
- Smaller units of failure can be created within an SM pubset. If one volume in an SM pubset fails only the unit of failure to which this volume belongs is affected.

- Like existing SF pubsets, SM pubsets are self-contained file containers which contain not only the files themselves but also all the relevant information and volumes required to access these files. Unlike SF pubsets, the background levels and backup configuration of SM pubsets is consistently subordinated to the container concept. The fact that SM pubsets act as containers is of particular significance in computer networks. It makes it possible to switch an SM pubset from one computer to another with the minimum administrative cost even if the pubset contains complex and extensive configurations.
- SM pubsets permit a more differentiated user administration than SF pubsets, thus making it possible to monitor the way in which users consume the various resources.
- SM pubsets permit the coexistence of users who want to use SMS features and users who want to perform their own data administration tasks. It is possible to set up pubset-internal, 'user-controlled' domains for special users who know and wish to make use of the physical features of the pubset. User-controlled domains may, for example, be attractive to previous private disk users since these form part of the pubset world and thus make it possible to use certain pubset functions which are not available with private disks (such as security functions, F5 reconstruction etc.). If SM pubset users who have previously performed their data management themselves wish to use SMS features, the transition within the SM pubset is simple to perform at the administrative level.

The central features of pubsets such as the container concept and the addressing unit are common to both SF pubsets and SM pubsets. It is thus possible for the user to change easily from SF pubsets to SM pubsets with SMS without the need for a hard break.

3.3 Structure of an SM pubset

The above-mentioned characteristics of SM pubsets are possible because the structure of SM pubsets is more complex than that of SF pubsets. This is illustrated in figure 5.
An SM pubset must contain a **processing level**. This consists of one or more **volume sets**. A volume set consists of one or more volumes. All files are located in a unique volume set and may not extend across volume set boundaries.

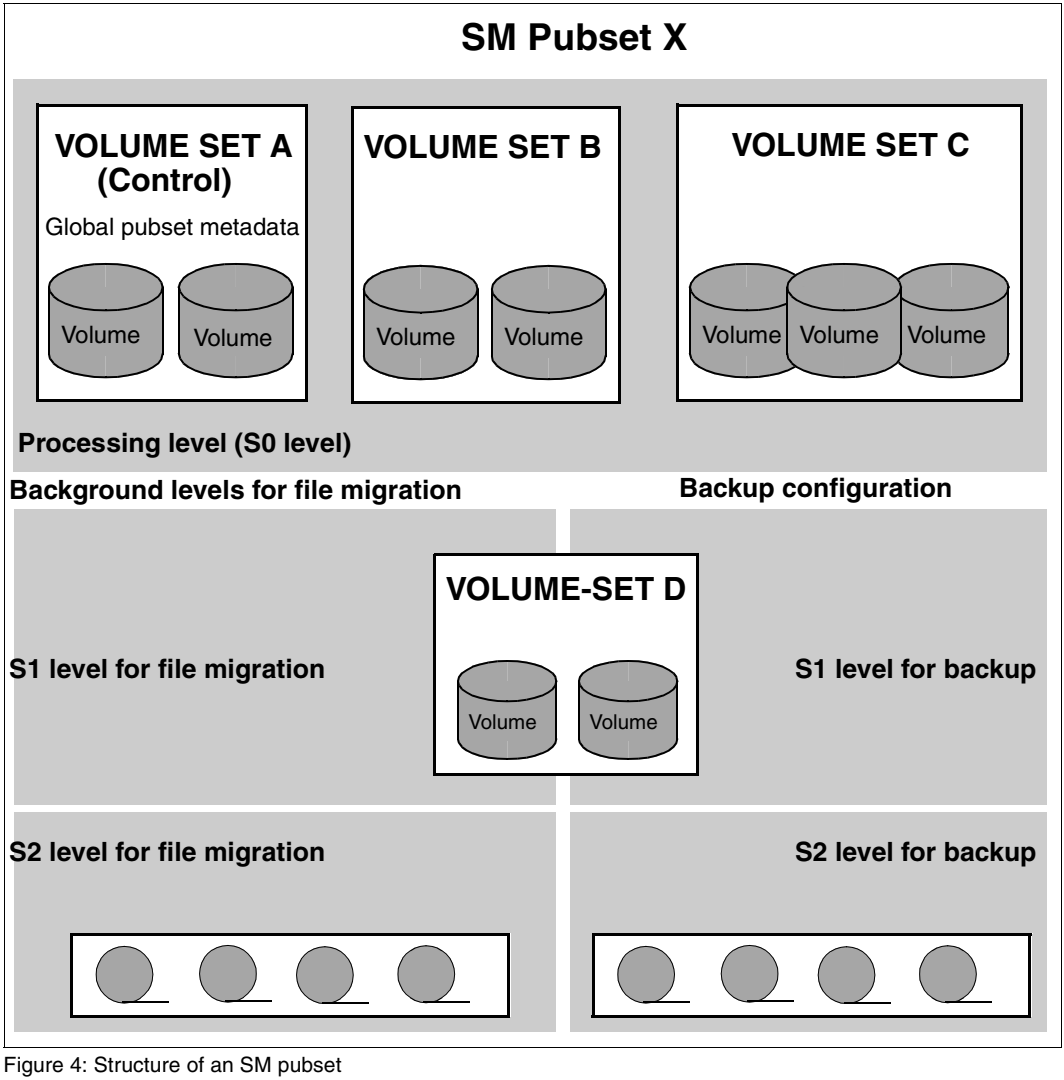


Figure 4: Structure of an SM pubset

Among these volume sets the **control volume set** plays a special role. It is an obligatory component of the SM pubset and contains all global pubset metadata, for example the user catalog. As the core of the SM pubset, it should be particularly failsafe, especially in the case of large SM pubsets which contain multiple volume sets. This can be achieved, for example by mirroring the control volume set at volume level. Users do not need to know the structure of the SM pubset. Of particular importance is the fact that files are addressed independently of their location within the pubset.

The volume sets within an SM pubset have a number of functions:

1. Resource type

   Volume sets form units in which it is possible to group various resource types within an SM pubset. The resource type is characterized by certain attributes which depend on the volumes belonging to the volume set. These attributes are taken into account when a suitable location is determined location is determined for a file. The volume set must be homogeneous so that it can represent a specific resource type, i.e. the volumes contained in the volume set must all have the same attributes. However, it is not necessary for all the volumes having the same attributes within an SM pubset to belong to a single volume set. It often makes sense to group them together in a number of different volume sets all having the same attributes (see below Unit of failure).

2. Reconfiguration unit

   With the exception of the control volume set, volume sets are all reconfiguration units, i.e. with the pubset online, it is possible to enlarge a pubset by adding volume sets or reduce a pubset by removing volume sets. There must be no files on the relevant volume sets when this reconfiguration work is carried out. In order to clear files from a volume set in an SM pubset (flushing a volume set) it is necessary to delete files contained in it or relocate these to other volume sets or background levels. Special functions are provided for this purpose. This operation is indirectly supported by the fact that the global pubset metadata files which can be neither deleted nor migrated are concentrated in the control volume set. The volume sets (including the control volume set) can be enlarged by adding volumes and reduced by removing volumes. However, the volumes affected must not contain any files or file extents. The flushing of individual volumes within a volume set is also supported by system functions. However, there are fewer options available than when flushing volume sets.

3. Unit of failure

   With the exception of the control volume set, the volume sets are self-contained units of failure within an SM pubset. If a volume fails, the worst that can happen is that the files of the volume set to which this volume belongs are lost. The volume set affected can be removed from the SM pubset as a defective volume set. The system creates a list of files that the volume set contained. The rest of the SM pubset is not affected by the failure of the volume, nor does pubset operation have to be interrupted.

   In the event of the failure of the control volume set, the entire SM pubset fails, but it can be restored on the basis of the remaining, intact volume sets. The data on the failed control volume set, in particular the metadata of the SM pubset itself, has to be restored from backups. The files on the other volume sets are preserved with their current statuses. As a result, as far as the data is concerned, the control volume set is not a “single point of failure” either.

Volume sets form distinct failure units in the SM pubset. There is a direct connection between this and the fact that a file cannot extend across a number of volume sets. When the volume set configuration is specified for an SM pubset that contains a large number of volumes with the same properties, the following factors have to be taken into consideration:

- An SM pubset consisting of a small number of large volume sets has the disadvantage that the failure of a volume affects a large proportion of the pubset and damage (number of files affected) is greater than in the case of the failure of a volume in an SM pubset consisting of a large number of small volume sets.
- Compared to an SM pubset consisting of a small number of large volume sets an SM pubset consisting of a large number of small volume sets has the disadvantage that the available space cannot be used as flexibly. For example, when the capacity of a small volume set has been exhausted the files contained in it can no longer be increased even if sufficient space is available in another suitable volume set. This loss of flexibility also affects other storage space management objectives such as the minimizing of file extent fragmentation.

SF and SM pubsets can be equipped **for file migration** with **background levels** by the product HSMS. When taken in conjunction with the processing level these represent a three level storage hierarchy in so far as access time, availability and costs are concerned. The position in the hierarchy is expressed in terms of levels: the **S0 level** (for the processing level), the **S1 level for background migration** (for the background level that is available online) and the **S2 level for background migration** (for the background level formed by volumes that are accessible offline). If files are not processed for a considerable time, it is advisable to migrate them to a lower cost background level. In the case of SM pubsets, the S1 level is implemented in the form of a volume set which belongs to the pubset and which must be exclusively reserved to HSMS for this purpose. The media used for the S2 level consist of a tape pool containing magnetic tapes, magnetic cartridges or optical disks. Background levels are exclusively assigned to an SM pubset and cannot be used by

multiple pubsets unlike the background levels of SF pubsets. The metadata required for the use of background levels, for example the migration archive directory is located in the SM pubset itself where it is stored in the control volume set.

In the case of both SF and SM pubsets, HSMS makes it possible to set up **backup archives** for backup operation. The **system backup archive** is of particular importance. SM pubsets differ from SF pubsets in the following ways:

- In the case of both SF and SM pubsets, either tapes or disks can be used as backup volumes (note: “tapes” is used in the following to mean magnetic tapes, magnetic tape cartridges and optical disks). In the case of SM pubsets, the disk backups must be stored within the pubsets on a dedicated volume set that is used only by HSMS and on which there are no normal user files. This ensures that original data and backup copies are in the same SM pubset (which acts as a container) but are stored on independent units of failure. The part of the backup configuration that consists of disks is referred to as the **S1 level for backup**, whereas the tapes form the **S2 level for backup**.
- The backup archives of an SM pubset can only be used to back up files from the SM pubset to which they are allocated.
- It is essential that the metadata required for the use of the backup archives, for example the backup archive directory is located in the SM pubset itself. The system creates this preferably on the control volume set in the SM pubset. For the directories of the backup archives, the volume set required as the storage location can be specified by the systems support operative. The control volume set should always be selected for the system backup archive.

3.4 File management in an SM pubset

File management in an SM pubset is illustrated in figure 5. In SM pubsets - as in SF pubsets - the pubset in which a file is created is implicitly specified by the path name assigned to the file by the user (catalog ID). There is no correlation between the pubset-internal location of a file and the way it is addressed. Users do not need to know anything about the internal structure. The facilities made available by SM pubsets are visualized at a logical interface. From the user's point of view, the SM pubset appears as a file container which provides certain file management services. These are referred to as **file services**. They can be subdivided into storage services (performance provided, availability,....) and HSMS management services (control of file migrations to background levels, control of backup creation,...). Users request the required file services with the help of file attributes to which values appropriate to the files must be assigned.

3.4.1 Storage services

Storage services can be requested in the two different ways listed below:

1. Direct attribution

   In the case of **direct attribution**, users specify the attributes which are relevant to the file's location such as performance, availability, etc. individually.

2. Assignment to a storage class

   Systems support staff can set up **storage classes** which represent a specific combination of the attribute values which are relevant for the file's storage location. Users can select the storage class which best meets their requirements from the available storage classes and allocate the file to this class.

The storage service requested for a file by the user is taken into account if the storage location in the processing level is defined for the file. This is the case if a new file is created, if a file is recalled from a background level to the processing level or if users modify the file attributes relevant to the storage location of an existing file in such a way that they are incompatible with the existing location. The system then automatically determines the most suitable volume set. However, system behavior can be influenced by systems support in order to implement certain data center-specific strategies. The tools that systems support staff use to do this are called **volume set lists**. These can be set up by systems support and be linked to storage classes. If a user assigns a file to a storage class which is assigned to a volume set list then the system attempts to create the file in a volume set which belongs to this volume set list. Files to which no storage class or a storage class without a volume set list is assigned are preferentially created in volume sets which do not belong to the volume sets created by systems support.

3.4.2 HSMS management services

Management services are requested via **HSMS management classes**. They must be set up by systems support. The HSMS management classes represent certain procedures for backup (e.g. backup frequency, life time of backup versions, etc.) together with rules which control the migration of files to background levels (e.g. eligibility for migration dependent on the time elapsed since the last access). If users assign a file to an HSMS management class, the file is then subject to the backup and migration procedures represented by this HSMS management class. **Migration locks** are additionally available to meet the special user requirement to exclude certain files from migration to background levels either globally or when certain circumstances occur.

The implementation of a service which is represented by an HSMS management class differs from the implementation of storage services in that the former is not performed using automated mechanisms which are already integrated into the system. Instead, systems support staff can use interfaces which support easy automation and which allow them to develop command procedures or programs for handling the planned processes.

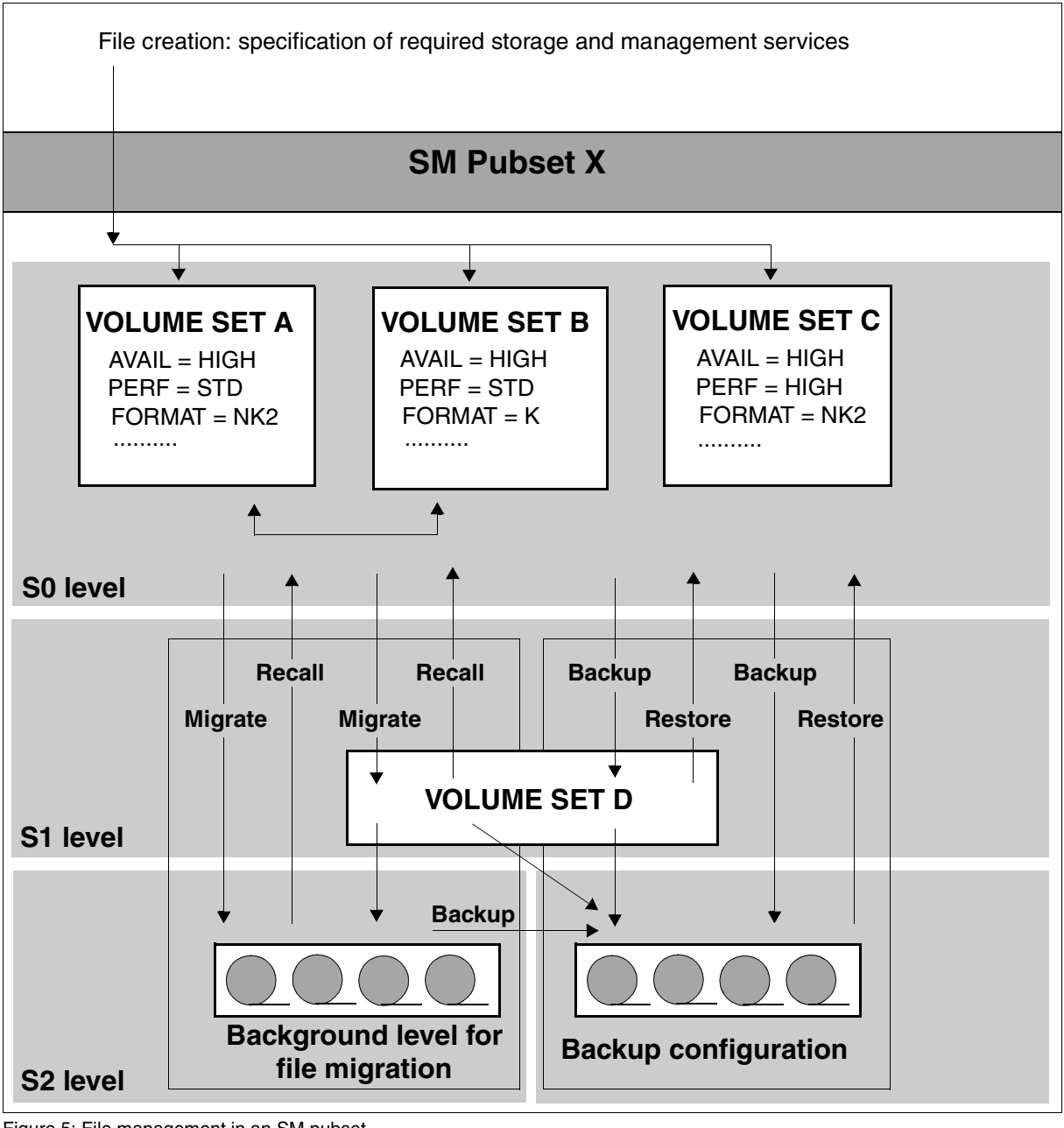


Figure 5: File management in an SM pubset

4 Characteristics of an SM pubset

This chapter contains a list of the characteristics of an SM pubset. These include

- the **structure** of the SM pubset
- the **statuses** of the SM pubset or its individual volumes
- the **settings** made by systems support for pubset operation.

4.1 Pubset identification and file name space

Like SF pubsets, SM pubsets are identified by the pubset identification. This is entered as the catalog ID in the path names of all the files and job variables cataloged in the pubset and therefore determines the (file) name space which is allocated to the pubset. The name space thus also comprises files stored on private disks or tapes which are cataloged within the pubset. The pubset ID is recorded within the pubset itself.

4.2 Structure of the processing level of an SM pubset

Figure 6 presents an example for the configuration of an SM pubset processing level. The processing level consists of one or more volume sets (maximum number 255). Each volume set is a container for complete files. The path name of a file does not specify the volume set in which the file is contained. Among the various volume sets of an SM pubset, the control volume set plays a special role as the container of the global pubset metadata files. For this reason it must be particularly failsafe. The volume sets belonging to an SM pubset are recorded together with their attributes in the pubset configuration file :<catalog ID>: $TSOS.SYS.PUBSET.CONFIG. This is implicitly generated during pubset generation and is always located in the control volume set.

A volume set normally consists of one or more volumes (maximum number 255). However, during pubset reconfiguration the following special status may occur: a volume set may be entered in the configuration file of an SM pubset even though it does not possess a physical configuration. Volume set C in figure 6 illustrates precisely this type of case. An intact physical configuration must always be assigned to the control volume set. Among the volumes in a volume set (with a physical configuration), the VOLRES has a special role to play. Its SVL (standard volume label) contains the volume catalog in which all the volumes belonging to the volume set are recorded. Each volume is identified by its VSN (volume serial number). The VSN is related to the ID of the volume set to which the volume belongs by means of a naming convention. This naming convention permits both the volume set ID to be obtained from the VSN and the VSNs of the volumes to be derived from the volume set ID. There is no relationship, on the other hand, between the VSN and the ID of the SM pubset.

Volume sets that do not belong to an SM pubset are known as free volume sets. They serve exclusively as pubset reconfiguration elements and cannot be used independently for file processing (note: free volume sets cannot be confused with SF pubsets). Free volume sets are normally empty. In other words, apart from the file catalog they do not contain any files. Not until they are incorporated in an SM pubset can user files be created on them. Free volume sets can be created by systems support with the help of SIR. They are also formed when volume sets are removed from an SM pubset (physically) during dynamic pubset reconfiguration. To add a free volume set to an SM pubset during pubset reconfiguration, the systems support operative has to do the following:

– enter the volume set in the configuration file
– make a free volume set available
– add the free volume set to the SM pubset

Free volume sets that are not empty only occur in special situations such as when there is a failure of the control volume set and the SM pubset thus destroyed as a unit or when a volume set is removed forcibly from an SM pubset. Free volume sets that are not empty cannot be used immediately; unlike empty free volume sets, they cannot be added to SM pubsets directly in pubset reconfiguration. Before the files of free volume sets that are not empty can be accessed again, a special startup process must take place with conversion to SF pubsets. One important application of this function is the restoration of an SM pubset after the failure of the control volume set.

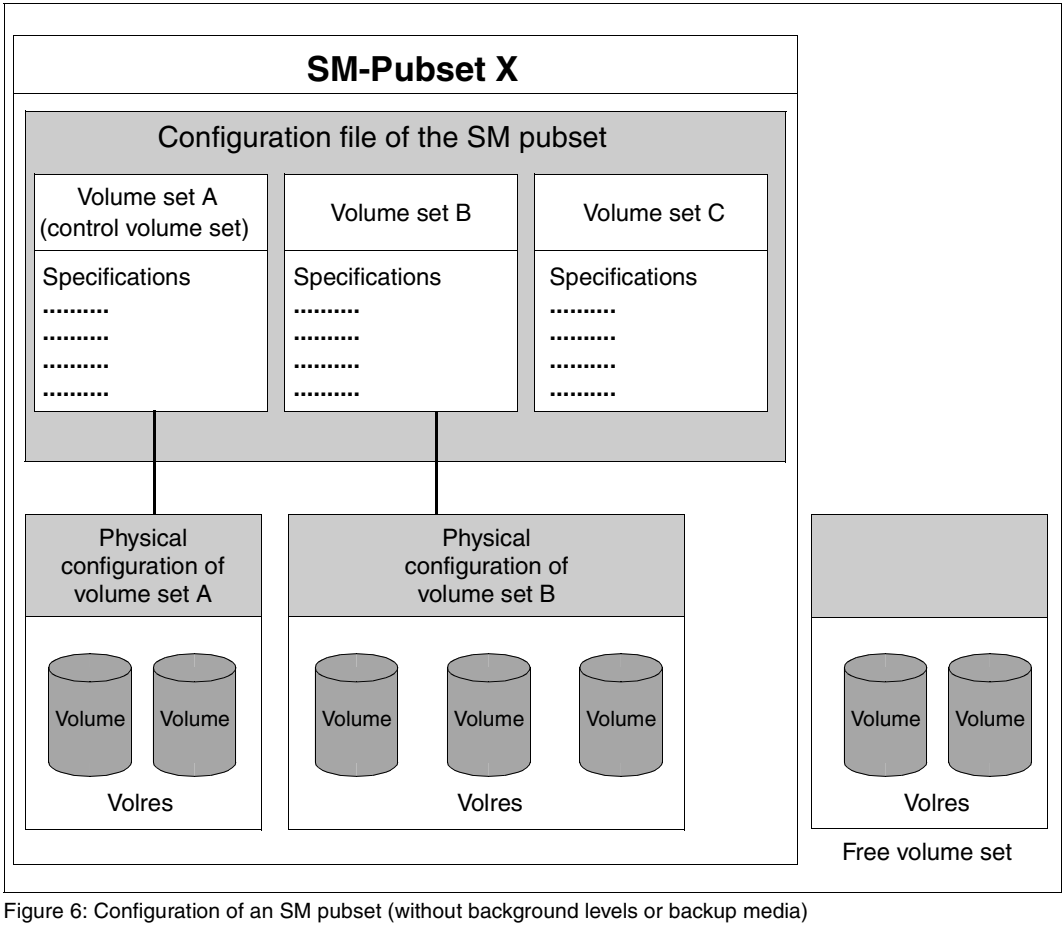


Figure 6: Configuration of an SM pubset (without background levels or backup media)

4.3 Characteristics of individual volume sets within an SM pubset

4.3.1 Configuration status of a volume set

Normally the volume sets of an SM pubset possess an intact physical volume set configuration, i.e. they consist of one or more volumes which can be accessed without problems. However, it is possible that the configuration of a volume set may be subject to special conditions, for example if access to individual volumes is impossible or only possible with restrictions as a result of disk defects or because of the failure of data paths. Another special status occurs during pubset reconfiguration. Two steps are involved in incorporating a volume set in an SM pubset:

– logical incorporation of the volume set

– incorporation of a free volume set

After the first step the volume set is entered in the pubset configuration file. However, no volumes are allocated to it. The reverse applies to the removal of a volume set from an SM pubset. As far as the physical configuration of a volume set is concerned, distinctions are drawn between the following configuration statuses:

normal use	The volume set possesses an intact physical configuration. The control volume set must always have the ‘normal use’ status.
defective	The physical configuration of the volume set can no longer be used as the result of a defect.
in hold	Temporarily only limited access to the volumes of the volume set is possible (e.g. because of problems in the access paths which, however, can be eliminated). The use of the entire SM pubset is impaired for as long as a volume set of this pubset has this status.
defined only	The volume set is entered in the configuration file. However, no physical configuration is allocated to it. The volume set characteristics which result from the physical configuration are undefined in this status. A free volume set does not belong to the configuration of a pubset, even when there is an entry for the same ID in the configuration file of the pubset.

The exact meaning of the individual statuses and the possible transitions between statuses are described as part of the discussion of dynamic pubset reconfiguration.

The configuration statuses of a volume set are not computer-specific, i.e. they can be defined independently of whether and at which computers the SM pubset to which the volume set belongs is online. The configuration statuses are recorded in the pubset configuration file. The pubset configuration file does not contain any information on free volume sets. They are not viewed as belonging to an SM pubset.

4.3.2 Attribute profiles for volume set selection

The format, allocation unit size, performance behavior and availability attributes of the individual volume sets of an SM pubset may vary. These attributes are evaluated by the system (volume set selection) when it selects the volume set which is best suited to a given file.

- The allocation unit size and the format are assigned to a volume set when it is initialized/formatted and remain assigned to it for as long as it continues to exist. These attributes are also defined for free volume sets.
- The performance and availability profile of the volume set associated with a pubset are determined by systems support when the pubset is formed or if a volume set is incorporated in the pubset as part of a pubset reconfiguration. These profiles are recorded in the SM pubset configuration file. They remain undefined for free volume sets.

Volume set performance and availability profiles can only be described at the level of detail permitted by volume set selection. Further differences (e.g. between the availability of DRV and REMOTE COPY disks) cannot be specified. We consider the individual attributes in greater detail below.

Format

The formats K, NK2, NK4 are possible for volume sets. Each volume set has a homogeneous format (as in the case of SF pubsets), i.e. the format of all volumes belonging to the volume set is the same.

Allocation unit size

The values 4KB, 6KB, 64KB are possible for a volume set allocation unit. The volume set is homogeneous in terms of the employed allocation unit (as in the case of an SF pubset), i.e. all its volumes have the same allocation unit.

Availability

The availability of a volume set is characterized by the AVAILABILITY attribute. The possible values are STD and HIGH. There are a number of ways of achieving increased volume set availability, e.g. DRV, REMOTE COPY etc. When you assign a value to the volume set attribute AVAILABILITY no check is performed to discover whether the actual characteristics of the volume set correspond to the assigned value. It is the responsibility of systems support to select values which match real circumstances. The assignment of the AVAILABILITY attribute is likewise the responsibility of systems support and is not automatically performed by the system for the following reasons:

– It must be possible for systems support to use their own experience in assessing availability. An inflexible assessment performed by the system may not correspond to the systems support staff's evaluation.
– It is advisable to assess media in relation to the characteristics of the other volumes present in the pubset. When new hardware is incorporated in a pubset it may be advisable to adapt earlier assessments.
– The availability value assigned by systems support corresponds to the long-term positioning intended for the volume set within the SM pubset. Modification of the AVAILABILITY value from HIGH to STD demands considerable organizational effort since before this is possible all the files in the volume set which have the file attribute AVAILABILITY=HIGH must be moved to another volume set with a high level of availability. It is therefore only advisable to change the availability level when the positioning of the volume set is to be modified in the long term not, however, in the case of temporary fluctuations (e.g. if the operating status of a volume is changed from 'Dual' to 'Mono' in the case of DRV and for a short period only). The need to assign these values explicitly during pubset generation or pubset reconfiguration should make the long-term nature of this specification clear to systems support.

Performance

The performance profile of a volume set is described through the specification of a performance range together with restrictions relating to the reliability of write operations. This complex structure is necessary in order, for the purpose of volume set selection, to be able to adequately characterize volume sets to which caches are assigned by means of PFA (performant file access). In PFA, users use file attributes for performance to control both the selection of a suitable volume set and active cache usage in file processing.

– A volume set configured with a cache can generally cover a range of differing user performance requirements (STD, HIGH, VERY-HIGH). The volume set is therefore suitable as a storage location for files with the requirements PERFORMANCE=*STD, files with the requirement PERFORMANCE=*HIGH and files with the requirement PERFORMANCE=*VERY-HIGH.

  Note

  It is not possible to allocate more than one different type of cache to a volume set simultaneously. The various performance values correspond to different types of cache utilization which are possible during file processing:

PERFORMANCE=*STD	no cache utilization
PERFORMANCE=*HIGH	caching with file migration
PERFORMANCE=*VERY-HIGH	caching without file migration

  The type of cache usage can be adjusted individually to suit the specific performance requirements of different files within the same volume set.

– The level of write reliability provided by different cache media (global storage, main memory) differs. A cache is considered to be reliable for write operations if the risk of failure for the output data buffered in the cache is no greater than if this data were to be written directly to disk. Non-reliable caches should generally be used to improve performance during read-only file access. However, they can only be used for write access if users accept limitations with regards to the reliability of write operations and explicitly specify this using the file attribute DISK-WRITE=*BY-CLOSE. Only if these conditions are fulfilled can volume sets with non-reliable caches provide the full performance range (STD, HIGH, VERY-HIGH).
  A non-reliable cache should not be used for write accesses for which limited reliability is not acceptable. In this case, the performance range of volume set with a non-reliable cache is reduced to the value PERFORMANCE=*STD. Volume sets configured with caches which are reliable for write operations provide the full performance range without restrictions (STD, HIGH, VERY-HIGH).

– If systems support presets the value WRITE-CONSISTENCY=*BY-CLOSE for a volume set the system assumes that no availability restrictions apply to the files located in the volume set if no enhanced performance is provided for them (i.e. no caching is requested). This means, for example, that the specifications PERFORMANCE= *VERY-HIGH and WRITE-CONSISTENCY=*BY-CLOSE have the same meaning for the performance profile of a volume set as the specification PERFORMANCE=(STD,VERY-HIGH) and WRITE-CONSISTENCY=*BY-CLOSE.

– Enhanced performance can be achieved not simply through the use of caches but also through the employment of volume sets which consist of special high performance volumes such as volumes emulated in global storage. The performance spectrum of such volume sets can be characterized adequately by a single performance value for volume set selection. Caches used by ADM PFA (administrator performant file access) represent a special case in which caching is not controlled by the users by means of file attributes; instead, specific precautions are taken by systems support (the cache administrator) for the caching of particular files. The storage location of the files must also be included in such caches. The storage location can be specified by systems support by means of physical allocation (see page 193). The performance-related file attributes and the performance profile of a volume set are of no significance in ADM PFA caching.

Notes on terminology

You can use the MODIFY-PUBSET-DEFINITION-FILE command to set the performance profile of a volume set. The terms used in this description correspond to the operand and operand values used for setting the performance profile in the command syntax in the following way:

Terminology used in this description	**Command syntax**
performance profile	PERFORMANCE-ATTRIBUTES
performance range	PERFORMANCE
write-reliability with enhanced performance yes (ensured) no (not ensured)	WRITE-CONSISTENCY IMMEDIATE BY-CLOSE

Performance profile recommendations

Like the availability profile, the performance profile of a volume set is not automatically determined from the physical configuration. It is the task of systems support to define the performance profile of volume sets in such a way that this corresponds to the actual circumstances. When describing the performance range it is particularly important to assign the correct level to each individual volume set with reference to the other volume sets present in the pubset. Consequently the settings for the various cache media proposed in table 1 can only serve as a guideline in so far as the performance range is concerned since they do not take account of the circumstances of real configurations. It is, however, possible to make recommendations which can be implemented in all configurations when defining the level of write reliability in cases where enhanced performance is required.

Volume set configuration	**Recommended performance profile**	
	Performance range **PERFORMANCE**	**Write reliability with enhanced performance (HIGH and VERY-HIGH) WRITE-CONSISTENCY**
Volume set consisting of volumes without any special performance characteristics and to which no cache is allocated	STD	irrelevant
Volume set to which a main memory is assigned as cache medium	STD, HIGH, VERY-HIGH	BY-CLOSE (no write-reliability exists)
Volume set to which a volatile global storage is assigned (VOLATILITY=*YES; command MODIFY-PUBSET-CACHE-ATTRIBUTES)	STD, HIGH, VERY-HIGH	BY-CLOSE (no write-reliability exists)
Volume set to which a non-volatile global storage is assigned (VOLATILITY=*NO; command MODIFY-PUBSET-CACHE-ATTRIBUTES)	STD, HIGH, VERY-HIGH	IMMEDIATE (write-reliability exists)
Volume set which consists of volumes emulated in global storage	VERY-HIGH	IMMEDIATE (write-reliability exists)

Table 1: Recommendations for performance profiles

4.3.3 Physical volume set characteristics

As mentioned above, the systems support staff must ensure that the physical volume set configuration responds to the attribute profiles of the volume sets. Currently there are a number of ways to do this at the ‘physical level’ and the range of options will increase in the future. Below we restrict ourselves to a number of examples illustrating certain aspects of this operation:

Physical attributes of individual volumes

The physical attributes of a volume set are determined by the attributes of the volumes allocated to it as well as its I/O configuration (Controller, access paths, etc.). In order to make it possible to assign meaningful attribute profiles to volume sets, the volume configuration of the volume sets should be homogenous, i.e. systems support should ensure that volume sets only contain volumes whose attributes are compatible with the projected attribute profile. In the case of certain attributes, the system imposes volume set homogeneity (allocation unit size, format, volumes emulated in global storage).

Maximum I/O length of a volume set

One of the physical attributes that characterizes a volume is its maximum I/O length, i.e. the maximum number of blocks that can be transferred during an I/O job. The lowest value of all the volumes belonging to the volume set determines the maximum I/O length of a volume set. If the pubset reconfiguration facility is used to add a new volume to a volume set, the maximum I/O length of the volume set may be reduced. The maximum I/O length of a volume set is used by certain privileged applications (ARCHIVE, CCOPY) in order to optimize input/output operations.

Volume sets as independent units of failure

Pubset management makes it possible for the individual volume sets to operate as independent units of failure, i.e. if a volume fails then only the volume set to which it belongs is affected. Provided that the control volume set does not fail, the rest of the SM pubset can remain in service. However, this also requires measures at the physical level: volume sets which are to be unaffected by a volume set failure must be mapped to physical configurations which are independent of one another in terms of failure. For example, the access paths (channels, controllers etc.) to the volumes of the volume sets which need to be independent of one another in terms of failure must be distinct. The cache configuration assigned to the volume set must also be taken into account. As the central metadata container of the SM pubset, the control volume set has a special role, but it does not represent a “single point of failure”. In the event of it failing, it is possible to convert the remaining, undamaged volume sets to SF pubsets during a special startup process and then to form an SM pubset from these by means of SMPGEN.

The data on the control volume set, which cannot be reconstructed automatically to the desired state when the SM pubset is formed (unlike, for example, the HSMS metadata, the catalogs for storage classes and volume set lists or the user catalog), must be restored from backups by the systems support operative.

Failsafe performance

It is possible to use the DRV and DUALCOPY functions to create volume sets which are particularly failsafe. For SM pubsets, these provide an operator interface at volume set level which is analogous to that provided for SF pubsets at pubset level.

Volume sets with assigned cache

Caches cannot be used unless certain cache configurations are provided. Different cache areas can be assigned individually to the different volume sets of the SM pubset. The precautions that need to be taken for the use of caches depend greatly on the cache medium involved. Volume sets to which global storage caches are assigned by means of PFA are considered by way of example below.

- *Allocating the cache configuration*

  For each volume set, systems support must determine whether it is to be operated with a cache, which cache medium is required and how large the cache area for the volume set is to be. If caches are used, it is also necessary to set certain cache operating parameters. The scope and the significance of the operating parameters differ between individual cache media. The specifications made by systems support determine the defined values for the cache configuration of the volume set. They are recorded in the configuration file of the SM pubset or the individual volumes. It is only possible to modify the defined values during a pubset session (unlike SF pubsets).

  The defined values for the volume set cache configuration become effective

  – if the SM pubset to which the volume set belongs is put into operation on a system

  – if a new volume set is physically added to an SM pubset by means of dynamic pubset reconfiguration, i.e. if a free volume set is allocated to a volume set which previously had the ‘defined only’ status

  – if the START-PUBSET-CACHING command is used to activate the cache for a volume set with the ‘normal use’ status.

  In such cases the system attempts to connect the volume set with a new cache area on the basis of the defined values. In certain situations deviations in size may occur. The size of the currently allocated cache area and the cache operating parameters currently active for the volume set are described by the current values of the (volume set) cache configuration.

  The cache area is disconnected from the volume set:

  – if the SM pubset to which the volume set belongs is taken out of service

  – if the physical volume configuration is removed from the volume set by means of dynamic reconfiguration,

  – if a volume set cache is deactivated using the STOP-PUBSET-CACHING command

Volume sets belonging to pubsets which are not currently online are normally not connected to cache areas. In special cases, e.g. after a system crash, cache areas may also be allocated to volume sets belonging to exported SM pubsets.

- *Setting up the GS partition*

  In order to ensure that GS cache areas can be provided for the volume sets of an SM pubset, systems support must set up GS partitions for the pubset. These are exclusively assigned to the SM pubset even if this is not in service. These act as reserves for all GS cache areas which are connected to one of the volume sets of an SM pubset. The correlation between GS partitions and the SM pubset is ensured by a GS partition naming convention.

- *Taking account of cache media availability*

  Caches are associated with a certain risk of failure. When files are processed in read-only mode caches can be used as buffers independently of their level of reliability without affecting the reliability of the file. However, when write accesses are buffered, the failure of the cache results in data loss which may lead to the destruction of the entire file. Therefore when evaluating the reliability of a volume set/SF pubset which is operated with a cache it is necessary not only to consider the reliability of the volume and its access paths but also the availability of the allocated cache.

  If GS caches are used it is necessary to consider the following aspects:

  – Reliability of the GS cache power (VOLATILITY)

    GS caches can be protected against power supply failures using special mechanisms (battery operation, uninterruptedly power supply etc.). If these measures are not taken, GS caches are volatile, i.e. the data stored in them is lost if the power supply fails. Files for which output data was stored buffered in the affected GS areas are then no longer available in their current processing state.

  – Reliability of the cache media themselves

    Not only the GS power supply but also the cache media themselves are subject to a certain risk of failure. It is possible to reduce the risk of failure for data that is buffered in a GS cache using redundant data recording methods.

– Reliability on system crashes

  Both nonvolatile and volatile GS caches generally offer failsafe performance in the event of a system crash, since all the metadata required for the use of the cache areas is stored in the cache. After a system crash or the forced removal from service of the pubset, what happens depends on whether the GS cache configuration assigned to the pubset is used only on the local system or globally across more than one system. If it is used only on the local system, the restoration of files that still have write data in the GS cache is not possible until the SM pubset is put into operation on the same system again. In the case of a distributed GS cache, the files can also be restored from other systems at which the pubset is put into operation, provided the GS cache can be accessed from them. In shared pubset operation, when a GS cache is used across more than one system and one of the systems involved crashes, the affected files are restored on the current master system or during the master change.

GS caches which do not provide the level of availability necessary for them to classify as reliable caches may not be used to buffer output data from files for which the user requires write-reliability (file attribute DISK-WRITE=*IMMEDIATE). In the case of GS caches, the system cannot, unlike the situation when the cache media main memory are used, decide whether the cache is reliable for write operations. For such a decision to be possible, systems support must inform the system of the volatility of the cache (using the MODIFY-PUBSET-CACHE-PROCESSING command).

The setting VOLATILITY=*YES means that the GS cache is only used to buffer files for which the user is prepared to accept limited write-reliability (file attribute DISK-WRITE=*BY-CLOSE). It is unconditionally recommended for GS caches which have no failsafe power supply. However, VOLATILITY=*YES should also be set for GS caches which do not appear to guarantee sufficiently reliable write operations for any other reason.

It is also necessary to take the write-reliability of the cache into consideration when selecting a suitable volume set. However, this is not achieved using the VOLATILITY specification but instead by taking account of the performance profile which systems support must adapt to the volume set requirements.

4.3.4 Usage methods for volume sets

The alternative usage types STD, WORK, HSMS-CONTROLLED are possible for the volume sets of an SM pubset:

- The STD usage type is the one normally used. This is the only usage type possible for control volume sets.
- Only volume sets with the WORK usage type can be used for storing work files.
- Volume sets with the HSMS-CONTROLLED usage type are required in order to set up the S1 level using HSMS. They are used exclusively for this purpose.

The usage type for the volume sets is recorded in the pubset configuration file. It is also defined for a volume set if no physical configuration is currently assigned to it, i.e. if the volume set has the configuration state 'defined only'.

4.3.5 Usage restrictions for volume sets

The usage of a volume set can be restricted using the MODIFY-PUBSET-RESTRICTIONS command. Unlike usage type definitions, short-term modifications to usage restrictions are also possible.

a) File creation restrictions

The creation of new files within a volume set can be restricted using the NEW-FILE-ALLOCATION restriction. The following settings are possible:

NOT-RESTRICTED	no restriction
PHYSICAL-ONLY	new files are not created in the volume set unless this is explicitly requested via physical allocation.
NOT-ALLOWED	it is not possible to create new files. This restriction is not permitted for the control volume set.

The NOT-ALLOWED setting helps you flush a volume set before removing it from the pubset. It is not possible for the control volume set since this setting would interfere with the system functions.

b) File processing restrictions

The VOLUME-SET-ACCESS restriction can be used to restrict the processing of files located in a given volume set. It is not permitted unless the new file allocation restriction has the value NOT-ALLOWED. The following settings are possible:

ADMINISTRATOR-ONLY	TSOS privileged tasks may open files without restrictions (or reserve them using SECURE). Tasks which do not possess the TSOS privilege should be able to terminate current file processing operations in a structured way, even if they have to open other files in order to do this. However, if possible the commencement of new file processing operations should be prevented. For this reason, tasks without the TSOS privilege may only open new files if files are already open for them.
NOT-RESTRICTED	There are no restrictions.

c) Temporary volume set shutdown

No input/output operations may be performed at a volume set which has been recognized as defective. The same is true of volume sets which are only temporarily unfit for operation but which can very probably be used later without any loss of data. Assigning the restriction PROCESSING-STATE=*HOLD to the volume set blocks any input/output relating to the execution of system functions. A restriction of system functionality alerts users to this status (e.g. restricted output information from SHOW-FILE-ATTRIBUTES,

files cannot be reopened, no secondary allocation etc.). Current processing operations relating to user files which are located in the affected volume set are not automatically halted. If access problems occur, the associated access methods supply the corresponding return codes.

The “in hold” status is brought about automatically by the system or explicitly by systems support (by means of the MODIFY-PUBSET-RESTRICTIONS command or when the pubset is imported). It is not possible for the control volume set. It is canceled when systems support staff physically remove the volume set from the SM pubset or explicitly reset the “in hold” restriction when a temporarily inoperable volume set becomes usable again. When a pubset is put into services, as well, the system implicitly attempts to cancel the “in hold” status. Systems support staff should ensure that the “in hold” status does not last long because it impairs the DMS functionality for the remaining pubset (e.g. side-effects when new files are created).

d) Usage restrictions for the individual volumes of a volume set

Allocation restrictions can be assigned to the individual volumes of a volume set (ALLOCATION-ON-VOLUME). The following settings are possible:

NOT-RESTRICTED	no restrictions
PHYSICAL-ONLY	no space is assigned in the volume unless it is explicitly requested using physical allocation; no secondary allocations are performed
NOT-ALLOWED	no allocations are performed

Setting volume-related restrictions reduces the number of freely available volumes and consequently the amount of freely available space in the volume set. Such restrictions cannot be set if the volume set already suffers from a serious storage bottleneck (saturation level 4 exceeded) or if setting the restriction would cause such a bottleneck.

The usage restrictions NEW-FILE-ALLOCATION, VOLUME-SET-ACCESS and PROCESSING-STATE are recorded in the pubset configuration file of the SM pubset. They also have defined values if no physical volume set configuration is assigned to the volume set (configuration status 'defined only'). The usage restrictions at volume level (ALLOCATION-ON-VOLUME) are stored in the volume catalog of the VOLRES SVL. They are not modified when the volume set is removed. They therefore remain associated with the volumes of free volume sets.

4.3.6 Saturation thresholds for volume sets

Systems support assigns definable threshold levels to the individual volume sets of an SM pubset in order to monitor storage bottlenecks. Here it is necessary to draw a distinction between defined and current values. Saturation monitoring for volume sets is performed on the basis of the current values. The highest exceeded (current) saturation level determines the current saturation status of the volume set. The system uses this status as a criterion to determine where to store files. For example, if the other attributes of two volume sets make them equally well suited for a file then the volume set with the lower saturation level is preferred. If the enlargement or recreation of a file in the volume set in which the file has so far been located, or which has been identified as the most suitable storage location for the new file by the system, would result in a higher saturation level than previously being exceeded, then this has the same effect on user requirements as is the case for SF pubsets (message output, acceptance or rejection depending on the TSOS privilege etc.).

When the pubset is taken into service in ‘exclusive’ or ‘shared-master’ mode, the current values for each volume set are determined from the defined values recorded in the configuration file with the exception of volume sets which have the ‘defined only’ status. In general, the defined values are taken over unchanged. The ‘logical’ setting BY-SYSTEM represents a special case. This setting is used to initialize the saturation threshold of the volume set following SM pubset generation. When the pubset is taken into service this setting results in the calculation of current saturation thresholds in accordance with a specific procedure which involves the system parameter L4SPDEF and the capacity of the volume set.

Only the defined values are relevant for volume sets with the status ‘defined only’. The current values are not determined until the physical volume set configuration is assigned. The procedure used is analogous to that used when the pubset is taken into service.

The systems support staff can modify both the defined and the current values with the pubset online. It is not possible to modify the defined values for the volume sets of exported SM pubsets (unlike SF pubset!).

4.4 SM pubset characteristics affecting multiple volume sets

4.4.1 Scope of the provided storage services

The attribute profiles, usage type and usage restrictions of the individual volume sets determine the scope of the storage services which the SM pubset is able to provide to users. The storage services appear to users as combinations of file attribute values for performance, availability, format, etc. which are available for the files located in the SM pubset or which are optimally supported. The systems support staff can assign names to storage services by setting up storage classes.

4.4.2 Maximum I/O length of an SM pubset

As is the case with SF pubsets the maximum I/O length of an SM pubset is determined by the smallest maximum I/O length of all the volumes belonging to the pubset.

4.4.3 Default pubset space settings

The selection of defaults for SM pubsets can be controlled by means of the pubset space defaults. The pubset space defaults affect:

– defaults for the size of primary, secondary and maximum allocation (SF pubsets also have these defaults)
– file format defaults

There is both a defined and a current variant. The defined values are administered across pubset sessions in the pubset configuration file and are used to initialize the current values on pubset startup in ‘exclusive’ or ’shared-master’ mode. While the default values defined for primary, secondary and maximum allocation can be taken across into the current values unchanged this is not true of the default file format unless it can be provided by the SM pubset. This means that a volume set with the STD usage type must be present in the SM pubset, that this volume set must not be subject to any restrictions affecting new file allocation (NOT-ALLOWED, PHYSICAL-ONLY) and that the volume set format is compatible with the default file format. If the defined value of the default file format cannot be provided in the SM pubset the system switches to a format which can be provided in the pubset when determining the current value (for K preferably to NK2, or to NK4 if necessary, for NK2 to NK4).

The systems support staff can modify the defined and current values with the pubset online. Unlike in the case of SF pubsets, it is not possible to modify the defined values for the pubset space default settings of exported SM pubsets.

4.4.4 Large objects attributes

Like SF pubsets, SM pubsets can have the LARGE-OBJECTS attribute. LARGE-OBJECTS is an attribute of the entire SM pubset rather than the individual volume sets.

If the SM pubset has the attribute LARGE-OBJECTS, associated volume sets may contain volumes with a capacity of ≥ 32 GB (LARGE-DISKS-ALLOWED=*YES). In addition, the creation of large files (≥ 32 GB) can also be permitted on a pubset with large volumes (LARGE-FILES-ALLOWED=*YES).
The attribute LARGE-OBJECTS is assigned either when the pubset is modified with the SIR utility routine or via the SET-PUBSET-ATTRIBUTES command. See also the manual “Files/Volumes > 32 Gbyte” [7].

4.5 System files in the SM pubset

Pubset configuration file

The volume sets and associated volume set characteristics belonging to the SM pubset are recorded in the pubset configuration file :<catalog ID>: $TSOS.SYS.PUBSET.CONFIG. It Pubset user catalog is implicitly created on pubset generation and is always located in the control volume set.

Pubset user catalog

The user catalog of an SM pubset is stored in the :<catid>:$TSOS.SYSSRPM file. This is always located on the control volume set. Since SM pubsets can only be used as data pubsets, not home pubsets, the user profiles stored in the user catalog refer exclusively to the pubset itself. Although system-related user specifications are possible here, they have no further effects. For the purpose of controlling resources, systems support staff can assign differentiated space quotas to individual users. The user right for physical allocation is also included in the user entries. It refers to all the volume sets of the SM pubset. Finally, the user entries also serve as the storage location for the user-specific assignments of default storage classes. These represent an important tool for systems support staff, allowing them to implement their own strategies for space usage on SM pubsets.

It is possible to create user groups on SM pubsets in the same way as on SF pubsets. For SM pubsets that cannot be used as the home pubset, the space limit specifications in the group definitions are of particular relevance. The user group definitions are saved in the :<catid>:$TSOS.SYSSRPM file, as is the user catalog.

SM pubset file catalog

The file catalog of an SF or an SM pubset serves as a container for catalog entries for the files and job variables associated with the pubset as well as for the catalog entries of private disk files and tape files. In order to support the use of volume sets as independent units of failure, the file catalog for SM pubsets is implemented using multiple catalog files, unlike in the case of SF pubsets. The structure is depicted in figure 7.

- Each volume set in an SM pubset contains a volume set-specific catalog file with the catalog entries for the files contained in it. The path name of the volume set-specific catalog file is composed of the pubset identification and volume set identification as follows :<catalog ID>:$TSOS.TSOSCAT.<volset-id>.
- In addition, catalog management requires further catalog files which are all stored in the control volume set of the SM pubset:
  - The catalog entries for files migrated to a background level as well as cataloged files which occupy no space are stored in the catalog :<catalog ID>:$TSOS.TSOSCAT.#MIN.
  - The catalog file :<catalog ID>:$TSOS.TSOSCAT.#PVT is used as container for the catalog entries for tape files and private disk files.
  - The catalog file :<catalog ID>:$TSOS.TSOSCAT.#JVC is used as the container for the job variables of the SM pubset.
  - Catalog management uses the catalog files :<catalog ID>:$TSOS.TSOSCAT.$NLO, :<catalog ID>:$TSOS.TSOSCAT.$NLC and :<catalog ID>:$TSOS.TSOSCAT.$PFI for the rapid location of catalog entries or to create a list of affected files when a defective volume set is removed

Note

As of BS2000/OSD V6.0B the special catalogs with the suffixes #MIN, #PVT and #JVC can be converted to the “extras large” catalog format. In this case they are renamed and assigned the new suffixes M00, P00 and J00. Up to 99 further sub-catalogs can be added to these catalogs (the counter in the suffix being continuously incremented). For details, please refer to the “Introductory Guide to Systems Support” [5].

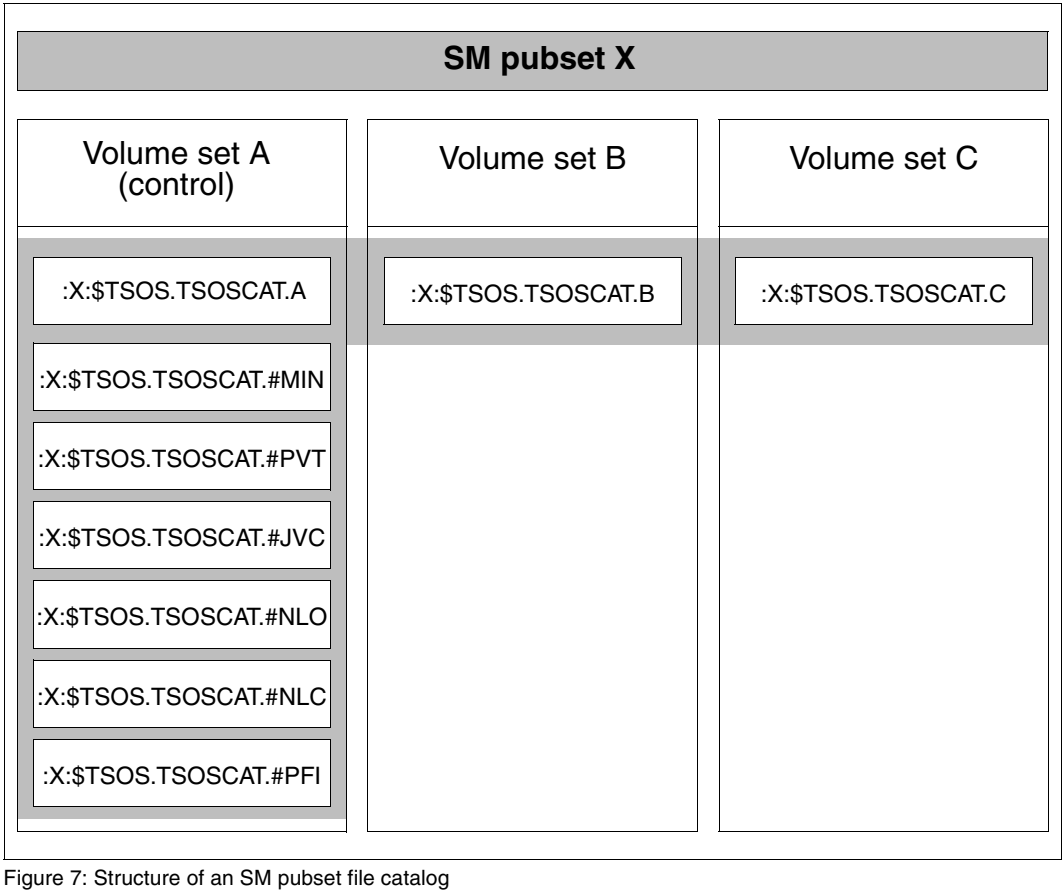


Figure 7: Structure of an SM pubset file catalog

Catalog files are implicitly created when the pubset is generated. The volume sets on which the individual catalog files are located are determined in accordance with the description above. The maximum size of the individual catalog files is 16384 PAM pages. During SM pubset generation, systems support can influence the size of catalog files and the way they are distributed across specific volumes.

Systems support can subsequently increase the size of catalog files using SIR (for a pubset which is not online) or the MODIFY-FILE-ATTRIBUTES command (during a pubset session). Catalog management can make direct use of any additional space made available during a pubset session.

GUARDS catalog

GUARDS protection profiles can be set up in a pubset in order to protect the files, storage classes, HSMS management classes, job variables etc. present in an SM pubset. This profile is stored in the GUARDS catalog of the pubset. It is implicitly created in the control volume set under the path name :<catalog ID>:$TSOS.SYSCAT.GUARDS when a GUARDS protection profile is first set up for the pubset. GUARDS protection profiles have the same significance for both SM pubsets and SF pubsets**.**

Catalogs for storage classes and volume set lists

The storage class catalog (:<catalog ID>:$TSOS.SYSCAT.STORCLS) and the volume set list catalog (:<catalog ID>:$TSOS.SYSCAT.VSETLIST) are implicitly created in the control volume set by storage class management. These catalogs record the definitions of the storage classes which have been set up for the pubsets or the volume set lists associated with these storage classes.

Watchdog file

The watchdog mechanism for computer monitoring in a shared pubset network requires the presence of the watchdog file :<catalog ID>:$TSOS.SYS.PVS.SHARER.CONTROL. It is automatically created in a pubset when this is first imported in ‘exclusive’ or ‘shared-master’ mode. It has the same function in both SM and SF pubsets. In the case of SM pubsets, the watchdog file is located in the control volume set.

Defect garbage file

Defect garbage files are used to record volume I/O errors. The system automatically creates a defect garbage file in a volume set when a defect is first recorded for one of the disks in the volume set. The volume has the path name :<catalog ID>:$TSOS.SYSDAT.DEFECT.GARBAGE.<volume set ID#>, where <volume set ID#> is the identification of the volume set. If this ID is less than 4 characters in length it is filled to 4 characters using #. The defect garbage file of a volume set has the same significance as in SF pubsets.

SYSEAM file

The file :<catalog ID>:$TSOS.SYSEAM is used as container for EAM files. It has the same significance for SM and SF pubsets. It can be located on any volume set in the SM pubset and does not have to be stored in the control volume set. If systems support uses the CREATE-FILE command to create this file before any EAM file is accessed then it is possible to assign certain file attributes to it (e.g. work file) or specify its location within the SM pubset using physical allocation. If systems support does not do this it is automatically created, assigned default file attributes and assigned an appropriate location by the system.

4.6 HSMS configuration of an SM pubset

By the HSMS configuration of an SM pubset in the narrow sense we mean the background levels and backup archives assigned to the pubset. Among the backup archives, the system backup archive of the SM pubset has a special role to play. More broadly, the HSMS configuration of an SM pubset also includes long-term archives and backup archives which are not allocated to the SM pubset but from which files can be restored to the SM pubset. Such backup archives may, for example, be created when an SF pubset is converted into an SM pubset. The HSMS configuration in the narrow sense is pubset-specific when SM pubsets are used, i.e. the configuration is allocated to precisely one SM pubset which contains the metadata required for its use. It therefore supports the SM pubset as a switchable unit. For a depiction of the HSMS configuration of an SM pubset, refer to figure 8.

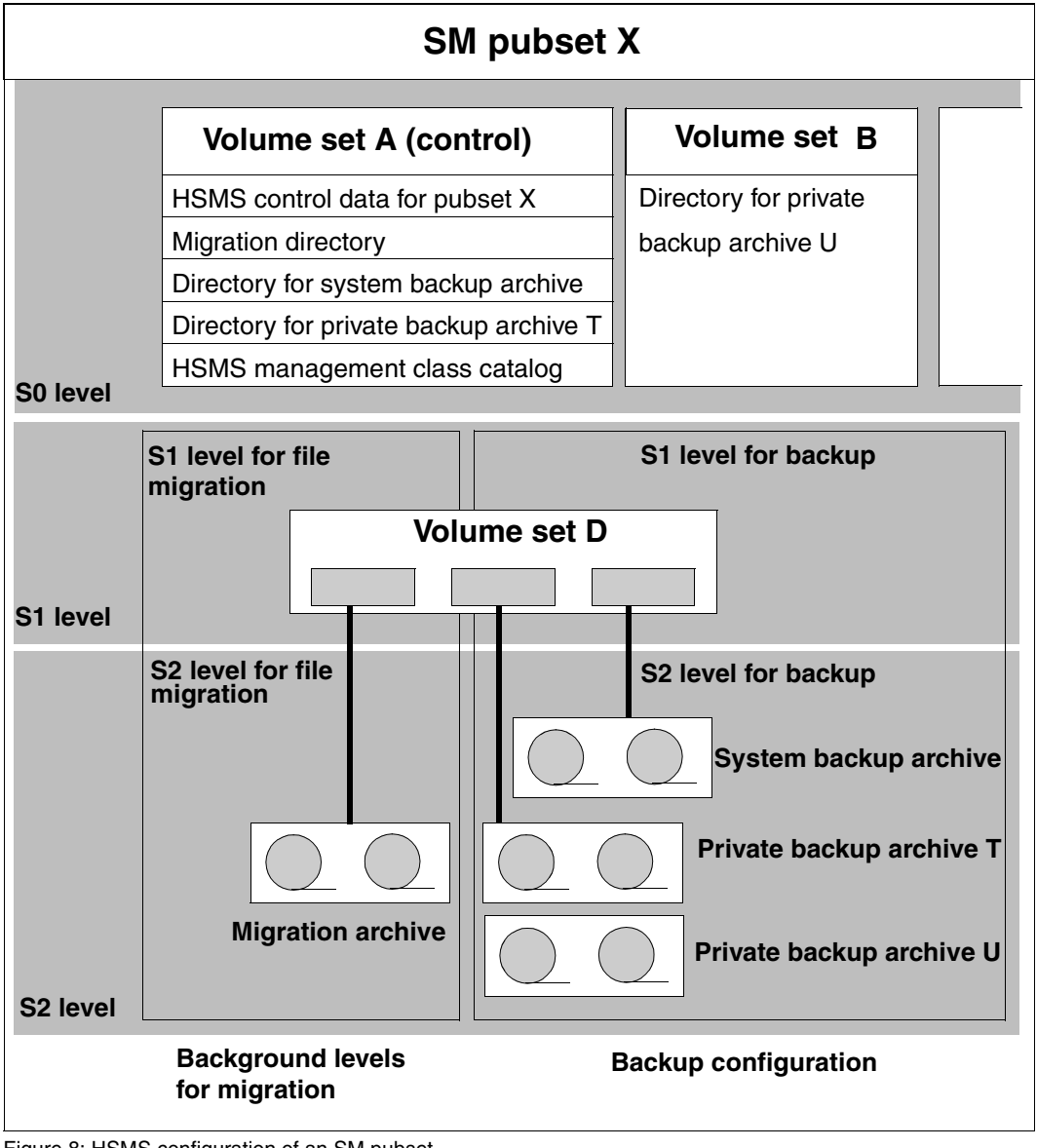


Figure 8: HSMS configuration of an SM pubset

If systems support sets up an HSMS configuration for an SM pubset (HSMS statement CREATE-SM PUBSET-PARAMETERS) the pubset acquires the status HSMS-SUPPORTED. This means that it is not possible to use the pubset without restrictions unless HSMS is available.

4.6.1 Configuring background levels

HSMS functions can be used to configure SM pubset background levels for file migration. These background levels are exclusively allocated to the SM pubset (unlike in SF pubsets) and cannot be used for other SF pubsets or SM pubsets. Unlike the processing level, these are not obligatory components of the pubset.

The most important characteristics of background level configurations are:

- the name of the migration archive
  If systems support does not specify a name explicitly HSMS assigns a default name to the migration archive.
- the migration directory
  If systems support does not specify a name explicitly, HSMS assigns a default name to the migration directory. It must be located in the SM pubset to which it is assigned (unlike SF pubsets!). By default HSMS creates it in the control volume set within the SM pubset unless systems support explicitly specifies another volume set as the storage location. This should only be done in special cases since it runs counter to the principle of concentrating pubset metadata in the control volume set (affects pubset availability and pubset reconfigurability).
- the name of the volume set which acts as the S1 level if an S1 level is required
- the default device type of the S2 volumes and name of the tape pool which acts as the S2 level if an S2 level is required
- the control parameters for migration

4.6.2 Backup configuration of an SM pubset

HSMS makes it possible to set up backup archives in an SM pubset. The backup archive of an SM pubset are pubset-specific (unlike SF pubsets). This means that:

- Files located in the SM pubset can only backed up in a backup archive belonging to the SM pubset.
- The backup archive of an SM pubset can only contain backup files belonging to the SM pubset to which it is allocated.

Restrictions analogous to those applying to backups do not exist when backups are restored. It is possible to restore backups from any available backup archive into an SM or SF pubset.

The backup archive of an SM pubset is characterized by:

- the name of the backup archive,
- the name of the allocated directory; this must be located in the SM pubset to which the backup archive is allocated,
- the default device type of the backup volumes and the name of the tape pool in which the volumes are located.
- Unlike in SF pubsets, the backup archives of SM pubsets may not contain disks as volumes, the volume set used for backups saved to disk. The volume set that also serves as the S1 level for background migration is used for this. Since original files and backup copies should be kept on different volumes, it is not suitable for backing up files that have been migrated to the S1 background level,
- backup control parameters.

Among the backup archives of an SM pubset the system backup archive has a special role to play. It is used by default if the user does not explicitly specify the backup archive during backup and restore activities. HSMS defines the names of the system backup archive and associated directory by default unless they are explicitly specified by systems support. The directories of the backup archives of an SM pubset must be located in the pubset itself. By default HSMS creates them in the control volume set of the pubset. It is particularly important that another volume set is explicitly specified as the location of the system backup archive directory in special cases only since this runs counter to the principle of concentrating pubset metadata in the control volume set (affects pubset availability and pubset reconfigurability).

4.6.3 Long-term archives

Long-term archives can have a longer lifetime than the pubsets whose archived files they contain. The long-term archives can therefore be set up that are not pubset-specific. They are not assigned to a specific SF or SM pubset and can be used both to create and restore file archives of different pubsets. However, it is also possible to define pubset-specific long-term archives.

4.6.4 HSMS management class catalog

HSMS management classes allow users to control migration and backup activities within an SM pubset. The HSMS management class catalog is used as the storage location for the HSMS management classes defined for an SM pubset. HSMS implicitly creates this catalog in the control volume set of the SM pubset.

4.6.5 HSMS files

Alongside the directories for the migration archive and the backup archives and the HSMS management class catalog, there are other HSMS files for SM pubsets, the most important of which are described below:

- In addition to the computer-specific (central) HSMS control files ($SYSHSMS.HSMS. 1.CONTROL.FILE) there is a pubset-specific (local) control file ($SYSHSMS.HSM.1.SM.CONTROL.FILE) for each SM pubset. The archive definitions for the migration archive and the backup archives for the SM pubset in question are recorded in this file.
- The systems support staff can specify an except file for an SM pubset. This file contains the names of files which may not be migrated to background levels. This file must be located in the SM pubset.
- The HSMS job file of an SM pubset ($SYSHSMS.HSM.2.SM.REQUEST.FILE) is used to administer the migration and backup jobs created for a pubset.
- The ARCHIVE checkpoint file of an SM pubset is used to save restart points for the HSMS jobs which are active in the pubset. This makes it possible to restart jobs following an interruption, e.g. because of a system crash.
- The result of an HSMS job is entered in the result files (result file and report file).
- Users can specify whether the result of an HSMS job is to be output at a printer or to be stored in print-ready form in a specified report file. The catalog ID in the file name specifies the pubset to which the report file is sent.
- A temporary auxiliary file is required when recalling a file from a background level to the processing level. It is set up in the pubset in which the recall is performed.

The HSMS files of an SM pubset provide particular support for pubset switching capabilities since, unlike the corresponding HSMS files for SF pubsets, they are not computer-specific but instead pubset-specific. With the exception of the report file, the location of which is specified by users, they are located in the SM pubset to which they have been allocated.

Within an SM pubset, the HSMS files which are automatically created by HSMS and which are important for the execution and restart capabilities of HSMS functions (control file, job file, checkpoint file, results files) are created in the control volume set.

The control volume set is the recommended location for the except file which is created by systems support. The HSMS functionality of the SM pubset is therefore not affected by pubset reconfiguration or the failure of individual volume sets. The control volume set is an obligatory component of the SM pubset and cannot be removed via pubset reconfiguration during the lifetime of the pubset.

4.7 Use of ARCHIVE

ARCHIVE supports SM pubsets in the same way as SF pubsets. In contrast to HSMS, however, ARCHIVE does not offer file migration or any special functions for the additional structures of SM pubsets. In particular, ARCHIVE does not permit file selection on the basis of management classes or storage classes, identification of an S1 volume set or restoration to a specific or original volume set.

Customers who are interested in the SM pubset in its capacity as a switchover unit should select distributed administration in ARCHIVE operation (directories per SM pubset and in the SM pubset) because this is more favorable for switchovers. They can then plan an easier transition to HSMS. Moreover, the simple takeover of the directories of SM pubsets when changing from ARCHIVE to HSMS operation is facilitated if the directories have a catid in archive operation (PARAM CATID=*YES).

4.8 SF pubset characteristics with no SM pubset equivalent

Use as home pubset

SM pubsets cannot be used as home pubsets. Consequently:

1. SM pubsets must not contain any files that are required for system initialization (IPL files, REP files, SLED files, IOCONF files, etc.). The volumes of an SM pubsets cannot have any boot blocks.
2. No SNAPSHOT files can be created on SM pubsets.

SPEEDCAT use

SPEEDCAT settings are not available for SM pubsets since, by default, these possess functionality which corresponds to that of SPEEDCAT.

General physical allocation authorization

The ability to permit all users to perform physical allocation in a pubset is not available for SM pubsets since it has been superseded by an improved functionality (physical allocation rights for individual users).

Converting back the file catalog

Converting the file catalog back to a format required in operating system versions < BS2000/OSD V3.0 is irrelevant because these versions are no longer supported.

4.9 Use of an SM pubset on a system

Before a pubset (SF or SM) can be put into service on a system, an MRSCAT entry must be created for the pubset on this system. Among other things, this stores information required by the system when the pubset is put into service by systems support and allows the specification of system-specific operating parameters and constraints on usage that apply for a pubset session. As a minimum requirement, the name of the control volume set must be entered for SM pubsets in the MRSCAT entry. The commands ADD/MODIFY-MASTER-CATALOG-ENTRY and MODIFY-PUBSET-CACHE-ATTRIBUTES are available to systems support staff for creating an MRSCAT entry and changing its contents.

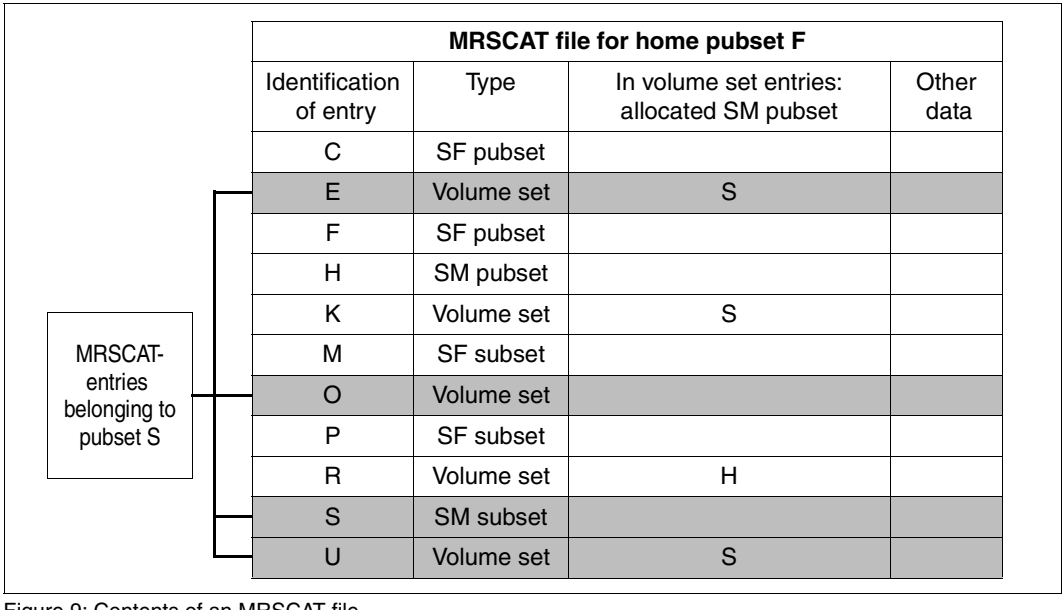
In addition to the MRSCAT entry of the SM pubset, the system requires an MRSCAT entry for each of its volume sets as a system-internal resource. This applies regardless of the mode in which the pubset is put into service (exclusive, shared-master or shared-slave). The MRSCAT entries for the volume sets are created by the system automatically when the pubset is put into services, unless they already exist. This is only possible provided there is no MRSCAT entry that is being used for another purpose whose ID collides with that of one of the volume sets. In this case, the putting into service of the pubset is aborted. An MRSCAT entry is said to be used for another purpose if it is not of the appropriate type (i.e. of the type SF or SM pubset) or if it is assigned to a volume set of another SM pubset. MRSCAT entries that lead to collisions must be deleted by the systems support staff before the SM pubset goes into service.

4.9.1 MRSCAT entries for SM pubsets

The MRSCAT file $TSOS.SYSTEM.MRSCAT on the home pubset of a system acts as a container for the MRSCAT entries created on a system. It can contain the following different types of MRSCAT entries: entries for SF pubsets, SM pubsets and volume sets. Figure 9 shows an example of an MRSCAT file. An MRSCAT entry is identified by the ID of the SM pubset, SF pubset or volume set. To ensure that the MRSCAT entries can be addressed uniquely, the IDs of all the SF pubsets, SM pubsets and volume sets entered in the MRSCAT file must differ from each other.

For as long as the SM pubset is in operation on a system, the MRSCAT entries of the volume sets that belong to the pubset are required there as resources and cannot be deleted by systems support staff. When the SM pubset is taken out of service on the system, the MRSCAT entries of the volume sets remain until they are removed. The system-specific status of a (volume set) MRSCAT entry describes the current situation. It is stored in the MRSCAT entry of the volume set and can take on the values *connected* (pubset to which the volume set belongs is in service) and *not connected* (pubset to which the volume set belongs is out of service).

The system also uses the MRSCAT entries of the SM pubset and of its volume sets internally to store specific pubset attributes anchored in the SM pubset itself redundantly in order to be able to access them quickly for the purpose of read access. For example, certain pubset information functions such as SHOW-MASTER-CATALOG-ENTRY or SHOW-PUBSET-PARAMETERS use the MRSCAT entries to determine the pubset and volume set attributes. During a pubset session, the system ensures the consistency of the data in the MRSCAT entries. After a pubset is taken out of service, the MRSCAT entries assigned to it on this system remain unchanged. If the pubset is then changed from another system, discrepancies can result. Consequently, whenever a pubset is put into service on a system, the MRSCAT entries there are automatically updated by the system to ensure consistency.



MRSCAT file for home pubset F			
Identification of entry	Type	In volume set entries: allocated SM pubset	Other data
C	SF pubset		
E	Volume set	S	
F	SF pubset		
H	SM pubset		
K	Volume set	S	
M	SF subset		
O	Volume set		
P	SF subset		
R	Volume set	H	
S	SM subset		
U	Volume set	S	

Figure 9: Contents of an MRSCAT file

4.9.2 Specifications for putting a pubset into service

We will restrict ourselves to a brief description of the specifications stored in the MRSCAT entry that are different for SM pubsets. You will find a detailed description in the “Introductory Guide to Systems Support” [5].

1. ID of the control volume set (difference to SF pubsets)

In order to put an SM pubset into service, the system requires specific data that is stored on the VOLRES of the control volume set. The VOLRES plays a similar role for an SM pubset to that played by the PUBRES for an SF pubset. The system obtains its name from the ID of the control volume set, which the systems support staff have to make known to the system by means of the SM pubset’s MRSCAT entry.

2. Specifications for the system when cache areas cannot be reconnected

When a pubset is put into service, a check is carried out to establish whether individual volume sets are still linked with cache areas as a result of the pubset not being taken out of service properly previously. The system attempts to reconnect these in order to avoid data losses. If this is not possible, in the case of volatile cache media (main memory, global storage with VOLATILITY=*YES) the system proceeds to put the pubset into service without reconnection.

In the case of nonvolatile cache media (global storage with VOLATILITY=*NO), the following alternatives are possible if reconnection is unsuccessful:

a) The process of putting the pubset into service is aborted. Before it is put into service again, the systems support staff identify and deal with whatever it was that prevented the cache areas from being reconnected.

b) The system asks the operator whether the process of putting the pubset into service is to be aborted or whether it should be continued without reconnection of the problematic cache areas.

The system finds out which of these alternatives the systems support staff want from the FORCE-IMPORT setting in the pubset’s MRSCAT entry. If cache areas are not reconnected, files that have output data stored in them temporarily can no longer be used normally. It should be noted that the FORCE-IMPORT setting always applies to the entire SM pubset (i.e. to all the associated volume sets). Differentiated treatment of individual volume sets is not possible.

3. *Specifications for the system in the event of problems reconnecting cache areas*

The reconnection of cache areas is only necessary in the exceptional case when the previous pubset session was not terminated with the exporting of the pubset in the normal way. Normally, when the pubset is put into operation, the system attempts to provide new cache areas for the different volume sets of the SM pubset provided the system support staff have specified this by means of the MODIFY-PUBSET-CACHE-ATTRIBUTES command. If the required cache media are either not available or not available to the extent required, or if there are any other problems in making them available, the following alternative courses of action are possible:

a) The process of putting the pubset into service is aborted. Before it is put into service again, the systems support staff should identify and deal with whatever it was that was preventing the required cache configuration from being made available.

b) The pubset is put into service, but individual volume sets may not receive any cache areas or may receive smaller cache areas than specified in the configuration file.

The systems support staff use the SIZE-TOLERANCE entry in the pubset's MRSCAT entry to tell the system how it should proceed in this case.

It can also happen when the START-PUBSET-CACHING command is executed for a volume set or when a free volume set is added to an SM pubset that the cache required for the volume set cannot be made available, or at least not to the extent required. In this case, as well, the response of the system can be controlled by SIZE-TOLERANCE.

The SIZE-TOLERANCE setting also always applies to the entire SM pubset and does not permit differentiated handling of individual volume sets.

4.9.3 Other specifications for pubset usage

There are no differences between SF and SM pubsets in terms of the specifications as to whether or how a pubset is to be used in the pubset network (see the ADD/MODIFY-MASTER-CATALOG-ENTRY command – particularly the operands SHARED-PUBSET, BATCH-WAIT-TIME, DIALOG-WAIT-TIME and PARTNER-NAME – and the MODIFY-PUBSET-ATTRIBUTES command). In particular, specifications for SM pubsets, as for SF pubsets, generally apply to the whole SM pubset rather than individual volume sets.

This also applies to the specifications of the system-specific operating parameters, of the buffer areas for catalog management and for the control of EAM (ADD/MODIFY-MASTER-CATALOG-ENTRY commands with the operands RESIDENT-BUFFERS, NUMBER-OF-BUFFERS and EAM).

For more detailed information, refer to the "Introductory Guide to Systems Support" [5].

5 Setting up an SM pubset

The generation of a fully configured SM pubset (with storage classes, background levels, backup archives, HSMS management classes, GUARDS protection profiles, etc.) requires a number of steps. It is possible to vary the scope, sequence and time of execution of these steps. The first step always consists of generating a basic pubset using SIR. This basic pubset can then be extended and modified to meet the specific requirements of individual users. The procedure with SIR is described in sections 5.1 through 5.3.

A special option for generating a renamed copy of an SM pubset is provided by PVSREN with the function “cloning an SM pubset” for mirrored pubsets (BCV mirror, see the “SHC-OSD” manual [8]. This option is described in section “Cloning an SM pubset with PVSREN” on page 63.

5.1 Overview of SIR functionality for SM pubsets

SIR offers functions for pubset generation and the maintenance of existing pubsets for both SF pubsets and SM pubsets. The following points apply specifically to SM pubsets:

- Because of their more complex structure, SM pubsets require more extensive functionality in SIR for pubset generation.
- That part of the functionality that is only of relevance to the generation and maintenance of home pubsets is not made available for SM pubsets because SM pubsets cannot be used as home pubsets. The functionality for creating and maintaining SNAPSHOT files and for setting up IPL capability on disks of the pubset is not supported for SM pubsets.
- The functions for maintaining existing SM pubsets allow new volumes to be added to volume sets (with implicit volume initialization, if appropriate), new volume sets to be added to an SM pubset and catalog files to be maintained. They represent an alternative to the functions for dynamic pubset reconfiguration.
- It is possible to create free empty volume sets for SM pubsets. Free volume sets are characterized by the fact that they do not belong to an SM pubset and that, except in certain special situations, they are empty. Apart from the their catalog file, they do not contain any other files. The free volume sets are an aid to pubset reconfiguration. SM pubsets can be expanded by adding free empty volume sets.

5.2 Generating a new SM pubset using SIR

5.2.1 Starting point

The starting point for the generation of an SM pubset may take the form of individual volumes or previously generated free volume sets:

a) Individual volumes

   During pubset generation, SIR forms a volume set from the individual volumes in accordance with systems support's specifications. This represents an internal intermediate processing step. When this is done the volumes are formatted if they have not already been formatted or if their existing formats do not correspond to the required format for the volume set. The volume sets are also assigned the DRV, RAID or REMOTE-COPY attributes required by systems support.

b) Previously generated free volume sets

   With the exception of the control volume set which is always regenerated as part of pubset generation, free volume sets can be taken over complete in the SM pubset. Fixed volume set characteristics (e.g. volume configuration, REMOTE-COPY, DRV, format, allocation unit) cannot be modified.

5.2.2 Preparation for pubset generation

As with SF pubsets, there does not have to be an MRSCAT entry for the pubset. In this case, it is created automatically by SIR. However, if an MRSCAT entry has already been created and its identification corresponds to the pubset identification, the information stored in the entry (pubset type, control volume set ID, VOLRES device type) is compared with systems support's specifications.

When compared to SF pubsets SM pubsets make it necessary to take account of new considerations relating to integration in a computer's existing pubset configuration. It is necessary to ensure that MRSCAT entries can be created not just for the pubset itself but also for each of its volume sets at all computers at which the SM pubset is to be used subsequently. This is only possible if there are no identically named MRSCAT entries which are required for other SF or SM pubsets or the volume sets of other SM pubsets. Before the SM pubset is generated, systems support should therefore ensure that there are no conflicts with existing configurations at any computer at which the SM pubset is to be used subsequently. This is checked by SIR at the computer at which the SM pubset is generated. Generation is not performed if this computer contains an MRSCAT entry which coincides with the ID of one of the volume sets of the SM pubsets.

At the computer at which the SM pubset is generated, an MRSCAT entry is created for each of its volume sets. A precondition here is that the maximum possible numbers of MRSCAT entries at the computer has not been exceeded (system parameter DMCMAXP). If this precondition is not met the SM pubset is not generated.

5.2.3 Specifying pubset characteristics

When specifying individual pubset characteristics using SIR it is necessary to distinguish between the following cases:

- Predefined by systems support
- Determined in accordance with a set procedure
- Not taken into account during SIR generation

Only a small number of pubset characteristics belong to the first category which provides explicit control facilities for systems support. This fact is based on the idea that SIR creates a basic pubset which is then processed in subsequent installation steps using the other pubset maintenance functions. Control facilities are primarily provided for basic pubset characteristics which cannot be modified later or whose modification is problematic. The individual characteristics and the way they are defined are discussed in brief below.

Pubset characteristics which can be influenced by systems support

Pubset identification

When specifying pubset identifications it is necessary to comply with the name conflict conditions specified above. In particular, the pubset ID must differ from the IDs of all the volume sets belonging to the pubset.

Physical structure

The following specifications are possible when defining the physical structure of the SM pubset:

– control volume set of the SM pubset

– number of other volume sets belonging to the pubset

– number of volumes belonging to the individual volume sets (for volume sets which have not been previously generated)

– format and size of the allocation unit of the individual volume sets (for volume sets which have not been previously generated)

– DRV, RAID or REMOTE COPY support for the individual volume sets (for volume sets which have not been previously generated)

Location and size of the catalog files

SIR allows systems support to specify the size and the location within the volume sets of the catalog files of the individual volume sets ($TSOS.TSOSCAT.<volset-id>). The size and location of the global pubset catalog files ($TSOS.TSOSCAT.#JVC/ #MIN/ #PVT/ $NLO/ $NLC/ $PFI) is automatically determined on the basis of the size of the catalog files for the individual volume sets. If systems support is not satisfied with the extent to which they are able to influence global pubset catalog files they should create small catalog files for the individual volume sets on generation. This also results in the creation of small global pubset catalog files. All catalog files can subsequently be enlarged with the pubset online. This also makes it possible to specify their location on the individual volume using the physical allocation facilities.

Volume set attribute profiles

Alongside the format and the size of the allocation unit which is directly assigned to the individual volume sets as part of initialization/formatting, the volume set attribute profiles also define the performance and availability profiles. Systems support should specify these in such a way that they correctly describe the physical attributes of the volume sets. If these are then modified as a result of measures taken after pubset generation - for example through the allocation of cache areas in order to enhance performance - then it is these subsequent physical attributes that have to be taken into account

Volume set usage types

It is possible to predefine the usage type for the individual volume sets (STD, WORK, HSMS-CONTROLLED).

Starting configuration with user files

Systems support may cause files from other pubsets, private disks or tapes to be copied to the SM pubset during pubset generation. However, unlike in SF pubsets, it is not possible to modify the pubset-internal location of files.

Large objects attribute

The systems support staff must specify the large objects attribute (see section “Large objects attributes” on page 38) for the SM pubset explicitly or implicitly. SIR validates the specifications on the basis of the physical configuration. An SM pubset which can incorporate volume sets with large volumes is not set up unless the systems support staff have specified LARGE-DISKS-ALLOWED=*YES. If the SM pubset is also to be configured for large files, the systems support staff must specify LARGE-DISKS-ALLOWED=*YES(LARGE-FILES-ALLOWED=*YES).

Pubset characteristics preset by SIR

The following pubset characteristics are determined using a procedure which cannot be modified:

Configuration status of volume sets

When an SM pubset is generated all its volume sets have the configuration status “normal use”.

Usage restrictions for volume sets and volumes

There are no usage restrictions attached to volume sets. The same applies to volumes of volume sets which are created as part of pubset generation. In the case of previously generated free volume sets the question of whether or not the associated volumes are affected by allocation restrictions depends on the volume set history.

Cache allocation for volume sets

During pubset generation, the required cache configuration and cache operating parameters of each volume set are preset with default values (CACHE-MEDIUM=*NO, CACHE-SIZE=...). Unlike in SF pubsets, in SM pubsets these default settings are not entered in the pubset’s MRSCAT entry but are stored in the pubset itself. If no pre-existing MRSCAT entry exists for the pubset on pubset generation then this is automatically created. The settings for FORCE-IMPORT and SIZE-TOLERANCE which systems support uses to handle cache problems when taking the pubset into service are initialized with default values (FORCE-IMPORT=*NO, SIZE-TOLERANCE=*YES).

Saturation thresholds of volume sets

The value BY-SYSTEM is assigned to the saturation threshold (defined values) of the individual volume sets. This setting means that when the pubset is taken into service the current values for the individual volume sets are determined by the system in accordance with a specific calculation procedure which takes account of both volume set capacity and the system parameter L4SPDEF.

Default pubset space settings (including the default file format)

The default values for primary, secondary and maximum allocation are set in accordance with the DMMAXSC, DMPRALL, DMSCALL system parameters which are effective at the computer at which generation is performed. The default value for the file format is the same as the format of the control volume set.

User catalog

During pubset generation, the pubset is initially taken into service with ACTUAL-JOIN=*FIRST. This creates a user catalog with default user IDs.

Location and size of system files

With the exception of the catalog files, systems support cannot influence the size and location of system files created by SIR ($TSOS.SYS.PUBSET.CONFIG, $TSOS.SYSSRPM,..).

Use of pubsets in networks

The required pubset master and backup master, shareability etc. are specified using the same default values as for SF pubsets.

Pubset characteristics not affected by SIR

The following are not set up during pubset generation:

– storage classes

– GUARDS profiles

– user groups

– SYSEAM file

– HSMS environment with background levels, backup archives and HSMS management classes (pubset status: HSMS-SUPPORTED=*NO)

5.3 Other installation measures

The pubset generated by SIR provides the basis for further installation measures. The sequence and timing of these measures is not predetermined. They need not necessarily be performed during pubset installation but may instead be conducted when required as part of pubset maintenance. In general it is advisable to take the newly generated pubset into service specifically for follow-up processing and then to modify the settings made by SIR in order to meet current requirements. It is also then possible to set up the pubset environment which is ignored by SIR. The most important interfaces and products which make this possible are listed below. The pubset must be online before these can be used.

5.3.1 Adapting the presettings made by SIR

Enlarging the file catalog

The catalog files set up by SIR can be enlarged. Physical allocation makes it possible to influence the location of catalog files on volumes.

Usage restrictions for volume sets

The usage restrictions desired for individual volume sets or volumes can be set using the MODIFY-PUBSET-RESTRICTIONS command.

Saturation thresholds for volume sets

The MODIFY-SPACE-SATURATION-LEVELS command is used to adapt the saturation thresholds required for the individual volume sets.

Pubset space default values (including the default file format)

The MODIFY-PUBSET-SPACE-DEFAULTS command is used to adapt the default values for pubset space.

Storage classes

The storage class management functions can be used to set up storage classes. The storage class catalog can also be restored from backups.

User entries

Systems support can use the user administration commands ADD-USER, MODIFY-USER-ATTRIBUTES) to create the required user entries. It is also possible to load backups of user catalogs that may have been created on other pubsets, including SF pubsets.

HSMS environment

The HSMS functions (CREATE-ARCHIVE, CREATE-SM-PUBSET-PARAMETERS, CREATE-MANAGEMENT-CLASS, etc.) can be used for the generation of HSMS environment, of HSMS configuration for background levels and backup as well as for the setting up of HSMS management classes.

Cache allocation for volume sets

In the case of PFA, the MODIFY-PUBSET-CACHE-PROCESSING command can be used for a volume set to change the default settings for the desired cache configuration and cache operating parameters. In addition, appropriate cache configurations must be set up at the physical level. The cache is connected the next time the pubset is put into service or when the START-PUBSET-CACHING command is called for the individual volume sets.

5.4 Cloning an SM pubset with PVSREN

In Symmetrix disk subsystems, the Symmetrix Multi Mirror Facility permits disks to be assigned additional mirrors as BCVs (Business Continuance Volumes) which, if required, can be separated and used independently (see the “SHC-OSD” manual [8]). PVSREN enables an independent SM pubset with new catalog IDs for pubset and volume sets to be generated from the BCV mirrors of the disks in an SM (and also an SF) pubset. This procedure is known as “cloning” a pubset. In terms of the data a copy of the source pubset is concerned here. Typical applications for cloned pubsets would be evaluations on the basis of a frozen data inventory or use for test purposes.

Further details on cloning a pubset are provided in the “PVSREN” chapter of the “Utility Routines” manual [1].

6 Generating an SM pubset from existing SF pubsets

6.1 Basic considerations

If users are to be able to use SM pubsets for an existing file store, the existing SF pubsets must be converted into an SM pubset. It is particularly simple to transform an SF pubset into a single SM pubset of the same name. However, it is also often desirable to combine multiple SF pubsets to form a single SM pubset. There are a number of different possible ways of performing the conversion and we consider these in detail below. The following basic considerations must be taken into account irrespective of the procedure chosen.

Modifying the addressing of files, job variables, GUARDS profiles

As the catalog ID, the pubset identification is a component of the path name of the files cataloged in the pubset. If an SF pubset is to form part of an SM pubset with an ID which differs from that of the SF pubset, the addressing of the files is modified (see figure 10). The same is true of job variables and GUARDS profiles. In this section we consider the problem of addressing using the example of files.

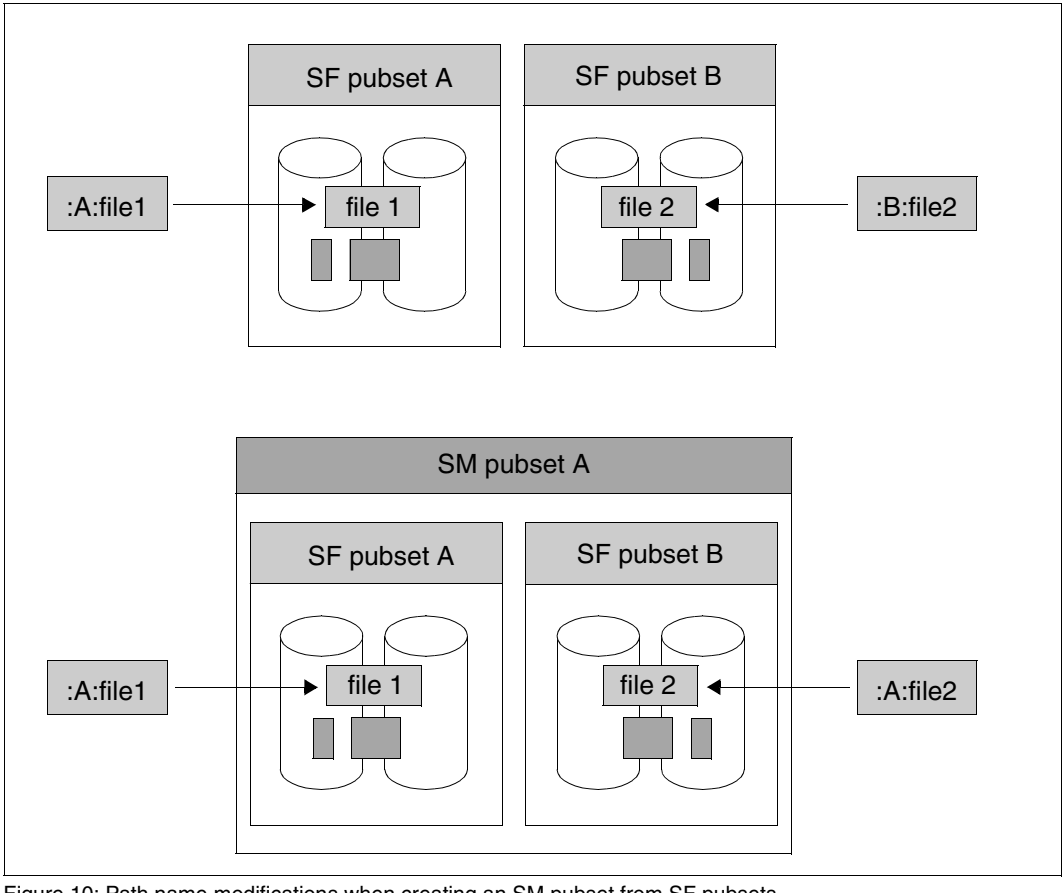


Figure 10: Path name modifications when creating an SM pubset from SF pubsets

The modifications to file addressing can be catered for as follows:

- If users have used the default catalog ID (i.e. the catalog ID of the default pubset) in order to address files, systems support simply needs to adapt the default catalog IDs in the user entries (of the potential home pubsets).
- If users have addressed files using the complete path names, i.e. with an explicit catalog ID specification (e.g. in command procedures, user programs or other user data structures), it is necessary to adapt the path names. Program names referenced in GUARDS profiles are one example of the use of complete path names.

- One reason for the use of complete path names may be that users choose to distribute their files across multiple SF pubsets in order to make use of the specific attributes of the individual SF pubsets (performance, availability, format). If command procedures or user programs contain statements which cause files to be set up with fully specified path names, it may be necessary to perform adaptations in order to ensure that the files are sent to a suitable volume set within the SM pubset. The required attributes (performance, availability, format) must be explicitly assigned to the file using file attributes.
- For files in SF pubsets which are created with a default catalog ID, certain attributes implicitly result from the attributes of the associated default pubset. In order to ensure that files which are created using the default catalog ID retain their previous attributes after being taken over into an SM pubset, systems support must allocate storage classes to the individual users which have the same attributes as the previously allocated default pubsets. The command procedures and programs do not have to be changed. If no file attributes are specified when a file is created, the values stored in the user's default storage class are used for the file.

Conflicts and excessively long path names

If differing SF pubsets which are to be combined in the same SM pubset contain files with path names which differ in the catalog ID only, name conflicts will occur during the formation of the SM pubset, i.e. the path names would no longer be unambiguous. This can only affect users whose files are distributed across multiple SF pubsets. If an SF pubset is incorporated in an SM pubset with an ID which contains more characters than that of the SF pubset then the path names of the files located in the SF pubset are extended. This may mean that the maximum path name length is exceeded. Name conflicts or excessive path name lengths are handled as follows:

- The affected files are renamed or deleted.
- Statements which reference renamed files are adapted in command procedures or programs.

Incompatible commands, statements, program interfaces

Some system commands, program interfaces and statements used in individual utilities cannot be used in SM pubsets in every form which is permitted in SF pubsets. Thus HSMS backup, restore, migrate and recall jobs must be pubset-specific when used for SM pubsets. Jobs cannot be active across different pubsets unlike the case with SF pubsets. This also affects the private disk files, for example. In order to back them up in their entirety, a separate backup request is required for each SM pubset in which a private disk file is cataloged. Command procedures and programs affected by such incompatibilities have to be adapted.

Adapting the MRSCAT entries

The MRSCAT entries must be adapted at all computers at which the SM pubset is to be taken into service:

- An SM pubset type MRSCAT entry must be created.
- The MRSCAT entries of the SF pubsets from which the SM pubset is formed are no longer required and should be deleted.
- For each volume set within the SM pubset it is necessary to make sure that no MRSCAT entry for an SF pubset, SM pubset or volume set of another SM pubset is present with an ID which corresponds to that of the volume set.

Other references to SF pubset IDs

All other references to the IDs of SF pubsets which are no longer valid following the generation of the SM pubset must be adapted. For example, the default catalog IDs in the user entries of potential home pubsets, pubset IDs in command procedures and user programs.

6.2 Overview of procedures

The processing level, background levels and backup archives must be included in the conversion of an SF pubset into an SM pubset. We outline the possible procedures below. They are described in greater detail in the sections that follow.

6.2.1 Adapting the processing level

The processing level can be adapted in the following ways:

a) In-place conversion

The in-place conversion of processing level is made possible by the SMPGEN utility. This makes it possible to generate a new SM pubset from one or more SF pubsets or to extend an existing SM pubset (see also the section “Extending an SM pubset with SF pubsets” on page 161). In this case, each SF pubset is transferred to a volume set. The user files located in the SF pubsets are retained unmodified during the merge operation. The metadata files of the SF pubsets such as file catalogs, user catalogs, GUARDS catalogs are converted into the corresponding SM pubset metadata files. Certain pubset and volume set characteristics, user quotas, etc. are automatically ascertained for the SM pubset by SMPGEN. If systems support wants to use differing settings then it is necessary to process the SM pubset following generation.

When SF pubsets are used, a certain level of availability or performance can be achieved for files by locating them in special SF pubsets (such as pubsets mirrored using DRV or REMOTE COPY, etc.) without it being necessary to specify these attributes explicitly using file attributes. Files located in SF pubsets with special attributes must be assigned corresponding file attributes during the generation of the SM pubset. In the case of SF pubsets which are transferred to volume sets with the attribute AVAILABILITY=*HIGH, this is automatically performed by SMPGEN. In the case of SF pubsets with caches which are used via PFA, appropriate performance attributes must be assigned to the files before the SM pubset is generated. In all other cases, e.g. in the case of SF pubsets which consist of volumes emulated in global storage, etc. systems support must assign attributes explicitly. If no appropriate attributes are assigned to the files, SMPGEN does not initially change their location during pubset generation. However, a new storage location is determined if they are subsequently backed up and restored or migrated and recalled to the processing level and the non-adapted file attributes are taken into consideration.

b) Backing up and restoring files

The files present in the processing level are backed up for all SF pubsets which are incorporated in the SM pubset. This backup may also be performed as part of a full backup which involves background levels. SIR is then used to generate the SM pubset. Systems support then creates user entries in the SM pubset for the existing SF pubset users.

This process is supported by HSMS which is used to restore the backed up SF pubset user catalog. The user attributes which have been newly introduced for SM pubsets (e.g. user quotas for high availability files) are assigned default values which ensure the highest possible compatibility with the SF pubsets. If other users were present for the remaining SF pubsets which are to be incorporated into the SM pubset then systems support must use the ADD-USER command to create user entries for these users explicitly. GUARDS catalogs are then set up and once again it is possible to use SF pubset backups. Finally the user files are restored.

Prior to backup, suitable file attributes must be assigned to files which were previously located in SF pubsets offering special performance or availability in order to ensure that they are written to suitable volume sets in the SM pubset when they are restored.

If systems support wants to affect the distribution of files in the SM pubset by means of default storage classes with assigned volume set lists when performing the restore operation, it is first necessary to set up appropriate storage classes and assign these to individual users as default storage classes. These classes are not taken into consideration by HSMS during the restore operation unless the specification STORAGE-CLASS=*STD is present in the HSMS statement RESTORE-FILES.

In-place conversion using SMPGEN makes migration to SM pubsets an extremely comfortable and convenient operation. This is also particularly true of the creation of the user catalog and GUARDS catalogs in cases where multiple SF pubsets are combined to form the SM pubset. However, if the SM pubset requires a different physical structure from that resulting from the in-place conversion of the SF pubsets, conversion must be performed by backing up and restoring the files. One example of this is the conversion of an SF pubset consisting of a large number of volumes into an SM pubset which consists of multiple small volume sets.

6.2.2 Adapting the background levels

The background levels can be adapted for file migration as follows:

a) In-place conversion of the S2 level

The HSMS utility DIRCONV can be used to create a new directory from the SF pubset migration directories. This directory can subsequently be used as the migration directory of the SM pubset. To do this it is not necessary to access the volumes containing the migrated files. However, certain conditions must be fulfilled before in-place conversion is possible and in certain circumstances the preliminary use of HSMS functions is necessary. For example, no files may be migrated to the S1 level in the affected SF pubset. In order to meet the conditions for in-place conversion of the migration directories it may also be necessary to access the volumes containing the migrated files.
The extent of these clean-up measures depends to a considerable extent on the application environment in question.
To transfer the S2 levels of the SF pubset to the SM pubset, systems support must adapt the processing level and set up an HSMS environment for the SM pubset and, in doing so, specify that the directory which was previously created using the conversion procedure is to be the migration directory.

b) Adapting the background level by backing up and restoring files

The files located in the background levels of the SF pubsets are backed up (if necessary as part of a full backup of the SF pubsets which are incorporated in the SM pubset including their background levels). Once the processing level has been adapted, an HSMS environment with (initially empty) background levels is set up in the SM pubset. The backups of the files which were previously located in the background levels of the SF pubsets are then restored to these background levels.

The in-place conversion of the S2 level is preferable to adaptation by means of backup and restore in cases when the preconditions for the conversion of the migration directories are already fulfilled or can be easily met.

6.2.3 Adapting the backup archives

The backup archives of the SF pubsets which are incorporated in the SM pubset can be adapted using a procedure similar to that employed for the migration archives by using DIRCONV to create directories suitable for the SM pubset from the existing directories. When the HSMS environment has been set up in the SM pubset, pubset-specific backup archives can be set up in it. The appropriately prepared backup directories can be assigned to these backup archives. This is particularly true for the system backup archive of the SM pubset.

DIRCONV adapts the backup directories in-place, i.e. without accessing the volumes on which the backups are stored. As in the case of migration directories, certain preconditions must be fulfilled before conversion is possible. In general these conditions are considerably more difficult to meet in the case of existing backup archives and in most cases it is necessary to access the volumes containing the backups. The volume of preparatory work required is highly dependent on the application environment in question.

In cases where it is too time-consuming or difficult to adapt the existing backup archives, it is advisable to set up a new backup archive immediately before generating the SM pubset (i.e. after the clean-up measures necessary for the adaptation of the processing level have been taken and the SMPGEN checks reveal no further violations) and to transfer current backups to this new backup archive.

In the case of the backup archive which is subsequently converted into the system backup archive of the SM pubset, it is advisable to make a current full backup of all the SF pubsets which are to be incorporated in the SM pubset. This procedure avoids the need for all the difficult clean-up measures involved in the adaptation of archives.

For example, no file name conflicts can occur since the preliminary SMPGEN checks mean that only files with non-conflicting names are admitted in the backup. The existing system backup archives of the SF pubsets which are incorporated in the SM pubset are retained unmodified. They can be used following SM pubset generation if it is necessary to access backups which were created before the SM pubset was formed and are not present in the system backup archive of the SM pubset. Indeed, HSMS makes it possible to restore backups from any SF pubset or SM pubset backup archives to an SM pubset (as is also the case for SF pubsets). However, backup archives for SF pubsets cannot be used to record SM pubset backups.

6.2.4 Measures for recovering initial states

If problems occur while adaptation is being performed, it is often important to be able to recover a given starting state. In general this requires additional preventive measures such as the creation of backup copies (e.g. of the directories of migration and backup archives). In order to make this presentation as simple as possible, recovery procedures are largely excluded from section “Using SMPGEN to convert the S0 level” on page 75 and from section “Adapting the migration and backup archives” on page 86 below which describe the adaptation of the processing level, background level and the backup archives. Instead they are discussed in section “Reverting from SM pubset to SF pubset” on page 117 which deals with related material. Systems support is strongly advised to schedule the steps necessary for recovery operations as part of the conversion procedure.

6.2.5 Preparations

Careful preparations are required before SM pubsets can be formed from SF pubsets. Both systems support and the users must participate in these preparations. The SMPGEN and DIRCONV utilities provide functions for these operations.

- If the SF pubsets are adapted using SMPGEN, the affected SF pubsets must be prepared in such a way that SMPGEN can merge them coherently to form an SM pubset. For example, there should be no conflicts between file names and no excessively long file names. SMPGEN also provides a CHECK function to support users and systems support in performing the required clean-up measures. Only if the CHECK function reveals no problems can the SM pubset be generated.
- It is also essential to perform a large number of the SMPGEN checks (e.g. check for conflicts and excessively long file names) when adapting the processing level by backing up and restoring files (e.g. check for name conflicts). AT an early stage, it is therefore possible to attenuate problems which would be very difficult to eliminate later when restoring files or adapting the migration and backup archives.
- Before forming the SM pubset, it is necessary to take steps which subsequently make it possible to set up the background levels (with the file contents of the SF pubset background levels) in it. These steps are independent of whether the background levels are adapted by backing up and restoring the files they contain or by means of in-place conversion.
- The backup archives of SF pubsets which are to be transferred to the backup archives of the SM pubset must be adapted using DIRCONV before the SM pubset is generated.
- Individual users must adapt file addressing and replace catalog IDs which are no longer valid in command procedures and programs.
- Individual users must adapt command procedures and programs which contain commands or statements which are not permitted in connection with SM pubsets.
- Program names referenced in GUARDS profiles must be adapted. If the SM pubset is generated using SMPGEN, systems support is supported during the adaptation of the GUARDS catalogs. However, this only applies to SF pubsets which are incorporated in the SM pubset.
- Systems support must adapt the MRSCAT entries at all computers at which the SM pubset is to be used. When the SM pubset is generated using SMPGEN adaptation at the computer where SMPGEN is running is performed automatically.

- Files for which there are special requirements regarding performance and availability must be assigned the appropriate files attributes. In the case of SF pubsets which are transferred to volume sets with AVAILABILITY=*HIGH, SMPGEN automatically adapts the availability file attribute.

The level of preparatory work required depends on the individual conversion scenarios. In some cases this may be very small (e.g. conversion of an SF pubset without an S1 level with its own migration and backup archive). In more complex cases a large volume of work may be required (e.g. creating an SM pubset from multiple SF pubsets with migration and backup archives which are used by other pubsets). Concrete conversion scenarios are discussed in section “Example migration scenarios and procedures” on page 122.

6.3 Using SMPGEN to convert the S0 level

6.3.1 Preconditions

Below we consider the conditions which SF pubsets must meet before SMPGEN can use them to generate an SM pubset.

SM pubset identification

The SM pubset identification must not conflict with the name of another SF or SM pubset or of a volume set. In particular, it must differ from the IDs of all the SF pubsets which are incorporated in the SM pubset. If the SM pubset is intended to assume the ID of an earlier SF pubset, this SF pubset must be renamed using the PVSREN utility before the SM pubset is generated. For example, this is necessary when an SF pubset is converted into an SM pubset of the same name which possesses precisely one volume set.

Name conflicts between files, job variable, GUARDS profiles

Name conflicts occur if multiple SF pubsets for incorporation in the SM pubsets possess files, job variables or GUARDS profiles with identical names. Identical names means that the path names differ in the catalog ID only. With the exception of certain system files which are automatically renamed or deleted by SMPGEN (e.g. $TSOS.TSOSCAT, $TSOS.SYSSRPM) it is necessary for users to clean up name conflicts before the SM pubset is generated. If name conflicts occur, SMPGEN rejects the generation job. Conflicts can be resolved by deleting or renaming the affected files. Migrated files must retrieved into the processing level before they can be renamed.

Excessive path name length

If one of the SF pubsets from which the SM pubset is to be formed has a longer ID than the SM pubset, SM pubset generation extends the path names of the files, job variables and GUARDS profiles cataloged in the SF pubset. SMPGEN checks whether this results in the maximum permitted path name length of 54 characters being exceeded. Like name conflicts, excessive name lengths can be corrected by renaming or deleting the elements involved.

Replacement of the catalog ID can also lead to illegal path name lengths in the case of program names which are referenced in GUARDS conditions. This may also affect GUARDS profiles which are located in SF pubsets which are not to be incorporated in the SM pubset. SMPGEN checks the path lengths of the program names in GUARDS profiles. However, this only applies to the SF pubsets from which the SM pubset is to be generated. The generation of an SM pubset is even permitted if SMPGEN finds uncorrected program names in GUARDS profiles (with the exception of warnings). This means that such names may not be corrected until after the SM pubset has been generated.

File names which are reserved for system files

Certain file names which are permitted for user files in SF pubsets (such as $TSOS.TSOSCAT.<volsetid>, $TSOS.SYS.PUBSET.CONFIG) are reserved for system files in SM pubsets. If SMPGEN detects a file with such a name in one of the SF pubsets to be incorporated in the SM pubset, SM pubset generation is rejected.

Paging files

The SF pubsets which are incorporated in the SM pubset must not contain any files with path names starting with $TSOS.SYS.PAGING.

Files for system start-up

System start-up files located in the SF pubsets do not have to be removed for pubset generation. However, they can no longer be used after generation of the SM pubset.

Files at the S1 level for file migration

If the SF pubset migration directories are to be converted in-place to form the SM pubset migration directory none of the files in the affected pubsets may be located on the S1 level. Systems support can use SMPGEN to check that this is the case (operand: S1-MIGRATED-FILES=*NOT-ALLOWED).

Connected cache areas

None of the SF pubsets to be incorporated in the SM pubset may be connected to a cache area. Any connection to a cache area can be canceled by importing the pubset (if necessary in the mode FORCE-IMPORT=*YES) and then taking it out of service.

Space for system files

When the SM pubset is generated system files are created on the SF pubset which acts as the control volume set. Sufficient space must be present for these system files.

Overflow problems

SM pubset generation may fail due to overflow problems. SMPGEN checks most overflow conditions (e.g. maximum number of SF pubsets, maximum number of users) during the preliminary stage. However, certain conditions (e.g. catalog file overflows) are not checked until generation is performed.

6.3.2 Preparation and execution

The SMPGEN statement CREATE-SYSTEM-MANAGED-PUBSET is provided to enable systems support and users to check that the conditions necessary for SM pubset generation are met. The execution of this function permits two variants: 'checks during an ordinary session' and 'checks in an exclusive mode reserved for systems support'. This makes it possible to take account of the various requirements in different phases of preparation.
The CREATE-SYSTEM-MANAGED-PUBSET statement is available to non-privileged users only in test mode. Systems support must explicitly request test mode using the OPERATIONAL-MODE=*CHECK-NAME-CONSISTENCY(...) operand, which offers further selection options for user IDs and the pubset status.

Checks which have to be performed by users

Initially users should ensure that there are no name conflicts or excessive name lengths for their IDs. To do this they can use SMPGEN checks as part of an active pubset session (see CREATE-SYSTEM-MANAGED-PUBSET, operand PUBSET-STATE=*IMPORTED). In order to perform the checks it is necessary to import all the SF pubsets which are to be incorporated in the SM pubset.

In the case of users without TSOS privilege, the checks performed by SMPGEN refer to the associated user ID. Systems support can also perform checks for multiple users and can thus determine the extent to which individual users have already performed the necessary clean-up measures. Even if no further violations are reported during checks in PUBSET-STATE=*IMPORTED mode, systems support cannot be certain that the subsequent activities performed by individual users have not resulted in new violations.

Checks to be performed by systems support

Once the individual users have taken the necessary clean-up steps, systems support must perform certain checks immediately before converting the SF pubsets. When doing this, systems support should not be disturbed by other user activities (e.g. this stage should represent a reliable check of whether all necessary clean-up measures have been taken). To do this systems support performs the SMPGEN checks in the mode PUBSET-STATE=*NOT-IMPORTED. When the check function is called in this mode, all the SF pubsets to be incorporated in the SM pubset must be exported. When the checks are performed, the pubsets are implicitly imported and exclusively reserved for systems support. If the checks performed under these circumstances reveal no further problems systems support can be certain that the subsequent SMPGEN generation of the SM pubset by SMPGEN will not be rejected. However, this does not guarantee successful generation since certain problems are not detected until conversion time (e.g. insufficient space in the control volume set).

The SM pubset is generated using the SMPGEN statement CREATE-SYSTEM-MANAGED-PUBSET in execution mode (OPERATIONAL-MODE=*OPERATION is implicitly predefined). Systems support must not only provide correctly prepared SF pubsets but must also ensure that the correct runtime environment is present. This is described in detail in the manual “Utility Routines” [1].

If SM pubset generation is affected by problems which prevent successful completion then the SF pubsets are generally restored in their original state. However, the possibility that certain abort situations (e.g. a system crash) may result in an irreparable intermediate status cannot be excluded. Before generating the SM pubset, systems support should therefore back up the SF pubsets involved (e.g. with FDDRL) in case it should subsequently be necessary to restore them.

6.3.3 Characteristics of an SM pubset generated using SMPGEN

When using SMPGEN to define the individual pubset characteristics it is necessary to distinguish between the following cases:

- Definition on the basis of the attributes of the SF pubsets which are incorporated in the SM pubset
- Characteristics predefined by systems support
- Automatic determination by SMPGEN

Below you will find a brief presentation of how the individual pubset characteristics are defined by SMPGEN.

Pubset identifications

The ID of the SM pubset is predefined by systems support.

Physical structure of the SM pubset

Systems support determines which SF pubsets are incorporated in the SM pubset and which of these is to act as the control volume set. These specifications clearly define the physical structure of the SM pubset since the volume configuration, the size of the allocation unit, the format as well as DRV- and REMOTE-COPY attributes are retained when the SF pubsets are converted into volume sets. The volume set IDs are assigned on the basis of the SF pubset IDs.

Large-Objects attribute

If the SF pubsets included in the SM pubset include a pubset with the attribute LARGE-OBJECTS, the SM pubset that results from this is also assigned this attribute (i.e. it becomes a LARGE-OBJECTS pubset).
If at least one of the SF pubsets collected under the LARGE-OBJECTS attribute supports both large volumes (LARGE-DISKS-ALLOWED= *YES) and also large files (LARGE-FILES-ALLOWED=*YES), the SM pubset will also have this attribute. If this is not the case, no large files can be created on the SM pubset (LARGE-FILES-ALLOWED=*YES).

Volume set cache configuration

The cache configuration of a volume set (cache medium, size, operating parameters) is determined from the definitions in the MRSCAT entry of the associated SF pubset.

Volume set configuration status

Once an SM pubset has been generated all its volume sets have the configuration status “normal use”.

Attribute profiles for volume set selection

The size of the allocation unit and the size of the volume sets are determined by the attributes of the SF pubset. The performance and the availability profiles are predefined by systems support.

Usage types for volume sets

By default, all volume sets are assigned the STD usage type. In addition, the systems support can also request the HSMS-CONTROLLED usage type for a volume set. It only makes sense to use this function when an SM pubset is restored after the failure of the control volume set. As a result, file backups and the contents of files migrated to S1 that are on the SF pubset derived from the previous S1 volume set are brought into the SM pubset again.

Usage restrictions for volume sets

There are no usage restrictions for volume sets.

Usage restrictions for volumes

If allocation restrictions apply to any individual volumes of one of the SF pubsets to be incorporated in the SM pubset then these restrictions are retained.

Saturation thresholds for volume sets

The individual volume sets retain the saturation thresholds recorded in the MRSCAT entries of the SF pubsets.

Default pubset space settings (including the default file format)

The default values for primary, secondary and maximum allocation are taken from the MRSCAT entry of the SF pubset which becomes the control volume set.
The default value for the file format is the same as the format for the control volume set.

Using pubsets in networks

The definitions regarding the use of pubsets in a computer network which are stored in the SVL of the VOLRES are determined in the SM pubset from the corresponding definitions for the SF pubset which becomes the control volume set.

MRSCAT entries

An MRSCAT entry is created for the SM pubset. Its contents correspond to that of an SM pubset MRSCAT entry which is initialized using the ADD-MASTER-CATALOG-ENTRY command. All the values are set by default with the exception of the pubset ID and control volume set ID as well as the VOLRES device type of the control volume set. For example, this results in the settings SHARE=*NO, FORCE-IMPORT=*YES, SIZE-TOLERANCE=*NO. The MRSCAT entries of the SF pubsets incorporated in the SM pubset are deleted.

Location of the catalog files and the pubset configuration file

The catalog files of the individual volume sets are created from the conversion of the SF pubset catalog files. SMPGEN creates the global pubset catalog files and the pubset configuration file in the control volume set. Systems support is not able to influence the location or size of these files.

GUARDS profiles

The GUARDS profiles which are set up in the SF pubsets are retained. The GUARDS catalogs of the SF pubsets incorporated in the SM pubset are internally merged to create the GUARDS catalog of the SM pubset.

When the GUARDS catalog for the SM pubset is generated references to program names present in the GUARDS profile are (generally) adapted. The catalog IDs of the SF pubsets are replaced by the catalog ID of the SM pubset. For the handling of special cases (name conflicts, handling of wildcard constructs in GUARDS profiles) please refer to the “SECOS” [2] manual.

User catalog

The user catalog of the SM pubset is generated by merging the user catalogs of the individual SF pubsets. The number of users is calculated by merging the number of users in the individual SF pubsets. SMPGEN creates a new user entry for each user. If multiple entries in individual SF pubsets are present for one and the same user, their contents are compared and combined. The values for the user quotas of an SM pubset for which there is no corresponding element in SF pubsets are increased as appropriate.

When determining the contents of the user entries it is possible to differentiate between the following data categories:

– Space limit values, space used values

  The values of the space limits which are already present in SF pubsets where they have an equivalent meaning are formed by totaling the space limit values which are entered for the relevant user in the affected SF pubsets (example: S0-LEVEL-SPACE-LIMIT corresponds to PUBLIC-SPACE-LIMIT).
  The values of space limits for which there is no corresponding limit in the SF pubsets (e.g. HIGH-PERF-SPACE-LIMIT) are set by default in such a way that they are compatible with the hierarchical quota structure of the SM pubset.
  The space used values are calculated from the space occupied by a user's files and the attributes assigned to the individual files.

– Current number of cataloged files/job variables and limit value

  The current number of files/job variables cataloged for individual users together with the permitted maximum values are calculated by totaling the corresponding values in the affected SF pubsets.

– User authorizations applicable to data pubsets

  The user authorizations which are relevant for data pubsets, PUBLIC-SPACE-EXCESS (space limit exceeded), DMS-TUNING-RESOURCES (enhanced performance requirements) and PHYSICAL-ALLOCATION (right to physical allocation) are specified in such a way that the least restrictive definitions are taken over from all the SF pubsets.

– Privileges and other user characteristics

  User privileges and other system related user characteristics (e.g. privileges) which are located in the home pubset are of secondary significance for SM pubsets provided that these characteristics cannot be used as the home pubset. When defining these user characteristics - in particular the LOGON password - SMPGEN allows you either to determine them using default values or to take them over from one of the SF pubsets which is incorporated in the SM pubset (operand KEEP-USER-ATTRIBUTES). The latter option is only possible for users for whom a user entry exists in the SF pubsets.

– Group structures

When the SM pubset is generated, the KEEP-USER-ATTRIBUTES operand also determines whether user group setup is omitted or whether the group structures are taken over from one of the SF pubsets which is incorporated in the SM pubset. The group entries for SM pubsets contain space limits for user quotas for which there is no corresponding value in the group entries for SF pubsets. Maximum values are assigned to these space limits on transfer. Users of SM pubsets who were not entered in the SF pubset from which the group definitions are transferred or who belonged to no group in the SF pubset are assigned the *UNIVERSAL group.

Files and file attributes

The location of the user files in the volumes is not modified by the generation of an SM pubset.
Files cataloged in the SF pubsets which do not themselves occupy any space in the SF pubset (files located in background levels, tape files, private disk files, cataloged files without a space allocation) remain cataloged. Their catalog entries are internally transferred to the associated special catalog files.

When defining file attributes, a distinction is drawn depending on whether or not there is already a corresponding definition in SF pubsets (for the significance of file attributes see the section “File attributes which are relevant for file location and their default values” on page 205).

a) File attributes with a corresponding definition in SF pubsets

With the exception of the AVAILABILITY file attribute the file attributes for which there is a corresponding definition in SF pubsets remain unchanged. The AVAILABILITY file attribute is affected by the availability profile of the volume set on which the file is located: All non-temporary files in a volume set with AVAILABILITY=*HIGH are automatically assigned the file attribute AVAILABILITY=*HIGH, all other files retain their previous AVAILABILITY value. Systems support should make sure that the SF pubsets which contain files with the attribute AVAILABILITY=*HIGH are assigned the availability profile AVAILABILITY=*HIGH when they are used as volume sets. Otherwise following the generation of the SM pubset, files with the file attribute AVAILABILITY=*HIGH would be located in volume sets with the availability profile AVAILABILITY=*STD (which would conflict with the binding nature of the AVAILABILITY file attribute).

b) File attributes with no corresponding definition in SF pubsets

The file attributes for which there is no corresponding definition in the SF pubsets are the preformat and the S0 migration lock.

– All the files automatically receive the attribute S0-MIGRATION=*ALLOWED. They are then subsequently no longer bound to a specific volume set. If users with physical allocation authorization want certain files to remain in the volume set in which they are located immediately after SM pubset generation they must explicitly assign the attribute S0-MIGRATION=*FORBIDDEN to them using the MODIFY-FILE-ATTRIBUTES command.

– The preformat is only relevant to the files taken over from the SF pubsets if space has already been assigned to them in SF pubsets but they have not yet been opened there. When these files are opened for the first time they are handled as if they had been assigned the format of the volume set in which they are currently located as their preformat.

6.3.4 Follow-up processing of the presettings made by SMPGEN

Following conversion, systems support should check whether the presettings automatically made by SMPGEN, for example concerning user quotas, default allocation values, usage restrictions are already appropriate or whether they need to be modified. Adaptations can be performed using the pubset maintenance functions (e.g. the commands MODIFY-USER-ATTRIBUTES, MODIFY-PUBSET-DEFINITION-FILE, MODIFY-PUBSET-RESTRICTIONS, MODIFY-MASTER-CATALOG-ENTRY).
In particular, systems support should analyze whether the assignment of default values to file attributes may result in undesirable incompatibilities for users. Such problems can be largely prevented by assigning appropriate default file format settings for the SM pubset and allocating appropriate default storage classes for the individual users.

Example

For a user who was previously assigned an SF pubset with the format K as default pubset, the default file format assignment remains the same if the default file format K is assigned to the SM pubset. However, if the SM pubset receives the value NK4 as the default file format it is possible to guarantee compatible behavior for this user by assigning him or her a default storage class with the Preformat K.

6.3.5 Reduction in functional scope as a result of pubset conversion

The SF pubsets which are incorporated in the SM pubset are subject to the following reductions in functional scope:

EAM files

SYSEAM files located in the SF pubsets are deleted.

Suitability as home pubset

When an SF pubset is converted for use in an SM pubset it can no longer be user as a home pubset.

Spool jobs

Spool jobs referring to files whose path names have changed as a result of SM pubset generation should be completed before conversion as they will otherwise be invalid and can no longer be executed. Systems support must explicitly delete temporary files belonging to spool jobs which have become invalid following SM pubset generation.

User file backup

The file $TSOS.SYSSRPM.BACKUP is lost.

6.4 Adapting the migration and backup archives

This section provides an overview of the HSMS and DIRCONV-functions which are used to adapt the backup and migration archives. It is assumed that readers are familiar with the HSMS statements (BACKUP-FILES, COPY-SAVE-FILE, etc.) and HSMS concepts (such as archives, save files, save versions) (see the “HSMS“ manual [4]). This also applies to DIRCONV operation. When using this function, note that (e.g. in the case of RENAME-CATALOG ID) the directories made available by the caller remain unchanged and an additional new directory containing the required modifications is created. It is advisable to rename the previous directories in such a way that their future role as backup copies is clear from their path names before calling DIRMERGE. The path name for the new directory should be selected in such a way that as little effort is necessary to adapt the existing archive definitions, e.g. by taking over the path name of the existing directory.

Unlike the case of the processing level, is often difficult to provide the right conditions for the in-place conversion of the migration archive and the system backup archive. The same applies to private backup archives. Before conversion, systems support or the owners of the private backup archives should examine whether the individual archives deviate from the conditions necessary for in-place conversion and what effort is necessary in order to create these conditions. The effort involved should be compared with the effort necessary when using alternative adaptation options. An alternative approach consists of setting up new archives (more precisely: archives with new directories and new volumes) for the SM pubset and to provide these archives with suitable contents by transferring files or job variables (save/restore, COPY-SAVE-FILE) from other archives. In contrast to in-place conversion, this process requires access to the volumes in which migrated files or file/job variable backups are located.

The execution of HSMS functions requires access to the MAREN catalog. If this is located in one of the SF pubsets which are incorporated in the SM pubset it cannot be used during SM pubset generation. Access should again be possible at the latest when the HSMS environment is set up in the SM pubset. If adaptation of the processing level is not performed in-place but by backing up and restoring files, special measures are necessary. In this case, it is advisable to move the MAREN catalog to an unaffected pubset (or private disks) before SM pubset generation.

6.4.1 Preconditions for in-place conversions

The following conditions must be met before in-place conversion is possible:

- Although the migration and backup archives of SF pubsets can be used by multiple SF pubsets simultaneously, it cannot be assumed that all these SF pubsets will be incorporated in the SM pubset. In contrast, the migration and backup archives of SM pubsets are pubset-specific and must not contain any files or backups belonging to other pubsets. The in-place conversion of the SF pubset archives is only possible if these are disconnected from ‘foreign’ pubsets, i.e. if they are not used for SF pubsets which are not incorporated in the SM pubset.
- In place conversion is only possible for backup and migration archives whose entire file store is stored on tapes. In the case of migration archives this is due to the fact that the in-place conversion of an S1 pubset into an S1 volume set is not supported. In the case of backup archives it is the result of a fundamental restriction which, unlike SF pubsets, only permits tapes as backup media for SM pubsets.
- In the path names of the files and job variables stored in the archives, the catalog IDs of the earlier SF pubsets must be replaced by those of the SM pubset. Before doing this it is necessary to ensure that the renaming operation does not lead to excessive name lengths are file name conflicts.

If an archive does not fulfill these conditions it must be cleaned up prior to in-place conversion. In general this means that it is necessary to access the volumes containing the migrated files or backups and this can considerably reduce the advantages of in-place conversion.

6.4.2 Adapting the migration archives

First we present the procedure for the in-place conversion of the migration archives. Then we describe how migration archives are adapted using backups. Some of the necessary steps must be taken before and some after the adaptation of the processing level. The migration archives can also be adapted in-place if the processing level is adapted by backing up and restoring files.

6.4.2.1 In place conversion of migration archives

We first provide an overview of the necessary measures followed by a detailed description of the individual steps.

a) Measures preceding the adaptation of the processing level

A1	Clean up name conflicts affecting files migrated to background levels
A2	Determine the affected migration archives and migration directories
A3	Disconnect the migration archives of SF pubsets which are not to be incorporated in the SM pubset
A4	Migrate the files located in the S1 level
A5	Remove obsolete entries from the affected migration directories
A6	Merge the affected directories to form a single directory
A7	Optional: Measures which make it possible to continue using the migration archive for SF pubsets for a certain period
A8	Adapt the directory for the future SM pubset by replacing the catalog ID.
A9	Adapt the MAREN catalog assigned to the archive
A10	Back up the migration directory (only necessary if the processing level is not adapted in-place)
A11	Clean up the HSMS definitions for the SF pubsets to be incorporated in the SM pubset

b) Measures following the adaptation of the processing level

A12	Set up the HSMS environment in the SM pubset
A13	Restore the migration directory (only necessary if the processing level was not adapted in-place, see A10)
A14	Set up the migration archive with S2 level and allocate the previously adapted migration directory
A15	Adapt the migration control parameters (e.g. Except files)
A16	Set up the S1 level and migrate files to the S1 level

a) Measures preceding the adaptation of the processing level

A1	**Clean up name conflicts affecting files migrated to background levels** SMPGEN checks performed as part of the adaptation of the processing level also recognize name conflicts between files located in background levels. These name conflicts can be cleaned up by renaming or deleting the files. Migrated files must be moved to the processing level before they can be renamed.
A2	**Determine the affected migration archives and associated directories allocated to the SF pubsets to be incorporated in the SM pubset** The migration archive assigned to a pubset can be determined using the HSMS statement SHOW-PUBSET-PARAMETERS. The archive directory can be determined using the HSMS statement SHOW-ARCHIVE-ATTRIBUTES.
A3	**Disconnect the migration archives of SF pubsets which are not to be incorporated in SM pubsets** To recognize connections between migration archives and foreign SF pubsets (i.e. SF pubsets which are not incorporated in the SM pubset) systems support can use the DIRCONV statement **//SHOW-CATID** **DIRECTORY-NAME = ... , ...** to display all pubsets for which the migration archive of the archive is used. A connection to a foreign pubset can be disconnected as follows: A new migration archive is assigned to the foreign pubset using the HSMS statement MODIFY-PUBSET-PARAMETERS **//MODIFY-PUBSET-PARAMETERS** **PUBSET-ID = <identification of foreign pubset > , STORAGE-LEVEL = S0 (SYSMIGRATE = <name of the new migration archive >) , ...** If necessary the new migration archive must be set up beforehand. Two different procedures are available for transferring the migrated files of the foreign pubsets to the new migration archives:

A3 cont.

Alternative a)

Each save file which contains a migrated file belonging to the foreign pubset is transferred selectively (i.e. accompanied only by the files of the foreign pubset) to the migration archive of the foreign pubset using the HSMS statement

//COPY-SAVE-FILE	**SAVE-FILE-ID = ... , SELECT-FILES = :< identification of foreign pubset>:$*.* , ...**

The localization information in the catalog entries must then be updated for the migrated files of the foreign pubset. This can be performed using the following HSMS statement:

//REPAIR-CATALOG-BY-EXCHANGE	**S0-PUBSET-ID = <identification of foreign pubset> , ...**

One difficulty may relate to the procedure used to determine the affected save files. This may occur as follows:

The HSMS statement

//SHOW-ARCHIVE	**SELECT = FILES (FILE-NAMES = :<identification of foreign pubset>:$*.*) , ...**

displays the save versions which contain save files with migrated files belonging to the foreign pubset. The affected save files can then be determined using the HSMS statement

//SHOW-ARCHIVE	**SELECT = SAVE-VERSIONS(...) , ...**

Alternative a) should only be considered if there is a low level of connection to other SF pubsets as otherwise considerable work is involved.

Alternative b)

First of all the migrated files of the foreign pubset are backed up. Before doing this it is advisable to determine the migrated files of the foreign pubset using SHOW-FILE-ATTRIBUTES and store their file names in a list. This can be performed using the BACKUP-FILES call.

//BACKUP-FILES	**FILE-NAMES = *FROM-FILE (<list of foreign files>) , SAVE-OPTIONS = PAR(SAVE-DATA = S2-S1-S0) , ARCHIVE-NAME = <name of backup archive> , ...**

A3 cont.

Next the HSMS statement

//REPAIR-CATALOG-BY-RESTORE	**S0-PUBSET-ID = <identification of foreign pubset>, ARCHIVE-NAME = <backup archive with backups of the migrated files from the foreign pubset> , ...**

is used to reconstruct the migration archive of the foreign pubset on the basis of the previously created backups. This call also updates the localization information in the catalog entries for the migrated files.

When the migrated files of the foreign SF pubset have been transferred to their new migration archives the references to the foreign pubset must be deleted in the original directory. This can be done using the DIRCONV statement

//REMOVE-CATID	**DIRECTORY-NAME = ... , CATID = ... , ...**

The contents of the volumes remain unchanged, i.e. the migrated files for the foreign pubset continue to occupy space (in the save files) even if they can be no longer used. If you wish to prepare not only the directory but also the S2 volumes (which is not absolutely necessary) it is necessary to reorganize the MRSCAT entries (see A5, alternative a).

A4

Migrate the files located in the S1 level

None of the SF pubsets involved in the formation of the SM pubset may contain a file migrated to S1. SMPGEN checks can be used to ensure that this condition is fulfilled (SMPGEN operand: S1-MIGRATED-FILES=*NOT-ALLOWED). If this condition is not fulfilled systems support can adopt one of the approaches below to fulfill it:

Alternative a)

For each migration archive which is assigned to an SF pubset which is to be incorporated in the SM pubset, all the files migrated to S1 are migrated from the S1 level to the S2 level by calling MIGRATE-FILES:

//MIGRATE-FILES	**FROM-STORAGE-LEVEL = S1-STORAGE-LEVEL , TO-STORAGE-LEVEL = S2-STORAGE-LEVEL , ARCHIVE-NAME = ... , ...**

Since the foreign SF pubsets entries have already been removed from the migration directories this statement relates only to the files of the SF pubsets involved in SM pubset generation which have been migrated to S1.

A4 cont.

Alternative b)

The files migrated to S1 are retrieved to the processing level using a Recall call:

//RECALL-MIGRATED-FILES	**FILE-NAMES = *FROM-FILE (e.g. output list from SHOW-FILE-ATTRIBUTES) , FROM-STORAGE-LEVEL = S1-STORAGE-LEVEL , ...**

Alternative c)

The files migrated to S1 are backed up and deleted:

//BACKUP-FILES	**FILE-NAMES = *FROM-FILE (e.g. output list from SHOW-FILE-ATTRIBUTES) , DELETE-FILES-AND-JV = YES , ...**

If the files which were previously located in the S1 level are subsequently be moved to the S1 level of the SM pubset, it is advisable to record their names in a file (e.g. output list from SHOW-FILE-ATTRIBUTES). A precondition for all these alternatives is that sufficient storage space is (or can be made) available in the processing level for the temporary storage of the S1 level files. In alternative b) this requirement relates to SF pubsets in alternatives a) and c) to the subsequent SM pubset.

A5 **Remove obsolete entries from the affected migration directories**

In order to avoid name problems when adapting the migration archives, files affected by name problems must be renamed or deleted (step A1) and obsolete entries must also be removed from the migration directories. The following methods of deleting obsolete entries are possible:

A5 cont.

Alternative a)

The entire migration archive is reorganized. To do this each save file is copied to a new save file using the statement

//COPY-SAVE-FILES	**SAVE-FILE = ... , MIGRATION-STATE = MIGRATED-FILES , ...**

Only the contents of currently migrated files are taken over. It is not necessary to copy save files which contain no current files. Systems support can obtain a list of affected save files using the following HSMS statement:

//SHOW-ARCHIVE	**ARCHIVE-NAME = ... , SELECT = SAVE-FILES(SAVE-FILE-ID = *BY-ATTRIBUTES (SAVE-FILE-STORAGE = TAPE)) , ...**

The previous save files are deleted using the HSMS statement

//MODIFY-ARCHIVE	**ARCHIVE-NAME = ... , SAVE-FILES = DELETE(SAVE-FILE-ID = ... , FORCED-DELETE=YES) , ...**

If all the save files were created before the day the reorganization is performed it is particularly easy to identify them by using the specification SAVE-FILES=DELETE(SAVE-FILE-ID=*BY-ATTRIBUTES (CREATED-BEFORE = <date of day before reorganization>). During reorganization, obsolete entries are deleted both from the migration directory and from the tapes.

Alternative b)

Reorganization of the migration archive requires access to the S2 volumes. This may be undesirable during in-place conversion. For this reason, the DIRCONV statement

//REMOVE-UNCATALOGED-FILES	**DIRECTORY-NAME = ... , ...**

makes it possible to clean up only the migration directory. The contents of the S2 volumes are not modified. However, it is not subsequently possible to access the obsolete entries which they contain. Such entries can be removed when the archive is reorganized.

A6	**Merge the affected directories to form a single directory** If different migration directories are assigned to the SF pubsets which are to be incorporated in the SM pubset they must be merged to form a single directory. This is possible using the following DIRCONV statement: **//MERGE-DIRECTORIES** **DIRECTORY-NAMES = ... ,** **NEW-DIRECTORY-NAME = ... , ...**
A7	**Optional: Adapt the archive for use with SF pubsets** The merging of migration directories may form a preparatory stage for later migration to an SM pubset. This does not have to be performed immediately after merging. In this case it is necessary to take the following steps to ensure that the migration directory created by MERGE can continue to be used in the interim: 1. It is necessary to define a migration archive to which the newly created directory is allocated. 2. This must be allocated as migration archive to all SF pubsets involved in the formation of the SM pubset. 3. The transition to the new directory must be consistently respected in the volume allocations of the MAREN catalog. DIRCONV provides the following statement for this purpose: **//UPDATE-VOLUME-CATALOG** **DIRECTORY-NAMES = ... ,** **NEW-DIRECTORY-NAME = ... , ...** In the MAREN catalog, this allocates all volumes which are entered in the directory specified using the NEW-DIRECTORY-NAME operand to this directory. The DIRECTORY-NAMES operand of the UPDATE-VOLUME-CATALOG statement also makes it possible to enter free tapes, which were previously reserved for the earlier directories, in the pool of free volumes for the new directory. The above-mentioned measures are not necessary if migration to the SM pubset follows directly after the merging of the migration directories.

A8	**Replace catalog ID in the migration directory** The DIRCONV statement **//RENAME-CATID** **DIRECTORY-NAME = ... ,** **OLD-CATID = *ALL ,** **NEW-CATID = PUBSET-ID , ...** is used to create a new migration directory in which references to the catalog IDs of the previous SF pubsets are replaced by the reference to the catalog ID of the future SM pubset. This step is not necessary if the SM pubset is generated from precisely one SF pubset and takes over the ID of the SF pubset. If the processing level is adapted using SMPGEN, the future SM pubset directory must be created on the SF pubset which later becomes the control volume set (e.g. by specifying a corresponding path name for the new directory in the DIRCONV statement). The directories which were previously used for migration purposes can be retained in order to make a subsequent recovery of the initial state possible. However, as of this moment they should no longer be used to execute migration and recall jobs as otherwise the newly created directory will be inconsistent.
A9	**Adapt the MAREN catalog** If the future migration directory path name in the SM pubset differs from the current path name then the new path name must be consistently respected in the MAREN catalog (see also A7). This is achieved using the DIRCONV statement **//UPDATE-VOLUME-CATALOG** **DIRECTORY-NAMES =... ,** **NEW-DIRECTORY-NAME = ... , ...**
A10	**Back up the migration directory (only necessary if the processing level is not adapted in-place)** If the processing level is not adapted in-place but by backing up and restoring files the migration directory must also be backed up. To do this it is also possible to copy it to a pubset which is not involved in SM pubset generation.
A11	**Clean up the HSMS definitions for the previous SF pubsets** If the SM pubset takes over the ID of an SF pubset, the HSMS definition of this SF pubset must be removed. Otherwise problems may occur when the HSMS environment is set up in the SM pubset. In the case of the other SF pubsets which are incorporated in the SM pubset and migration archives which are not required in the future, the definitions can also be cleaned up after migration.

b) Measures following adaptation of the processing level

A12	**Set up the HSMS environment in the SM pubset** It order for it to be possible to use HSMS functions following adaptation of the processing level, the HSMS environment must be installed in the pubset using the HSMS statement CREATE-SM-PUBSET-PARAMETERS. This confers the status HSMS-SUPPORTED on the pubset. MAREN catalog accesses must also be possible if HSMS functions are to be used without problems.
A13	**Restore the migration directory (only necessary if the processing level was not adapted in-place, see A10)** If the processing level was not adapted in-place but by backing up and restoring files then the migration directory must also be restored or, if it has been migrated to another pubset, copied from there to the SM pubset (see A10). It should be noted that the directory is sent to the control volume set.
A14	**Set up the migration archive and allocate the migration directory** In order to restore the operability of the S2 level it is necessary to set up a migration archive and allocate it to the directory which was prepared appropriately before SM pubset generation and which may have been restored in step A13. This can be performed when CREATE-SM- PUBSET-PARAMETERS is called or by a subsequent call to MODIFY-SM-PUBSET-PARAMETERS.
A15	**Set up the MIGRATION-CONTROL parameters** When setting up the archive, systems support must set the MIGRATION-CONTROL parameters correctly. When doing this, special attention should be paid to the control facilities which make it possible to prevent undesired migrations to background levels. It should be noted that the earlier Except file loses its effect for the SF pubsets incorporated in the SM pubset. In order to restore protection against unwanted file migration systems support can set up a pubset-specific Except file in the SM pubset, enter the SM pubset files for which migration locks are required in this Except file and allocate these files to the SM pubset. Allocation can be performed on the CREATE-SM-PUBSET-PARAMETERS call or during a subsequent MODIFY-SM-PUBSET-PARAMETERS call. Another way of preventing unwanted migrations is to use the file-specific migration lock MIGRATE=FORBIDDEN which can be set using the MODIFY-FILE-ATTRIBUTES command.

A16	**Set up the S1 level and migrate files to the S1 level** If systems support wants to restore the S1 level which was dissolved for SM pubset generation following SM pubset generation, he or she can do this in the following ways: 1. The SM pubset must first be configured with an S1 volume set. To do this the dynamic pubset reconfiguration facilities are used to incorporate a volume set with usage type HSMS-CONTROLLED in the pubset. This can be done as follows: – Enter the volume set in the pubset configuration file with usage type HSMS-CONTROLLED (MODIFY-PUBSET-DEFINITION-FILE command) – Add the volume set to the physical volume set configuration (MODIFY-PUBSET-PROCESSING command) This volume set must be reported to HSMS as an S1 volume set (HSMS statements CREATE-SM-PUBSET-PARAMETERS or MODIFY-SM-PUBSET-PARAMETERS, operand S1-DEFINITION, S1-VOLUME-SET-ID). 2. The files migrated to S1 prior to SM pubset generation can be moved to the S1 level of the SM pubset. First of all the files must be moved to the processing level if alternative a) or c) was chosen in step A4: if the files migrated to S1 were relocated to the S2 level (alternative a) they must be retrieved into the processing level using a RECALL job. If they were backed up and then deleted (alternative c) the backup must be restored. When doing this, systems support must make sure that sufficient space is available in the processing level, for example by temporarily enlarging the pubset by adding volumes or a volume set. If alternative b) was selected the files are already present in the processing level at this point. A migration job can now be used to move the files from the processing level to the S1 level. These steps are easier to perform if the names of the files migrated to S1 in the SF pubsets are recorded in a file during step A4.

The background levels should be made available for the SM pubset before it is released for general use. It is then possible both to initiate new migration jobs and to recall to the processing level the files which were located in background levels prior to SM pubset generation.

The above description of the in-place conversion of migration archives clearly shows that the problems which have to be overcome depend to a very large extent on the application environments of individual users. The ideal situation occurs when none of the migration archives possesses references to pubsets which are not to be incorporated in the SM pubset. The disconnection of foreign pubset is then guaranteed from the outset. Considerable preparatory work must be performed if the migration archives for adaptation are also used by foreign pubsets. Here it may often be preferable to adapt the migration archives by means of backup operations.

6.4.2.2 Adapting the migration archives using backups

Below we present an overview of the necessary measures followed by a detailed description of the individual steps.

a) Measures preceding the adaptation of the processing level

B1	Perform the SMPGEN checks using the option ‘files permitted in the S1 level’
B2	Backup the files located in background levels
B3	Clean up the existing migration archives for the SF pubsets
B4	Adapt the backup archive in which the backups are stored

b) Measures following adaptation of the processing level

B5	Set up the HSMS environment in the SM pubset
B6	Make the backup archive containing the backups available
B7	Set up the migration archive with S2 level
B8	Set the MIGRATION-CONTROL parameters for the SM pubset (Except file)
B9	Set up the S1 level
B10	Restore backups in the S1 and S2 levels

a) Measures preceding the adaptation of the processing level

B1	**Perform the SMPGEN checks** SMPGEN checks performed as part of the adaptation of the processing level also recognize name conflicts between files located in background levels. These name conflicts can be cleaned up by renaming or deleting the files. Migrated files must be moved to the processing level before they can be renamed. Unlike the procedure for in-place conversion of the migration archives no special measures are required for the files which have been migrated to S1 (such as migration to the S2 level, see A4). To prevent these files from being evaluated as inconsistent during the SMPGEN checks, the checks must be performed using the option S1-MIGRATED-FILES=*ALLOWED.
B2	**Back up the files located in background levels** The files which are located in background levels are backed up. This backup can be performed explicitly or as part of a full backup of all the SF pubsets involved in SM pubset generation. It should be performed after the SMPGEN checks have run and before the processing level is adapted. In order to prevent uncoordinated user activities resulting in inconsistencies in the backup version (e.g. as a result of file processing during the backup) it is advisable to import the SF pubsets which are to be backed up in such a way that only systems support is able to access them (command ADD-MASTER-CATALOG-ENTRY, operand ACCESS-CONTROLLED).

B3 **Clean up the HSMS definitions and the migration archives**

If the SM pubset takes over the ID of an SF pubset, the HSMS definition of this SF pubset must be removed. Otherwise problems may occur in the SM pubset when the HSMS environment is subsequently set up.

The other clean-up measures listed in step B3 may also be deferred to a later point, e.g. following successful migration to the SM pubset. This makes it easier to recover the starting situation if problems occur during migration.

The HSMS definitions for the SF pubsets which are incorporated in the SM pubset can be removed. The assigned migration archives and directories can also be deleted if they are no longer required for other SF pubsets. If they have to be retained for other SF pubsets it is advisable to clean them up:

1. The DIRCONV statement REMOVE-CATALOG ID is used to delete references to the SF pubsets which are to be incorporated in the SM pubset in the affected directories.
2. The reorganization of the archives should also be considered in order to remove the migrated files for the SF pubsets incorporated in the SM pubset from the save files and thus from the volumes. This can be performed by copying each save file to a new save file using

//COPY-SAVE-FILES	**SAVE-FILE = ... ,** **MIGRATION-STATE = MIGRATED-FILES , ...**

When this is done only the contents of the currently migrated files are taken over. systems support can use the HSMS statement

//SHOW-ARCHIVE	**ARCHIVE-NAME = ... ,** **SELECT = SAVE-FILES(SAVE-FILE-ID =** ***BY-ATTRI(SAVE-FILE-STORAGE = TAPE)) , ...**

to generate a list of the affected save files. The earlier save files are removed using the HSMS statement

//MODIFY-ARCHIVE	**ARCHIVE-NAME = ... ,** **SAVE-FILES = DELETE (SAVE-FILE-ID =** ***BY-ATTR (CREATED-BEFORE..)) , ...**

B4 **Perform adaptation measures for the backup archive containing the backups of the migrated files**

In order to make it possible to use the backup archive containing the backups of the migrated files to restore the backups following SM pubset generation, certain measures must be performed prior to SM pubset generation. These are discussed in section “Adapting the backup archives” on page 103.

b) Measures following adaptation of the processing level

B5	**Set up the HSMS environment in the SM pubset** It order for it to be possible to use HSMS functions following adaptation of the processing level, the HSMS environment must be installed in the pubset using the HSMS statement CREATE-SM-PUBSET-PARAMETERS. This confers the status HSMS-SUPPORTED on the pubset. MAREN catalog accesses must also be possible if HSMS functions are to be used without problems.
B6	**Make the backup archive containing the backups available in the SM pubset** The backup archive in which the backups of the migrated files are stored is made available as the backup archive of the SM pubset. The necessary steps are discussed in section “Adapting the backup archives” on page 103.
B7	**Set up the migration archive** The migration archive is set up (either directly by CREATE-SM-PUBSET-PARAMETERS or using MODIFY-SM-PUBSET-PARAMETERS). Unlike the procedure for in-place conversion, the migration archive is allocated a new directory which is initially empty.
B8	**Set the MIGRATION-CONTROL parameters** When setting up the migration archive, systems support must set the MIGRATION-CONTROL parameters correctly. When doing this, special attention should be paid to the control facilities which make it possible to prevent undesired migrations to background levels. It should be noted that the earlier Except file loses its effect for the SF pubsets incorporated in the SM pubset. In order to restore protection against unwanted migration, systems support can set up a pubset-specific Except file in the SM pubset, enter the SM pubset files for which migration locks are required in this Except file and allocate these files to the SM pubset. Allocation can be performed on the CREATE-SM-PUBSET-PARAMETERS call or during a subsequent MODIFY-SM-PUBSET-PARAMETERS call. Another way of preventing unwanted migrations is to use the file-specific migration lock MIGRATE=FORBIDDEN which can be set using the MODIFY-FILE-ATTRIBUTES command.

<table>
<tr><td>B9</td><td>Set up the S1 level

If the files which were previously located in the S1 level of the SF pubsets are to be moved to the S1 level of the SM pubset, the SM pubset must first be configured with an S1 volume set. To do this the dynamic pubset reconfiguration facilities are used to incorporate a volume set with usage type HSMS-CONTROLLED in the pubset. This can be done as follows:

– Enter the volume set in the pubset configuration file with usage type HSMS-CONTROLLED (MODIFY-PUBSET-DEFINITION-FILE command)

– Add the volume set to the physical volume set configuration (MODIFY-PUBSET-PROCESSING command)

This volume set must be reported to HSMS as an S1 volume set (HSMS statements CREATE-SM-PUBSET-PARAMETERS or MODIFY-SM-PUBSET-PARAMETERS, operand S1-DEFINITION, S1-VOLUME-SET-ID).</td></tr>
<tr><td>B10</td><td>Restore backups in the S1 and S2 levels

The HSMS statement

<table><tr><td>//REPAIR-CATALOG-BY-RESTORE</td><td>S0-PUBSET-ID = <ID of SM pubsets> ,
TO-STORAGE-LEVEL = ... ,
ARCHIVE-NAME = <backup archive with backups of migrated files > , ...</td></tr></table>
is used to reconstruct the migration archive using the backups created earlier and to update the localization information in the catalog entries of the migrated files. The option TO-STORAGE=*SAME can be used to reconstruct the original distribution to the S1 and S2 levels. If the files which were previously located in the S1 level are to be transferred to the S2 level, for example if the SM pubset contains no S1 level or if this has not yet been set up, the option TO-STORAGE= S2-STORAGE-LEVEL must be selected.</td></tr>
</table>

The background levels of the SM pubset are now operative. The files migrated prior to SM pubset generation can now be accessed again.

6.4.3 Adapting the backup archives

In some respects the adaptation of the backup archives differs considerably from the adaptation of the migration archives:

- More effort is involved in identifying the archives affected. The system backup archive can be determined in the same way as the migration archives using the HSMS statement SHOW-PUBSET-PARAMETERS. However, it is also necessary to take account of all private backup archives which contain backups for one of the involved SF pubsets or which are to be used for backups from the SM pubset at a later stage.
- Backups which are stored in SF pubset archives can also be restored to SM pubsets. The necessity of converting existing SF pubset archives into SM pubset archives is therefore greatly reduced. The backups which they contain continue to be accessible even if they are not adapted. However, in this case they can no longer be used for new backups from the SM pubset. The decision whether or not to convert a backup archive can be taken by the archive's owner in the light of any special conditions which may apply.
- The adaptation of the backup archives allows considerably greater room for maneuver between the various procedures than is available when adapting the migration archive. For example, while the contents of the migration archive following conversion are predetermined by the files which are migrated at this time, the backups which are taken over into the SM pubset backup archives can be defined to meet individual user needs. Transfer need also not be performed at a specific time. For example, it may be performed in a number of steps depending on needs.
- During the in-place conversion of the existing backup archives the number of conflicts and excessive file name lengths is more difficult to determine since all the files which have previously been stored in the archives in question are relevant. Consequently the preparations required for in-place conversion depend to a very great extent on the history of the archive.

By way of example, we present two procedures in greater detail below:

1. The first case deals with the in-place conversion of existing backup archives for SF pubsets. We examine the example of the system backup archives. Here the system backup archive for the SM pubset is formed from the existing backup archives of all the SF pubsets which are to be incorporated in the SM pubset. The objective is to ensure that all the backups contained in the system backup archives of the affected SF pubsets are transferred to the system backup archive of the SM pubset and that this results in little or no modification to access to backups or to the relation between backups, save file and save version. As far as system backups are concerned the transition to the SM pubset is largely concealed from users.
2. In the second case, the file store of the SF pubsets for incorporation in the SM pubset is backed up in a newly created backup archive immediately before SM pubset generation. This new backup archive is used to create the system backup archive of the SM pubset by means of in-place conversion. The preceding system backup archives of the SF pubsets are retained as private backup archives along with the backups they contain. For users this means that accesses to system backups may differ depending on the time (before or after SM pubset generation) they were created.

The procedure described in the first case is especially well suited for user scenarios which are easy to manage (i.e. user scenarios in which name conflicts between files and therefore also between backups cannot occur and in which the archives in question are not used by pubsets which are not incorporated in the SM pubset).
The alternative presented in the second example should be preferred for more complex situations. As when adapting the migration archives, systems support and the owners of the private backup archives should perform an analysis to determine which alternative is most suitable in the light of any special conditions.

6.4.3.1 Adapting existing system backup archives

Below we present an overview of the necessary measures followed by a detailed description of the individual steps.

a) Measures preceding the adaptation of the processing level

C1	Determine the system backup archives and their directories
C2	Disconnect the backup archives of SF pubsets which are not incorporated in the SM pubset
C3	Migrate save files which are located on private disks or pubsets
C4	Check the backed up files for name conflicts and excessive name lengths
C5	Clean up any name conflicts and excessive name lengths occurring in backed up files
C6	Merge the directories
C7	Replace the catalog IDs in the directory
C8	Perform adaptations in the MAREN catalog
C9	Back up the backup directory and, if necessary, the MAREN catalog (only necessary if the processing level is not adapted in-place)
C10	Clean up the archive definitions for the SF pubsets

b) Measures following SM pubset generation

C11	Set up the HSMS environment in the SM pubset
C12	Restore the backup directory and, if necessary, the MAREN catalog (only necessary if the processing level is not adapted in-place)
C13	Set up the system backup archives using the directory which has been adapted for the SM pubset

It is only advisable to convert the existing system backup archives for the SF pubsets in-place if there are no or only very few connections between these and foreign pubsets and if it is necessary to clean up no or only a few name conflicts and excessive name lengths. Otherwise in-place conversion (steps C2,C5) is very difficult and time-consuming. However, in many cases the circumstances will be favorable for in-place conversion.

a) Measures preceding the adaptation of the processing level

C1 **Determine the affected system backup archives and their directories**

The system backup archive assigned to a pubset can be determined using the HSMS statement SHOW-PUBSET-PARAMETERS. The archive directory can be determined using the HSMS statement SHOW-ARCHIVE-ATTRIBUTES.

C2 **Disconnect the backup archives of SF pubsets which are not incorporated in the SM pubset**

The affected backup archives cannot contain file backups of pubsets that are not included in the SM pubset. In order to identify connections with foreign pubsets, systems support can use the DIRCONV statement

// SHOW-CATID	**DIRECTORY-NAME = ... , ...**

to display all the pubsets for which a given backup directory is used. Existing connections to a foreign pubset can be disconnected as follows:

1. A different system backup archive is assigned to the foreign pubset. It may be necessary to set up this system backup archive first.
2. The backed up files of the foreign pubset are transferred to this archive. To do this proceed as follows:

   The save files which contain a backup of a foreign pubset are identified. First the HSMS statement

//SHOW-ARCHIVE	**SELECT = FILES(FILE-NAMES = :<identification of foreign pubset>:$*.*) , ...**

   is used to identify all the save versions which contain save files with backups of the foreign pubset. The HSMS statement

//SHOW-ARCHIVE	**SELECT = SAVE-VERSIONS(...) , ...**

   identifies the affected save files. While respecting the ascending sequence of save versions, each save file which contains a backup of the foreign pubset is selectively (i.e. only accompanied by files of the foreign pubset) transferred to its new system backup archive:

//COPY-SAVE-FILE	**SAVE-FILE-ID = ... ,SELECT-FILES = :<identification of foreign pubset>:$*.* , ...**

C2 cont.

3. When the backups of the foreign SF pubset have been transferred to their new backup archive the following DIRCONV statement must be used to delete the references to the foreign pubset in the original directory:

//REMOVE-CATID	**DIRECTORY-NAME = ... ,CATID = ... , ...**

The contents of the volumes remain unchanged, i.e. although the backups for the foreign pubsets continue to occupy space in the save files they can no longer be used. If not only the directory but also the volumes are to be cleaned up (which is not absolutely necessary) it is necessary to reorganize the archive.

C3

Migrate save files which are located on private disks or pubsets

The HSMS statement

//SHOW-ARCHIVE	**SELECT = SAVE-FILES(SAVE-FILE-ID =*BY-ATTR (SAVE-FILE-STORAGE = PUBLIC-DISK / PRIVATE-DISK)) , ...**

determines whether the affected backup archives contain save files which are located in a pubset or on a private disk. If these save files contain backups which are to be incorporated in the system backup archive of the SM pubset they must be copied to tapes for the associated backup archive. This can be done using the HSMS statement

//COPY-SAVE-FILE	**SAVE-FILE-ID = ... , ARCHIVE-NAME = ... , TO-ARCHIVE-NAME = ..., TO-STORAGE = S2-STORAGE-LEVEL , ...**

Since only the specification SAVE-FILE-ID = *LATEST is permitted in association with a COPY-SAVE-FILE call within a given backup archive (ARCHIVE-NAME = *SAME) it is usually necessary to perform a detour via another backup archive. All the save files for the affected backup archives which are located on pubsets or private disks are then deleted:

//MODIFY-ARCHIVE	**SAVE-FILES = DELETE , ...**

C4 **Check the backed-up files for name conflicts and excessive name lengths**

The DIRCONV functions MERGE-DIRECTORIES and RENAME-CATALOG ID are available for the conversion of archives. They can only be executed successfully if there are no conflicts between file names or names of job variables and no excessive name lengths. To ensure that this condition is fulfilled, users are very strongly advised to display all violations prior to conversion using the following DIRCONV functions (SILENT-RUN):

//MERGE-DIRECTORIES	**DIRECTORY-NAMES = ... ,** **SILENT-RUN = YES , ...**
//RENAME-CATID	**OLD-CATID = ... ,** **NEW-CATID = ... ,** **SILENT-RUN = YES , ...**

C5 **Clean up name conflicts and excessive name lengths**

A simple way of correcting violations is to delete the entries which result in conflicts or excessive name lengths from the directories. This can be performed using the DIRCONV statement REMOVE-CATALOG ID. This does not modify the contents of the save files in the backup volumes:

//REMOVE-CATID	**DIRECTORY-NAME = ... ,** **NEW-DIRECTORY-NAME = ... ,** **CATID = ... , USER-ID = ... ,** **FILE-NAME = ... , JV-NAME = ... , ...**

The situation is more complex if backups of files or job variables are to be retained under a different name. For any given backup this can be achieved as follows:

1. Restore the file/job variable backup (if an identically named file or job variable already exists this must be renamed and the original name subsequently restored)
2. Rename the file/job variable which is created by the restore operation
3. Back up the renamed file/job variable, if necessary with an implicit delete (BACKUP-FILES, operand DELETE-FILES);
4. Use REMOVE-CATALOG ID to remove the entry for the original backup from the directory

It should be noted that this renaming procedure modifies the relation between the backup, save file, save version and backup date.

C6	**Merge the directories** If the SF pubsets which are incorporated in the SM pubset are assigned different system backup directories these must be merged to form a common directory using the following DIRCONV statement: **//MERGE-DIRECTORIES** **DIRECTORY-NAMES = ... ,** **NEW-DIRECTORY-NAME = ... , ...**
C7	**Replace the catalog IDs in the directory** The DIRCONV statement **//RENAME-CATID** **DIRECTORY-NAME = ... ,** **OLD-CATID = *ALL = ... ,** **NEW-CATID = <identification of SM pubset> , ...** creates a new system backup directory in which all the references to the previous SF pubsets are replaced by the reference to the future SM pubset. If the processing level is adapted using SMPGEN then the new directory must be located (by specifying the catalog ID in the path name) in the SF pubset which later becomes the control volume set. Existing directories can be retained in order to make it possible to recover the existing situation if problems occur during migration. However, these directories may no longer be used for backup jobs as this would result in inconsistencies between the newly created directory and the contents of the volumes.
C8	**Perform adaptations in the MAREN catalog** If the future path name of the system backup directory in the SM pubset differs from the current path name, the new path name must be taken into account in the volume allocations in the MAREN catalog **//UPDATE-VOLUME-CATALOG** **DIRECTORY-NAMES = ... ,** **NEW-DIRECTORY-NAME = ... , ...** In the MAREN catalog, this statement assigns all the volumes which are entered in the directory specified using the NEW-DIRECTORY-NAME operand to this new directory. The DIRECTORY-NAMES operand of the UPDATE-VOLUME-CATALOG statement also makes it possible to enter free tapes, which were previously reserved for the earlier directories, in the pool of free volumes for the new directory.

C9	**Backup the backup directory** **(only necessary if the processing level is not adapted in-place)** If the processing level is not adapted in-place but by backing up and restoring files, the system backup directory must also be backed up. To do this it is also possible to copy it to a pubset which is not involved in SM pubset generation.
C10	**Clean up the archive definitions for the SF pubsets** Transition to the SM pubset means that the archive definitions for the old system backup archives and the allocations of these archives to the SF pubsets have no further significance and they can therefore be deleted. This step can also be performed at a later stage, e.g. following successful migration. This makes it simple to recover the starting situation if problems occur during migration to the SM pubset.

b) Measures following adaptation of the processing level

C11	**Set up the HSMS environment in the SM pubset** It order for it to be possible to use HSMS functions following adaptation of the processing level the HSMS environment must be installed in the pubset using the HSMS statement CREATE-SM-PUBSET-PARAMETERS. This confers the status HSMS-SUPPORTED on the pubset. MAREN catalog accesses must also be possible if HSMS functions are subsequently to be used without problems.
C12	**Restore the directory of the system backup archive (only necessary if the processing level is not adapted in-place)** If the processing level was not adapted in-place but by backing up and restoring files then the backup directory must also be restored in the SM pubset or, if it has been migrated to another pubset, copied from there to the SM pubset (see C9). It should be noted that the directory is sent to the control volume set.
C13	**Set up the system backup archive** In order to ensure that the backups in the system backup archives of the previous SF pubsets can be used for the SM pubset it is necessary to set up a system backup archive for the SM pubset and assign to it the appropriately prepared directory which may have been restored in step C12. This may be performed on the CREATE-SM-PUBSET-PARAMETERS call or in a subsequent MODIFY-SM-PUBSET-PARAMETERS call.

The system backup archive should be made available before the SM pubset is released for general use. It can subsequently be used for new backups and for accessing the backups generated prior to SM pubset generation. If the system backup archive is used to restore background levels it must first be made available for use (see B6).

6.4.3.2 Creating a new system backup archive for the SM pubset

The usual standard recommended procedure for adapting the system backup archives takes the form of setting up a new (private) backup archive immediately before SM pubset generation (i.e. when the SMPGEN checks reveal no further violations), backing up the current file store of the SF pubsets for incorporation in the SM pubset in this archive, and then later using it as the system backup archive of the SM pubset. This backup archive is automatically disconnected from foreign pubsets and therefore does not contain the backups for any files which might cause name conflicts or excessive name length during adaptation. This procedure deserves special consideration if the in-place conversion of the existing backup archives presents difficulties (e.g. because of the disconnection of foreign pubsets or the elimination of name conflicts) and the transfer of the backups it contains to the system backup archive of the SM pubset is only of subordinate importance.

First we present an overview of the measures necessary for this procedure followed by a detailed description of the individual steps.

a) Measures preceding the adaptation of the processing level

D1	Set up a new backup archive which later takes on the role of system backup archive for the SM pubset
D2	Create a current full backup of the SF pubsets to be incorporated in the SM pubset
D3	Replace the catalog IDs in the directory
D4	Perform adaptations in the MAREN catalog
D5	Back up the backup directory (not necessary if the processing level is adapted in-place)
D6	Prepare archive definitions
D7	Record the names of the system backup archives and their allocations to the previous SF pubsets

b) Measures following adaptation of the processing level

D8	Set up the HSMS environment in the SM pubset
D9	Restore the system backup archive directory (not necessary if the processing level is adapted in-place)
D10	Set up the system backup archive

a) Measures preceding the adaptation of the processing level

D1	**Set up a new backup archive** When the SMPGEN checks have been successfully performed a new shared (SF) backup archive is set up for all the SF pubsets incorporated in the SM pubset. Tapes are the only volumes allocated to this shared backup archive. This archive can subsequently play the role of system backup archive for the SM pubset.
D2	**Create a current full backup of the SF pubsets to be incorporated in the SM pubset** Each SF pubset incorporated in the SM pubset is fully backed up in this backup archive. This step should be performed immediately after the SMPGEN checks have been successfully completed.
D3	**Replace the catalog IDs in the directory** The DIRCONV statement **//RENAME-CATID** **DIRECTORY-NAME = ... ,** **OLD-CATID = *ALL ,** **NEW-CATID = <identification of the SM pubset> , ...** is used to create a new directory in which all references to the previous SF pubsets are replaced by the reference to the future SM pubset. Problems resulting from name conflicts or excessive name length cannot occur since only files and job variables are backed up which have already been checked for name conflicts and excessive name lengths during the SMPGEN checks. If the processing level is adapted in-place, the future directory must be created in the SF pubset which later becomes the control volume set.
D4	**Perform adaptations in the MAREN catalog** If the future path name of the system backup directory in the SM pubset differs from the current path name, the new path name must be taken into account in the volume allocations in the MAREN catalog: **//UPDATE-VOLUME-CATALOG** **DIRECTORY-NAMES = ,** **NEW-DIRECTORY-NAME = ... , ...** In the MAREN catalog, this statement assigns all the volumes which are entered in the directory specified using the NEW-DIRECTORY-NAME operand to this new directory. The DIRECTORY-NAMES operand of the UPDATE-VOLUME-CATALOG statement also makes it possible to allocate free tapes, which were previously reserved for the earlier directories, to the new directory, i.e. enter them in the pool of free volumes for the new directory.

D5	**Backup the backup directory (not necessary if the processing level is adapted in-place)** If the processing level is not adapted in-place but by backing up and restoring files, the system backup directory must also be backed up. To do this it is also possible to copy it to a pubset which is not involved in SM pubset generation.
D6	**Clean up archive definitions** On transition to the SM pubset, the previous (SF) archive definition is no longer required for the newly created archive and should be deleted.
D7	**Record the names of the system backup archives and their allocations to the previous SF pubsets** The previous system backup archives are retained as user backup archives, i.e. they are no longer allocated as system backup archives. Systems support should create a directory in which the name of the earlier system backup archive of each SF pubset incorporated in the SM pubset is recorded. This directory should be made available to users.

b) Measures following adaptation of the processing level

D8	**Set up the HSMS environment in the SM pubset** In order for it to be possible to use HSMS functions following adaptation of the processing level the HSMS environment must be installed in the pubset using the HSMS statement CREATE-SM-PUBSET-PARAMETERS. This confers the status HSMS-SUPPORTED on the pubset. MAREN catalog accesses must also be possible if HSMS functions are to be used without problems.
D9	**Restore the system backup archive directory (not necessary if the processing level is adapted in-place)** If the processing level was not adapted in-place but by backing up and restoring files then the backup directory must also be restored in the SM pubset (see D5). It should be noted that the directory is sent to the control volume set.

D10 **Set up the system backup archive**

The system backup archive to which the appropriately prepared directory was assigned prior to SM pubset generation is set up for the SM pubset. This can be performed at the time of the CREATE-SM-PUBSET-PARAMETERS call or as part of a later MODIFY-SM-PUBSET-PARAMETERS call. It contains current backups of all the files located in the SM pubset but no earlier backup versions.

Backups which are stored in the system backup archives of the earlier SF pubsets can be read in using the HSMS function

//RESTORE-FILES	**ARCHIVE-NAME = ... , NEW-FILE-NAMES = *BY-RULE(NEW-CATALOG-ID=<ID of the SM pubset>) ,..**

Here it is necessary to specify the explicit archive name. To determine this, users must know the SF pubset in which the file was located prior to SM pubset generation and the archive which was allocated to it as system backup archive.

In order to ensure that the necessary information is available when required, systems support staff or users should record the following data during the transition to the SM pubset:

- The names of the system backup archives of the earlier SF pubsets. These should be logged by systems support and made known to the users (see D7).
- The ID of the SF pubsets in which the files were located prior to SM pubset generation. Consequently before creating the SM pubset it is advisable to log all the files to be incorporated in the SM pubset together with their complete path names (e.g. in a SHOW-FILE-ATTRIBUTES command output list). This step can be taken by individual users for their associated user IDs. Systems support should also create a log which includes all users.
- If users rename files in order to clean up name conflicts before the generation of the SM pubset, the original file names should be recorded.

6.4.3.3 Adapting private backup archives

SM pubset generation affects all private (SF) backup archives in which SM pubset backups are to be stored in the future. If they are not adapted, then although it is still possible to access the files or job variable backups contained in them after SM pubset generation, it is not possible to create new backups of SM pubset files/job variables in these archives.

It is more difficult to determine the affected private archives than is the case with system backup archives. Here support is provided by the HSMS statement SHOW-ARCHIVE-ATTRIBUTES. Systems support should inform the owners of the affected private backup archives prior to SM pubset generation. Private backup archives can be adapted in the same way as system backup archives.

6.4.4 Adapting the long-term archives

As before, long-term archives can be used both to create and retrieve files archives involving multiple pubsets. When an SM pubset is generated from SF pubsets it is not therefore necessary to adapt the long-term archive.

6.4.5 Using ARCHIVE functions

Backup stores which contain files from SF pubsets and which were created using ARCHIVE can be read into SM pubsets using ARCHIVE or the HSMS statement IMPORT-FILES.

6.5 Reverting from SM pubset to SF pubset

6.5.1 Fundamental considerations

When reverting from an SM pubset to SF pubsets the considerations necessary for the migration to the SM pubset which are described in chapter “Generating an SM pubset from existing SF pubsets” on page 65 (modifications of path names of command procedures, adaptation of the MRSCAT entries) must be approached in the ‘opposite direction’. Reversion affects the pubset’s processing level, background levels and backup configuration. Reversion to the starting situation prior to SM pubset generation represents a special case in which modifications to the file store and modifications to files themselves which have been performed since the creation of the SM pubset become invalid. It is particularly important when problems occur during the migration to the SM pubset.

6.5.2 Adapting the processing level

Converting back

The following steps refer to the general case in which the SM pubset is subdivided into multiple SF pubsets when it is converted back. The conversion of an SM pubset into a single SF pubset with the same pubset ID represents a particularly simple special case.

- The files present in the SM pubset’s processing level, the user, and GUARDS catalogs as well as the catalog entries for files located in background levels and files located on private disks or tapes are backed up. It should be noted that the file names, including the CATID of the SF pubset, must not be longer than 54 characters. If they are, they have to be renamed beforehand.
- The SF pubsets are generated using SIR.
- The backup archive with the backups of the SM pubset is made available for the SF pubsets (import the pubset, read in the directory, define a backup archive for SF pubsets, and assign the previously read in directory).
- The backup of the user catalog which is created in the SM pubset is read back into the SF pubsets. Any SM pubset user definitions for which there is no corresponding definition in the SF pubsets (e.g. SPACE limit for WORK files) are lost. It is necessary to export and then reconstruct the SF pubsets before the user catalogs which have been read in can become effective. In general, it is necessary for systems support to edit the user catalog following this operation (e.g. adaptations of space limits, removal of individual user IDs).

- The backed up SM pubset file store is read into the SF pubsets. Systems support must determine how the files are distributed to the individual SF pubsets. If, in the SM pubset, files have been assigned files attributes for which there are no corresponding attributes in the SF pubsets (e.g. WORK files) they are automatically mapped to file attributes which are valid for SF pubsets when they are read into the SF pubset environment.

Recovery

In order to recover the status of the SF pubsets prior to SM pubset generation the SF pubsets are first recreated using SIR. The SF pubset backup store which was created prior to SM pubset generation is read back into these pubsets (including user, FACS and GUARDS catalogs, migration directory, etc.).

6.5.3 Adapting the backup archives

Converting back

HSMS makes it possible to convert an SM pubset back into one or more SF pubsets by permitting the conversion of the backup archive of an SM pubset into a backup archive for SF pubsets without it being necessary to access the backup volumes.

To do this the directory is backed up in the SM pubset and then restored to an SF pubset which may have been newly generated in the interim using SIR. It is then necessary to define a backup archive for SF pubsets to which the restored directory is then allocated. If the path name of the directory changes when it is transferred from the SM pubset to an SF pubset (e.g. because the ID of the SF pubset differs from that of the SM pubset) it is necessary to perform adaptations in the MAREN catalog. These adaptations can be performed using the DIRCONV statement UPDATE-VOLUME- CATALOG.

Initially the backups are still recorded under the SM pubset catalog ID in the backup archive of the previous SM pubset. The DIRCONV statement RENAME-CATALOG ID can be used to replace the catalog IDs in all backups either globally or for individual users only. Backups which are stored in the backup archive under a catalog ID which differs from the ID of the pubset to which they are restored must be referenced with their complete path name in the backup call, i.e. with catalog ID specification.

In the special case where the SM pubset is converted into a single SF pubset with the same ID it is not necessary to adapt the MAREN catalog or replace the catalog IDs in the directory.

Recovery

In order to make it possible to restore the original backup configuration of the SF pubsets if problems occur during the transition to the SM pubset, the directories of the backup archives are backed up prior to SM pubset generation or are transferred to a pubset which is not involved in the generation of the SM pubset. They can be used for recovery provided that they remain consistent with the contents of the backup volumes, i.e. provided that the contents of the volumes are unmodified once the directories have been fixed. In order to restore a backup archive to its status prior to SM pubset generation the following steps must be performed:

- The original backup directory (as it was prior to SM pubset generation) must be restored to an SF pubset if it was lost during SM pubset generation.
- The original backup archive must be redefined if the definition has been deleted in the interim. When this is done, the restored directory is allocated to it.
- If the path name of the original directory has been replaced in the MAREN catalog this must be reentered. This is done using the DIRCONV statement:

//UPDATE-VOLUME-CATALOG	**DIRECTORY-NAMES = *NONE , NEW-DIRECTORY-NAME=<name of the old directory> , ...**

It is only possible to restore the backup directories to their original status prior to SM pubset generation if the original backup store has not yet been modified by backups for the SM pubset.

6.5.4 Adapting the migration archive

In the special case in which an SM pubset in which no files have been migrated to S1 is converted into precisely one SM pubset, the migration archives can be converted back in-place. Even if problems have occurred during the conversion of the SF pubsets, the migration archives can be restored to their original status prior to SM pubset generation without it being necessary to access the backup volumes. In contrast, if an SM pubset is subdivided into multiple SM pubsets or if files have been migrated to S1, adaptation must be performed by backing up and restoring the files.

Converting back by backing up and restoring files

The files located in background levels of the SM pubset and the catalog entries for the migrated files are backed up, for example as part of the full backup of the SM pubset. When the SF pubsets have been set up, the backup archives containing the backed up files have been made operational and initially empty migration archives have been set up for the SF pubsets, the HSMS statement REPAIR-CATALOG-BY-RESTORE is used to restore the background level files from the backups. An alternative method consists of first reading the files which were previously located in the background levels of the SM pubset into the SF pubsets' processing level and then migrating them to background levels by means of migration jobs. The processing level of the SF pubsets must possess sufficient space for this to be possible.

Converting back by in-place adaptation

In the case of an SM pubset which is converted into precisely one SF pubset, the migration archive can be converted in-place as follows provided that no files have been migrated to S1.

- If the catalog ID of the previous SM pubset differs from the catalog ID of the future SF pubset, it is replaced in the directory using the following DIRCONV statement:

//RENAME-CATID	**DIRECTORY-NAME = ... ,** **OLD-CATID = *ALL ,** **NEW-CATID = <identification of the SM pubset> , ...**

- If the path name of the directory changes during conversion of the SM pubset into an SF pubset (e.g. because the SF pubset ID differs from that of the SM pubset) the new path name must be taken into account in the MAREN catalog. This is performed using the following DIRCONV statement:

//UPDATE-VOLUME-CATALOG	**DIRECTORY-NAMES = ... ,** **NEW-DIRECTORY-NAME =** **<path name of directory in the SF pubset> , ...**

- The migration directory and the catalog entries of the migrated files are backed up in the SM pubset.
- SIR is used to generate the SF pubset.
- The file store of the S0 level (including the migration directory) and the catalog entries of the migrated files are restored in the SF pubset from the backups.
- The migration archive for the SF pubset must be defined and allocated to the SF pubset. The restored migration directory is allocated as the directory for this archive.

Recovery

To make it possible to restore the migration directories of the SF pubsets (without S1 level) to their original status if problems occur during the transition to the SM pubset, the directories of the migration archives are backed up prior to SM pubset generation or are relocated to a pubset which is not involved in the formation of the SM pubset. They can be used for recovery provided that they remain consistent with the contents of the backup volumes, i.e. provided that no files are migrated to the background levels in the SM pubset. In order to restore a migration archive the following steps must be performed:

- The original migration directory (as it was prior to SM pubset generation) must be restored in an SF pubset if it was lost during SM pubset generation.
- The original migration archive must be redefined. When this is done the restored directory is allocated to it.
- If the path name of the original directory has been replaced in the MAREN catalog (step C7), this must be reentered. This is done using the DIRCONV statement:

//UPDATE-VOLUME-CATALOG	**DIRECTORY-NAMES = *NONE , NEW-DIRECTORY-NAME= <name of old directory> , ...**

During the transition phase, it is only possible to restore the migration directories to their original status prior to SM pubset generation during the conversion of the SF pubsets.

6.6 Example migration scenarios and procedures

6.6.1 Migration scenario 1

Migration scenario 1 is outline in figure 11. It is characterized by the fact that an SF pubset is replaced by an SM pubset with almost identical attributes. A precondition for this is that the migration and backup archives of the SF pubset have been set up to be pubset-specific. Initially the functionality of the SM pubset is largely similar to that of the previous SF pubset. It can be gradually extended (addition of further volume sets, setting up of HSMS management classes, etc.) in order to make use of the possibilities available for SM pubsets.

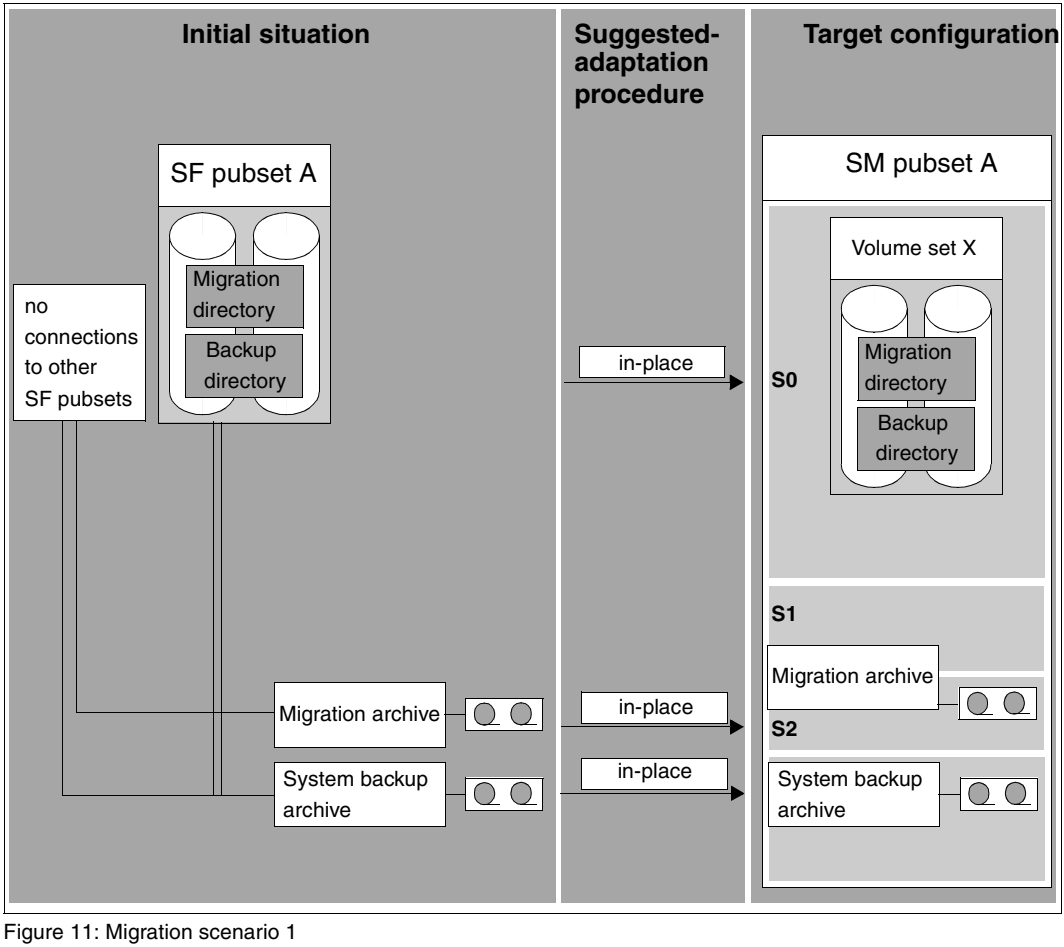


Figure 11: Migration scenario 1

Starting situation and target configuration

1. Processing level

   The physical structure of the volume set corresponds to that of the SF pubset. The file store in the processing level remains unchanged.

2. Background levels

   An S2 level but no S1 level is assigned to the SF pubset. The migration archive is not used by other SF pubsets and the migration directory is located in the pubset itself. The S2 level together with its entire file store is transferred to the S2 level of the SM pubset.

3. Backup configuration

   The SF pubset is assigned a system backup archive to which only tapes are allocated as backup volumes. It is not used for other SF pubsets and the backup directory is located in the SF pubset itself. The system backup archive together with its complete backup store is transferred to the system backup archive of the SM pubset.

Suggested procedure

Measures preceding the adaptation of the processing level	Comments
Clean up conflicts and excessive path name lengths (using SMPGEN checks)	Not necessary since the SM pubset takes on the ID of the SF pubset
Adapt command procedures and programs which contain commands/statements which are not permitted for SM pubsets	Affects systems support staff and the individual users
Adapt the private backup archives	Affects systems support staff and the individual users; the procedure depends on any special conditions (connections to other SF pubsets)
as of now: pubset operation in ACCESS-CONTROLLED mode!	
SMPGEN checks by systems support: //CREATE-SYSTEM-MANAGED-PUBSET called explicitly in test mode (OPERATIONAL-MODE=*CHECK-NAME-CONSISTENCY)	
General recommendation: perform full backup of the SF pubsets with HSMS including the background levels	D1, D2
If necessary, FDDRL backup of the SF pubsets (rapid recovery)	
Identify the directories of the migration archive and the system backup archive of the SF pubset	A2, C1
Adapt the migration archive (directory)	Not necessary since there are no connections with foreign pubsets and the SM pubset takes on the ID of the SF pubset
Adapt the system backup archive (directory)	Not necessary since there are no connections with foreign pubsets and the SM pubset takes on the ID of the SF pubset
Clean up the HSMS definitions for the SF pubsets; if necessary, following SM pubset generation (recovery)	A11, D6

Adapt the processing level	**Comments**
Rename the SM pubset (PVSREN). The new pubset ID (future volume set ID) should not be longer than the previous ID in order to avoid excessive path name lengths.	
Convert the pubset using SMPGEN: //CREATE-SYSTEM-MANAGED-PUBSET (called implicitly in execution mode)	
Adapt the presettings made by SMPGEN	
If necessary, set up storage classes and allocate default storage classes	
Adapt the MRSCAT entries at other computers	
Measures following the adaptation of the processing level	
Take pubset into service in ACCESS-CONTROLLED mode	
Set up the HSMS environment in the SM pubset	A12 (=C11)
Set up the system backup archive, allocate the directory	C13
Set up the migration archive, allocate the directory	A14, A15
Define the migration control parameters (e.g. Except file)	
Pubset is again available in general access mode	

6.6.2 Migration scenario 2

As in migration scenario 1, in migration scenario 2 (see figure 12) an SF pubset is converted into an SM pubset of the same name which, however, consists of multiple volume sets. This means that in-place conversion of the processing level is not possible. Despite this the effort required for conversion is low (as in migration scenario 1). For example, it is not necessary to adapt the addressing of the files located in the pubset. This scenario makes it possible, for example, to replace an SF pubset containing a large number of volumes (large unit of failure) by an SM pubset consisting of multiple volume sets (smaller failure subunits). Another possible application for this scenario is the replacement of an SF pubset with format K by an SM pubset which possesses one volume set with format K and one volume set with format NK2. Within the SM pubset it is possible to move the files with format NK2 to the volume set with format NK2. This means that in general they require less storage space in the SM pubset than in the SF pubset.

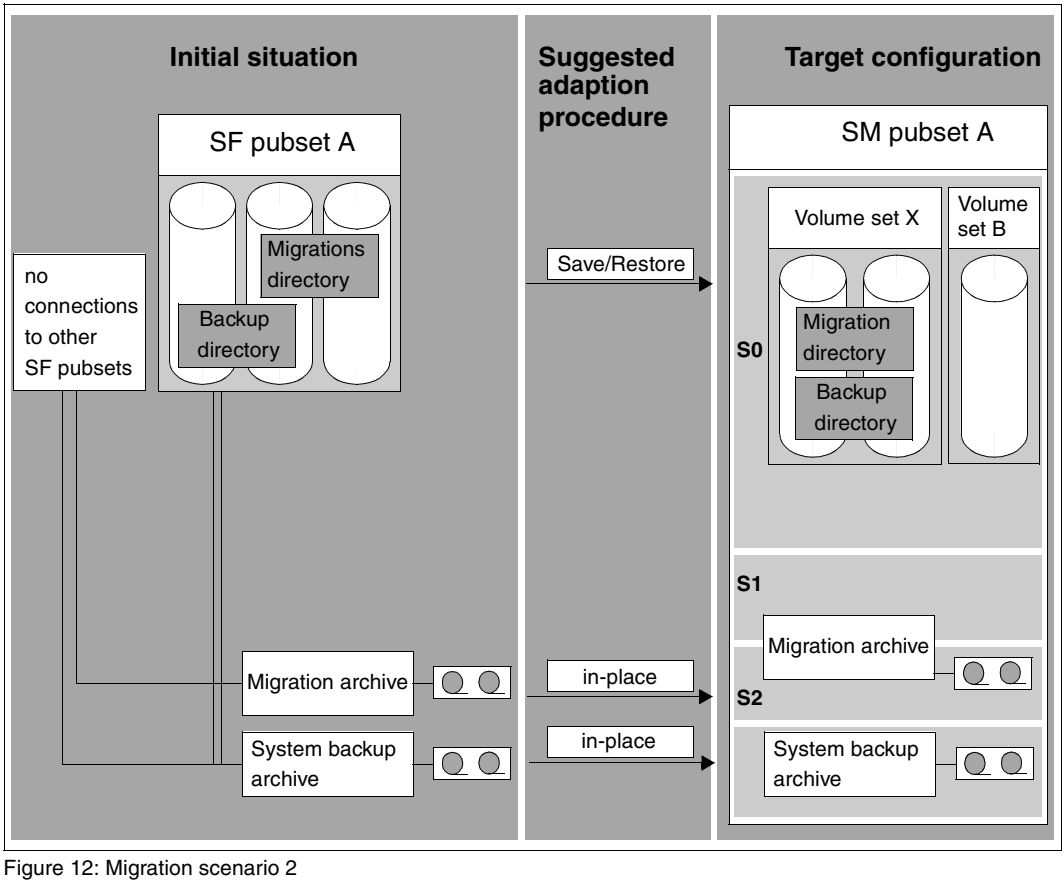


Figure 12: Migration scenario 2

Starting situation and target configuration

1. Processing level

   The physical structure of the processing level is redefined. The file store in the processing level remains unchanged.

2. Background levels

   An S2 level but no S1 level is allocated to the SF pubset. The migration archive is not used by other SF pubsets and the migration directory is located in the pubset itself. The S2 level together with its entire file store is transferred to the S2 level of the SM pubset.

3. Backup configuration

   The SF pubset is allocated a system backup archive for which only tapes are permitted as backup volumes. It is not used for other SF pubsets and the backup directory is located in the SF pubset itself. The system backup archive together with its complete backup store is transferred to the system backup archive of the SM pubset.

Suggested procedure

Measures preceding the adaptation of the processing level	**Comments**
Clean up conflicts and excessive path name lengths (using SMPGEN checks)	Not necessary since the SM pubset takes on the ID of the SF pubset
Adapt command procedures and programs which contain commands/statements which are not permitted for SM pubsets	Affects systems support staff and the individual users
Adapt the private backup archives	Affects systems support staff and the individual users; the procedure depends on any special conditions (connections to other SF pubsets)
as of now: pubset operation in ACCESS-CONTROLLED mode!	
SMPGEN checks by systems support: //CREATE-SYSTEM-MANAGED-PUBSET called explicitly in test mode (OPERATIONAL-MODE=*CHECK-NAME-CONSISTENCY)	
If necessary, FDDRL backup of the SF pubsets (rapid recovery)	
Identify the directories of the migration archive and the system backup archive of the SF pubset	A2, C1
Adapt the migration archive (directory)	Not necessary since there are no connections with foreign pubsets and the SM pubset takes on the ID of the SF pubset
Adapt the system backup archive (directory)	Not necessary since there are no connections with foreign pubsets and the SM pubset takes on the ID of the SF pubset
Clean up the HSMS definitions for the SF pubsets; if necessary, following SM pubset generation (recovery)	A11, D6

Adapt the processing level	**Comments**
Full backup of the files located in the processing level; here it is particularly important to take account of the directory files of the pubset's system backup archive and migration archive and, if necessary, the MAREN catalog; in the case of the MAREN catalog, relocation to another pubset should be considered.	
Generate the SM pubset using SIR	
Adapt the presettings made by SIR	
If necessary, set up storage classes and define default storage classes	
Restore full backup of the processing level; at the same time, restore directories of the system backup archive and the migration archive	
Adapt the MRSCAT entries at other computers	
Measures following the adaptation of the processing level	
Take pubset into service in ACCESS-CONTROLLED mode	
Set up the HSMS environment in the SM pubset	A12 (=C11)
Set up the system backup archive, allocate the directory	C13
Set up the migration archive, allocate the directory	A14, A15
Define the migration control parameters (e.g. Except file)	
Pubset is again available in general access mode	

6.6.3 Migration scenario 3

In migration scenario 3 (figure 13) an SM pubset is formed from multiple SF pubsets. All the affected SF pubsets share the same migration archive and system backup archive which is not used by any other pubsets. No disks are assigned as volumes to the migration archive or the system backup archive. The SM pubset ID must correspond to the ID of one of the SF pubsets which is to be incorporated in it. To a large extent the effort required for conversion is dependent on the scale of the necessary adaptations to user data structures (e.g. catalog ID replacement in command procedures) and modifications to file names (e.g. because of name conflicts).

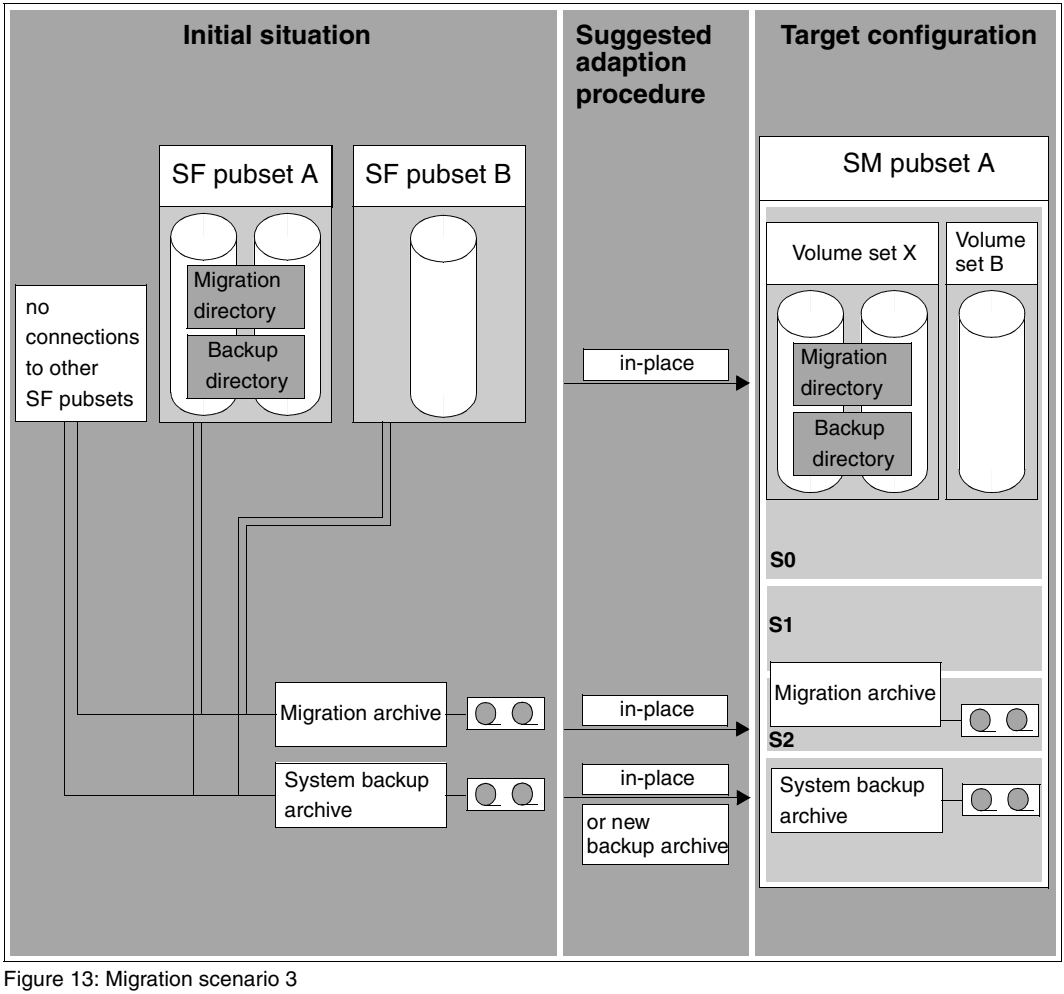


Figure 13: Migration scenario 3

Starting situation and target configuration

1. Processing level

   The processing level of the SM pubset results from the merging of the processing levels of the individual SF pubsets. The file store which it contains is not modified.

2. Background levels

   A shared migration archive is allocated to all the SF pubsets incorporated in the SM pubset. This migration archive is not used for other pubsets. The directory is located in the SF pubset which becomes the control volume set. No S1 pubset is assigned to any of the SF pubsets. The S2 level together with its entire file store is transferred to the S2 level of the SM pubset.

3. Backup configuration

   A shared backup archive is assigned to all the SF pubsets for incorporation in the SM pubset as the system backup archive. This is not used by other pubsets. Only tapes are permitted as backup volumes. The directory is located in the SF pubset which becomes the control volume set. The backup archive together with its entire backup store is transferred to the system backup archive of the SM pubset.

Suggested procedure

Measures preceding the adaptation of the processing level	Comments
Clean up conflicts and excessive path name lengths (using SMPGEN checks)	Affects systems support staff and the individual users; this step also includes A1 or B1
Take account of modified path names in command procedures. programs, GUARDS catalogs and other user data structures; allocate suitable file attributes in the file creation statements	Affects systems support staff and the individual users
Adapt command procedures and programs which contain commands/statements which are not permitted for SM pubsets	Affects systems support staff and the individual users
Adapt the default catalog IDs (in all potential home pubsets)	Affects systems support staff only

Measures preceding the adaptation of the processing level (cont.)	**Comments**	
Adapt the private backup archives	Affects systems support staff and the individual users; the procedure depends on any special conditions	
as of now: pubset operation in ACCESS-CONTROLLED mode!		
SMPGEN checks by systems support: //CREATE-SYSTEM-MANAGED-PUBSET called explicitly in test mode (OPERATIONAL-MODE=*CHECK-NAME-CONSISTENCY)		
Assign attributes to existing files: files with the attribute PERFORMANCE=STD which are located in SF pubsets consisting of high performance disks should receive the attribute PERFORMANCE=HIGH or VERY-HIGH; the availability attribute is automatically determined by SMPGEN	Can be performed centrally by systems support staff for all user files	
General recommendation: full backup of the SF pubsets with HSMS inclusive of the background levels	D1, **D2**	
If necessary, FDDRL backup of the SF pubsets (rapid recovery)		
Adapt the migration archive (directories)	A1, A5, A6, A7(optional), A8, A9	
Adapt the system backup archive (directories)	in-place	C1, C4, **C5**, C6, C7, C8, C10
	new archive	(D1, **D2** see above), D3, D4, D7
Clean up the HSMS definitions for the SF pubsets; if necessary, following SM pubset generation (recovery)	A11, D6	

Adapt the processing level	**Comments**	
Rename the SF pubset (PVSREN) whose ID is to be transferred to the SM pubset. The new pubset ID (future volume set ID) should not be longer than the previous ID in order to avoid excessive path name lengths.		
Convert the pubset using SMPGEN: //CREATE-SYSTEM-MANAGED-PUBSET (called implicitly in execution mode)		
Adapt the presettings made by SMPGEN		
If necessary, set up storage classes and define default storage classes		
Adapt the MRSCAT entries at other computers		
Measures following the adaptation of the processing level		
Take pubset into service in ACCESS-CONTROLLED mode		
Set up the HSMS environment in the SM pubset	A12 (=C11=D8)	
Set up the system backup archive, assign the directory	in-place	C13
	new archive	**D9**, D10, D11
Set up the migration archive, assign the directory	A14, A15	
Define the migration control parameters (e.g. Except file)		
Pubset is again available in general access mode		

Note

Special attention should be paid to the steps printed in bold C5, D2, D9 when selecting the procedure (in-place adaptation of the backup archive or new backup archive) since they are either difficult to implement or require tape accesses.

6.6.4 Migration scenario 4

Migration scenario 4 (figure 14) describes the most difficult scenario for the generation of an SM pubset from SF pubsets. As in migration scenario 3, multiple SF pubsets are incorporated in the SM pubset. Unlike migration scenario 3, differing migration archives are assigned to the individual SF pubsets. These migration archives can also be used for SF pubsets which are not incorporated in the SM pubset. The SF pubsets have an S1 level in addition to the S2 level. The system backup archives also have connections to SF pubsets which are not incorporated in the SM pubset. It is possible to use both tapes and disks (e.g.private disks) as backup volumes.

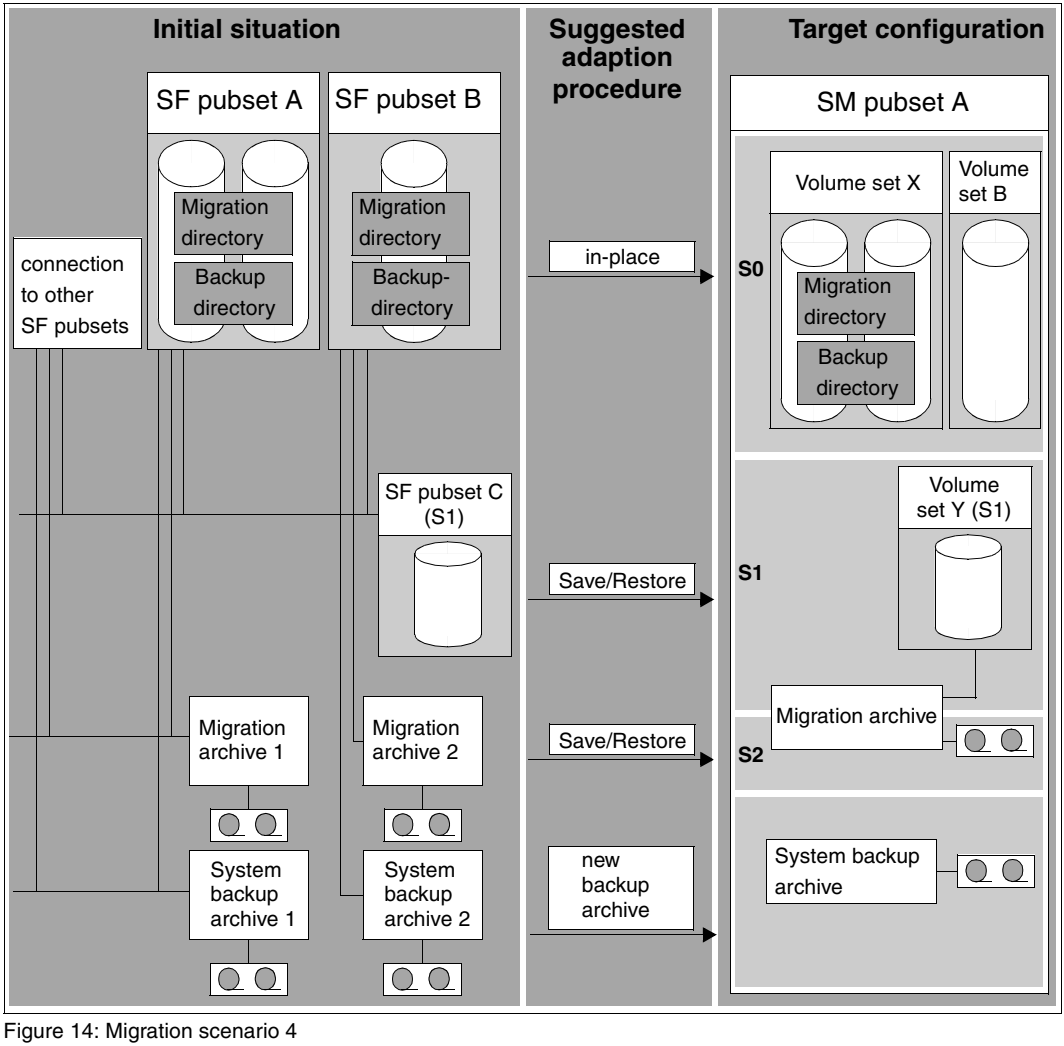


Figure 14: Migration scenario 4

Starting situation and target configuration

1. Processing level

   The S0 level of the SM pubset results from the merging of the S0 levels of the individual SF pubsets. The data store it contains is not modified.

2. Background levels

   The data store located in the S1 level or S2 level of the SF pubsets is located in the S1 level or S2 level of the SM pubset following SM pubset generation.

3. Backup configuration

   A new system backup archive is created for the SM pubset. The SF pubset backups which are created immediately before SM pubset generation are entered in this archive. Part of the backup store is taken over from the previous system backup archives. The scale of this transfer is determined by systems support in consultation with the users. The archives are retained as private backup archives for SF pubsets. This means that it is still possible to access backups contained in them which have not been transferred to the system backup archive of the SM pubset.

Suggested procedure

Measures preceding the adaptation of the processing level	Comments
Clean up conflicts and excessive path name lengths (using SMPGEN checks)	Affects systems support staff and the individual users; this step also includes A1 or B1
Take account of modified path names in com-mand procedures. programs, GUARDS catalogs and other user data structures; allocate suitable file attributes in the file creation statements	Affects systems support staff and the individual users
Adapt command procedures and programs which contain commands/statements which are not permitted for SM pubsets	Affects systems support staff and the individual users
Adapt the default catalog IDs (in all potential home pubsets)	Affects systems support staff only
Adapt the private backup archives	Affects systems support staff and the individual users; the procedure depends on any special conditions

Measures preceding the adaptation of the processing level (cont.)	**Comments**	
as of now: pubset operation in ACCESS-CONTROLLED mode!		
SMPGEN checks by systems support:		
//CREATE-SYSTEM-MANAGED-PUBSET called explicitly in test mode (OPERATIONAL-MODE=*CHECK-NAME-CONSISTENCY)		
Assign attributes to existing files: files with the attribute PERFORMANCE=STD which are located in SF pubsets consisting of high performance disks should receive the attribute PERFORMANCE=HIGH or VERY-HIGH; the availability attribute is automatically determined by SMPGEN	Can be performed centrally by systems support for all user files	
General recommendation: full backup of the SF pubsets with HSMS inclusive of the background levels	D1,**D2**	
If necessary, FDDRL backup of the SF pubsets (rapid recovery)		
Adapt the migration archives (directories)	in-place	A2, **A3**, **A4**, A5 A6, A7 (optional), A8, A9
	save/restore	**B2** (contained in D2, see above), B3 (optional)
Adapt the system backup archives (directories)	in-place	C1, C4, **C5**, C6, C7, C8, C10;
	new archive	(D1, **D2** see ab.), D3, D4, D7
Clean up the HSMS definitions for the SF pubsets; if necessary, following SM pubset generation (recovery)	A11, D6	

Adapt the processing level		
Rename the SF pubset (PVSREN) whose ID is to be transferred to the SM pubset. The new pubset ID (future volume set ID) should not be longer than the previous ID in order to avoid excessive path name lengths.		
Convert the pubset using SMPGEN: //CREATE-SYSTEM-MANAGED-PUBSET (called implicitly in execution mode)		
Adapt the presettings made by SMPGEN		
If necessary, set up storage classes and define default storage classes		
Adapt the MRSCAT entries at other computers		
Measures following the adaptation of the processing level	**Comments**	
Take pubset into service in ACCESS-CONTROLLED mode		
Set up the HSMS environment in the SM pubset	A12 (=B5=C11=D8)	
Set up the system backup archive, allocate the directory	in-place	C13
	new archive	**D9**,D10,D11
Set up the migration archive, allocate the directory	in-place	A14, A15,**A16**
Define the migration control parameters (e.g. Except file)	save/restore	B7, B8, B10, **B11**
Pubset is again available in general access mode		

Note

Particular attention should be paid to the steps printed in bold A3, A4, A16, B2, B11, D2, D9 when selecting the procedure (adaptation of backup archive: in-place or new archive, adaptation of migration archive: in-place or using save/restore) since they are difficult to implement or require tape accesses.

6.6.5 Migration scenario 5

Migration scenario 5 (figure 15) describes a case in which an SF pubset has to be integrated into an existing SM pubset. Depending on their location within the SF pubset the SF pubset files should be moved to the processing level, S1 level or S2 level of the SM pubset. As far as possible the backup store of the system backup archive should be transferred to the system backup archive of the SM pubset.

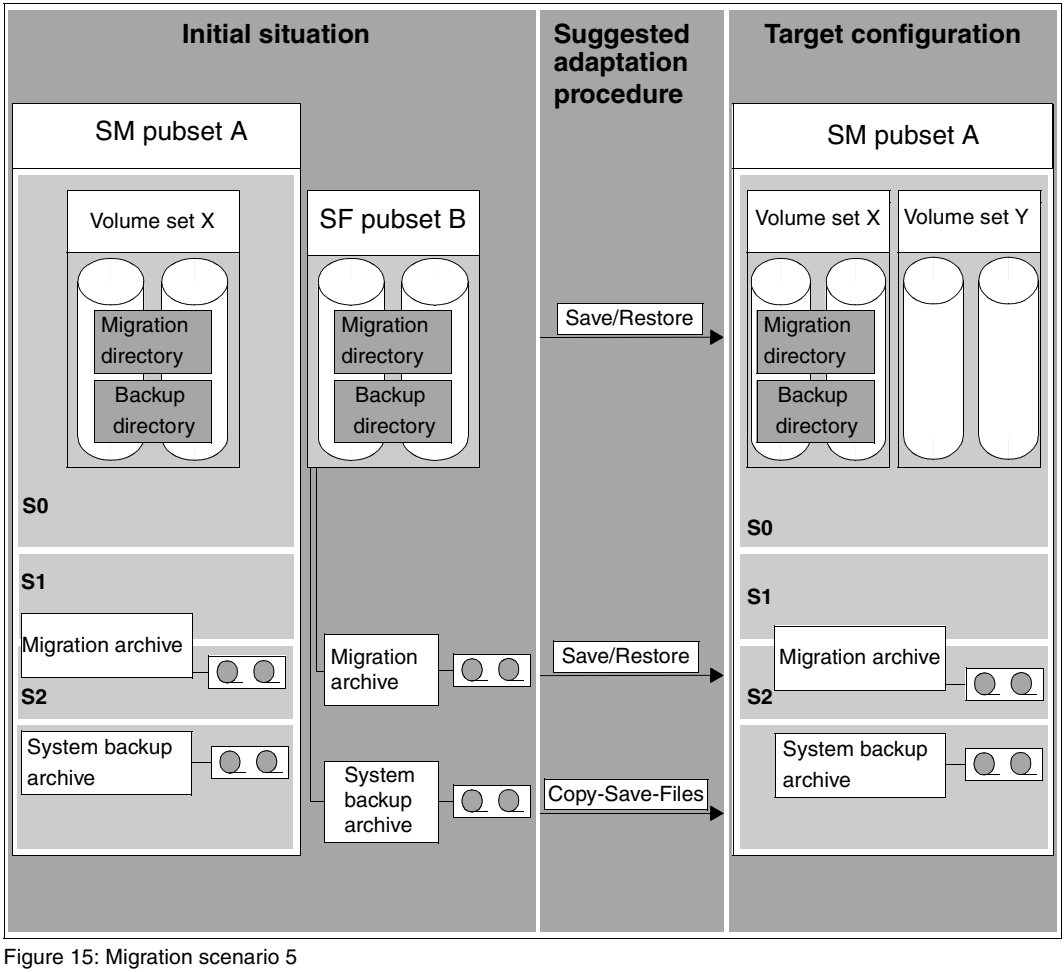


Figure 15: Migration scenario 5

Suggested procedure

The SF pubset files located in the processing level and the background levels are backed up. Systems support can use the SF pubset disks which this liberates to increase the capacity of the SM pubset. To do this it is possible either to enlarge existing volume sets through the addition of volumes or form new free volume sets from the volumes which are subsequently taken over into the SM pubset. In both cases the files located in the SF pubset are lost. In the SM pubset they must be retrieved by restoring the backup. For the transfer of files which are located in the background levels it is possible to use the HSMS function COPY-SAVE-FILE instead of backing up and restoring the files.

The HSMS function COPY-SAVE-FILE can be used to transfer the backup store of the SF pubset to the system backup archive of the SM pubset.

7 Information functions for SM pubsets

We restrict ourselves here to listing information functions that provide information on the properties of SM pubsets. You will find more detailed information in the relevant manuals.

The following pubset management information functions relate to pubset configuration and pubset usage:

- SHOW-MASTER-CATALOG-ENTRY (for user and systems support staff)
- SHOW-PUBSET-PARAMETERS (only for systems support staff)
- SHOW-PUBSET-CONFIGURATION (for user and systems support staff)
- SHOW-PUBSET-SPACE-ALLOCATION (only for systems support staff)
- SHOW-PUBSET-ATTRIBUTES (only for systems support staff)
- SHOW-PUBSET-FILE-SERVICES (for user and systems support staff)

Much of the information provided by the above commands is also available via the STAMCE program interface.

The following information functions are also relevant for the management of SM pubsets:

- SHOW-USER-ATTRIBUTES
- Information functions for storage classes
- Information functions for HSMS management classes
- Information functions for the HSMS configuration (SHOW-ARCHIVE,.....)
- SHOW-PUBSET-USAGE (HSMS statement)
- Information functions of the SPCCNTRL utility
- Information functions for GUARDS protection profiles
- Information functions at the physical level (e.g. for partitions in global storage, DRV configuration, etc.).

8 Modifying SM pubset characteristics

This chapter describes the most important options available for modifying the characteristics of SM pubsets.

8.1 Renaming pubsets and volume sets

The PVSREN utility enables you to rename SM pubsets and volume sets belonging to SM pubsets. The renaming of free volume sets is not supported.

Before you can use PVSREN, certain conditions must be fulfilled (e.g. concerning pubset status, configuration statuses of the volume sets, exclusive allocation of volumes, number of MRSCAT entries). They are described in detail in the manual “Utility Routines” [1]. In particular, the new ID of the pubset or the volume set must not conflict with IDs of other pubsets or volume sets. This applies to all computers at which the renamed pubset or pubset with a renamed volume set will be taken into service.

The basic functionality of PVSREN consists of the consistent modification of the internal pubset data structures to which the user has no direct access (e.g. SVL, pubset configuration file, file catalog). In addition, PVSREN also provides a certain amount of support for the adaptation of the pubset environment. Systems support staff can do this directly with the help of commands, for example (to change MRSCAT entries or user catalogs).

8.1.1 Renaming pubsets

Most of the points to note when renaming a pubset are the same for both SF pubsets and SM pubsets:

- Renaming a pubset changes the path names of files and job variables. If the ID of the pubset is made longer, you must ensure that path names do not subsequently exceed the maximum permissible length.
- In order to access the migrated files of a pubset after renaming, the previous pubset ID must be replaced in the entries of the migration directory with the new pubset ID. If the path name of the migration directory changes when a pubset is renamed (which is always the case with SM pubsets), the path name for the directory must be changed in the volume assignments of the MAREN catalog.
- The steps for adapting the backup archives are similar although it is not imperative to modify the CATID in the directory. We do, however, strongly recommend it because it is otherwise difficult to access backups that were created before the pubset was renamed (addressing of the backups with the previous CATID; RESTORE with RENAME).
- Existing references to the pubset ID must be adapted. This is the case, for example, for the MRSCAT entries, the default CATIDS in the user entries of potential home pubsets, program names in GUARDS profiles, IMON declarations, as well as command procedures and user programs in which files with complete path names (i.e. an explicit CATID specification) are addressed.
- Among the data structures in which the pubset ID is included, the MRSCAT entries are particularly important. In the case of SM pubsets, the MRSCAT entries of the volume sets have to be taken into account as well as those of the pubset. They have to be removed because they are no longer consistent with the pubset entry after the pubset has been renamed. The MRSCAT entries are automatically adapted by PVSREN at the computer at which PVSREN performs renaming. Systems support must explicitly adapt them at all other computers (e.g. in a shared pubset network). If no adaptation is performed at a computer, the next attempt to take the pubset into service will fail.

8.1.2 Renaming volume sets

When a volume set is renamed, the effects on its environment must again be taken into account.

- If the control volume set is renamed, the control volume set name must be modified in the MRSCAT entries at all computers at which an MRSCAT entry has been set up for the associated SM pubset (MODIFY-MASTER-CATALOG-ENTRY command). The MRSCAT entries are automatically adapted by PVSREN at the computer at which PVSREN performs renaming. At all other computers (e.g. in a shared pubset network), systems support must explicitly adapt them. If no adaptation is performed at a computer, the next attempt to take the pubset into service will fail.
- If you rename a volume set, the MRSCAT entries allocated to it become invalid. At the computer running PVSREN, the invalid MRSCAT entry is automatically deleted by PVSREN. At all other computers, the entries are automatically removed the next time the pubset is taken into service, provided they have not already been deleted explicitly by systems support using the REMOVE-MASTER-CATALOG-ENTRY command.
- If an S1 volume set is renamed, the S1 volume set allocation performed via the HSMS statement MODIFY-SM-PUBSET-PARAMETERS must be modified.
- Volume set lists containing the volume set to be renamed should be adapted.
- In certain cases, during recall HSMS tries to transfer a file back to the volume set in which it was located before migration (e.g. files with the attribute S0-MIGRATION=*FORBIDDEN). If this volume set has since been renamed, it can no longer be recognized as the original volume set for the file. In this event, HSMS ascertains the storage location in the same way as when recall with the specification NEW-DATA-SUPPORT=*BEST-VOLUME-SET. The same applies when backing up and restoring files.
- Renaming a volume set has no effect on the path names of files/job variables. The effects on command procedures and user programs are therefore less significant than when the pubset ID is changed. Adaptation is required, for example, in procedures that create files with physical allocation on a particular volume set.

8.2 Adding and removing volume sets

The disk configuration of an SM pubset can be modified. It is possible to add or remove complete volume sets, or add or remove individual volumes to enlarge or reduce volume sets already allocated to the SM pubset.

The addition of volume sets is supported by SIR as part of static pubset maintenance and also by functions of the dynamic pubset reconfiguration. For static pubset maintenance, the pubset must be offline whereas dynamic pubset reconfiguration is only possible with the pubset online. You can only use the functions of dynamic pubset reconfiguration to remove pubsets. If the SM pubset volume set configuration is modified, global pubset characteristics such as the maximum I/O length and the service spectrum of the pubset which is visible to the user are therefore also affected.

8.2.1 Pubset reconfiguration and volume set configuration statuses

Pubset reconfiguration on the volume set level must be considered in connection with the changes to the configuration status of a volume set. Changes to the configuration status can either be initiated by systems support or by the system, which automatically initiates transitions to the statuses “in hold” or “defective” when certain error situations occur. Below you can find an overview of the status transitions that occur when there are reconfiguration measures. They do not apply to the control volume set. This must always have the status “normal use” and must not be removed from the SM pubset. The pubset reconfiguration functions are described in detail in section “Adding a volume set with the pubset online” on page 150 to section “Pubset reconfiguration in the event of problems at pubset startup” on page 159.

Initial status after pubset generation

After an SM pubset has been created by SIR or SMPGEN, all the volume sets belonging to it have the status “normal use”.

Entering a new volume set in the configuration file of the SM pubset

The MODIFY-PUBSET-DEFINITION-FILE command with the operand VOLUME-SET-ENTRY= *ADD allows you to create a new entry for a volume set in the configuration file of the SM pubset. The volume set is then defined from the SM pubset's point of view but it is not allocated a physical configuration. This status is referred to as "defined only". It has the role of an intermediate status that generally bridges the phase between two consecutive pubset reconfiguration steps.

Adding a free volume set

You can allocate physical volumes to a volume set with the status "defined only" using the MODIFY-PUBSET-PROCESSING command with the operand VOLUME-SET-SUPPORT=*ADD. To do this, they must already be preconfigured in a free (and empty) volume set. This can be created with SIR, for example. Allocating the physical configuration changes the status of the volume set to "normal use" as far as the SM pubset is concerned.

Transitions to the status "in hold"

The status "normal use" of a volume set can be changed to "in hold" either implicitly by the system (when problems occur) or explicitly by systems support staff using the MODIFY-PUBSET-RESTRICTIONS command with the operand RESTRICTION=*PROC-STATE(MODE=*HOLD). The status "in hold" is also obtained when the systems support staff specify when the pubset is put into service that the volume set is to assume the status "in hold" (IMPORT-PUBSET command with the IN-HOLD-VOLUME-SET operand). The status "in hold" indicates that the SM pubset can still be used even though some of its volume sets are not functional but are not to be forcibly removed because their functionality is only disrupted temporarily.

Transitions from “in hold” to “normal use”

The status “in hold” can be canceled by means of the MODIFY-PUBSET-RESTRICTIONS command with the operand RESTRICTION =*PROC-STATE(MODE=*RESTART) operand when the volume set is intact again. If problems occur when the command is executed, the status of the volume set may remain “in hold” or it may change to “defective”. If the command is executed successfully, the status is “normal use”. When an SM pubset is put into service, the system assumes that all the volume sets of the SM pubset are functional with the exception of those for which the systems support operative explicitly specifies that they are subsequently to have the status “in hold” or be forcibly removed when the pubset is put into service (IMPORT-PUBSET command, DEFECT-VOLUME-SET or IN-HOLD-VOLUME-SET operands). If there are other volume sets that are not working in addition to those specified, pubset startup must be aborted. It must be repeated by the systems support staff, possibly with changed settings. When a pubset is put into service successfully, the status “in hold” is canceled for volume sets that previously had the status “in hold” but are now intact again. In other words, their status reverts to “normal use”..

Transitions to the status “defective”

The status “defective” is brought about by the system when it detects irreparable inconsistencies in the volume set metadata. These can be caused by software or hardware problems. For a volume set with the status “defective”, the only thing that can be done is to remove the physical configuration from the SM pubset.

Removing the physical volume set configuration

A volume set can be physically removed in the following ways:

- The physical configuration of a volume set with the status “normal use” that contains only the volume set-specific file catalog is removed from the SM pubset by means of the MODIFY-PUBSET-PROCESSING command with the operand VOLUME-SET-SUPPORT= *REMOVE(CONDITION= *EMPTY-VOLUME-SET). It then forms a free (and empty) volume set.

- The physical configuration of a volume set with the status “defective” or “in hold” can be removed from the SM pubset by means of the MODIFY-PUBSET-PROCESSING command with the operand VOLUME-SET-SUPPORT= *REMOVE(CONDITION=*VOLUME-SET-DEFECTS), even if there are still files on the volume set. The system stores the names of these files, which then also disappear from the SM pubset, in a file. The physical configuration of the removed volume set subsequently forms a free but not empty volume set. Whether it can be used subsequently depends on the condition of the associated volumes. If they are still intact, the removed volume set can be converted to an SF pubset in a special startup procedure. As a result, the files it contains become accessible again. It is not possible to add to an SM pubset a free volume set that is not empty.
- The forced removal of a volume set can also take place when the SM pubset is put into service. The systems support staff must specify the volume sets whose physical configuration is to be removed in the IMPORT-PUBSET command with the DEFECT-VOLUME-SET operand. Unlike forced removal during pubset operation, an initial status of “defective” or “in hold” is not necessary here.

When the physical configuration of a volume set is removed, the volume set entry remains in the configuration file. In other words, the volume set (as far as the pubset is concerned) has the status “defined only”.

Removing a volume set entry from the configuration file of the SM pubset

The MODIFY-PUBSET-DEFINITION-FILE command with the operand VOLUME-SET-ENTRY=*REMOVE removes the entry of a volume set from the configuration file of the SM pubset. For this to happen, the volume set must have the status “defined only”.

8.2.2 Adding a volume set with the pubset online

A number of steps are necessary to add a volume set to a pubset by means of dynamic pubset reconfiguration.

Entering a volume set in the configuration file

You use the command MODIFY-PUBSET-DEFINITION-FILE, operand VOLUME-SET-ENTRY= *ADD(...) to enter a new volume set into the configuration file of the SM pubset without initially allocating a volume set configuration to it. This step is also known as the logical addition of a volume set. This step sets the volume set to the “defined only” status.

When a volume set is entered in the pubset configuration file, MRSCAT entry is created for the volume set at all the computers at which the pubset is currently online, provided that there is no MRSCAT entry with the same ID which is already used differently. An MRSCAT entry is considered to be differently if it is not of the correct type (i.e. type SF or SM pubset) or is allocated to a volume set of another SM pubset. In this way the system automatically ensures that no name conflicts occur with the rest of the configuration at any computers at which the pubset is currently online. Systems support must explicitly ensure that no name conflicts occur at all other computers at which the pubset is to be used.

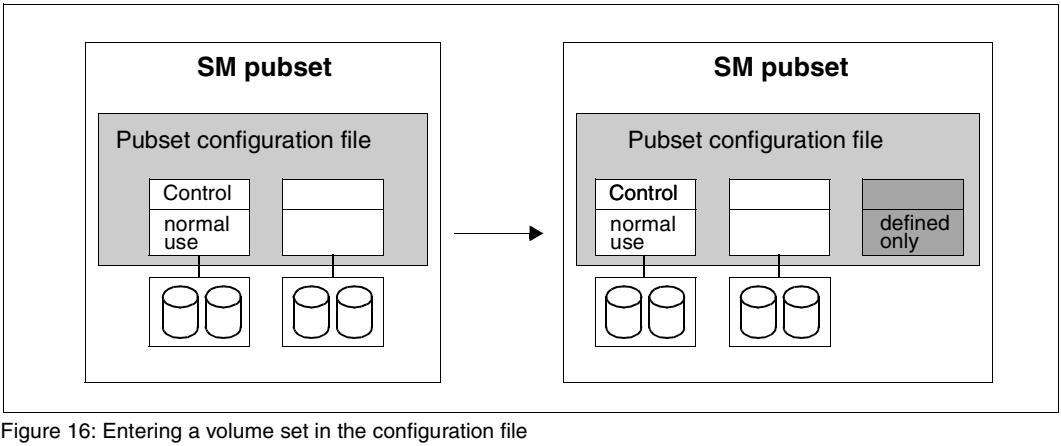


Figure 16: Entering a volume set in the configuration file

In the “defined only” status, only the volume set characteristics that are stored in the configuration file are defined. These include the performance profile, the availability profile, the usage type, the usage restrictions, the saturation thresholds, the static cache configuration, etc. When the volume set is entered in the configuration file, these characteristics are partly specified by systems support and partly initialized with default values. They can subsequently be queried and (usually) modified by systems support.

The default values largely correspond to the values which are allocated by SIR to volume sets during the generation of an SM pubset. The attributes which result directly from the physical volume set configuration - format, size of allocation unit and number of associated volumes - have the ‘defined only’ status and are thus undetermined (from the pubset’s point of view).

The ‘defined only’ status is an intermediate status suitable for the performance of certain preparation steps for the subsequent use of the volume set. If, for example, files are to be created in a volume set by means of physical allocation only then the volume set can be assigned the suitable usage restrictions while in the “defined only” status.

Adding a free volume set

A volume set with the status “defined only” is equipped with a physical configuration by assigning it a free (and empty) volume set with the same ID by means of the MODIFY-PUBSET-PROCESSING command with the operand VOLUME-SET-SUPPORT=*ADD(...). This is referred to as the physical addition of a volume set.
A free volume set is a volume set that does not belong to an SM pubset. It consists of at least one volume and is characterized by its ID, its VOLRES, the other volumes that belong to it, its format and the size of the allocation unit. Apart from the volume set-specific catalog file, it contains no further files except in special cases, which will be considered later. The volume set characteristics that are stored not in a volume set itself but in the configuration file of the SM pubset (such as the performance and availability profile, the usage type, usage restrictions and saturation thresholds) are undefined for a free volume set. Free empty volume sets can be created by SIR (BEGIN-VOLUME-SET-DECLARATION statement with the operand ACTION=*INSTALL without a preceding DECLARE-PUBSET statement). They are also formed when empty volume sets are physically removed from an SM pubset. Free volume sets with “large” volumes can only be added to LARGE-OBJECT pubsets.

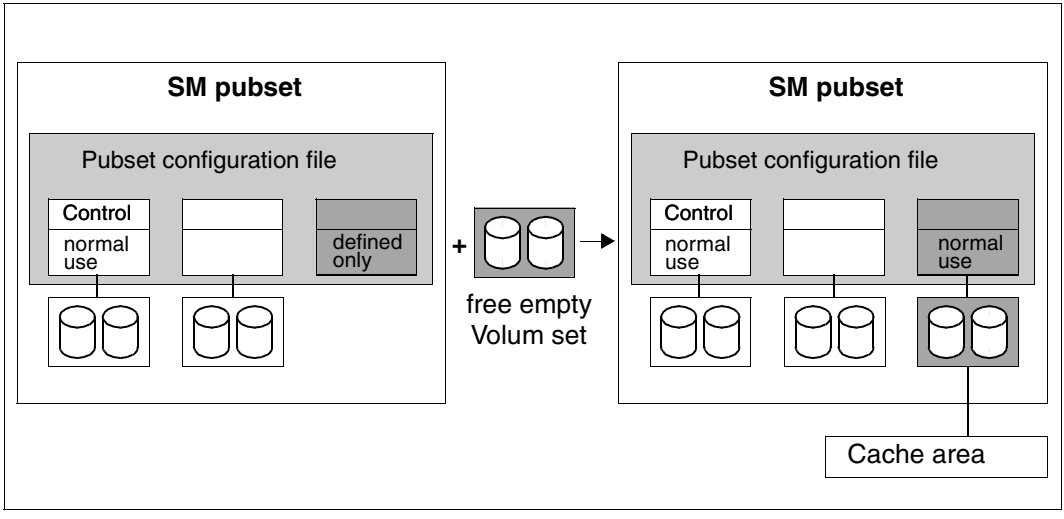


Figure 17: Adding a free volume set in the SM pubset

When you physically add a volume set, all its volumes are connected to each computer at which the SM pubset is currently online. The volumes which belong to the volume set are ascertained from the volume catalog in which the VOLRES is stored. The VSN of the VOLRES is obtained from the ID of the volume set.

If a volume set is to be operated with a cache (static cache attribute of the volume set), as specified in the configuration file, the system tries to provide an appropriate cache area and connect it to the volume set. If problems occur (e.g. no cache area available or not available in sufficient size) then the SIZE-TOLERANCE specification (see the MODIFY-PUBSET-CACHE-ATTRIBUTES command) made by systems support determines whether the volume set is added nevertheless or whether the operation is aborted.

After the volume set has been physically added to the SM pubset it has the ‘normal use’ status. In this status, the volume set characteristics also have defined values that are determined by the physical configuration (from the pubset’s point of view).

8.2.3 Using SIR to add volume sets

The SIR statement DECLARE-PUBSET together with the statement BEGIN / END-VOLUME-SET-DECLARATION can be used to add volume sets to an SM pubset. This permits a more extensive grouping of individual steps (forming the volume set from the volumes, logical and physical addition of the volume set) than is possible using dynamic reconfiguration. A number of different cases are supported:

The volume set to be added does not yet exist as a free volume set

The basic material consists of single volumes which may not yet be formatted or suitably formatted. The following SIR statements are used to form them into a volume set which is then added to the SM pubset:

```
//DECLARE-PUBSET    PUBSET-TYPE=*S-M(ACTION=*EXTEND(...))
...
//BEGIN-VOLUME-SET-DECLARATION    ACTION=*INSTALL(...) , ...
//CREATE-VOLUME    ...
...
//END-VOLUME-SET-DECLARATION
...
```

An existing free volume set is added to the SM pubset:

This case is implemented using the following SIR statements:

```
//DECLARE-PUBSET    TYPE=*S-M(ACTION=*EXTEND(...))
...
//BEGIN-VOLUME-SET-DECLARATION    ACTION=*ADD(...) , ...
//END-VOLUME-SET-DECLARATION
...
```

The prerequisites and necessary boundary conditions for the addition of volume sets using SIR are described in detail in the manual “System Installation” [3]. The following considerations must be taken into account:

- The SM pubset to which a new volume set is to be added must not be online (static pubset reconfiguration).
- Systems support must ensure that access paths exist to each volume of the volume set to be added from all computers at which the pubset is to be taken into service. This is checked implicitly at the computer at which SIR is to be run.
- If SIR is used to add an already generated volume set, this volume set must be available as a free volume set.
- Systems support must ensure that the ID of the volume set to be added does not conflict with the IDs of other SF pubsets, SM pubsets, or volume sets at any computer at which the pubset will be taken into service. This is automatically checked at the computer at which SIR is to be run since an MRSCAT entry is created for the volume set. However, this is only possible if no MRSCAT entry yet exists whose ID conflicts with the ID of the volume set. Otherwise, the addition of the volume set is rejected. At other computers where the pubset is to be taken into service, systems support must explicitly ensure that the ID of the volume set does not cause any conflicts.
- When a volume set is added to the SM pubset, the volume set is also assigned the characteristics which are not already determined by its physical volume configuration, such as performance profile, availability profile, usage type, usage restrictions, saturation threshold, specification of the cache configuration etc. While systems support can specify the performance and availability profile and the usage type via the SIR user interface (operand VOLUME-SET-ATTRIBUTES), the remaining volume set characteristics are determined by the assignment of default values. The default values are identical to the initial values after the generation of an SM pubset using SIR. Once volume set has been added, the dynamic pubset reconfiguration commands can be used to adapt the values to concrete requirements.

8.2.4 Removing volume sets with the pubset online

With the exception of the control volume set, volume sets can be removed from an SM pubset. As for the addition of a volume set, a number of steps are required.

1. **Physically removing a volume set**

   The MODIFY-PUBSET-PROCESSING command, operand VOLUME-SET-SUPPORT=*REMOVE(...) is used to physically remove a volume set in ‘normal use’ status from an SM pubset. If the volume set is connected to a cache area, the cache area is released as well. Subsequently, the volume set has the ‘defined only’ status (from the pubset’s point of view), i.e. there is still an entry for the volume set in the pubset configuration file, but it has no physical configuration in the pubset. The format and the size of the allocation unit are then undefined for the volume set (from the pubset’s point of view). The volume set characteristics (such as performance and availability profile, saturation thresholds) stored in the configuration file remain unchanged. Removing a volume set can reduce the services provided by the SM pubset. Particular attention should be paid to the fact that the recall of files which are stored in background levels may be impaired (e.g. if files in background levels have a format which is no longer supported in the processing level, or if the volume set to be removed is a pubset which is the original location of files with the attribute S0-MIGRATION=*FORBIDDEN). Various modes are available for the physical removal of a volume set:

   a) Regular mode

      In regular mode (operand CONDITION=*EMPTY-VOLUME-SET), the volume set to be removed must be empty. Empty means that no other files besides the volume set-specific file catalog are present in the volume set. Additionally, the volume set must previously have been locked (MODIFY-PUBSET-RESTRICTIONS command, RESTRICTION=*NEW-FILE-ALLOCATION(MODE=*NOT-ALLOWED) operand) against reallocation by systems support. This avoids new files being created during the removal of the volume set. Furthermore, the lock facilitates the flushing of the volume set by systems support. The procedure is discussed in more detail in section “Flushing a volume set” on page 159. After a volume set has been physically removed in regular mode, it forms a free volume set (outside the SM pubset).

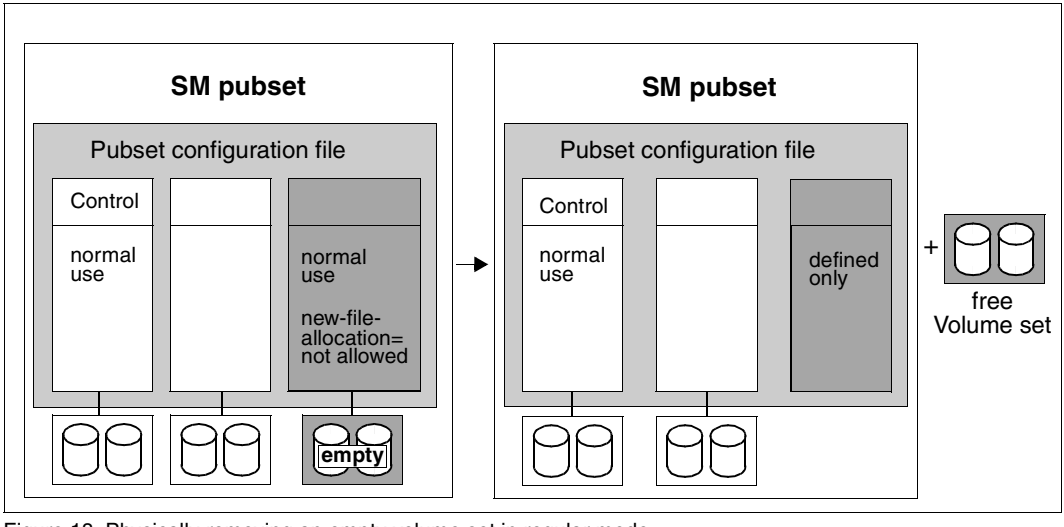


Figure 18: Physically removing an empty volume set in regular mode

b) Forced removal

It is not always possible for a systems support operative to flush a volume set, e.g. if the volume set is defective and no disk access is possible. In order to enable the removal of volume sets in such situations, a forced mode is available. This mode is specified using the operand CONDITION=*VOLUME-SET-DEFECTS. The files on the volume set are removed from the pubset implicitly. In order to inform the users of this and enable the files that have disappeared to be restored, their names are stored in the file :<catid>:$TSOS.SYS.PUBSET.DEFECT.<volsetid>.<date>.<time>. This can be used for the restoration with HSMS.

Forced removal of a volume set is only possible if the volume set has the “in hold” or ‘defective’ status. The “in hold” status is automatically set by the system when certain errors are detected or can be set by systems support using the command MODIFYPUBSET-RESTRICTIONS, operand RESTRICTION=*PROCESSING-STATE(MODE=*HOLD). The “defective” status can only be set by the system. When a volume set is removed in forced mode, it has the “defined only” status (from the pubset’s point of view).

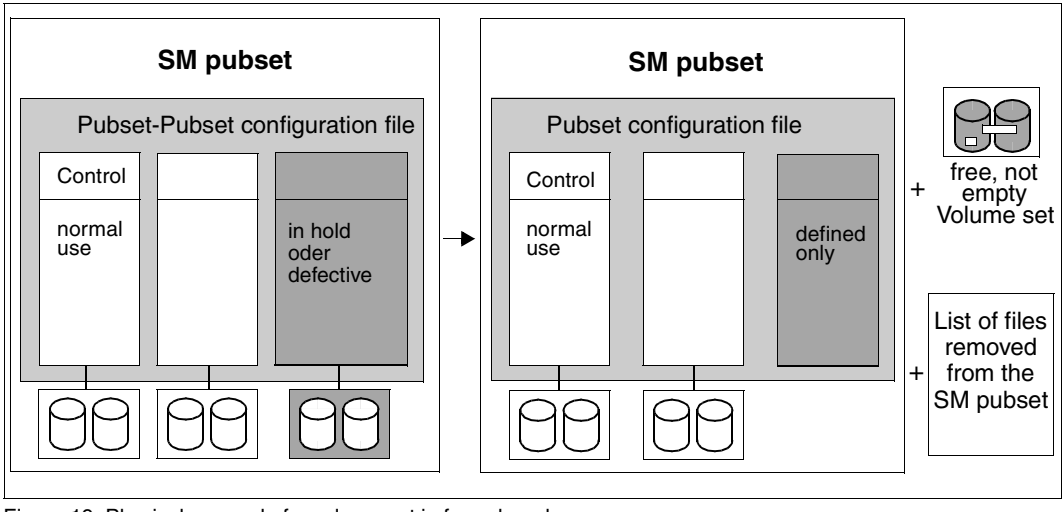


Figure 19: Physical removal of a volume set in forced mode

The volumes that have belonged to the volume set up to now form a free, but not empty, volume set. There are a number of possible applications for them.

- The main application for the forced removal of a volume set is when a volume has failed and consequently a volume set is no longer operable. In this case, the free volume set resulting from the forced removal can no longer be used as a unit. The volumes that belong to it that are still intact can only be used individually in order, for example, to create a new free (and empty) volume set with SIR that replaces the failed volume set. The lost files can be restored from backups.
- Another possible application for the forced removal of a volume set is that there is a new use planned for its volumes, but it is either not possible to empty the volume set or it would require too much effort. This situation can arise, for example, when a work volume set made available to users temporarily is to be taken away again, but individual applications do not terminate the processing of files on it or do not cancel the reservation of such files (SECURE-RESOURCE-ALLOCATION). In this case, as well, the free but not empty volume set obtained as a result of the forced removal is not suitable to be continue to be used as a unit.
- Finally, forced removal allows a volume set and its files to be removed from the pubset “in place” and then used elsewhere. To enable it to be used subsequently, the free but not empty volume set must first be converted to an SF pubset in a special startup procedure. The “in place” integration of an SF pubset in an existing SM pubset is currently not possible, however.

2. Removing a volume set entry from the configuration file

A volume set with the status “defined only” is still entered in the configuration file of the SM pubset. If the systems support operative intends to assign a physical configuration to the volume set shortly, the volume set entry with the corresponding specifications for the volume set can be retained for the time being. This is the case, for example, when a defective volume set is replaced by an intact volume set. However, if a volume set is to be removed from an SM pubset permanently, it is advisable to delete the entry from the configuration file as well.

This is enabled by the MODIFY-PUBSET-DEFINITION-FILE command with the operand VOLUME-SET- ENTRY=*REMOVE. The volume set must have the status “defined only” for this to happen. When the command is executed, the MRSCAT entries of the volume set are deleted on all systems at which the pubset is currently in operation. The removal of a volume set entry from the configuration file of an SM pubset is also referred to as the logical removal of a volume set.

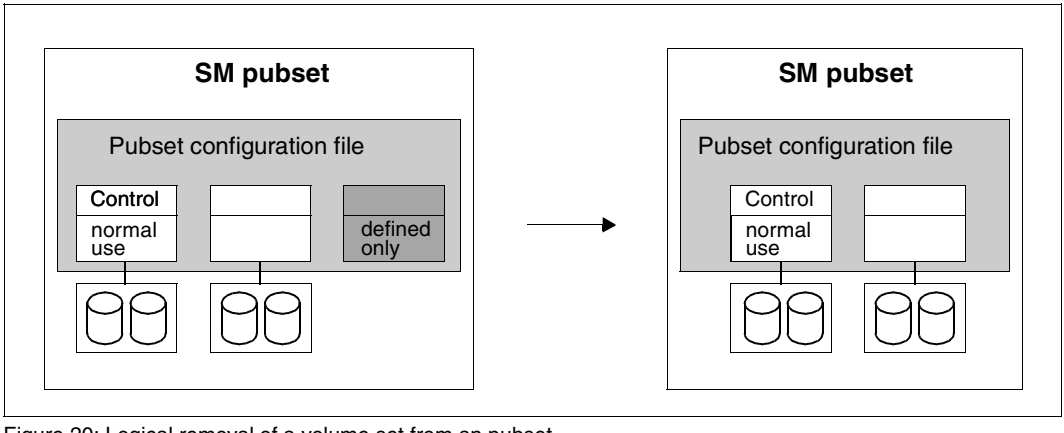


Figure 20: Logical removal of a volume set from an pubset

8.2.5 Pubset reconfiguration in the event of problems at pubset startup

When an SM pubset is put into service, this normally includes all the volume sets that belong to the SM pubset. If some of them are not accessible, this must be taken into account by the systems support staff or it will not be possible to put the pubset into service. A distinction must be drawn depending on whether a volume set is permanently damaged or whether the problems are temporary. In the former case, it is advisable to declare the affected volume set as defective when putting the SM pubset into service and thus to remove it from the pubset. This can be done by means of the IMPORT-PUBSET command with the DEFECT-VOLUME-SET operand. In the latter case, it makes sense to leave the volume set in the SM pubset but to set its status to “in hold” when the pubset is put into service. This can be done by means of the IN-HOLD-VOLUME-SET.

8.2.6 Flushing a volume set

Below we use an example to illustrate the steps which systems support may perform in order to flush a volume set. In order for this to be possible, the pubset must be online.

1. **Restricting new user activities on the volume set**

   First systems support must use the MODIFY-PUBSET-RESTRICTIONS command to set the following locks for the volume set:

   a) Lock to prevent the creation of new files

      RESTRICTION=*NEW-FILE-ALLOCATION(MODE=*NOT-ALLOWED)

      Following this command, files located on the volume set can be processed as normal. However, it is not possible to create new files in the volume set.

   b) Lock volume set access

      RESTRICTION=*VOLUME-SET-ACCESS(MODE=*ADMINISTRATOR-ONLY)

      Here the command order has to be taken into account. Before the volume set access lock can be set, the lock against new file creation must be set.
      When these locks are set, access facilities to the files of the volume set are very restricted for users without TSOS privilege. Current file processing tasks can be continued, but new processing tasks or the creation of new files are no longer possible. Moreover, users without TSOS privilege can also no longer delete files which are located in the volume set. If users are intended to help systems support flush the volume set by deleting files which are no longer required, they must be asked to do so before the volume set access lock is set.

2. Removing files

Files are removed from the volume set either through deletion, migration to a background level or relocation to another volume set within the processing level.

In general, to relocate the files to another location in the processing level, systems support first migrates the files to a background level (using the HSMS statement MIGRATE-FILES with the selection criterion VOLUME-SET-ID) and then recalls them from there into the processing level. When doing this, it is important to note that in order to be able to recall the files into the SM pubset, suitable volume sets must be available which are compatible with the file attributes of the migrated files.

Before files can be migrated to a background level, certain conditions must be fulfilled and it may be necessary to take the required steps in advance. In order to determine which files of a volume set cannot be migrated, systems support can first initiate a migration job for all files of the volume set and then either read the HSMS-REPORT-FILE or use the SHOW-FILE-ATTRIBUTES (selection criterion VOLUME-SET-ID) command to find out which of the files have not been migrated. Systems support can then create the conditions necessary for the migration of these files and start another migration job. However, it should be noted that certain files can never be migrated to background levels. A different procedure must be used to relocate these files.

It may not be possible to relocate a file for the following reasons:

a) The file is currently being processed and is reserved by a SECURE lock or affected by another lock. Systems support can use the SHOW-FILE-LOCK command to obtain information about the locks set for a file. It addition, the SHOW-PUBSET-CONFIGURATION command can be used to output the tasks and associated user IDs which occupy space in the volume set. Systems support must use this information to identify the applications currently active in the volume set and cause them to close the associated applications and release any reservations or other locks. A radical method, which has the same effect but should only be acceptable in special situations, is to abort all user tasks occupying the volume set.

b) The file may never be migrated. Examples are files with the attribute AVAILABILITY=*HIGH, temporary files, work files, files subject to the MIGRATION=*FORBIDDEN lock, files listed in the except files for migration, etc. In this case it is necessary that systems support and the users agree on a detailed procedure. One possibility is to back up the affected files, delete them and restore them again later. When the files are restored, volume sets which support the file attributes must be made available. Before these measures are performed, the owners of the affected files must be informed that special circumstances will occur temporarily and that during this period the files are not cataloged in the pubset and can therefore not be accessed. In cases where the migration locks are associated with physical allocations, file owners should be able to determine the new physical location of the files. These requirements must be taken into account by systems support when the files are restored.

8.3 Extending an SM pubset with SF pubsets

SMPGEN allows an SM pubset to be extended by one or more (not empty) SF pubsets. When the SF pubsets are added to the SM pubset a conversion takes place to equivalent volume sets. The procedure and general conditions which are described for adding SF pubsets to an SM pubset apply analogously (see the chapter “Generating an SM pubset from existing SF pubsets” on page 65ff).

This function provides additional flexibility for the assignment of data inventories to SM pubsets:

- Migration to SM pubsets can take place step by step in that the SF pubsets are included successively in an SM pubset.
- Volume sets can be transferred between SM pubsets:
  In the first step the volume sets are converted to SF pubsets with a special import (/IMPORT-PUBSET with USE=*EXCLUSIVE(CONVERT-VOLUME-SET=*YES)). The general conditions to be observed here are described in the “Commands” [10] manual under the IMPORT-PUBSET command. In a further step the SF pubsets thus created can be included in another SM pubset.
  Data backup for the SF pubsets here is also advisable (cf. the chapter “Generating an SM pubset from existing SF pubsets” on page 65).

8.4 Adding and removing volumes

The most important functions for adding and removing volumes are provided by the dynamic pubset reconfiguration commands. In addition, the pubsets can be enlarged by means of SIR and by adding further volumes when the pubset is put into service.

8.4.1 Adding volumes using SIR

Adding a volume to a volume set

The following SIR statements are used to add further volumes to a volume set in an SM pubset

```
//DECLARE-PUBSET    TYPE=*S-M(ACTION=*EXTEND(...))
//BEGIN-VOLUME-SET-DECLARATION    ACTION=*EXTEND
//CREATE-VOLUME
...
//END-VOLUME-SET-DECLARATION
....
```

Free volume sets can also be enlarged by adding further volumes. In this case, the DECLARE-PUBSET statement is omitted.

Conditions

When adding volumes to volume sets by means of SIR, the following points must be noted:

1. There must be access paths to the volumes from all computers at which it is to be possible to put the SM pubset into service. On the computer at which volumes are added by means of SIR, this is checked by the system.
2. The disk type of the volumes must permit initialization/formatting with a format and an allocation unit that correspond to the format and size of the allocation unit of the volume set. As regards the attributes of dualization with DRV and volume emulation in global storage, the homogeneity of the pubset must be preserved.

3. For the dualization of the volumes by RAID or REMOTE-COPY, homogeneity is not forced by the system. The systems support operative himself must decide whether a pubset that is not homogeneous in terms of these attributes meets the desired availability requirements.
4. There must not already be a volume with the same VSN in the volume set. The addition of volumes must not also lead to the maximum number of volumes for the volume set being exceeded. This depends on the length of the ID of the SF pubset or volume set and the size of the allocation unit. It can be increased under certain circumstances when the volume set is renamed by PVSREN.
5. If the newly added volume is a “large” volume, the pubset to be enlarged must be a LARGE-OBJECTS pubset. If it is not, the enlargement is rejected and a corresponding message is output.

   Note

   SIR automatically ensures that the VSNs of the newly added volumes fit the ID of the volume set in accordance with the naming conventions for volumes.

8.4.2 Adding volumes when the pubset is taken into service

When an SM pubset is put into service in “exclusive” mode or “shared-master” mode, all volumes in the configuration of the computer are obtained whose VSNs according to the naming conventions for volumes fit a volume set of the pubset but are not entered in the volume catalog on the VOLRES. For each of these volumes, the operator/systems support operative is asked whether it is to be added to the pubset.

A volume cannot be added unless the system’s homogeneity requirements as regards format, allocation unit, DRV and volume emulation in global storage are met. “Large” volumes are only added when the SM pubset to be enlarged is a LARGE-OBJECTS pubset. The systems support operative must also ensure that there are access paths to the newly added volumes from all the computers on which it is to be possible to put the pubset into service.

A volume that is added is considered to be empty. In other words, certain metadata areas (SVL, headers of the F5 labels) must be supplied correctly; the rest of the volume’s contents are ignored. Appropriately initialized empty volumes can be created by means of the VOLIN utility.

8.4.3 Adding volume sets by means of dynamic pubset reconfiguration

The operand VOLUME-SET-SUPPORT=*MODIFY(VOLUME-ASSIGNMENT=*ADD(...),...) of the MODIFY-PUBSET-PROCESSING command can be used to enlarge a volume set by adding a volume with the pubset online. Certain considerations must be taken into account just as when SIR is used to add volumes as well as when the pubset taking the pubset into service:

- In order for it to be possible to add a volume to a volume set, the volume set must have the ‘normal use’ status.
- Access paths to the newly incorporated volume must exist from all the computers at which the SM or SF pubset is to be taken into service. This is automatically checked by the system at all computers at which the pubset is online at the time when the command is executed. This has to be checked explicitly by systems support on all other computers.
- The volume must be suitably formatted and initialized:
  - The VSN of the volume to be added must be compatible with the ID of the pubset or the volume set in accordance with the name conventions for volumes.
  - The format and size of the volume allocation unit must correspond to the format and size of the allocation unit of the volume set or SF pubset.
  - The homogeneity of the volume set with regard to the attributes of dualization with DRV and volume emulation in global storage must be ensured.
  - For the dualization of the volume set using RAID or REMOTE-COPY the homogeneity is not forced by the system. It is the task of the systems support staff to ensure that only volumes that do not reduce the applicable availability level are added.
  - If the new volume is a “large” volume, the pubset to which it is added must be a LARGE-OBJECTS pubset. If it is not, the addition of the new volume is rejected with a corresponding message.
- If the maximum I/O length of the newly incorporated volume is smaller than that of the other volumes of the volume set, the maximum I/O length of the volume set is reduced. Privileged applications (ARCHIVE,CCOPY), which use the maximum I/O length, take account of this modification.
- No volume with a VSN identical to that of the new volume may already exist in the volume set. In addition, the maximum possible number of volumes in a volume set must not be exceeded.

Since adding a volume increases the free space available in a volume set, this operation has a (beneficial) effect on the saturation status. It may result in a reduction of the current saturation level.

If the space on the newly added volume is not assigned automatically by the system, it can in certain cases be desirable to make a suitable space available by means of physical allocation. This can be necessary, for example, when certain system files are added.This is made possible by the operand ALLOCATION-ON-VOLUME=*NOT-ALLOWED of the MODIFY-PUBSET-PROCESSING command. No space is then occupied in the newly added volume until systems support releases the allocation lock entirely using the MODIFY-PUBSET-RESTRICTIONS command or replaces it by the soft lock ALLOCATION-ON-VOLUME=*PHYSICAL-ONLY.

8.4.4 Removing an empty volume from an SF pubset or volume set

A volume can be removed from a volume set using the MODIFY-PUBSET-PROCESSING command with the operand VOLUME-SET-SUPPORT=*MODIFY(VOLUME-ASSIGNMENT=*REMOVE(...),...) while the pubset is online.

When removing a volume the following conditions must be fulfilled:

1. The SM pubset to which the affected volume belongs must online at the computer at which the commands are issued.
2. A volume set must have the ‘normal use’ status when it is reduced in size.
3. The volume to be removed must not be the VOLRES of the volume set.
4. The volume to be removed must be empty. In order to ensure that no further space is subsequently assigned in the volume, an ALLOCATION-ON-VOLUME=*NOT-ALLOWED lock should be set for the volume when it is flushed. The utilities for flushing a volume are described below.
5. The system does not allow the volume to be removed if the reduction in the capacity of the volume set would result in saturation level 4 being exceeded or if saturation level 4 is already exceeded. This prevents systems support from unintentionally causing a problematic saturation level in the volume set or SF pubset. However, if systems support still wishes to remove the volume, he or she can first set saturation level 4 to a suitable value (MODIFY-SPACE-SATURATION-LEVELS command).

8.4.5 Flushing a volume

In order to flush a volume, all the files which occupy space in the volume must first be relocated or deleted. This is usually not possible for system files such as the catalog files, the user catalog, the storage class catalog, etc.

In SM pubsets, the pubset metadata - with the exception of the catalog files of the volume set - is stored centrally in the control volume set and may be distributed across all the volumes of the control volume set. Flushing and removing the volumes of the control volume set is therefore only possible under special circumstances. The only system file present in the other volume sets of an SM pubset is the volume set-specific file catalog. When it is created using SIR or subsequently enlarged through physical allocation, its location can be restricted to certain volumes of the volume set. In this way volumes which are eligible for removal from the volume set can be kept free from the file extents of system files.

The following steps are necessary to empty a volume:

- Systems support uses the MODIFY-PUBSET-RESTRICTIONS command to set the general allocation lock (ALLOCATION-ON-VOLUME =*NOT-ALLOWED) for the volume. The effect of this is that space occupied in the volume can be released but new space cannot be made available in the volume. If saturation level 4 is exceeded as a result of the reduction of available space on the unlocked volumes, the command is rejected. This prevents systems support from unintentionally causing a problematic saturation level in the volume set or SF pubset. However, if systems support still wishes to remove the volume, he or she can first set saturation level 4 to a suitable value (MODIFY-SPACE-SATURATION-LEVELS).
- All open files that occupy space on the volume to be emptied are closed.
- The open files that occupy space on the volume to be emptied are moved around the volume set as appropriate by means of the SPACEOPT utility.

8.5 Modifying the attribute profiles for volume set selection

The attribute profiles (see section “Attribute profiles for volume set selection” on page 23) of a volume set (with an operational physical configuration) belonging to a pubset are used by the system to determine a suitable file storage location. They also define the service scope which is made available to users by the SM pubset. Modifying the attribute profiles therefore usually automatically modifies the service scope offered by the pubset. Modifications to attributes which result directly from the physical configuration of the volume set and modifications to attributes which are explicitly described by systems support are performed differently.

Modifying the format and the allocation unit

The format and the allocation unit size are associated with the physical volume set configuration. They can only be modified by regenerating the volume set.

Modifying the performance and availability profile

The performance and availability profile for a volume set are defined by systems support and can be modified with the pubset online using the MODIFY-PUBSET-DEFINITION-FILE command, operand VOLUME-SET-ENTRY=*MODIFY. This allows systems support to reevaluate the characteristics of volume sets over time. When new files are created and migrated files are recalled into the processing level, the new specification immediately becomes effective. However, it should also be taken into consideration for the distribution of existing files and files located in the processing level. To some extent systems support is forced to redistribute the files:

1. If a volume set which has previously been specified as offering high availability does not provide the expected availability, files with the attribute AVAILABILITY=*HIGH located in the volume set should be relocated to volume sets which are actually provide high availability. In order to force systems support to perform this step, the modification of the availability profile of a volume set from HIGH to STD is only permitted by the system if no file with the file attribute AVAILABILITY=*HIGH is located in the volume set. Files with high availability requirements can be moved by first backing up the files and then restoring them. The HSMS functions Migrate and Recall cannot be used for relocation since files with AVAILABILITY=*HIGH cannot be migrated to background levels. In order to prevent the system creating new files with high availability requirements in the volume set during relocation, systems support must set the lock NEW-FILE-ALLOCATION=*NOT-ALLOWED (normal volume set) or NEW-FILE-ALLOCATION=*PHYSICAL-ONLY (control volume set) for the volume set. The lock can be released as soon as it has been possible to assign AVAILABILITY=*STD to the volume set.

2. Modifications to the performance profile and an increase in the availability profile from STD to HIGH are possible at any time. When this is done, the files initially retain the same location in the volume set, even if their current distribution within the SM pubset is no longer optimal in the light of the modified volume set attributes. It is the responsibility of systems support to decide whether and when to reorganize the SM pubset in order to improve its condition.

The file attributes of existing files are not affected when the performance and availability profiles of the pubset are modified. This also applies to files which have been created in the volume set by means of physical allocation and which are bound to the volume set by the S0-MIGRATION=*FORBIDDEN migration lock. However, the owners of such files should consider whether a volume set which has been re-assessed in this way can continue to meet the requirements of the files which have been specifically assigned to it.

8.6 Modifying the usage type and usage restrictions

The usage type of a volume set can be modified with the pubset online using the VOLUME-SET-ENTRY=*MODIFY operand of the MODIFY-PUBSET-DEFINITION-FILE command. For this to be possible, the volume set must have the ‘defined only’ status. In particular, no files may be present in the volume set. This ensures that the modification of the usage type does not result in impermissible combinations of file and volume set attributes. For example, no work files may be located in volume sets with usage type STANDARD.

Systems support can use the MODIFY-PUBSET-RESTRICTIONS command to modify the usage restrictions for volume sets or individual volumes of volume sets with the pubset online. When doing this, the following points must be taken into account:

1. No usage restrictions are defined for free volume sets either at the volume set level or at the volume level. Consequently they cannot be modified.
2. Apart from the ‘in hold’ lock, usage restrictions on volume set level can be defined and modified even if the volume set has the 'defined only' status and therefore no physical configuration is currently allocated to it.
3. Usage restrictions for the individual volumes can only be modified if the volume belongs to a pubset.

4. Setting the volume set locks RESTRICTION=NEW-FILE-ALLOCATION(MODE= *NOT-ALLOWED) or RESTRICTION=NEW-FILE-ALLOCATION(MODE=*PHYSICAL-ONLY) reduces the number of generally usable volume sets. This can result in the reduction of the service range which is provided by the pubset for general use. The system does not permit the setting of these locks if it would consequently not be possible to support the default format (current value) of the SM pubset on any of the volumes sets available for general use. A similar problem exists for the 'in hold' lock. However, since this lock is primarily used to lock defective volume sets, and the system must therefore set this lock automatically when error situations occur, the default format (current value) is newly determined by the system in such cases (see section "Default pubset space settings" on page 37). When these locks are set, files which have been migrated to the background levels must also be taken into consideration. When they are recalled into the processing level a suitable, unlocked volume set must be available for them.
5. The lock RESTRICTION=*VOLUME-SET-ACCESS(MODE=*ADMINISTRATOR-ONLY) can be considered as a stronger version of RESTRICTION=NEW-FILE-ALLOCATION(MODE=*NOT-ALLOWED). This is illustrated by the following relationship:

   VOLUME-SET-ACCESS = *ADMINISTRATOR-ONLY

   implies:

   NEW-FILE-ALLOCATION = *NOT-ALLOWED

   Modifications which violate this relationship are rejected by the system.
6. Setting the ALLOCATION-ON-VOLUME=*NOT-ALLOWED and ALLOCATION-ON-VOLUME=*PHYSICAL-ONLY volume locks reduces the number of volumes available for general use and therefore the space available for general use in a volume set or SF pubset. The setting of the lock is rejected if (for the space available for general use) saturation level 4 (current value) is exceeded or would be exceeded as a result of setting the lock. If the lock is to be set despite this, systems support must first set the saturation level to a suitable value.

8.7 Modifying the cache configuration of volume sets

There is a difference between the defined values and the current cache configuration of a volume set . The defined values can be modified using the MODIFY-PUBSET-CACHE-ATTRIBUTES command. When this command is used, the SM pubset to which the volume set belongs must be imported (unlike SF pubsets!) and the volume set must have the ‘normal use’ or ‘defined only’ status. The system automatically uses the newly defined values the next time the pubset is taken into service if cache areas are provided for the individual volume sets. In addition, using the commands STOP-PUBSET-CACHING, START-PUBSET-CACHING and MODIFY-PUBSET-PROCESSING, modifications to the defined values affect the current cache configuration during the currently active pubset session. The following cases are possible:

1. Setting up a new cache configuration

   Cache use is to be activated for a volume set (in ‘normal use’ status) which has not yet been assigned a current cache configuration, i.e. since the pubset was taken into service or since the volume set was physically incorporated in the SM pubset.

   This can be achieved by first setting defined values which match the desired cache configuration and then activating caching for the volume using the START-PUBSET-CACHING command.

2. Dynamic adaptation of cache size

   The size of the cache area should be dynamically adapted for a volume set for which caching is already active. The cache medium and the cache operating parameters remain the same.

   To do this, the defined value for the cache area of the volume set is first re-specified and then caching is deactivated for the volume set (STOP-PUBSET-CACHING command). Finally volume set caching is reactivated (START-PUBSET-CACHING command).

3. Modifying the cache medium

   Following the connection of a volume set to a cache area, the currently used cache medium and the cache operating parameter VOLATILITY cannot be modified until the pubset is taken out of service or the volume set is physically removed from the pubset. This means that if systems support issues the START-PUBSET-CACHING command, it is only accepted if the currently defined values for the cache configuration are consistent with the volume set history.

   The following steps are therefore necessary to modify the current cache medium or the VOLATILITY setting for the volume set in the current pubset session:

   - Re-specify the defined values for the cache configuration
   - Flush the volume set
   - Physically remove the volume set (regular mode)
   - Physically reincorporate the volume set

In order to modify the cache configuration, systems support must make sure that the necessary cache media are available in sufficient quantities. The SIZE-TOLERANCE setting is used to tell the system how to react if the defaults for the defined value cannot or can only be partially fulfilled by the system when the pubset is taken into service, during the physical incorporation of a volume set or during the execution of the START-PUBSET-CACHING command.

8.8 Modifying the saturation thresholds of volume sets

The saturation thresholds for the individual volume sets of an SM pubset can be modified using the MODIFY-SPACE-SATURATION-LEVELS command. For this to be possible, the pubset must be imported (unlike SF pubsets!). The SCOPE operand can be used to determine whether the modifications apply to the current values (which control current saturation monitoring) only, to the defined values only (used to initialize the current values), or to both values. This specification determines the start time and the period during which the modifications are effective for saturation monitoring:

- If SCOPE=*TEMPORARY is specified, only the current values are modified. This is only permitted if the volume set does not have the 'defined only' status. The modification influences saturation monitoring immediately, but is only effective until the end of the pubset session or until the volume set is physically removed from the SM pubset. The next time the pubset is taken into service or when the volume set is physically added, the current values are determined from the unmodified defined values.
- If SCOPE=*NEXT-PUBSET-SESSION is specified, only the defined values are modified. This can also be used when the volume set has the ‘defined only’ status. The modifications do not affect saturation monitoring until the next pubset session or the next physical volume set addition, at which point the current values are determined from the modified defined values. The modifications remain effective until systems support explicitly changes them again.
- If SCOPE=*PERMANENT is specified, both the current values and the defined values are normally modified. However, for volume sets with ‘defined only’ status, the modifications only affect the defined values since the current values are not defined in this case. The modifications affect saturation monitoring either immediately or as soon as the volume set is physically incorporated in the SM pubset. The effective period is unlimited (provided that systems support does not make any subsequent modifications!).

8.9 Modifying the pubset space default values

The default pubset space settings (default settings for file format and the size of primary, secondary and maximum allocation) for an SM pubset can be modified using the MODIFY-PUBSET-SPACE-DEFAULTS command. For this to be possible, the pubset must be imported (unlike SF pubsets!). Only values which can be supported in the SM pubset are permitted as the default file format. The SCOPE operand (values: TEMPORARY, PERMANENT, NEXT-PUBSET-SESSION) can be used to specify the period of validity for the settings. The meaning of the values is analogous to the MODIFY-SPACE-SATURATION-LEVELS command. Depending on the required period of validity only the defined values, the current values or both are modified.

In certain cases, the dynamic value for the default file format is implicitly modified by the system. If a volume set switches to the 'in hold' or ‘defective’ status, the system checks whether the dynamic default file format value can still be supported in the SM pubset. If this is not the case, the system determines a current value which can be supported in the SM pubset and which is as close as possible to the previous current value (with previous current value K, NK2 is preferred; if NK2 cannot be supported, NK4 is selected; if the previous value was NK2 only NK4 can be supported). Once the previous current value can again be supported by the SM pubset (e.g. because the volume set is operational again or a new volume set has been incorporated), the system does not automatically switch back. If the previous current value is to become effective again, systems support must explicitly initiate this.

8.10 Modifying storage classes and volume set lists

Systems support can provide storage classes in order to designate the storage services provided by an SM pubset and allow users easy access to these storage services via a convenient user interface. Storage classes in connection with the volume set lists enable systems support to implement specific strategies for storage usage. The user administration commands can be used to allocate a user a default storage class which is used if the user does not explicitly request a storage service for a file. The possible applications are described in detail in section “File attributes which are relevant for file location and their default values” on page 205. Storage class management provides commands which permit the setting up, modification and deletion of storage classes and volume set lists. The allocation of a default storage class and the subsequent modification of this allocation can be performed using the MODIFY-USER-PUBSET-ATTRIBUTES command. A modification to storage classes, volume set lists or default storage classes usually becomes necessary when the strategies for storage utilization are modified in an SM pubset. This often results from the removal or addition of certain volume sets. When performing modifications, the systems support staff must remember that interdependencies exist between command procedures, files, storage classes, volume set lists, volume sets and default storage classes and that these may be affected by the modification. The following rules apply:

1. Storage classes referenced in files

   The creation of a file to which a non-existent storage class is allocated is rejected by the system. Deleting a storage class can therefore impair the executability of command procedures. It is possible to delete storage classes even if files still have references to them. A file which has a non-defined storage class allocated to it is handled like a file which has no storage class allocated to it when the file’s storage location is redetermined (on a recall).

2. Storage classes which are allocated as default storage classes

   A default storage class can be allocated even if it is not defined. It is also possible to delete a storage class which is still allocated to certain users as the default storage class. However, systems support is strongly urged to avoid allocating users non-defined storage classes as default storage classes since the affected users can then only create files by explicitly specifying the file attributes (using direct attribution or explicit specification of a storage class). This can seriously impair the executability of command procedures.

3. Volume set lists referenced in storage classes

   If storage classes are created or modified it is not possible to assign non-defined volume set lists to them. However, it is permissible to delete a volume set list which is referenced in storage classes. A storage class which has a non-defined volume set list assigned to it handled treated like a storage class without a volume set list when files are created or recalled.

4. Volume sets referenced in volume set lists

   When volume set lists are created or modified they may contain references to volume sets which are not defined in the configuration of the SM pubset, have the status 'defined only', 'in hold' or 'defective', or are subject to one of the locks NEW-FILE-ALLOCATION=*NOT-ALLOWED or NEW-FILE-ALLOCATION=*PHYSICAL-ONLY'. If a volume set is removed or if a lock is set for it, the system again does not check whether a reference to this volume set exists in a volume set list. References to these volume sets are ignored when file locations are determined.

In order to ensure uniform, comprehensible allocation behavior, systems support is strongly urged to avoid inconsistent references to storage classes, volume set lists and volume sets. In particular, when these elements are deleted, any references to them should be cleaned up. The references can be identified using the commands SHOW-FILE-ATTRIBUTES (selection criterion STORAGE-CLASS), SHOW-STORAGE-CLASS (selection criterion VOLUME-SET-LIST) and SHOW-VOLUME-SET-LIST (selection criterion VOLUME-SET). No direct information functions are available for detecting inconsistent references.

8.11 Modifying other pubset characteristics

Modifying the HSMS configuration and the HSMS management classes

The options made available by HSMS for modifying backup archives, background levels and HSMS management classes for an SM pubset are presented in the “HSMS” manual [4]. Dynamic pubset reconfiguration provides basic mechanisms for adapting the S1 level (adding, removing, enlarging and reducing S1 volume sets).

Modifying user entries and user groups

The options for setting up, modifying and removing user entries are described in the manual “Introductory Guide to Systems Support” [5]. The special conditions which apply when modifying user quotas in SM pubsets and which result from their complex structure are also described in this manual.

Modifying protection profiles

The possibilities for setting up, modifying and removing GUARDS protection profiles are described in detail in the manual “SECOS” [2].

Modifying the physical availability of volume sets

For the ways in which the availability of a volume set can be influenced at the physical level, refer to the manuals “SHC-OSD” [8] and “DRV” [6].

Modifying the specifications for the use of a pubset at a given computer

The specifications which refer to the use of a pubset at a given computer (size of the buffer areas for catalog management, EAM parameters, pubset access control, etc.) can be modified in SM pubsets using the MODIFY-MASTER-CATALOG-ENTRY command as was previously the case for SF pubsets.

Modifying the specifications for the use of a pubset in a computer network

The specifications which refer to the use of a pubset in a computer network can be modified in SM pubsets using the MODIFY-MASTER-CATALOG-ENTRY and SET-PUBSET-ATTRIBUTES commands in the same way as in SF pubsets.

9 Managing file life cycles

The user is the person who knows the processing and backup requirements for a file as well as the time and duration of its processing phases etc. and who should therefore be able to control its life cycle correspondingly. The systems support operative is the person who knows what resources are available in a data center, is specially trained in the use of the commands and utilities available for data administration and who possesses global information, i.e. information applying to multiple users. Systems support is therefore in the best position to implement the concrete measures which best meet user requirements.

SMS in BS2000/OSD makes it possible to design the interfaces between systems support and the user in such a way that they support the distribution of roles as outlined above. In addition, they make a significant level of automation of systems support tasks possible. These interfaces take account of the fact that, in the SMS world, it may still be desirable for certain users to intervene in specific ways at the physical level. Although this requirement may at first appear to go against the idea behind SMS, the SM pubsets in BS2000/OSD are designed in such a way that users who want to do this can be accommodated.

Below we consider the following areas in greater detail: use of the processing level, use of background levels, backups, use of unneeded space. These are considered in terms of the logical interface between user and systems support staff, potential for automation and direct control facilities for users.

9.1 Use of the processing level

The following considerations are important for the use of the processing level:

- Space in the processing level (S0 level) is allocated to a file when it is created. While the file is being processed it cannot leave the processing level. If the file is not accessed for an extended period, it is often advisable to migrate it to a lower cost background level in order to ensure efficient resource utilization. The file must be recalled into the processing level before it can be processed again.
- The media used in the processing level may vary greatly in their characteristics. Their suitability for a given file will differ depending on the attributes the user wishes to assign to the file. Defining the right location is therefore of considerable importance when allocating space in the S0 level. The key storage characteristics for the allocation of a suitable location are the performance behavior, reliability, format (K, NK2, NK4) and the size of the allocation unit (6K, 8K, 64K). These correspond to the file attributes: required performance, required reliability, required file format and file size. In addition, when determining a file's location it is necessary to consider whether the file is a work file which should be stored on a medium which is specially intended for work files.
- The file attributes required by a user may change during a file's lifetime. A previously well chosen location in the S0 level may then prove to be unsuitable. In such a case, it is advisable to relocate the file to a more suitable volume in the S0 level.

9.1.1 Determining the location of a file in an SM pubset - overview

A number of functions are available that fulfill the wishes of users for storage locations for files at the processing level that are as suitable as possible:

Direct attribution

The basic functionality of an SM pubset, which is automatically provided by the system, includes a standard user interface which maps the physical structure of the pubset to the logical level. This makes the possibilities made available by the pubset visible as a combination of file attributes which users can allocate to their files.

Storage classes without volume set lists

Storage classes can be set up in SM pubsets. One example application is storage classes without volume set lists. These can be used to provide users with an interface which is functionally largely equivalent to direct attribution, but which is more convenient to use. Unlike the case with direct attribution, systems support must set up and maintain the storage classes themselves.

Storage classes with volume set lists

Storage classes with volume set lists allow systems support to influence the system in the selection of volume sets and to apply specific strategies for memory utilization in an SM pubset. One advantage of this is, for example, that the resources in an SM pubset can be used in a more differentiated way than is possible with direct attribution, without it being necessary to forego the separation of logical view (for the user) and physical view (for systems support).

Physical allocation

The capability of performing physical allocation can be given to individual users within an SM pubset. This means that these users can directly influence the location of their files.

The individual possibilities and their coexistence within an SM pubset are described below.

9.1.2 Direct attribution

In an SM pubset, the systems support staff can withdraw the right to direct attribution from any user who is not capable of performing physical allocation by assigning the user a default storage class.

This section discusses the special case in which only direct attribution is used within an SM pubset (i.e. an SM pubset without storage classes, volume set lists and physical allocation). Figure 21 illustrates the relation between systems support, the user and the system in such an application.

An SF environment, in which systems support makes multiple SF pubsets with differing attributes available to users, corresponds to an SM environment with one SM pubset consisting of volume sets with differing attributes. When setting up or extending the SM pubset, systems support assigns attribute profiles for performance and availability to the individual volume sets and also defines whether or not these are volume sets for work files (usage type). Together with the allocation unit size and the format which are assigned to the volume sets when they are initialized or formatted, this defines the volume set attributes which are relevant for volume set selection.

Together with their associated attribute profiles, the volume sets of an SM pubset which have the status ‘normal use’ and are not affected by allocation locks determine the scope of the storage services which the SM pubset provides to users. A storage service is represented by a certain combination of the file attributes performance (i.e. PERFORMANCE, USAGE, DISK-WRITE), availability, format and usage type. When considering the quality which an SM pubset provides for a given storage service, it is necessary to differentiate between various levels: not achievable, achievable, almost optimally supported, optimally supported. The SHOW-PUBSET-FILE-SERVICES command allows users to display the storage services of an SM pubset and the level to which they can be achieved. If a pubset is modified by means of reconfiguration, e.g. if systems support physically adds or removes volume sets or assigns allocation locks to them, this automatically affects the service spectrum for the SM pubset which is displayed using SHOW-PUBSET-FILE-SERVICES.

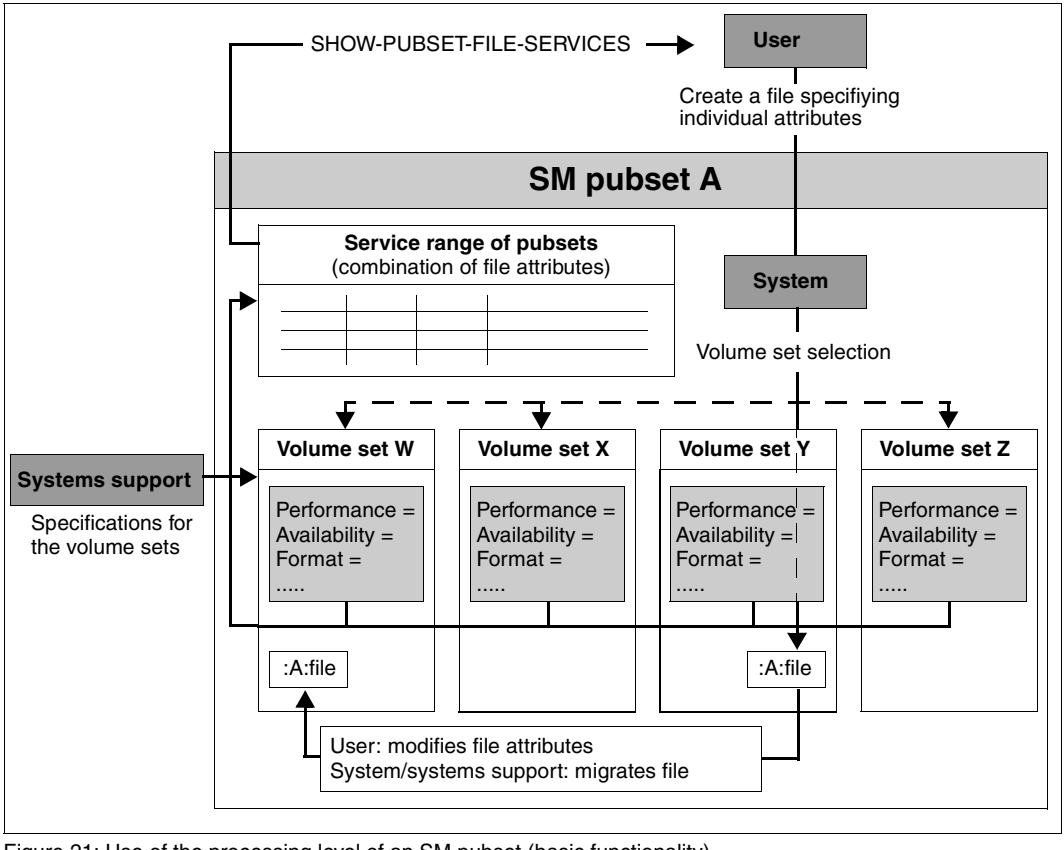


Figure 21: Use of the processing level of an SM pubset (basic functionality)

Users do not need to know the volume sets from which an SM pubset is constructed. When they create a file they determine which storage service within the SM pubset service spectrum best meets their requirements, and allocate the service to this file by specifying the corresponding file attributes (direct attribution). The system (volume set selection) uses these specifications to determine the most suitable volume set for the file while taking account of both the attributes of the volume set and the current space allocation. Volume sets which do not have the ‘normal use’ status or which are affected by allocation locks are not considered. One criterion which affects volume set selection is file size. To enable the system to evaluate this correctly, users should specify a primary allocation value on file creation which reflects the future space requirements. In general, users should specify sensible space requirements in order to make sure that the pubset can be maintained in an ordered state. For example, one inadvisable approach is to perform frequent secondary allocations. These increase the number of file extents and cause greater fragmentation of the free storage area.

If the requirements for an existing file change, users can take account of this by adapting the file attributes which affect its storage location as required (MODIFY-FILE-ATTRIBUTES command) and thus allocating it a different storage service from the storage service spectrum provided by the SM pubset. If the new file attributes are not compatible with the attributes of the volume set in which the file was previously located, it is automatically relocated by the system (e.g. new file attribute AVAILABILITY=*HIGH, availability of volume set AVAILABILITY=*STD). In all other cases, the file initially continues to be stored in the same location. The modified attributes take effect when the file is recalled into the processing level after being migrated to a background level and the volume set for the file has to be newly determined. This may, for example, take place as part of pubset reorganization which systems support should perform regularly in order to ensure the efficient use of pubset resources. The relocation of files within an SM pubset has no effect on their path names.

User quotas allow systems support to control the space occupied by individual users. SM pubsets possess a number of quota types which correspond to the different file attributes (see also chapter “Controlling resources in SM pubsets” on page 253).

9.1.3 Storage classes without volume set lists

Storage classes without volume set lists are pubset-specific named combinations of file attributes which are set up by systems support. The storage class definitions are stored in the SM pubset’s storage class catalog (see table 2).

Storage class catalog								
		Performance related file attributes						
Name of storage class	Preformat	Performance	Usage	Disk-Write	Availability	Work	allocated Volume set list	Comment
.....	.....	.....	.....	.....	.....	.....	.....	.....
STANDARD	K	STD	.....	.....	STD	NO	none	.....
HIGHAVAIL	K	STD	.....	.....	VERY-HIGH	NO	none	.....
HIGHPRF1	NK2	HIGH	READ-WRITE	IMMEDIATE	STD	NO	none	.....
HIGHPRF2	NK2	HIGH	READ-WRITE	BY-CLOSE	STD	NO	none	.....
WORK	K	STD	.....	.....	STD	YES	none	.....
.....	.....	.....	.....	.....	.....	.....	.....	.....

Table 2: Examples for storage classes without volume set lists

Figure 22 illustrates the role of storage classes without volume set lists and their effect on the interaction between users, systems support and the system.

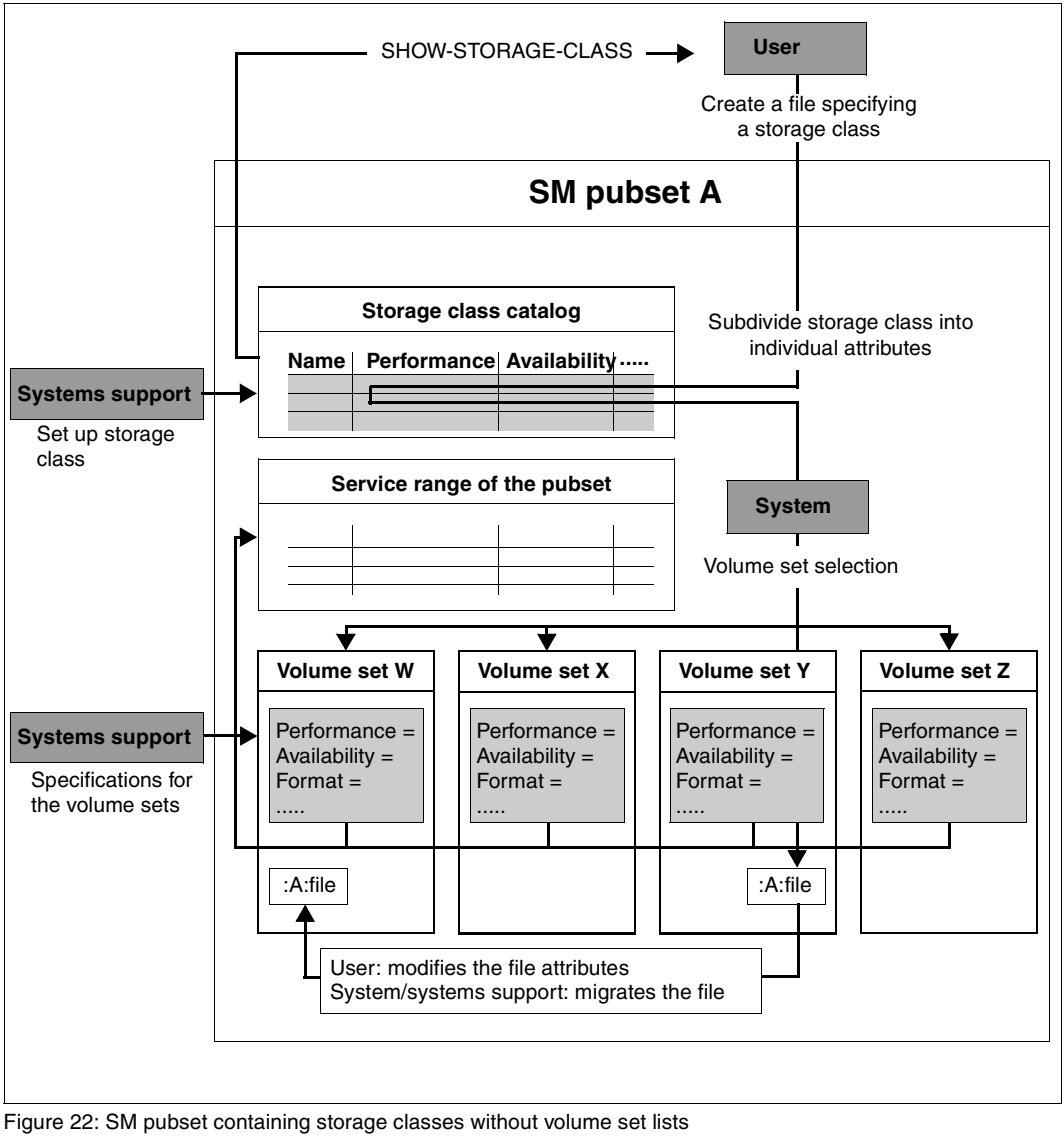


Figure 22: SM pubset containing storage classes without volume set lists

Clearly only storage classes are provided whose attributes are available in the SM pubset. This can be checked using the SHOW-PUBSET-FILE-SERVICES command. Users can obtain information about the available storage classes and their attributes using the SHOW-STORAGE-CLASS command. When files are created, storage classes can be used as an alternative to direct attribution in order to specify the required storage service. It is possible to take account of changing requirements for existing files by allocating different storage classes.

In so far as storage location selection, control of file relocation and space quotas are concerned, storage classes without volume set lists are largely equivalent to the direct attribution procedure. However, they have the advantage that user command procedures can be maintained in a clearly structured and compact way. When compared to direct attribution they result in a higher level of linkage between command procedures and specific pubsets, a feature which sometimes may be unwanted. Command procedures in which storage classes are referenced can only run in SM pubsets in which these storage classes are defined. Storage classes require more intensive maintenance on the part of systems support than direct attribution. If the configuration of an SM pubset changes, systems support should ensure that the effects on the service spectrum provided by the pubset are also taken into account in the storage classes provided. Systems support should consult users before deleting storage classes since this operation may make it necessary for them to adapt their command procedures.

One special, but very important, application for storage classes is their use as default storage classes. If a user does not explicitly specify the required storage service for a file (i.e. by specifying a storage class or individual attributes), the system checks whether a default storage class is allocated to this user. If this is the case then the user requirements are processed in precisely the same way as they would be if this storage class had been explicitly allocated to the file. If storage classes are exclusively used as default storage classes, they are not entered in command procedures. In this case they also do not result in command procedures being connected to specific SM pubsets.

The allocation of GUARDS profiles makes it possible to restrict storage classes to individual users. This function is primarily used to protect resources in the case of storage classes with allocated volume set lists (see section “Storage classes with volume set lists” on page 185). For storage classes without volume set lists, the use of GUARDS profiles is only relevant to those users from whom the ability to perform direct attribution has been withdrawn by means of a default storage class.

9.1.4 Storage classes with volume set lists

The systems support staff can influence the way in which the system selects volume sets by setting up volume set lists in an SM pubset, and linking these to storage classes. The volume set lists are stored in the SM pubset volume set list catalog (see table 3 and table 4).

Storage class catalog								
		Performance related file attributes						
Name of the storage class	Preformat	Peformance	Usage	Disk-Write	Availability	Work	allocated volume set list	Comment
.....	.....	.....	.....	.....	.....	.....	.....	.....
STDUSR1	K	STD	.....	.....	STD	NO	USR1LST	.....
STDUSR2	K	STD	.....	.....	STD	NO	USR2LST	.....
SPCPERF1	K	VERY-HIGH	WRITE	IMMEDIATE	STD	NO	GSVS1LST	xxxx
HIGHPRF2	K	VERY-HIGH	READ		STD	NO	GSVS2LST	xxxx
AVAIL	K	STD	.....	.....	HIGH	NO	none	.....
WORK	K	STD	.....	.....	STD	YES	none	.....
.....	.....	.....	.....	.....	.....	.....	.....	.....

Table 3: Example for the definitions of storage classes with volume set lists

Volume set list catalog	
Name of volume set list	associated Volume sets
.....	
USR1LST	VS1, VS2
USR2LST	VS3, VS4
GSVS1LST	GVS1
GSVS2LST	GVS2
.....	.....

Table 4: Example for volume set lists

It is necessary to distinguish between two categories of volume sets in SM pubset with volume set lists:

1. Volume sets which are present in (at least) one of the volume set lists which have been set up by systems support (note: a volume set may be present in more than one of the volume set lists defined by systems support).
2. Volume sets which do not belong to any of the volume set lists defined by systems support. These volume sets are considered to be part of the default volume set list which is thus implicitly defined. In SM pubsets in which no volume set lists are defined, all the volume sets belong to the default volume set list.

Figure 23 illustrates how user's requirements are taken into account during the selection of a volume set in SM pubsets with volume set lists. Here it is necessary to distinguish between the following cases:

1. Specification of a storage class with volume set list

   If a user allocates a storage class to a file on creation, the system first uses the storage class definition to determine the combination of file attributes this represents. If the storage class is linked to a volume set list (on the basis of its definition) then all the associated volume sets are identified. From these the system selects the volume set which most closely corresponds to the file attributes allocated to the storage class as the storage location for the file. When doing this, the system also considers the occupancy of the individual volume sets. Volume sets which are affected by an allocation lock or volume sets which do not have the status 'normal use' are not considered. If it not possible to create the file on any volume set within the volume set list, the system also considers the other volume sets of the SM pubset as part of volume set selection provided that these have the status 'normal use' and are not affected by an allocation lock. When creating or modifying a storage class, systems support must ensure that it is allocated a combination of file attributes which can be fulfilled by the volume sets of the associated volume set list. If this does not happen, the behavior of the system when it comes to volume set selection can be difficult to comprehend because volume sets of other volume set lists have to be used. The SHOW-PUBSET-FILE-SERVICES command (VOLUME-SET-LIST operand) helps you find out which combinations of file attributes are supported by a volume set list.

2. Specification of a storage class without a volume set list

   At volume set selection, storage classes without a volume set list are treated in exactly the same way as storage classes with volume set lists, except that the implicitly defined volume set list is used instead of a volume set list created by the systems support staff. Accordingly, when creating or modifying a storage class without a volume set list, the systems support staff have to ensure that the combination of file attributes assigned to it can be fulfilled by volume sets in the default volume set list. To find out the combinations of file attributes that are possible on the volume sets of the default volume set list, you can use the SHOW-PUBSET-FILE-SERVICE command (VOLUME-SET-LIST= *NONE operand).

3. Direct attribution

   In the case of direct attribution as well, volume sets are selected in exactly the same way as if the default volume set list were specified as the volume set list. From the volume sets it contains (i.e. all volume sets that do not belong to a volume set list defined by the systems support staff), the one that most closely matches the attributes specified by the user is selected. If no suitable volume set is found on which the file can be created, the other volume sets of the SM pubset that have the status “normal use” and are not subject to an allocation lock are included in the selection. Users from whom the right to direct attribution has not been withdrawn should be encouraged by the systems support staff to limit themselves in direct attribution to those combinations of file attributes that are supported by the volume sets of the default volume set list. They can obtain this information by using the SHOW-PUBSET-FILE-SERVICE command. If other combinations of file attributes are specified in direct attribution, the allocation strategies used by the systems support staff may be disrupted. The files then end up on volume sets that belong to explicitly defined volume set lists and are thus generally assigned for a specific use.

4. Modifications to the required storage service

   If a new storage class is allocated to an existing file or if the file attributes are modified by means of direct attribution, this may implicitly result in the allocation of a different volume set list. If the new file attributes are not compatible with the volume set in which the file has previously been located, the system automatically relocates the file to a different volume set. The new volume set list is also considered as part of this operation. In other cases the file initially continues to be stored at its existing location. The modified file attributes and the new volume set list do not take effect until the file is recalled into the processing level following migration to a background level. At this point the most suitable volume set for the file is determined (e.g. during pubset reorganization).

If no volume set lists have been set up in an SM pubset then all the volume sets of the SM pubset automatically belong to the default volume set list. The scenarios portrayed in sections below can thus be viewed as a special case of an SM pubset with a single volume set list to which all volume sets belong and that is always used for volume set selection.

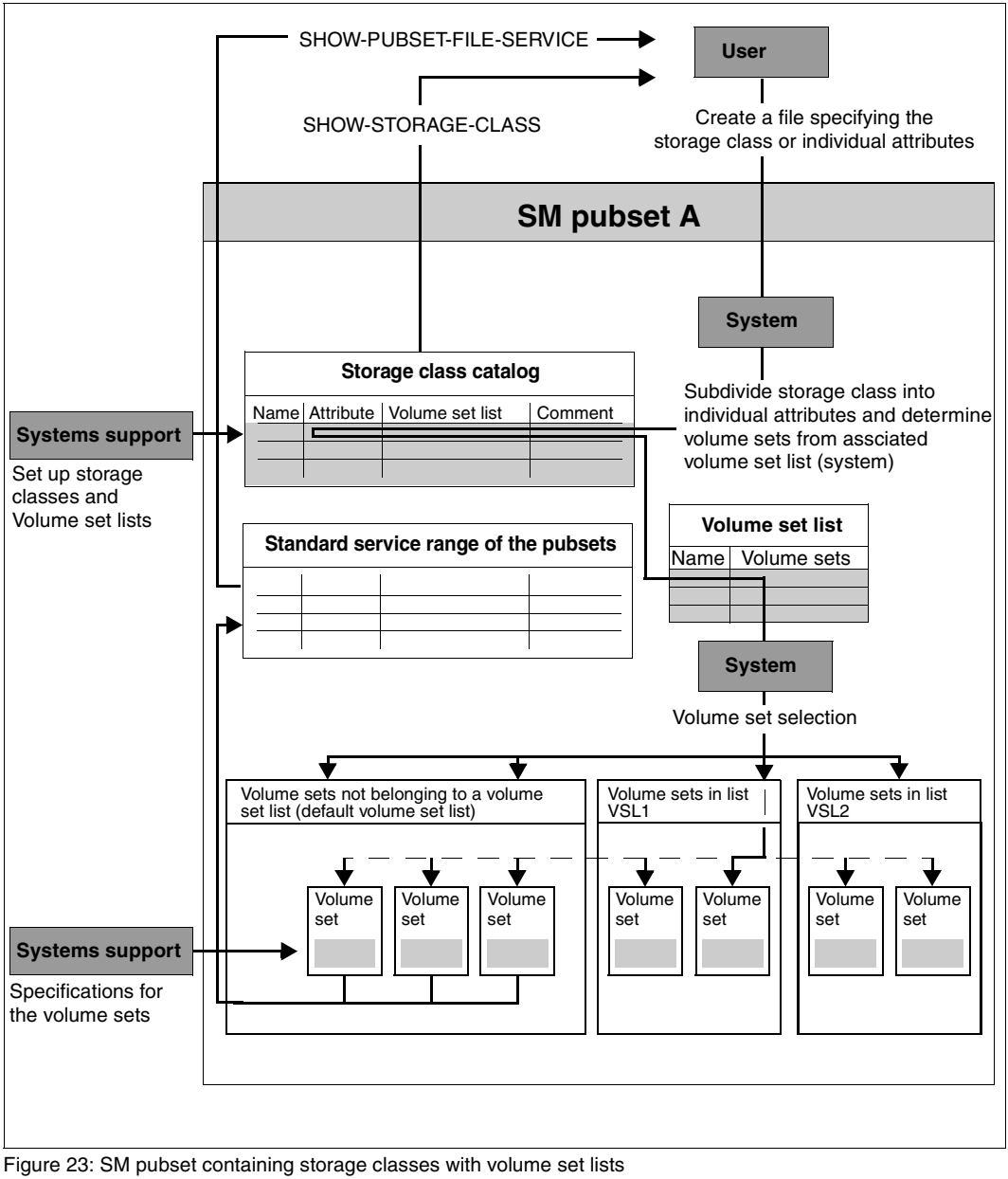


Figure 23: SM pubset containing storage classes with volume set lists

Example applications for storage classes with volume set lists

Differentiated resource utilization

It is only possible to take account of the various characteristics of different media to a limited extent in the volume set attribute profiles which systems support provides for volume set selection. For example, it is not possible to specify whether very high performance is achieved by allocating a GS cache which is reliable for write operations or by using high performance volumes such as volumes emulated in global storage. The accuracy of the file attributes which are available for file requirement specification is correspondingly limited. In consequence, targeted use of the varying characteristics of different media is only possible at a limited level of precision when direct attribution or storage classes without volume set lists are used. This represents a particular problem in the case of large SM pubsets in which a large number of different media are used simultaneously.

In order to permit a greater degree of differentiation, systems support groups volume sets with different attributes in volume set lists. This means that systems support is able to determine the required level of detail. For example, if it is necessary to distinguish between volume sets consisting of fast disks and volume sets consisting of volumes emulated in global storage (see figure 24), a separate volume set list is created for both types of volume set. Systems support then sets up storage classes to which the volume set lists are allocated, e.g. storage class SPCPRF1 and SPCPRF2. Depending on which of the storage classes is allocated to a file, this will preferably be created in volumes emulated in global storage and it will exhibit the specific performance characteristics associated with the selected medium.

If storage classes with volume set lists (such as GSVS1LST and GSVS2LST) are used to differentiate between services, then the services are not sufficiently characterized by the allocated file attributes since these are unable to express the differences between services in detail. The service characteristics which cannot be described by means of file attributes must be described by systems support in the comments section of the storage class. Users can view these comments using the SHOW-STORAGE-CLASS command. When describing storage classes, systems support staff should restrict themselves to those characteristics which users need to know about when selecting a suitable storage class, i.e. as far as possible the systems support staff's physical view should be mapped to a logical user interface.

User quotas for allocated space take account of the varying quality of the storage services provided by an SM pubset only in so far as these are reflected by the performance and the availability file attributes. To permit a greater level of differentiation, it is possible to use GUARDS protection profiles to restrict the usage of a storage class to certain users. This means that volume sets which belong to volume set lists normally only receive files from users who are authorized to use those storage classes to which the corresponding volume set lists are allocated. The limits imposed by volume set lists are ignored if they would make it necessary to reject user requirements - e.g. because of the current occupancy level.

This is also the case if users use direct attribution to specify a combination of file attributes which are incompatible with the volume sets of the default volume set list. In such cases the system also considers volume sets which belong to the volume set lists, defined by systems support when performing volume set selection. Users should also avoid such specifications since these may conflict with the storage utilization strategies planned by systems support. A similar problem may occur if systems support defines storage classes incorrectly, e.g. by allocating a combination of file attributes which cannot be provided in the default volume set list to a storage class without a storage class list. Storage class administration therefore requires considerable care.

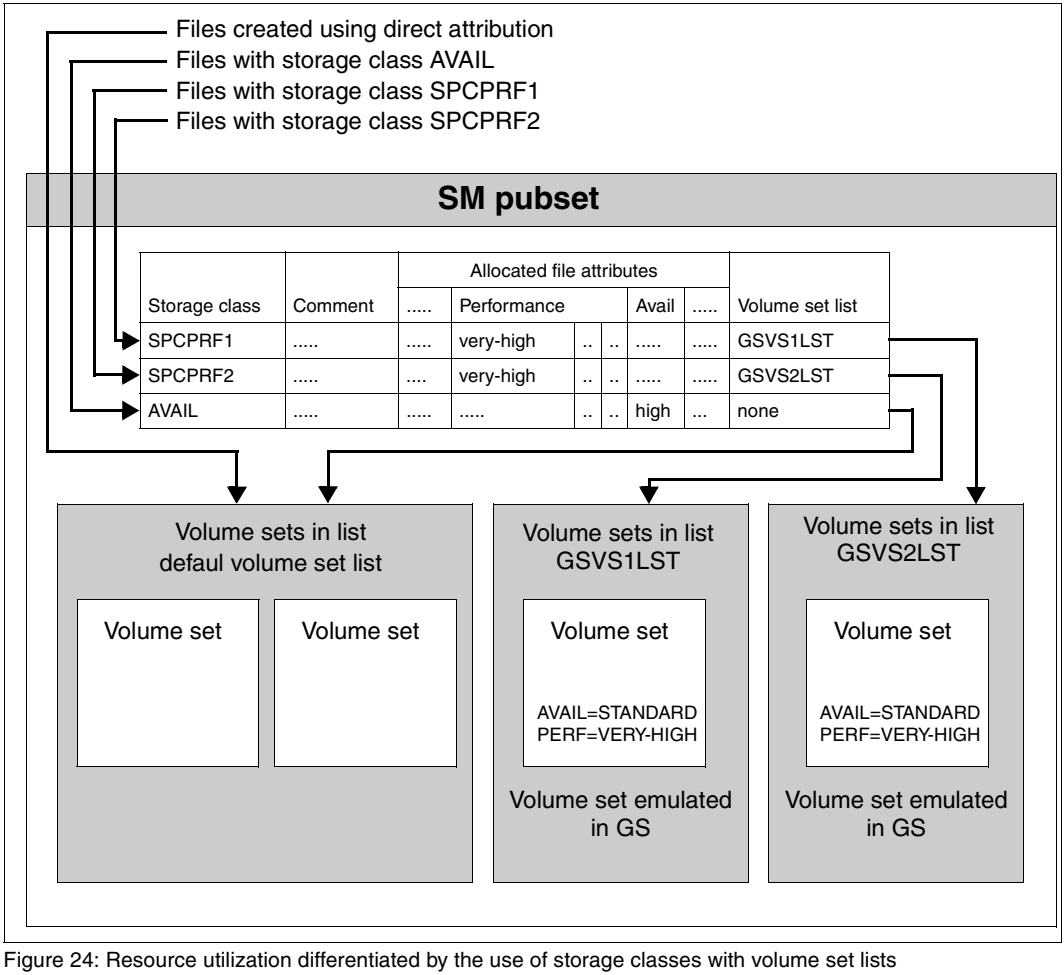


Figure 24: Resource utilization differentiated by the use of storage classes with volume set lists

SM pubset volume sets with attributes which can be properly considered during volume set selection without any special measures being necessary do need to be entered in volume set lists. They therefore form part of the default volume set list and are automatically used to store files whose attributes are specified using direct attribution. Systems support can also set up storage classes without volume set lists to simplify the use of the services provided by the default volume set list. Direct attribution, storage classes without volume set lists and storage classes with volume set lists can therefore be used simultaneously within an SM pubset in such a way that they complement one another.

Users do need to know how storage services are mapped to the volume sets of the SM pubset. In particular they do not require any information about volume set lists and the way these are allocated to storage classes. They can use the SHOW-PUBSET-FILE-SERVICES and SHOW-STORAGE-CLASS commands to obtain information about the service spectrum provided by the SM pubset and should take a note of the associated storage class descriptions. The SHOW-STORAGE-CLASS command only displays those storage classes for which the user has GUARDS authorization.

Separating pubset users

The use of storage classes within volume set lists is extremely flexible in order to make it possible to use specific space allocation strategies within an SM pubset. This is illustrated by the following example, even if it does not correspond to a typical application:

We consider the case of an SM pubset which consists of a large number of ‘normal’ volume sets which have the same attributes and smaller number of volume sets with special attributes (see figure 25). While the optimum location for files with special requirements can usually be determined, there are a large number of equivalent alternatives for files for which there are no special requirements. Systems support now wishes to influence volume set selection in such a way that the standard files created by individual users are only sent to certain volume sets wherever this is possible. The intention is to ensure that if a normal volume set fails then only a small number of users are affected.

To do this a number of volume set lists with ‘normal’ volume sets are first created. Then storage classes are set up and the volume set lists are allocated to these. Each storage class therefore represents a number of normal volume sets. Depending on the volume set to which a user’s standard file should preferably be sent, a corresponding default storage class is allocated to the user. If no explicit requirements are specified for a file (by means of direct attributes or the explicit specification of a storage class), it is allocated to its owner’s default storage class. The system therefore attempts to send it to a volume set which is associated with the user in question. Files with special requirements are sent to volume sets with special attributes. These volume sets therefore usually contain files belonging to a large number of different users. If the volume set fails then a large number of users are affected. Systems support should therefore consider configuring them to be particularly reliable.

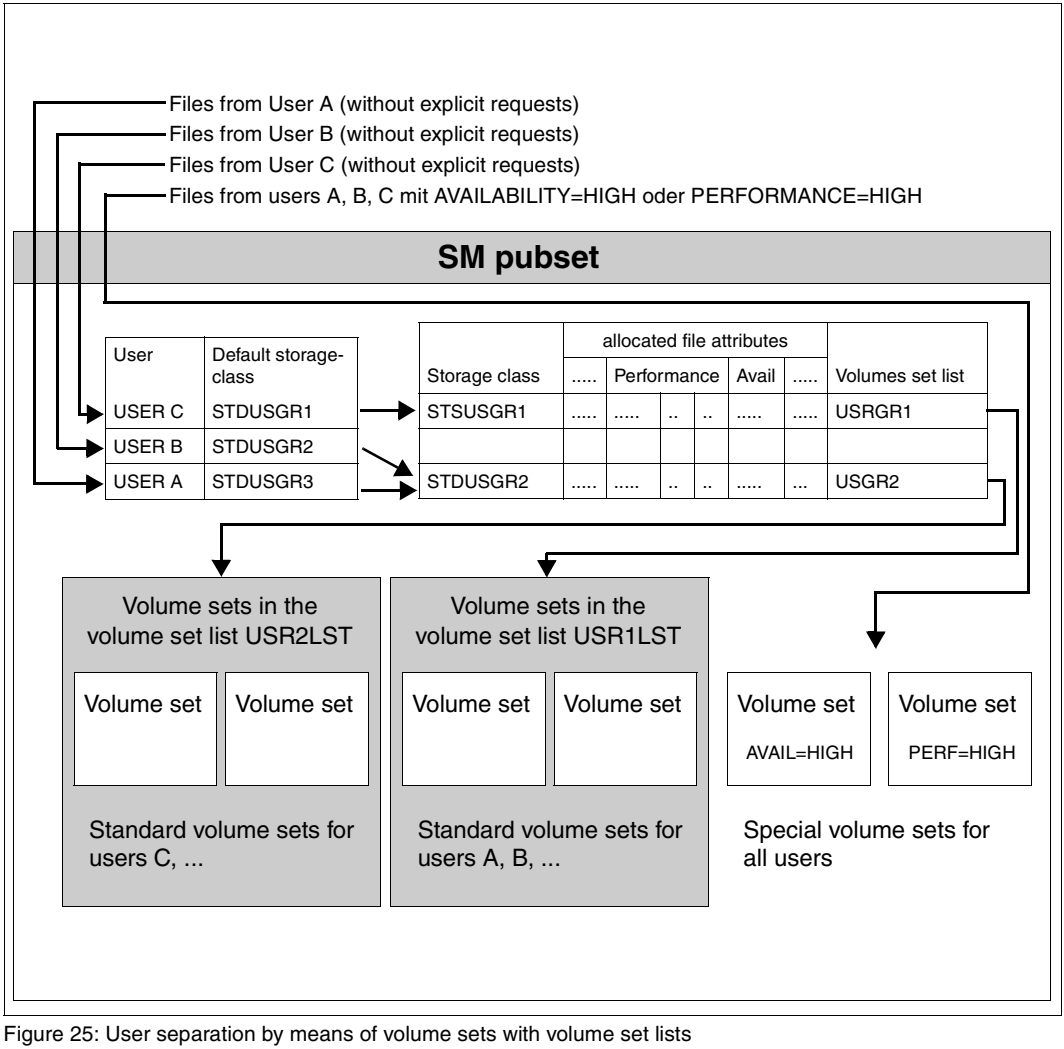


Figure 25: User separation by means of volume sets with volume set lists

9.1.5 Physical allocation

SM pubsets also support physical allocation, in particular. This may seem surprising, since the primary objective of SMS is to provide users with interfaces at the logical level that enable them to describe their requirements and thus conceal from them the details of the physical configuration. However, it can be assumed that in certain situations it will continue to be desirable when specifying the storage locations for files to take into account criteria that cannot be supported adequately by the options available at the logical level. One example of this is the specific distribution of database files to different units of failure. When users are able to specify how the files are arranged themselves, these kinds of aspects can be taken into account as well. It is important that users are not disturbed by parallel system activities while they request space through physical allocation. Although physical allocation is possible on SF pubsets, because of the lack of partitioning between parallel spaces, they are difficult to use. Their use is generally restricted to the systems support staff. Private disks are often used for applications whose files have to be created on particular disks.

Unlike the use of private disks, physical allocation in an SM pubset is characterized by the fact that it does not represent a separate world there; on the contrary, it is fully embedded in the concept of the SM pubset. Except for the specification of the storage location, files created by the system and files created physically by the user are treated in the same way; there are no other functional differences as regards, for example, the use of protection profiles, the use of hardware or file addressing. This uniform handling makes it easy for users to use files that are allocated physically and other files at the same time. Often there is a need for physical allocation only for individual files, whereas for the other files it is preferable for space to be provided by the system. In the case of files that have already been allocated physically, it is easy to unbind these from their storage location, where this appears to be useful.

The following functions make it possible to integrate physical allocation into SM pubset utilization:

- Allocation locks affecting individual volumes or volume sets make it possible to avoid interference between the system and users who wish to make use of physical allocation.
- Using physical allocation it is possible to allocate volume sets to individual applications within the SM pubset. Only the files of the corresponding applications are sent to these volume sets. As far as these applications are concerned, these volume sets represent delimited units of failure, i.e. an application is only affected if disks belonging to its volume set fail (provided that the control volume set of the SM pubset is operational). The failure of any other volumes does not affect the application. This means that the size of an SM pubset can be extended as required without increasing the risk of failure for the application, i.e. the size of a unit of failure and the size of the addressing unit can be defined independently of one another (as with private disks).

- No restrictions apply to files which are created using physical allocation (unlike private disks) in terms of residual pubset functionality (use of GUARDS protection profiles, resource protection through user quotas, hardware restrictions, etc.). Users can also use functions such as file migration to background levels if they so require.
- The boundary between automatic system allocation and physical allocation is not rigid, but can be flexibly adapted. Users who have previously used physical allocation in an SM pubset and who wish to make greater use of SMS functionality in the future benefit from a simple (and smooth) transition path. This can be achieved within the SM pubset without it being necessary to relocate files or modify file addressing.
- Since physical addressing gives users a powerful tool for influencing space allocation within an SM pubset, it can be restricted to certain users.

Level of detail for the specification of storage location

The operands VOLUME-SET, VOLUME and SPACE=*ABSOLUTE (..) in the CREATE-FILE and MODIFY-FILE-ATTRIBUTES commands allow users to specify the location of a file within the pubset at various levels of detail when files are created or extended:

- physical allocation at block level
  The user defines both the volume and the blocks within this volume in which space is to be allocated.
- physical allocation at volume level
  The user defines the pubset volumes in which the file is to be created. However, the location at which space is to be made available within the volume is not relevant to the user.
- physical allocation at volume set level
  The user defines the volume set in which the file is to be stored. However, the location within the volume set is stored is not relevant to the user.

Authorizations for physical allocation

In general, the TSOS privilege authorizes all user IDs to make use of physical allocation. Additionally systems support is allowed to authorize further users to use physical allocation using the ADD-USER and MODIFY-USER-ATTRIBUTES command. This means that users may physically create files on the all the disks of the pubset under the corresponding user ID (user ID in path name). When a user ID is set up, it is initialized with the value ‘not authorized for physical allocation’. All users are allowed to perform physical allocation on private disks as well as on the disks in work volume sets.

Information functions for users of physical allocation

Detailed knowledge of the physical configuration of the SM pubset is generally required for physical allocation. The display of the services offered at the logical level is generally not enough. Information functions are available for this. Some of these functions are only available to systems support staff (TSOS privilege). Users who want to allocate space physically but do not have the TSOS privilege have to get in touch with systems support.

- The SHOW-PUBSET-CONFIGURATION command gives users (even those who do not have TSOS privilege) a view of the physical configuration of the pubset. The information which this provides is grouped into information blocks.
  - The SUMMARY information block provides summary information for the pubset.
  - The PHYSICAL-CONFIGURATION information block lists the volume sets of the SM pubset, the volumes associated with them, together with the device type and allocation restrictions for individual volumes.
  - The VOLUME-SET-PARAMETERS information block displays the usage type (STANDARD, HSMS-CONTROLLED, WORK), allocation and usage restrictions and attribute profiles for volume set selection (performance, availability, structure, allocation unit) for the individual volume sets. The physical configuration which implements the systems support staff’s performance and availability specifications is only partially displayed (such as DUALCOPY, cache allocation, volumes emulated in global storage).
- The SHOW-MASTER-CATALOG-ENTRY command can also be used for SM pubsets and volume sets. It is also available to users.
- Physical allocation at block level requires knowledge of space allocation at block level. The SPCCNTRL utility provides information about this allocation. However, this utility may only be used by systems support (TSOS privilege).

- User programs obtain knowledge of the physical structure of SM pubsets via the STAMCE interface. Pubset-global statuses (accessible, inaccessible, etc.) for SM pubsets are displayed by means of the pubset's MRSCAT entry. Configuration data as regards cache allocations, format, etc. is volume set-specific for SM pubsets and is located in the MRSCAT entries of the individual volume sets. The STAMCE interface provides appropriate selection criteria for determining which volume sets belong to an SM pubset. Corresponding references are contained in the MRSCAT entries of the volume sets.

Coexistence of physical allocation and automatic system allocation

Figure 26 illustrates the coexistence of physical allocation and automatic system allocation on file creation. For users who wish to use physical allocation, it is often important to possess exclusive allocation control for one or more volumes for a given period. The period may vary depending on the application in question. We consider two examples below:

- Short-term usage of volumes for physical allocation

  It may be important to place individual files (e.g. system files) in volumes which are not normally reserved for physical allocation. One example here is the targeted extension of the file catalog. Before allocation is performed, systems support should usually analyze the space which has already been allocated (e.g. using SPCCNTRL) in order to determine the most suitable storage location. If the system performs further allocations at the affected volumes in the interim, this may obviate the results of the analysis and the planned physical allocation will fail.

  In order to avoid this, the MODIFY-PUBSET-RESTRICTIONS command – with the operand RESTRICTION=*ALLOCATION-ON-VOLUME(MODE=*PHYSICAL-ONLY(VOLUME=...)) – to prevent all allocations (including secondary allocation) on individual volumes of an SM subset, with the exception of physical allocations. This can be done before space analysis is carried out and canceled again after the allocation is carried out. Only systems support staff are permitted to do this, because it has effects on pubset usage as a whole (e.g. the monitoring of saturation thresholds). The information functions offered by SPCCNTRL providing information on the occupation of the volumes at block level are also available only to systems support staff. If users without the TSOS privilege want to specify the position of individual files on a volume that is otherwise in general use, they should liaise with systems support staff.

- User-controlled domains

  A different method of using physical allocation, which may occur frequently in connection with private disks, is the long-term reservation of certain volumes for the files of a specific application. To achieve this, only files which are explicitly created in these volumes are located in them. The targeted distribution of files across volumes may, for example, result in the creation of mutually independent units of failure. When volumes are used in this way, it is normally not necessary to specify the precise storage location within the set of volumes reserved for the application and, indeed, the requirement to make a detailed specification may often be undesirable. The files should also be automatically enlarged by the system during processing if necessary (secondary allocation). However, when this is done the space allocation should be performed within the volumes allocated to the application.

  In the case of SM pubsets, systems support can meet such requirements by setting up a separate volume set for an application and reserving it using the MODIFY-PUBSET-RESTRICTIONS command with the allocation restriction RESTRICTION=*NEW-FILE-ALLOCATION (MODE=*PHYSICAL-ONLY). The effect of this restriction is that the only files that can be written to this volume set are files which users have created there by means of physical addressing. No restrictions apply to secondary allocations. If users specify storage locations at volume set level, they implicitly use system routines for space allocation within the volume set. However, if necessary they can also specify the storage location in greater detail at the level of the individual volumes or even at block level. Since users can control space allocation within the volume set, they can prevent their detailed space allocations from being disrupted by parallel system activities. Any impairments caused by other users authorized to perform physical allocations must be prevented through prior consultation. If an application’s files are concentrated in a specific volume set, the application is not affected by the failure of other volumes. If some of an application’s files are to be distributed across mutually independent units of failure, it is necessary to provide multiple volume sets for them.

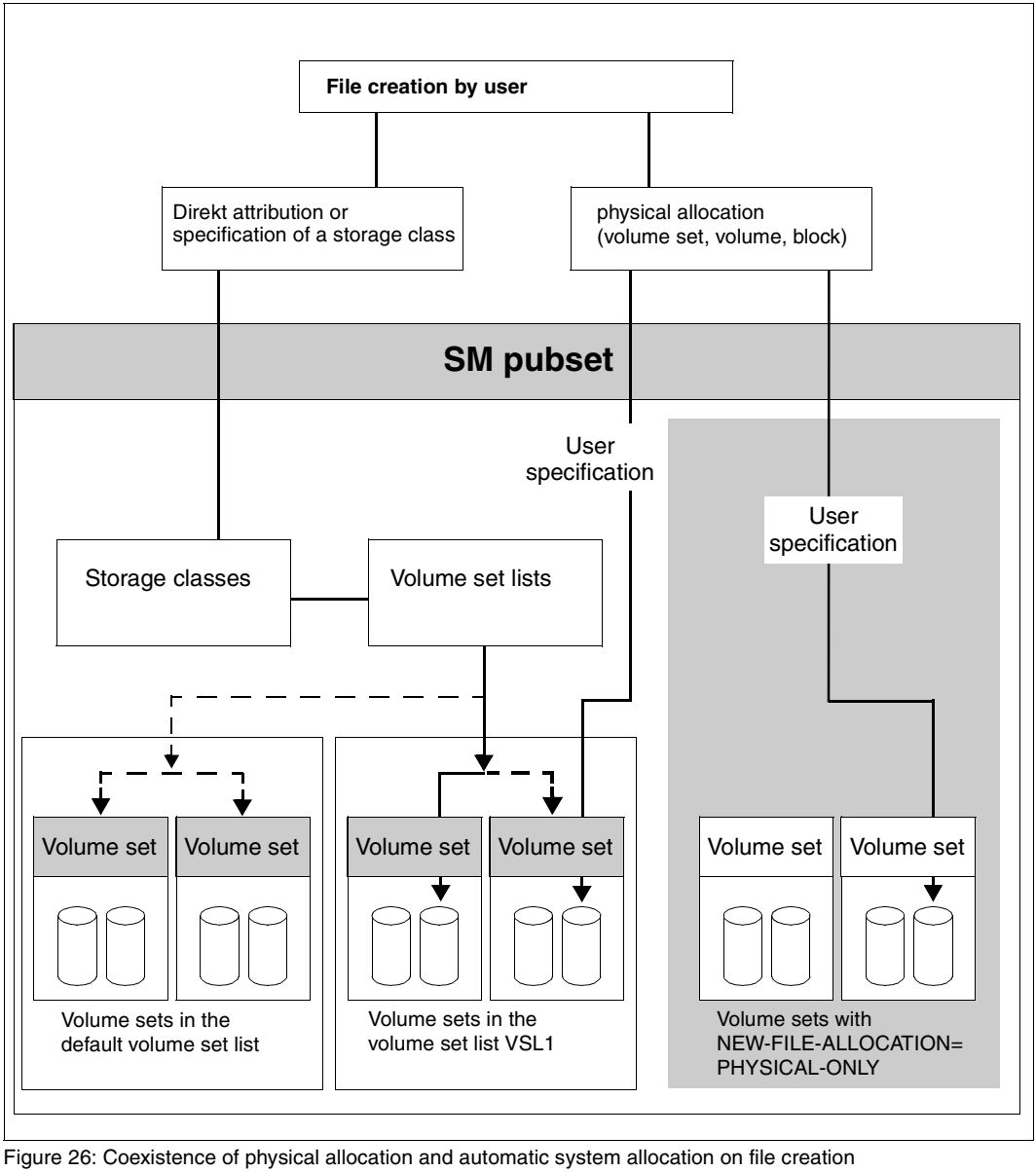


Figure 26: Coexistence of physical allocation and automatic system allocation on file creation

It is also possible to set the allocation lock NEW-FILE-ALLOCATION=*PHYSICAL-ONLY for volume sets which are contained in volume set lists. Since in this case the volume sets are not considered by the system during volume set selection, setting this lock has the same effect as removing this volume set from the volume set list.

Volume sets which are locked against reallocation are therefore not considered when the combination of file attributes which are supported by the volume sets in a volume set list is determined (via the SHOW-PUBSET-FILE-SERVICES command, VOLUME-SET-LIST operand). The same applies when determining the SM pubset service spectrum, which is accessible via direct attribution and which results from the default volume set list. If the reallocation lock is reset, the volume set list allocation is again considered both in the information functions and in the system's allocation behavior.

The role of file attributes during physical allocation

When they perform physical allocation, users themselves determine the volume set in which a file is created. The specification of the required performance, availability,... is therefore irrelevant for volume set selection by the system. However, the file attributes fulfill other functions which are also important for files which are created by means of physical allocation:

- If a volume set is equipped with a cache, the performance-related file attributes control cache utilization.
- The format of a volume set which contains a file does not generally provide an unambiguous identification of the format of the files contained within it (K, NK2, NK4). Further user specifications are necessary before the format can be specified precisely.
- From the file attributes of performance, availability and work files it is possible to determine the user quotas to which the space occupied in the pubset is charged.

For this reason it is necessary to specify the file attributes for performance, availability, work file, even in the case of files which are created using physical allocation. The user specifications are specially validated and assigned default values in order to take account of the special characteristics of physical allocation:

- By placing a file in a volume set which possesses certain attributes, users make certain implicitly defined file requirements explicit. If they do not specify the file attributes explicitly, these are automatically determined by the system which takes account of the characteristics of the specified volume set.
- Users are prevented from escaping the constraints of quota monitoring when performing physical allocation, for example, by allocating the attribute AVAILABILITY=*STD to a file which is created on a high availability volume set. In this case the user specifications are modified if necessary.
- If the attributes specified by the user are not compatible with the attributes of the predefined volume set, the user request is rejected.
- In the case of physical allocation, the only way to specify the file attributes which are relevant for the file storage location is direct attribution. Physical allocation and the use of storage classes are mutually exclusive.

Table 5 illustrates these principles for the file attribute AVAILABILITY.

AVAILABILITY of volume set in which file is created	File attribute AVAILABILITY specified by user	Value resulting from file attribute AVAILABILITY
HIGH	not explicitly specified HIGH STD	HIGH (a) HIGH HIGH (b)
STD	not explicitly specified HIGH STD	STD specification is rejected as it is incompatible with volume set (c) STD

Table 5: Determining the AVAILABILITY file attribute when using physical allocation

Physical allocation and automatic file relocation

Files which have been created using physical allocation require special handling, when they are migrated to background levels for long periods or when they are relocated in the processing level in order to prevent the storage space specification set by the user from becoming invalid. This is possible due to the fact that when files are created using physical allocation, the migration locks S0-MIGRATION and MIGRATE are initialized accordingly. This operation is dependent on the level of detail with which the user specifies the storage location:

- If the user only specifies the volume set, the system assumes that during the processing phase the file should always be located in the specified volume set, but that it may be relocated within the volume set, e.g. as part of reorganization. Although the file may be migrated to background levels during inactive phases, it must be recalled into the original volume set if it is subsequently retrieved into the processing level. This requirements profile corresponds to the settings S0-MIGRATION=*FORBIDDEN and MIGRATE=*ALLOWED. These settings are made automatically.
- If the user specifies the location at volume or block level, the system assumes that the file may not be relocated within the volume set, and that migration to background levels is also purposeless. To ensure that the file would be recalled to its original location on retrieval, it would be necessary to reserve this for the file while it was stored in a background level. Since the space could not be used in any other way, more cost effective storage in background levels would be of no benefit. For these reasons the settings S0-MIGRATION=*FORBIDDEN, MIGRATE=*FORBIDDEN are made when physical allocation is performed at volume or block level.

The migration locks S0-MIGRATION and MIGRATE can also be explicitly set or modified using the CREATE-FILE or MODIFY-FILE-ATTRIBUTES commands if the automatic presettings are not suitable or require subsequent modification. As this gives users the ability to influence pubset space utilization to a similar extent as when using physical allocation, the settings S0-MIGRATION=*FORBIDDEN and MIGRATE=*FORBIDDEN can only be used by users who are authorized to perform physical allocation.

Systems support does not need to implement any special measures in order to take account of migration locks during a migration job. This is automatically performed by HSMS which does not permit the migration of files with MIGRATE-FORBIDDEN to background levels and recalls files which are located in background levels to their original volume set on retrieval to the processing level, providing that this is still present in the pubset.

Transition from physical allocation to the use of SMS functions

Although physical allocations allow users to exert greater control, it restricts the performance and convenience offered by systems support and the system. It may therefore happen that users of physical allocation decide that they no longer wish to perform user-controlled data administration or wish to restrict its use and avail themselves more fully to SMS functions. Transition within an SM pubset can be performed in the following way:

- The command procedures must be adapted in such a way that no further physical allocation is performed and no explicit migration locks are set. When files are created they must be allocated suitable storage services (by allocating suitable file attributes or storage classes), so that the system can create them on suitable volume sets. The migration locks must be reset for existing files.
- If volume sets were previously affected by allocation locks in order to reserve them exclusively for a given application, these locks can be released by systems support. The affected volume sets are then available for general use. Since the migration locks for the files which they contain have been reset, the files are no longer bound to the volume set. They can therefore now be moved to other volume sets of the SM pubset.

9.1.6 Replacing private disks with SM pubsets

Physical allocation is often related to the use of private disks and is used for selective file distribution in order to keep the units of failure that are affected by a disk failure as small as possible. An application whose files are on private disks is only affected by the failure of the disks on which its files are located. Because its files are concentrated on a small number of disks, the risk of it being affected by disk failures is kept low. However, small units of failure do not mean that addressing units are small. Files located on different private disks can be addressed by means of a shared CATID (if they are imported into a particular SF pubset), even when they are on disks that are assigned to different units of failure. The size of the units of failure need bear no relation to the size of the addressing units.

SF pubsets do not provide an equivalent alternative for this. For applications whose files are located on an SF pubset, each disk in the SF pubset is critical in the event of a failure. Consequently, the use of large pubsets increases the risk of failure for all applications that store their files on the pubset. However, if a larger number of small SF pubsets are used instead of a large SF pubset in order to distribute the files to smaller, mutually independent units of failure, different CATIDS have to be used to address the files, which is generally not desirable. A large number of small SF pubsets also requires considerable administration effort, because each SF pubset has to be maintained separately (user IDs, GUARDS protection profiles, etc.).

SM pubsets with the entire pubset as the addressing unit, the individual volume sets as separate units of failure and the physical allocation as the instrument of file distribution meet the requirements of private disk users. Because the restrictions that apply to private disks do not apply to them, they offer an attractive alternative to private disks. However, a transition to SM pubsets does not necessarily mean that private disks have to be replaced. For reasons of compatibility and to ensure that conversion is possible at all times, private disks are supported by SM pubsets in the same way as SF pubsets.

The example of private disk replacement below indicates how an existing private disk configuration can be mapped to a volume set configuration in an SM pubset. Figure 27 provides a graphical illustration of the example.

Starting configuration

In the starting configuration, the private disks are used in conjunction with the private SF pubset X. The files they contain are imported into the SF pubset catalog. They are addressed via path names which include the ID of pubset X as the catalog ID. The files of application A are freely distributed over multiple volumes. The files of application B are arranged in such a way that the failure of one of the two volumes does not affect all the files simultaneously.

Target configuration

In the target configuration, SF pubset X is replaced by SM pubset X. The control volume set of the SM pubset is configured to be particularly reliable. The SM pubset contains a three volume set for application A and two volume set of one volume each for application B. The volume sets for applications A and B have the lock NEW-FILE-ALLOCATION=*PHYSICAL-ONLY. Users must agree between themselves to ensure that the two applications use only their own volume sets and therefore do not interfere with one another. The required volume set is specified whenever a file is created.

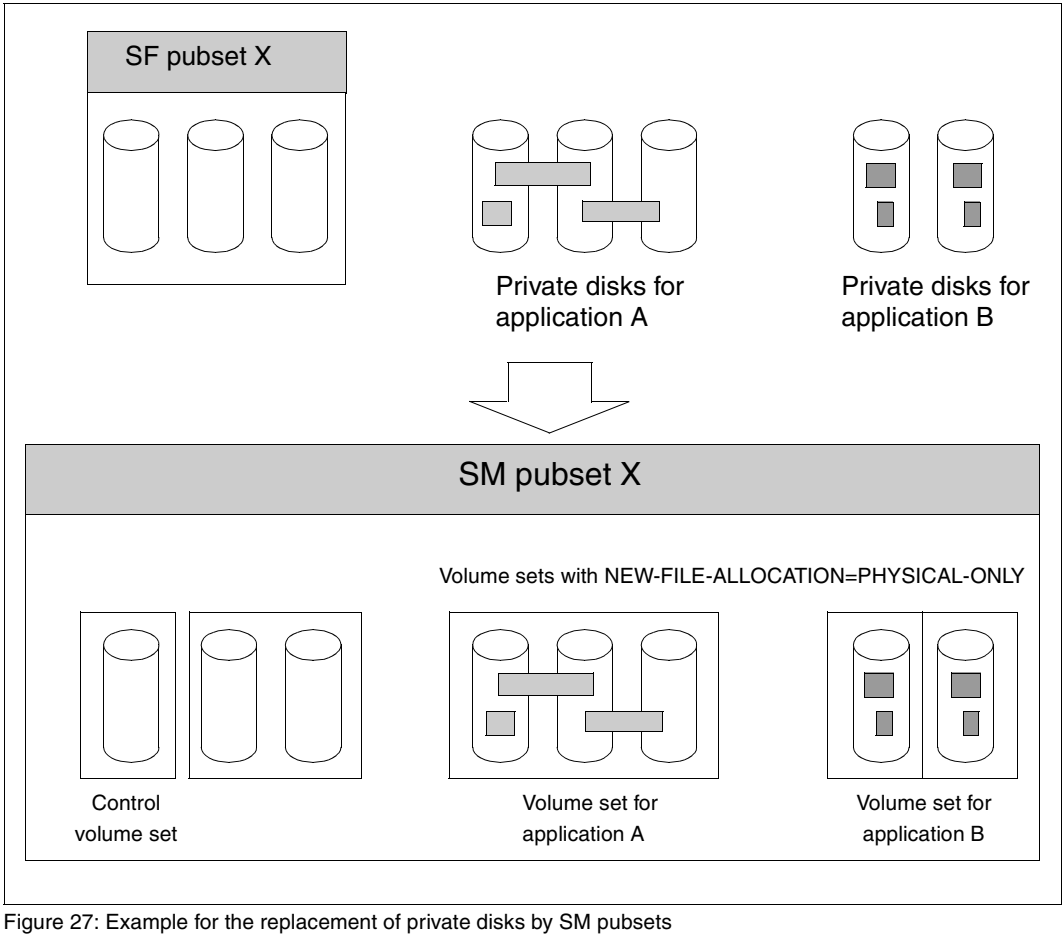


Figure 27: Example for the replacement of private disks by SM pubsets

Steps necessary for the transition

The transition from the starting configuration to the target configuration can be performed using the following steps:

- First the SF pubset is converted into an SM pubset, e.g. using SMPGEN. This does not affect the use of private disks.
- Next the files located on the private disks are backed up.
- SIR is used to create free volume sets on the earlier private disks. These volume sets are incorporated in the SM pubset and they are locked against new allocations. This results in the loss of the files located on the private disk.
- The file backups for the individual applications are read into the associated volume sets (specify the volume set on restore).
- File addressing does not change. However, it is usually necessary to adapt the command procedures. For example, it is necessary to replace statements in which the private disks were specified as the file location by statements containing the corresponding volume sets.

9.1.7 File attributes which are relevant for file location and their default values

The following file attributes are relevant for the selection of a suitable location for a file: the performance attributes (PERFORMANCE, USAGE, DISK-WRITE), availability, format, work file (WORK-FILE) as well as the implicit attribute of file size.

Defining file attributes on file creation (CREATE-FILE)

When a file is created, the user's explicit specifications are first considered in order to determine the file attributes, and allocate a storage class if necessary. If these are incomplete they are completed by the system. In certain cases these explicit specifications may be modified by the system. Overall, SM pubsets permit an extremely differentiated procedure for file attribute definition, which can be influenced by systems support using the default storage classes and the (pubset) default file format in order to achieve user- or pubset-specific goals. This permits the flexible use of the various services provided by an SM pubset without it being necessary to adapt the user's command procedures. The migration from an SF pubset to an SM pubset is also simplified since file attributes can be set to default values, thus ensuring that the command procedures and user programs in SM pubsets benefit from system behavior which is (largely) compatible to that found in SF pubsets. The way in which default values are assigned is also generally of interest to users. They can use the commands SHOW-USER-ATTRIBUTES (default storage class, DMS-TUNING-RESOURCES), SHOW-STORAGE-CLASS (definition of storage class) and SHOW-PUBSET-CONFIGURATION (default file format of pubset) to obtain information about the default allocation parameters which are relevant to them.

File attributes are determined from the interaction of storage classes, default storage classes and the defaults assigned to individual attributes.

If the ability to perform direct attribution has been withdrawn from the user, PERFORMANCE, USAGE, DISK-WRITE, AVAILABILITY, WORK-FILE=*NO and STORAGE-CLASS=*NONE with the CREATE-FILE command are ignored.

The CREATE-FILE command permits the following alternatives for the allocation of file attributes:

*STORAGE-CLASS=*NONE (direct attribution)*

This case corresponds to direct attribution, i.e. the user specifies the file attributes WORK-FILE, PERFORMANCE, USAGE, DISK-WRITE, FILE-PREFORMAT and AVAILABILITY separately. The associated attribute-specific default values are allocated to any attributes which are not explicitly specified. The user specifications for performance may be modified by the system in the light of the user's DMS-TUNING-RESOURCES authorization.
If direct attribution is used, it is also possible to specify the operands VOLUME-SET, VOLUME, DEVICE-TYPE and S0-MIGRATION which are of significance for physical allocation. They also influence the default values assigned to the attributes PERFORMANCE, AVAILABILITY and WORK-FILE and may result in modifications to the user's explicit specifications for these attributes.

STORAGE-CLASS=<composed-name> (explicit specification of a storage class)

The user specifies the file attributes by specifying a storage class which must be set up by systems support. The values stored in the storage class for the attributes WORK-FILE, PERFORMANCE, USAGE, DISK-WRITE, FILE-PREFORMAT and AVAILABILITY are assigned to the file and, once again, the PERFORMANCE value may be reduced in the light of the DMS-TUNING-RESOURCES setting. It is not possible to specify a storage class and simultaneously perform physical allocation, and this is not permitted by the command syntax. The file attributes which are relevant for physical allocation are implicitly assigned the following values: VOLUME-SET=*STD (no volume set specification), VOLUME=*STD (no volume specification), DEVICE-TYPE=*BY-VOLUME (no device type specification), S0-MIGRATION=*ALLOWED.

*STORAGE-CLASS=*STD (no explicit attribute specifications, default value)*

This allocation is performed automatically if the file attributes are not explicitly defined, either by the specification of the individual values or through the specification of a storage class. If the owner of the file has been allocated a storage class set up by systems support as the default storage class then this storage class is allocated to the file. The file attributes are then determined in the same way as when a storage class is explicitly allocated, i.e. the file attributes WORK-FILE, USAGE, DISK-WRITE, FILE-PREFORMAT and AVAILABILITY are taken over unchanged from the default storage class definition, and the performance value stored there may be reduced in the light of the defined DMS-TUNING-RESOURCES value. If no default storage class has been allocated to the owner of the file, the individual file attributes are determined using the associated attribute-specific default procedures. The case in which the storage class entered as the default storage class in a user entry is undefined is identified as an inconsistency by the system and results in the rejection of the CREATE-FILE command.

In the case of files which are cataloged in an SM pubset but which occupy no space either in the processing level or a background level, the storage class allocations and the attributes which are relevant for the storage location are undefined with the exception of the performance attributes. They are not determined until space is first requested for the file which previously only existed as a catalog (e.g. using the MODIFY-FILE-ATTRIBUTES command). The same procedure is used as when creating a file using CREATE-FILE.

Modifying file attributes (MODIFY-FILE-ATTRIBUTES)

The MODIFY-FILE-ATTRIBUTES command is used to modify file attributes and storage class allocations. Like the CREATE-FILE command, a number of alternatives are also possible for MODIFY-FILE-ATTRIBUTES. These can be selected using the STORAGE-CLASS operands. If the ability to perform direct attribution has been withdrawn from the user, PERFORMANCE, USAGE, DISK-WRITE, AVAILABILITY, WORK-FILE=*NO and STORAGE-CLASS=*NONE and S0-MIGRATION=*ALLOWED with the MODIFY-FILE-ATTRIBUTES command are ignored.

*STORAGE-CLASS=*UNCHANGED (default value)*

With the exception of the file size, it is not possible to specify any file attributes which are relevant for the storage location, i.e. they retain their existing values. Any existing storage class allocation also remains unchanged.

*STORAGE-CLASS=*NONE (direct attribution)*

Any existing storage class allocation for the file is canceled. Any file attributes explicitly specified by users are modified correspondingly and, in certain cases, the performance and availability specifications are automatically modified (depending on the user's DMS-TUNING-RESOURCES authorization or in the case of files which are bound to a specific volume set by the attribute S0-MIGRATION=*FORBIDDEN). An exception here is the modification of the attribute S0-MIGRATION=*ALLOWED to S0-MIGRATION= *FORBIDDEN. This may result in an implicit adaptation of the performance and availability attributes.

STORAGE-CLASS=composed-name (explicit specification of a storage class)

This specified storage class is assigned to the file and the file attributes are redetermined on the basis of the storage class definition. The procedure used is the same as when the CREATE-FILE command is used with an explicit specification of a storage class.

*STORAGE-CLASS=*STD (explicit allocation of the default storage class)*

This specification only results in a modification to file attributes if a default storage class is allocated to the file's owner and this has not already been allocated to the file as a storage class. In this case the owner' s default storage class is allocated as the storage class for the file and the file attributes are redetermined on the basis of the storage class definition in the same way as in CREATE-FILE with STORAGE-CLASS=*STD.

*STORAGE-CLASS=*UPDATE (update of file attributes)*

The storage class allocation does not change but the file attributes are redetermined on the basis of the current storage class definition. This makes it possible to eliminate discrepancies between the storage class allocated to a file and the file attributes. Such discrepancies may arise if the storage class definition has been modified since file creation. In such cases, the attributes of the files which belong to the storage class are not automatically adapted.

File attributes at the FILE interface

The FILE program interface[1)] also supports direct attribution, explicit storage class specifications and the use of default storage classes as alternative ways of allocating file attributes. The distinction between direct attribution and the allocation of the default storage class is somewhat different from in the CREATE-FILE command. If the ability to perform direct attribution has been withdrawn from the user and if no explicit specifications are made for the STOCLAS operand on file creation and no file attributes which are relevant for file location are specified except for BLKCTRL and BLKSIZE, the system checks whether a default storage class has been allocated to the file's owner. If this is the case, the file attributes performance, availability and work file are set in accordance with the default storage class definition (as in STORAGE-CLASS=*STD). However, the user's BLKSIZE and BLKCTRL specifications are taken into account when defining the preformat. If a volume set list is allocated to the default storage class, this is considered when the storage location is selected. This special treatment of the format specifications by the FILE program interface makes it possible to influence the allocation of file attributes and, consequently, file distribution within the SM pubset in the case of files which are created by existing programs with explicit BLKSIZE or BLKCTRL specifications without it being necessary to modify the programs.

We consider the individual file attributes in greater detail below.

[1)] The ISP command FILE behaves - especially with regard to the BLKCTRL and BLKSIZE specifications - in the same way as the FILE program interface. Exception: The options for attribution are restricted when a file is created using the FILE command, e.g. the explicit specification of a storage class is not possible.

Performance attributes

The PERFORMANCE, USAGE, DISK-WRITE file attributes can be used to specify the required performance for files.

Performance attributes	Values
PERFORMANCE	STD, HIGH, VERY-HIGH
USAGE (operations for which enhanced performance is requested)	READ read operations only WRITE write operations only READ-WRITE read and write operations
DISK-WRITE (enhanced performance even at the risk of reduced write reliability)	IMMEDIATE not required BY-CLOSE required

Table 6: Significance of performance attributes

In SM pubsets, the static values of the performance attributes allocated using the CREATE-FILE, MODIFY-FILE-ATTRIBUTES commands cause the system to search for a suitable volume set when the file is created. This need not always be a volume set with a cache since it is also possible to achieve enhanced performance in other ways, e.g. by using fast disks. If a file with enhanced performance requirements is created in a volume set with a cache then the system determines the extent to which the cache will be used when the file is opened. When this is done, the user (ADD-FILE-LINK command or FCB interface) or systems support (DMS-TUNING-RESOURCES) may dynamically reduce the performance attribute values which determine cache utilization from the specified static values. This dynamic modification of the performance values has no effect on the location of the file. It therefore has no effect on files which are located in volume sets without a cache since their performance characteristics are already fully determined by their location.

Users are not guaranteed that their performance specifications will be met. A volume set which does not provide the required performance or only provides it to a reduced extent is considered to be compatible with the user's requirements and may be used as the storage location of the file if no other more suitable volume set can be found. The requested write-reliability is always guaranteed although this may mean that enhanced performance cannot be provided.

Systems support can use the DMS-TUNING-RESOURCES right to restrict the maximum performance value permitted for the files of a given user. Here systems support can set NONE, CONCURRENT-USE or EXCLUSIVE-USE. These settings are correlated with the maximum performance value as set out in table 7: if in the CREATE-FILE or MODIFY-FILE-ATTRIBUTES commands, users allocate higher to their files which are higher than those for which they are actually authorized, the request is not rejected but is reduced to the maximum permitted value.

DMS-TUNING-RESOURCES values	Permitted performance attributes
NONE	STD
CONCURRENT-USE	STD, HIGH,
EXCLUSIVE-USE	STD, HIGH, VERY-HIGH

Table 7: Relation between DMS-TUNING-RESOURCES and performance attributes

In order to prevent the effect of user quotas being circumvented when physical allocation is used - e.g. by a user physically binding a file with the attribute PERFORMANCE=*STD to a volume set which consists of high-performance volumes - the performance values explicitly specified by the user may be increased. In the case of physical allocation, the attributes of the specified volume set are also considered when determining the default values.
This is illustrated in table 8. Correspondingly, in the case of files which have already been physically bound to a given volume set (S0-MIGRATION=*FORBIDDEN), reductions in performance values are monitored by the system and may be ignored if appropriate. If a file is explicitly bound to the volume set in which it is currently located through the setting of the migration lock S0-MIGRATION=*FORBIDDEN, the file's previous performance value may be automatically increased.

PERFORMANCE spectrum of the volume set	User-specified value for the PERFORMANCE file attribute	Resulting value for the PERFORMANCE file attribute
STD [,HIGH] [,VERY-HIGH]	not explicitly specified STD HIGH VERY-HIGH	STD STD HIGH VERY-HIGH
HIGH [,VERY-HIGH]	not explicit STD HIGH VERY-HIGH	HIGH HIGH HIGH VERY-HIGH
VERY-HIGH	not explicit STD HIGH VERY-HIGH	VERY-HIGH VERY-HIGH VERY-HIGH VERY-HIGH

Table 8: Determining the performance attribute in the case of physical allocation

The automatic increase of the performance attribute value in the case of physical allocation is not dependent on the file owner's DMS-TUNING-RESOURCES authorization. Since in the case of SM pubsets the DMS-TUNING-RESOURCES authorization is primarily suitable for cache resource protection (see chapter 10), this does not cause any problems concerning resource protection. Volume sets with caches generally possess a performance spectrum which also covers the value STD. In such cases no automatic increase is performed (see table 8).

Availability

The file attribute AVAILABILITY allows to specify the required availability.
A file with AVAILABILITY=*HIGH can only be created in a volume set to which systems support has allocated the availability profile AVAILABILITY=*HIGH. The combination of the file attribute AVAILABILITY=*HIGH and the volume set attribute AVAILABILITY=*STD is considered to be incompatible for volume set selection. Systems support must ensure that a volume set with the attribute AVAILABILITY=*HIGH actually provides high availability.

The file attribute AVAILABILTY=*HIGH is not permitted for work files and temporary files. Attribute allocations and conversion of permanent files into temporary files which infringe this condition are rejected. A special procedure is necessary when a user is allocated a default storage class for which AVAILABILITY=*HIGH is set. The creation of a temporary file for which the attributes are not explicitly specified (STORAGE-CLASS=*STD) is permitted and, with the exception of AVAILABILITY, the file attributes are set in accordance with the default storage class definition. AVAILABILITY=*STD is implicitly allocated to the availability attribute. The reason for this special processing of default storage classes is that behavior compatible with that in SF pubsets must be guaranteed for temporary files in SM pubsets, even if default storage classes with AVAILABILITY=*HIGH are used.

As is the case with the performance attributes, when files are created using physical allocation, the default value for availability is determined on the basis of the characteristics of the volume set in question. Explicit user specifications may be modified (see table 9). The same applies to modifications of the availability attribute for existing files which have already been physically bound to a specific volume set (S0-MIGRATION=*FORBIDDEN) and the setting of the migration lock S0-MIGRATION=*FORBIDDEN.

AVAILABILITY of volume set in which file is created	Value specified by the user for the AVAILABILITY file attribute	Resulting value for the AVAILABILITY file attribute
HIGH	not explicitly specified	HIGH
	HIGH	HIGH
	STD	HIGH
STD	not explicit	STD
	HIGH	specification incompatible
	STD	STD

Table 9: Determining the AVAILABILITY attribute in the case of physical allocation

Work files

The paradigm for the use of work files (i.e. files with the attribute WORK-FILE=*YES) specifies that these can only be located in work volume sets (i.e. volume sets with the usage type WORK) which systems support makes available to users for limited periods only. This period must be agreed between systems support staff and the users. When this period expires, systems support has the right to remove the work volume set and delete all the files located in it. This can be achieved through the forced removal of the volume set (CONDITION=*VOLUME-SET-DEECTS operand in MODIFY-PUBSET-PROCESSING command). This operation implicitly deletes all the files located in the volume set. If, at this time, files located in the volume set are being processed by user tasks, systems support can force their termination using the option TERMINATE-JOB=*YES. Since work volume sets only contain work files, this only affects users which have not kept to the agreement made with systems support. Conversely, since work files are always created in work volume sets and cannot be migrated to background levels, the entire capacity occupied by work files can be regained by removing the work volume sets from the users.

When a file is created using CREATE-FILE, the specifications *STD, *YES and *NO are possible for the WORK-FILE attribute. In the case of non-physical allocation, the *STD specification is equivalent to *NO. When physical allocation is performed, the WORK-FILE attribute is determined from the user specification and the volume set usage type as shown in table 10. Once the WORK-FILE attribute has been determined for a file it can no longer be modified using the MODIFY-FILE-ATTRIBUTES command.

Usage type of volume set in which the file is created	User-specified value for the WORK-FILE file attribute	Resulting value for the WORK-FILE file attribute
not WORK	STD NO YES	NO NO user specification incompatible
WORK	STD NO YES	YES user specification incompatible YES

Table 10: Determining the WORK-FILE attribute in the case of physical allocation

Work files and work volume sets provide an alternative to the widespread use of private disks as work disks. The user must specify the VSN when creating a file on a private disk. In order to make it possible to use existing command procedures in SM pubsets, it is sufficient to replace the VSN of the private disk with the VSN of a volume in the work volume set. The appropriate WORK-FILE attribute is automatically determined. In order to guarantee behavior which is equivalent to private work disk use, the right to perform physical allocation is not required in the case of physical allocations in work volume sets.

File format

The significance of the file format (K, NK2, NK4) for file processing is independent of whether the file is located in an SF pubset or SM pubset. However, there are differences in the ways in which the file comes to have its format. In the case of SM pubsets, it is important that users can already prepare information relating to the file format when space is requested for the file for the first time. This information can then be taken into account during the selection of the most suitable volume set.

For this reason the file attribute ‘temporary file format’ (preformat) is provided for files in SM pubsets. It is allocated to a file when space is provided for the file for the first time (see CREATE-FILE, MODIFY-FILE-ATTRIBUTES commands and FILE program interface). This usually occurs when the file is first created. However, a space request using the MODIFY-FILE-ATTRIBUTES command for a file which has previously only been cataloged is also considered to be an initial space provision. In the case of files which already occupy space, it is not possible to modify the allocated preformat and the attempt is rejected. The preformat is undefined for files which occupy no space, i.e. which are only cataloged. The preformat specifications are irrelevant for files in SF pubsets and they are ignored. If an SM pubset is created through the conversion of SF pubsets using SMPGEN, the files it contains have the special preformat value NONE. This has the same significance as if the format of the volume set containing the file had been assigned as the file’ s preformat.

If space is provided using the CREATE-FILE or MODIFY-FILE-ATTRIBUTES commands, the following alternatives are possible:

- The user specifies the (complete) preformat by specifying the direct attribute FILE-PREFORMAT or by explicitly allocating a storage class.
- The system determines a default value for the preformat. To do this, it considers the default file format of the pubset, the preformat stored in the user's default storage class and, in the case of physical allocation, the format of the volume set to which the file is sent.

The procedure for determining the preformat is illustrated in table 11. The space requested by the user is only provided via the CREATE-FILE and MODIFY-FILE-ATTRIBUTES commands if the preformat is possible in combination with the other file attributes (e.g. availability) in the pubset, i.e. in this case, the preformat specification is binding and must be fulfilled.

Different behavior to that for CREATE-FILE has been specified for the FILE program interface[1)] and the FILE command. This has been done so that files designed originally for SF pubsets can be used without changes (or with as few changes as possible) on SM pubsets. The specifications for file format (BLKCTRL) and block size (BLKSIZE) are used to determine the preformat. For files on SM pubsets it is good if the desired format is specified in full when the file is created. For file processing on SF pubsets, it is also possible to use the options described below for format specification, although they are not ideal for SM pubsets and are only permitted for compatibility reasons:

- The users are not obliged to specify anything for file format (BLKCTRL) and block size (BLKSIZE) when a file is created. In this way, they avoid doing anything that would definitively specify the format required subsequently at processing.
- Even if it is clear that specifications cannot be fulfilled at the corresponding pubset, they are not rejected. The format specifications are not validated until the file is opened.
- Alongside user specifications which unambiguously describe the format and user specifications in which the format is completely open, it is also possible to make specifications which only partly describe the format. Such specifications must then be appropriately completed.

[1)] Or the ISP command FILE

User specifications that are unfavorable for SM pubsets must be interpreted as well as possible while preserving compatibility as far as possible with the behavior for SF pubsets as regards the preformat and thus volume set selection. table 12 on page 217 shows for the FILE program interface[1)] how the preformat is derived from the specifications for file format (BLKCTRL) and block size (BLKSIZE).

If it emerges with a FILE program call that the preformat obtained on the basis of the user specifications cannot be fulfilled, the file is assigned a different preformat (in contrast to the CREATE-FILE command) that can be fulfilled on the pubset and is as similar as possible to the original one. This is the case, for example, if there is no suitable volume set in the SM pubset or if the volume set specified by the users through physical allocation does not support the preformat.

If preformat K is originally determined but is not possible, then NK2 is preferred and if NK2 is also not possible then NK4 is chosen. If NK2 is originally determined then NK4 is the only possible alternative.

Users who wish to obtain information about the parameters relevant for the default file format settings may use the commands SHOW-PUBSET-CONFIGURATION (default file format of pubset), operand SHOW-USER-ATTRIBUTES INFORMATION=*PUBSET-ATTRIBUTES (default storage class) and SHOW-STORAGE-CLASS (preformat in the default storage class).

[1)] The procedure is analogous for the ISP command FILE, although the command only offers some of the functionality of the FILE-program interface (e.g. no specifications of storage classes).

Conditions			**Resulting value for the preformat file attribute**
User specification for STORAGE-CLASS in CREATE-FILE:			
*NONE Direct attribution	**Specification for FILE-PREFORMAT in CREATE-FILE:**		
	K		K
	NK2		NK2
	NK4		NK4
	BY-PUBSET-DEFAULT (default value)	**VOLUME- or VOLUME-SET specification in CREATE-FILE:**	
		yes	Preformat as format of volume sets
		no	Preformat as default file format of pubset
<composed-name> specification of a storage class	**Preformat in storage class:**		
	K		K
	NK2		NK2
	NK4		NK4
	BY-PUBSET-DEFAULT		Preformat as default file format of pubset
*STD (default) no explicit file attribute specification	**Is a default storage class allocated to the user:**		
	yes	**Preformat in the default storage class:**	
		K	K
		NK2	NK2
		NK4	NK4
		BY-PUBSET-DEFAULT	Preformat as default file format of pubset
	no		Preformat as default file format of pubset

Table 11: Determining the preformat when space is first requested for a file using CREATE-FILE or MODIFY-FILE-ATTRIBUTES

Conditions							**Resulting preformat**
BLKCTRL:							
PAMKEY							K
DATA2K							NK2
DATA4K							NK4
DATA/NO	**BLKSIZE:**						
	(STD,n) n odd						NK2
	(STD,n) n even						NK4
	else	**Physical allocation:**					
		no	**Storage class specified:**				
			no	**Default storage class allocated**			
				no	**Pubset default file format**		
					K or NK2		NK2
					NK4		NK4
				yes	**Preformat in default storage class:**		
					K or NK2		NK2
					NK4		NK4
					BY-PUBSET-DEFAULT	**Pubset default file format:**	
						K or NK2	NK2
						NK4	NK4
			yes	**Preformat in the storage class:**			
				K or NK2			NK2
				NK4			NK4
				BY-PUBSET-DEFAULT		**Pubset default file format:**	
						K or NK2	NK2
						NK4	NK4
		yes	**Format of volume set in which the file is stored:**				
			K or NK2				NK2
			NK4				NK4

Table 12: Determining the preformat when space is first requested for a file either via the FILE program interface

Conditions					**Resulting preformat**
not specified	**Physical allocation:**				
	no	**Storage class specified:**			
		no	**Default storage class allocated**		
			no		Pubset default
			yes	**Preformat in default storage class:**	
				K	K
				NK2	NK2
				NK4	NK4
				BY-PUBSET-DEFAULT	Pubset default
		yes	**Preformat in storage class:**		
			K		K
			NK2		NK2
			NK4		NK4
			BY-PUBSET-DEFAULT		Pubset default
	yes				Volume set format

Table 12: Determining the preformat when space is first requested for a file either via the FILE program interface

The term preformat makes it clear that this has only a provisional significance for the file. The format is not finally determined (in SM pubsets as previously in SF pubsets) until the file is opened for creation. In the case of the procedures which already exist for SF pubsets as part of OPEN processing, the most important input values are the format of the SF pubset in which the file is located and the current values for file format (BLOCK-CONTROL-INFO) or block size (BUFFER-LENGTH or BLKSIZE) and access method (ACCESS-METHOD or FCBTYPE) stored by the user in the task file table (supplied via ADD-FILE-LINK, FILE program interface[1)]) or in the file control block (FCB) at the time OPEN is called. In the special case in which users choose not to make an explicit format specification on file creation, the (pubset) format of the SF pubset containing the file is taken over as the file format (K, NK2, NK4). This procedure is modified for SM pubsets which have no pubset format. Here, the role of the format of the SF pubset is largely taken over by the file preformat (see figure 28).

[1)] Or ISP command FILE

Here it is necessary to distinguish between the following two starting situations:

– A file is defined for the first time, i.e. it possesses no CREATION-DATE. Here the role of the SF pubset format is played in the SM pubset by the file preformat which has previously been allocated to the file (i.e. when space is provided for it for the first time). If the user chooses not to make explicit specifications for the file format (BLOCK-CONTROL-INFO or BLKCTRL) and block size (BUFFER-LENGTH or BLKSIZE) when opening the file, the preformat is taken over as the file format.

– The contents of an existing file are recreated and the space which the file has occupied in the past continues to be allocated to it. In this case it is not the preformat, but the previous format which is considered in the procedure used to determine the format. If the user chooses not to make explicit specifications for the file format (BLOCK-CONTROL-INFO or BLKCTRL) and block size (BUFFER-LENGTH or BLKSIZE) when opening the file, the previous file format is retained (K, NK2, NK4). In such cases the system guarantees that the format (of the file which is to be created) is compatible with the file's previous storage location and that the space previously assigned to the file can continue to be used.

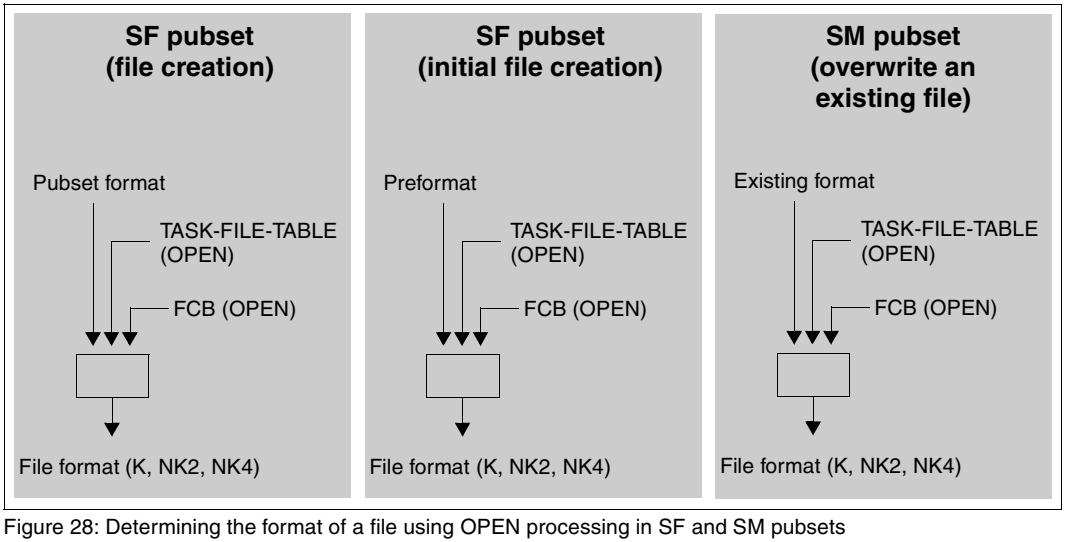


Figure 28: Determining the format of a file using OPEN processing in SF and SM pubsets

If file format and block size are explicitly specified at the time of OPEN processing, it may happen that the volume set on which a file has so far been allocated space in an SM pubset is not compatible with the required file format. In such cases, OPEN processing implicitly relocates the file to a suitable volume set, provided that such a volume set exists and that relocation is permitted (S0-MIGRATION). In order to keep system load low and exclude the risks which may be associated with relocation (space bottlenecks etc.), users are very strongly urged to specify the format required for subsequent processing as the preformat for files in SM pubsets at the time when space is first made available.

In the special case where a user makes no explicit format specifications, either when creating a catalog entry for a file or when filling a file with contents, but instead allows the format to be fully determined by the system, the system behaves as follows:

- For files which are not created using physical allocation, the format is usually determined directly from the pubset's default file format.
- The only exception to the above occurs if the file's owner is allocated a storage class which contains an explicit value for the preformat (i.e. not the value BY-PUBSET-DEFAULT) and the default storage class is used when the file is created (i.e. no direct attribution of other file attributes such as performance). In this case the value in the default storage class determines the file format.
- In the case of files which are explicitly created in a given volume set by means of physical allocation, the format is determined directly from the format of the volume set.

User programs which process SF pubset files use the pubset format in different ways. One approach is to determine the pubset format via the STAMCE program interface in order to permit an initial identification of the file formats which are possible in the pubset. Appropriate file format and block size specifications are then used to select the required format from these possible formats. In certain applications it is possible that the required format is only partially specified and that full specification is left to the system's default allocation procedure which obeys certain rules for the allocation of default values. This approach is not optimal for SM pubsets since these may contain multiple volume sets with differing formats.

In order to ensure that existing user programs continue to be executable in SM pubsets without modification, an SM pubset appears as an SF pubset with a format identical to the default file format of the SM pubset at the STAMCE program interface with the layout for BS2000 versions < BS2000/OSD-BC V3.0.

Provided that files are not created using physical allocation in volume sets whose format differs from the default file format of the pubset and no default storage classes are used whose preformat differs from the default file format of the pubset, an SM pubset behaves almost identically to a corresponding SF pubset for the purposes of determining the format of a file. One difference, for example, is that it is not generally possible to create K files in an SF pubset with the format NK2 whereas this is possible in an SM pubset with the default file format NK2, provided that this contains a K volume set. In the special case that all the volume sets in an SM pubset have the same format then full compatibility with SF pubsets is guaranteed in terms of format definition.

The above-mentioned differences relating to the use of default storage classes and physical allocation rather than the existing procedures for the assignment of default values should not usually cause any problems affecting the executability of existing user programs since these should not generally require the detailed implementation of the existing procedures for default value assignment.

One example application for the pubset default file format in conjunction with default storage classes is the conversion of file processing from K to NK2. The conversion phase is characterized by the fact that certain programs require the same behavior as in SF pubsets with the format K when default values are assigned to file attributes. To make this possible, the value K is selected for the pubset default file format, and users who run such programs are either allocated no default storage class or a default storage class with PREFORMAT= BY-PUBSET-DEFAULT. For others users it is possible to allocate a default storage class with the preformat NK2.

S0-MIGRATION file attribute

The file attribute S0-MIGRATION=*FORBIDDEN physically binds a file to a specific volume set. However, it does not prevent the file from being migrated to a background level. When the file is retrieved into the processing level, it is moved to its original volume set. However, its location in the volume set is newly determined. S0-MIGRATION=*FORBIDDEN can only be allocated by users who have the right to perform physical allocation. When a file is created, the value of the S0-MIGRATION attribute depends whether or not the file has been physically created (see table 13).

Conditions	Resulting value for the file attribute S0-MIGRATION
Physical allocation	
Yes	FORBIDDEN
No	ALLOWED

Table 13: Determining the S0-MIGRATION attribute on file creation

The S0-MIGRATION value can be explicitly modified for an existing file using the MODIFY-FILE-ATTRIBUTES command. However, the allocation of additional space to a file that has already been created, does not result in any implicit modification of the S0-MIGRATION attribute, even if the file is created using physical allocation.

MIGRATE file attribute

The MIGRATE attribute can be used with the values *STD, *INHIBITED (soft lock against background migration by HSMS) and *FORBIDDEN (hard migration lock). Unlike the soft lock, the hard lock cannot be deactivated by systems support. When a file is created, the value assigned to the MIGRATE attribute depends on whether or not the file is physically created (see table 14). If, in the case of physical allocation, the storage location is specified at volume or block level, it is assumed that the file is to retain the location assigned to it in the processing level permanently. If such a file were to be migrated to a background level, it would not be possible to guarantee that it would be returned to its original location on a subsequent recall. The implicit assignment MIGRATE=*FORBIDDEN prevents this.

Conditions			**Resulting value for the MIGRATE file attribute**
MIGRATE specification in CREATE-FILE			
STD	**Physical allocation at volume or block level:**		
	yes		FORBIDDEN
	no	**File type:**	
		Permanent file	ALLOWED
		Temporary file	INHIBITED
*ALLOWED	**File type:**		
	Permanent file		ALLOWED
	Temporary file		Illegal specification
*INHIBITED	**File type:**		
	Permanent file		INHIBITED
	Temporary file		INHIBITED
*FORBIDDEN	**File type:**		
	Permanent file		FORBIDDEN
	Temporary file		INHIBITED

Table 14: Determining the MIGRATE attribute on file creation

To free the file from this fixed location specification, users can modify the MIGRATE attribute as required using the MODIFY-FILE-ATTRIBUTES commands. MIGRATE=*FORBIDDEN can only be allocated by users who have the right to perform physical allocation, irrespective of whether the allocation is performed explicitly or implicitly (by means of physical allocation).

Work files, temporary files and files with high availability requirements cannot generally be moved to background levels. HSMS ensures this independently of the MIGRATE allocations. When temporary files are created or when permanent files are converted into temporary files, these files also receive the allocation MIGRATE=*INHIBITED unless a different value is specified by the user.

File size

The file size, which is of significance for the selection of a volume set with a suitable allocation unit, is calculated automatically from the associated user specifications when the file is created or enlarged.

9.2 Use of background levels for file migration

The background levels are suitable as cost-effective storage locations for files which users do not expect to access for an extended period and for which they must expect longer access times on the next access. The background levels can also be used as temporary storage for files which do not meet the conditions for long-term migratability, when the S0 level is reorganized or when acute storage problems occur in the S0 level. These files should be automatically returned to the processing level when reorganization has been completed or the storage bottlenecks have been eliminated (from the user's point of view). When relocating files to background levels, it should be noted that certain files, such as system files, cannot be migrated in this way.

9.2.1 User/systems support staff interface

Users know which of their files are suitable for long-term migration to background levels. In an SMS-administered data center, it is the responsibility of systems support to organize and perform migration since this requires the provision of centralized resources (such as tape devices) and migration is particularly efficient if it is performed for multiple users. This means that there must be an easy-to-use interface between the user and systems support in order to enable users to communicate their needs and allow systems support to meet differing user requirements with the minimum of administrative effort. Special functions are provided for this through the use of HSMS management classes.

Criteria relevant for migration to background levels

If a file is not accessed for an extended period, it is also then often not processed for an extended period. Therefore the period which has elapsed since the file was last accessed is a suitable criterion for the automatic determination of the eligibility of the file for migration. However, different values may be suitable for individual files. Periods of differing lengths may also prove to be desirable for one and the same file during its lifetime. Since users are the people best informed about the time frame within which they require their files they should be able to influence the period after which a file is considered to be eligible for migration on a file-specific basis.

If both an S1 level and S2 level are present for an SM pubset, it is necessary to decide to which of these the file will be migrated. The S1 level differs from the S2 level in that the files it contains can be recalled into the processing level more quickly, thus permitting shorter access times the next time the file is accessed. In contrast, storage in the S2 level is less costly. In the case of files which are initially transferred to the S1 level, it may be required that they are migrated from there to the S2 level after a specified period of time. Users should be able to express their file-specific requirements concerning the selection of background level and the relocation of files from the S1 level to the S2 level.

The file size and the number of file extents are further dimensions which should be taken into consideration when determining the file-specific eligibility of files for migration.

HSMS management classes

The HSMS management classes support the interaction of systems support staff, users and the system in the use of background levels. This is illustrated in figure 29.

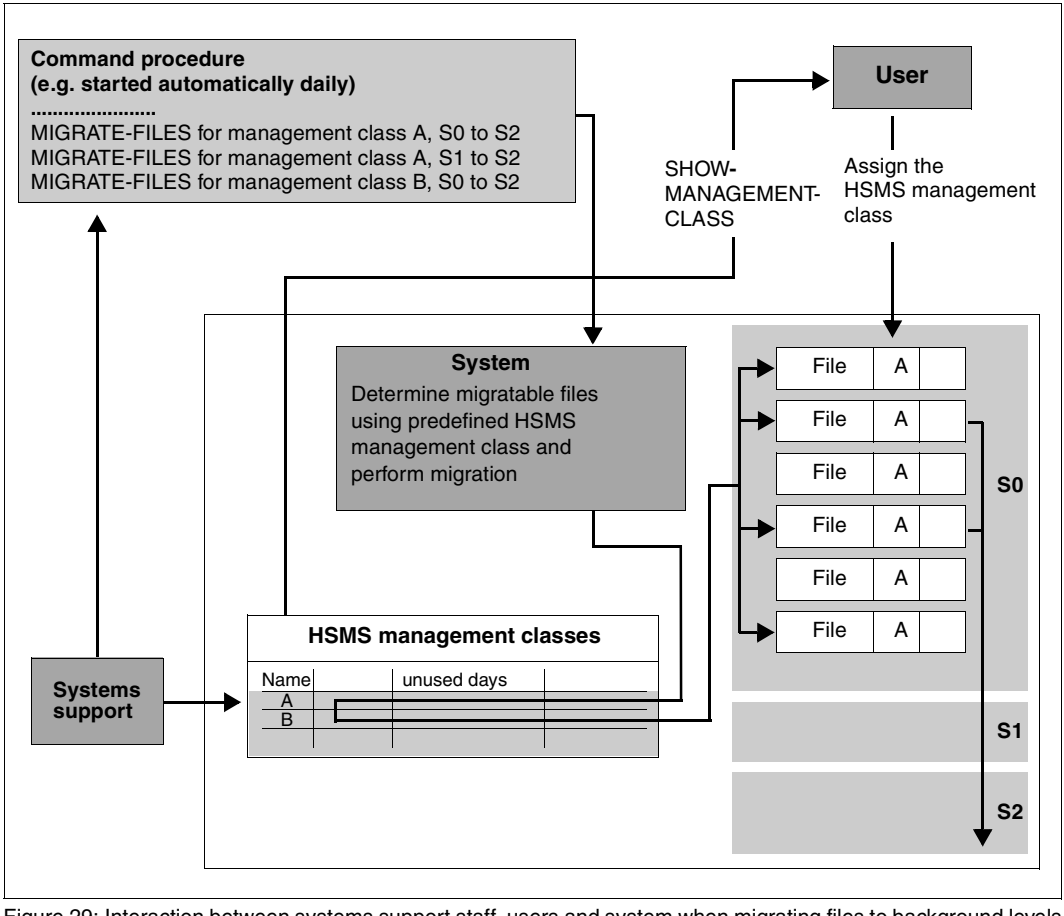


Figure 29: Interaction between systems support staff, users and system when migrating files to background levels

The HSMS management classes are set up by systems support (see the HSMS statements CREATE-MANAGEMENT-CLASS, MODIFY-MANAGEMENT-CLASS). They are allocated attributes which correspond to the various requirement profiles for file migration supported in the SM pubset.

The following attributes are used to control file migration from the S0 level:

- UNUSED-DAYS (period after which the file is considered to be eligible for migration)
- MAXIMUM-SIZE, MINIMUM-SIZE, MINIMUM-EXTENTS (migration restrictions which depend on the file size and number of file extents)
- TO-STORAGE (background level to which the file should be migrated).

The following attributes are used to control file migration from the S1 level to the S2 level:

- MINIMUM-DAYS-ON-S1 and MAXIMUM-DAYS-ON-S1
- MINIMUM-SIZE
- TO-STORAGE=*S2-STORAGE-LEVEL

Users can issue the HSMS statement SHOW-MANAGEMENT-CLASS to obtain information about the HSMS management classes permitted to them and the associated attributes. They can express their requirements concerning file migration to background levels by using the MODIFY-FILE-ATTRIBUTES command to allocate the most suitable HSMS management class to each individual file. GUARDS profiles can be used to restrict the use of individual HSMS management classes to certain users.

It is advisable for systems support to provide a number of special HSMS management classes, e.g. ‘NO LONG-TERM MIGRATION CLASS’ (by selecting a very high value for UNUSED-DAYS), or an 'IMMEDIATELY MIGRATABLE’ class (UNUSED-DAYS=0, MAXIMUM-SIZE=*NONE, MINIMUM-SIZE=*NONE, MINIMUM-EXTENTS=1,..). If users want a file to be migrated as soon as possible, e.g. because they know that from now on they will not need this file for an extended period, they can achieve this by allocating the HSMS management class ‘IMMEDIATELY MIGRATABLE’. This means that storage space in the S0 level is not unnecessary occupied for long periods.

Migration locks

To reorganize the S0 level systems support can migrate files with a large number of extents or files which are not located on optimally suited volume sets to background levels and thus free space in the S0 level. The files are then recalled into the processing level and their distribution is redetermined. For efficient reorganization, it is also advisable to use background levels for the relocation of files which do not currently meet the criteria for long-term migratability. In this case, the attributes for long-term background migration stored in the HSMS management classes can be ignored. The accessibility of the involved files is restricted for the entire period of reorganization. A similar situation occurs if files are temporarily transferred to background levels in the event of temporary bottlenecks.

Certain files may never be moved to background levels, i.e. not even temporarily as part of pubset reorganization or in the event of temporary storage bottlenecks. These files must be appropriately identified by the user. This is possible with the hard migration lock which can be set using the CREATE-FILE and MODIFY-FILE-ATTRIBUTES commands (MIGRATE=*FORBIDDEN operand). HSMS automatically takes account of the hard migration lock which cannot be deactivated by systems support. It is, for example, important in connection with physical allocation since files which are physically allocated at the block or volume level are fixed at their storage location which may not change even on a short-term basis. This lock can also be used to identify certain files which are constrained to remain in the S0 level, because of fundamental considerations. This property means that it can be used as an alternative to except file lists. While the latter must be maintained by systems support (HSMS administrator) in consultation with the users, users themselves are able to set the hard migration lock. As this gives them a very great level of ability to influence pubset allocation, as is also the case with physical allocation, such users must be authorized to perform physical allocation.

In addition to the hard migration lock, there is also a soft migration lock (MIGRATE=*INHIBITED file attribute). It can be disabled by systems support (see the HSMS statement MODIFY-HSMS-PARAMETERS, operand MIGRATION-CONTROL). Its meaning is not predefined, but depends on the circumstances under which systems support takes account of it or ignores it. The soft migration lock can be handled as follows: files for which a soft migration lock is set are not subject to any longer term background migration, irrespective of whether or not an HSMS management class is allocated to them, and how any allocated HSMS management class is defined. When short-term migration are performed, e.g. as part of pubset reorganization or in the event of temporary storage bottlenecks, the soft migration lock is ignored. It therefore has largely the same significance as assigning an HSMS management class ‘NO LONG-TERM MIGRATION’ (see above). Unlike HSMS management classes, the migration locks can be used for files located in both SF pubsets and SM pubsets. In the case of SF pubsets, they provide users with a simple tool for controlling file migration to background levels and may also be useful in SM pubsets in cases where the setting up of HSMS management classes is considered too time-consuming.

Relationship between eligibility for migration and other file attributes

Temporary files, work files (file attribute WORK-FILE=*YES) and files with high availability requirements (file attribute AVAILABILITY=*HIGH) cannot generally be migrated to background levels. In the case of temporary files and work files, this is because they are assumed to have only a short lifetime. Files with high availability requirements are not moved to background levels, because it cannot be assumed that the lower-cost media, which can be used for background levels, provide the necessary level of reliability. As a consequence of these restrictions, users cannot assign the attribute AVAILABILITY=*HIGH to a migrated file or convert a migrated file to a temporary file. The file must first be retrieved into the processing level before these attributes can be modified.

9.2.2 Performing file migration to background levels

In this section we primarily consider file migration in terms of the long-term migration of files. Short-term file migration in the event of temporary storage bottlenecks and pubset reorganization is described in detail in chapter “Controlling resources in SM pubsets” on page 253.

HSMS management classes not only make it possible to define a varied range of services for the use of background levels, but also provide systems support with easy-to-use functions which make it possible to do this with little effort. Here considerable importance is attached to the ability to automate migration jobs. Below we present the tools for the automation of file migration provided by BS2000 and, most importantly, HSMS.

Automatic activation of migration jobs

A key precondition for automatability is that migration jobs can be initiated automatically. This can, for example, be achieved by means of ENTER jobs:

- The REPEAT-JOB functionality can be used to start ENTER procedures for migration jobs at intervals which can be specified by users.
- The job variables for the current saturation values provided by HSMS can be used to ensure that certain files are automatically migrated to background levels when certain saturation values are reached. In the case of SF pubsets, these are pubset-specific, whereas in SM pubsets saturation monitoring is performed for each volume set separately. The job variables for saturation monitoring are correspondingly volume set-specific for SM pubsets. CJC (functionality of the Job Variables product) can be used to set an ENTER job to a wait state which is quit when the saturation job variables reach certain levels. The ENTER job can start migration jobs when it exits this wait state.
- At program level, interfaces exist which signal status modifications to the job variables for saturation monitoring as well as the occurrence of certain time-dependent events (calendar functions). Together with program interfaces provided by HSMS for the activation of migration jobs, this allows systems support to develop event-driven programs for migrations to background levels which can be executed in ENTER jobs.

Determining migratable files

Systems support should be freed from the need to monitor the complex criteria which govern the eligibility of files for migration. Instead, it must be possible to entrust this task to the system This is made possible by the MANAGEMENT-CLASS parameter in the HSMS statement MIGRATE-FILES.

If the migration job is initiated for files located in the processing level (FROM-STORAGE= *S0-LEVEL) and if a given HSMS management class is specified in this migration job then it automatically relates to all files which meet the following conditions:

- The specified HSMS management class is allocated to the files.
- The files meet the criteria relating to migration from the processing level specified in the HSMS management class (UNUSED-DAYS, MINIMUM-SIZE, MAXIMUM-SIZE, MINIMUM-EXTENTS).
- The files are not excluded from migration by any other conditions, such as migration locks (see below).

The background level to which the files are moved is determined from the HSMS management class definition (TO-STORAGE operand).

Other conditions which prevent the background migration of files include the following:

- The file is a work file, a temporary file or a file with high availability requirements.
- The file has the user ID SYSHSMS or is a system file.
- The file is present in the except file list of the SM pubset (HSMS statement MODIFY-SM-PUBSET-PARAMETERS, operands MIGRATION-CONTROL, EXCEPT-FILE) or in the except file list which is explicitly specified for the migration job (see HSMS statement MIGRATE-FILES, operands S0-STORAGE-LEVEL, EXCEPT-FILE-NAMES).
- The file is a file generation group file.
- The file is currently open.
- The file is subject to a SECURE lock.
- The file has to be repaired.
- The file is affected by a hard migration lock (MIGRATE=*FORBIDDEN).
- The file possesses a soft migration lock (MIGRATE=*INIBITED) which cannot be ignored in the corresponding SM pubset (HSMS statement MODIFY-SM-PUBSET-PARAMETERS, operands MIGRATION-CONTROL, FILE-INHIBIT).
- A backup class other than E is allocated to the file, there is no current backup for the file and only currently backed up files may be migrated within the corresponding SM pubset (HSMS statement MODIFY-SM-PUBSET-PARAMETERS, operands MIGRATION-CONTROL, BACKUP-MANDATORY).

- Migration to background levels is not permitted in the corresponding SM pubset (see HSMS statement MODIFY-SM-PUBSET-PARAMETERS, operands MIGRATION-CONTROL, MIGRATION).
- In the corresponding SM pubset, migration to background levels is restricted to the S2 level (HSMS statement MODIFY-SM-PUBSET-PARAMETERS, operands MIGRATION-CONTROL, MIGRATION), whereas the S1 level is specified as the required background level in the HSMS management class.

Migration from the S1 level to the S2 level is performed using a migration job with the specification FROM-STORAGE=*S1-STORAGE-LEVEL. If an HSMS management class is specified then the job affects all the files located in the S1 level which meet all following conditions:

- The specified HSMS management class is allocated to the files.
- The files meet the criteria relating to migration from the processing level specified in the HSMS management class (MINIMUM-DAYS-ON-S1, MAXIMUM-DAYS-ON-S1 and MINIMUM-SIZE).
- The files are not excluded from migration by any other conditions (e.g. certain settings for MIGRATION-CONTROL).

In order to achieve the migration response defined for the individual HSMS management classes, it is sufficient to provide each HSMS management class with one (for HSMS management classes with direct migration from the processing level to the S2 level) or two (for HSMS management classes with two-stage migration via the S1 level) migration jobs and ensure that these are started as frequently as necessary (e.g. daily). The more frequently the migration jobs are executed, the shorter the average time in which files continue to occupy space in the processing level or S1 level, once they meet the conditions for migration. The periods specified in the HSMS management class definitions, which are used to determine the eligibility of files for migration, do not be taken into account when the frequency with which migration jobs are executed is determined.

In order to ensure that the command procedures with the migration jobs can be executed at the required times, systems support must, of course, also ensure that the HSMS environment required in each instance has been set up and is operational and that the necessary resources (tapes, tape devices etc.) can be made available. The preparations necessary can be simplified by the use of robot-operated tape archives.

User-controlled file migration

In an SMS administered data center, the organization and execution of file migration is the task of systems support (HSMS administrator). However, systems support can also give users the right to initiate migration jobs themselves.

9.2.3 Recalling files into the processing level

Files which have been migrated for long periods are usually recalled on the initiative of the user. As in the case of migration jobs, systems support must ensure that the HSMS environment necessary to make recall possible has been set up and is operational and that the necessary resources (tapes, tape devices etc.) can be made available.

Files can be recalled either explicitly or implicitly:

- An explicit recall is performed using the HSMS call RECALL-MIGRATED-FILES.
- An implicit call is performed if a user opens a file which is located in a background level or reserves it using the SECURE-RESOURCE-ALLOCATION command. Systems support can define computer-specific restrictions for implicit recalls from the S2 level (HSMS statement MODIFY-HSMS-PARAMETERS, operand MIGRATION-CONTROL).

Even when SF pubsets are used, a number of problems may occur when a migrated files is retrieved. For example, the pubset may be affected by saturation problems or the user's space quota may not be sufficient. Even more complex situations may occur in the case of SM pubsets since the user quotas are more highly differentiated and, instead of global saturation monitoring as in SF pubsets, monitoring is performed for the individual volume sets, thus making it necessary to consider the saturation level of the individual volume sets. In addition, it is not certain that an SM pubset will always contain a volume set which is compatible with the file attributes of a migrated file and to which the file can be retrieved - for example, if after background migration of all the K files, the only K volume set is removed from the pubset by means of pubset reconfiguration or is locked for allocation. It is the responsibility of systems support to maintain the pubset in such a way that file recall does not normally fail, provided that the user possesses a sufficiently large user quota. In special situations, e.g. when error situations or bottlenecks occur etc., this cannot always be guaranteed. In order to make sure that the files required for processing can be made available and to avoid undesirable recall wait times during processing, users should issue the SECURE-RESOURCE-ALLOCATION command to recall the files into the processing level before processing. They are then also subsequently protected from migration to processing levels. If multiple migrated files are required for processing, it may be advisable to use SECURE-RESOURCE-ALLOCATION to generated collective jobs since these permit particular efficient recall operation.

9.2.4 The effects of migration and recalls on file attributes

Migrating a file to background levels and subsequently recalling it does not usually change the file's attributes. However, the following exceptions apply:

File location (extent lists)

Both the location of the file in the processing level, i.e. the volume set to which the file is sent, and the location of the file within the volume set are redetermined. The volume set which best corresponds to the file attributes is selected unless the user explicitly specifies a volume set in the RECALL-MIGRATED-FILES call (NEW-DATA-SUPPORT operand) or the file has to be returned to the volume set in which it was located before being migrated to a background level (file attribute S0-MIGRATION=*FORBIDDEN). Users must be authorized to perform physical allocation before they can explicitly specify the volume set. Unlike the CREATE-FILE command, explicitly specifying the volume set for the file does not result in S0-MIGRATION=*FORBIDDEN. If the file is to be fixed in the specified volume set, the user must specify this following recall using the MODIFY-FILE-ATTRIBUTES command. The redetermining of the volume set as part of a migrate/recall is a tool to improve poor file distribution in which file attributes and volume set attributes do not correspond. One cause of poor distribution is that the modification of file attributes by the user does not result in the immediate relocation of the file to a more suitable volume set unless such relocation is forced by the incompatibility of the new file attributes and the previous volume set. After modifying the attributes of a file, users can force it to be relocated to a suitable volume set by ensuring that the affected file is migrated to a background level as rapidly as possible (e.g. by means of an explicit migration job in the case of users who are so authorized or by allocating a corresponding HSMS management class). However, it is only purposeful for users to perform this type of reorganization measure in special cases, and pubset reorganization should usually be performed centrally by systems support.

File size

If a file is migrated to a background level, the system records the amount of space it previously occupied in the processing level. When a migrated file is recalled, the system attempts to make precisely this amount of space available in the processing level. When this is done it may prove necessary to round up the value so that is represents a multiple of the allocation unit of the volume set into which the file is recalled.

Certain of the file attributes which are relevant for the storage location in the processing level (performance attribute, file size, storage class allocation...) can be modified by users while the file is migrated. These modifications do not take effect until the file is recalled into the processing level.

9.3 File lifetimes

Files are usually deleted on the initiative of users. However, there are mechanisms which allow users to delegate file deletion to the system or systems support:

Temporary files

Temporary files in an SM pubset are normally handled like temporary files in an SF pubset. At the end of a user session, all the temporary files which were created during this session and which have not yet been deleted by the user are (normally) automatically removed by the system.

Work files

Work files can only be created in SM pubsets. When using work files it is advisable for systems support and users to agree on a time at which systems support is authorized to remove the files or, if necessary, force their removal by terminating tasks at the work volume sets. On removal, the work files located at the work volume sets are lost.

FREE-FOR-DELETION attribute

This attribute (CREATE-FILE, MODIFY-FILE-ATTRIBUTES commands) allows users to allocate a period to a file after which it can be deleted. The fact that files for which this period has expired can be identified using the SHOW-FILE-ATTRIBUTES and DELETE-FILE selection criteria means that it is simple to create clean-up procedures. These clean-up activities can be decentralized and performed by individual users or centralized and carried out by systems support. Centralized delete operations may, for example, allow systems support to free space if storage bottlenecks occur.

RETENTION-PERIOD

The RETENTION-PERIOD is relevant to file lifetimes. Although it protects files against accidental, premature deletion, it does not automatically ensure that the file contents of files migrated to background levels are retained until the expiry of the retention period. To make this possible, systems support must specify the retention period of the save files mechanisms appropriately.

9.4 Backing up and restoring files

9.4.1 Backup application scenarios

It is necessary to distinguish between different cases when performing backups:

- In one possible scenario, it may be necessary for users to have considerable influence on the time of backup, the selection of the backup tapes, backup procedures etc. - for example, in order to back up all the files belonging to an application jointly at a given synchronization point, thus distributing the backups over the smallest possible number of tapes and making a rapid restore possible if necessary. Private backup archives are suitable for such applications. Users maintain these themselves and are responsible for initiating the backup jobs. The entire range of HSMS backup functions is available to users when performing these operations.
- In a different example scenario, the precise time of backup is not important to users. They also want to be involved to the smallest possible extent in the technical implementations of backups. However, file-specific methods of influencing the frequency of backups and the lifetime of backups are still required. In such cases, centrally managed standard backup archives maintained by systems support are suitable. The users inform systems support of their wishes regarding the frequency of backups, lifetime of backups, backup procedures (full backup/partial backup) etc. and systems support is responsible for performing backups to a standard archive in a way which meets users' requirements.

9.4.2 The user/systems support staff interface

The second example above is of particular importance in an SMS-administered data center. Below we wish to outline the ways in which this is supported by the system and HSMS.

The SAVE file attribute and the HSMS management classes (SM pubsets only) provide users with interfaces which they can use to formulate their backup requirements. Both can be allocated to a file using the CREATE-FILE and MODIFY-FILE-ATTRIBUTES commands:

- The SAVE attributes make it possible to specify a backup class and determine whether the next backup should take the form of an incremental backup (affecting only those blocks modified since the last backup) or a full backup. The file's backup class determines the backup operations in which it is included. It therefore affects the backup frequency. The actual significance of a backup class depends on how it is treated when backups are performed and, in particular, on how often systems support performs backup operations for the corresponding backup class. If users are to be able to use the backup classes sensibly, it is necessary for them to reach an agreement with systems support concerning their actual meaning.
- The HSMS management classes can be used as an alternative to the backup classes in order to specify the backup frequency for files. However, this is not a fixed component of the HSMS management class; instead, it must be agreed between the systems support staff and the user in the same way as for a backup class. The systems support staff can describe the service that an HSMS management class represents for the backup in the comment section. In addition, the only backup-relevant attribute of the HSMS management classes (as things stand) is the RETENTION-PERIOD. This is specified for the different HSMS management classes of an SM pubset by the systems support staff by means of the HSMS statements CREATE-/MODIFY-MANAGEMENT-CLASS. The HSMS statement SHOW-MANAGEMENT-CLASS allows users to obtain information on the characteristics of the various HSMS management classes specified by the systems support staff.

9.4.3 Backups performed by systems support

The following facilities make it easier for systems support to provide the services agreed with users:

- Similar procedures to those available for background migration make it possible to perform backup runs at various intervals (ENTER procedures, calendar functions etc.).
- Various selection criteria available for (HSMS) backup jobs make it possible to ensure that the correct files are backed up, e.g. SELECT-FILES, MAXIMUM-BACKUP-CLASS, MANAGEMENT-CLASS. For example, in order to ensure that files with BACKUP-CLASS=A are backed up daily, files with BACKUP-CLASS=B weekly and files with BACKUP-CLASS=C monthly, it is possible to perform daily migration jobs with MAXIMUM-BACKUP-CLASS=A, weekly jobs with MAXIMUM-BACKUP-CLASS=B, and monthly jobs with MAXIMUM-BACKUP-CLASS=C. Specifying SELECT-FILE=*MODIFIED-FILES makes it possible to prevent identical backups being created unnecessarily often as part of different backup runs.
- If an HSMS management class is specified for BACKUP-FILES, all the files belonging to the corresponding HSMS management class and which are not excluded because of any other selection criteria are backed up. The backups are written to a save file to which HSMS automatically assigns the retention period specified for the HSMS management class.

9.4.4 Restoring files

Depending on the application in question, the decision whether or not to restore a file may depend on systems support or the user. If files have been destroyed as a result of a disk failure, it is advisable for systems support to restore backups as a service for users. If users have destroyed files through their own errors then it is their responsibility to initiate restore operations.

9.5 Long-term archiving

Long-term archives serve as containers for backup copies of files which have to be retained for long periods (e.g. because of legal requirements) and which will probably rarely or never be accessed. It is the responsibility of the users to perform long-term archiving. The same also applies when a file is restored from a long-term archive. Currently no special tools are available to help users delegate the technical implementation of long-term archiving to systems support. However, special arrangements can be discussed to make this possible. For example, if files with a FREE-FOR-DELETION period are automatically deleted, systems support may specify that backup copies are to be created in a long-term archive. There are no differences in functionality between SF pubsets and SM pubsets.

10 Pubset monitoring and pubset maintenance

10.1 Overview

In order to keep a pubset in good condition, pubset maintenance should be carried out on a continuous basis. The essential prerequisites for this are that it can take place while the pubset is in operation and users are disturbed as little as possible. The system provides options specifically for this purpose, in particular for SM pubsets.

Pubset monitoring and maintenance involve the recognition and elimination of the following situations:

- Storage bottlenecks in the S0 level
- Poor organization of the S0 level (fragmentation of the storage space and file extent fragmentation, space incorrectly recorded as occupied, poor distribution of files over the individual volume sets)
- Discrepancies between the performance and the availability attributes described in the volume set attribute profiles and the real behavior of the volume sets
- Poor organization of the background level and backup archives
- Storage bottlenecks in the background levels and backup archives
- Disk defects; the systems support staff should develop a recovery concept in advance so that an SM pubset can be restored to a status that is as up to date as possible after a partial or complete failure. The implementation of a recovery concept generally requires precautions to be taken constantly when the pubset is still intact and running normally
- Replacement of hardware, replacement of old storage media

In the sections below, some of the above points are examined more closely. We restrict ourselves, however, to SMS-specific aspects (i.e. to problems and procedures for dealing with them that are specific to SM pubsets).

10.2 Storage bottlenecks on the S0 level

Storage bottlenecks at the S0 level always become visible as storage bottlenecks on individual volume sets. Distinctions must be drawn between different situations:

1. Storage bottlenecks that can be relieved while retaining the desired allocation strategies through redistribution of the files in the SM pubset (including the background levels). Retaining the allocation strategies means, among other things, that the following must be retained at redistribution:
   – existing allocation locks
   – the storage classes assigned to the files with volume set lists
   – the compatibility of file attributes and volume set attributes
   – the thresholds for storage saturation defined by the administrator
   – the deadlines agreed with users for file migration at background levels
2. Storage bottlenecks that cannot be dealt with by redistribution alone. These can also occur when there still seems to be enough space on the SM pubset as a whole. They are caused by insufficient pubset resources. If these bottlenecks last, they can in the end only be dealt with by resource-saving behavior on the part of users or by the provision of additional resources. Short-term bottlenecks can often be dealt with by means of special measures that do not significantly impair users.

Examples of bottlenecks that can be eliminated by redistribution

Although the system considers occupancy levels when selecting the volume set by respecting exceeded saturation thresholds and attempting to avoid extremely divergent allocations within individual volume sets, it is still not always possible to avoid local volume set bottlenecks, as the following examples show:

1. The volume sets in an SM pubset all have a high saturation level. Initially, occupancy is balanced out. Users then delete files, thus randomly affecting the volume sets in extremely different ways.
2. Individual files on a volume set are enlarged unexpectedly by users (unfavorable user behavior) after being created (as a result of a large number of secondary allocations, for example). The volume set contains files that could be moved to other volume sets or to background levels, while retaining the desired allocation strategy in the SM pubset.
3. An additional volume set is added to the pubset in response to a global saturation situation. Distribution to the remaining volume sets initially remains equal.

Examples of bottlenecks caused by insufficient resources

In the following two examples, we assume that file migration to background levels is no longer possible as a means of obtaining space at the processing level. The overloaded volume sets contain only files that cannot be migrated (for longer periods).

1. The volume set on which the bottleneck occurs is the SM pubset's only volume set with the format K. It contains K files exclusively.
2. A particular volume set is intended exclusively for requests for space by specific users. However, the capacity of the volume set is not sufficient to cover their requirements. It is possible to use individual volume sets as resources for particular users with the help of storage classes and volume set lists and by assigning default storage classes.

In both examples, the redistribution of the files in the processing level is ruled out either because the existing pubset configuration does not permit it or because allocation strategies implemented by the systems support staff with the help of storage classes, volume set lists, allocation locks, etc. have to be overridden.

Detection of storage bottlenecks

Since storage bottlenecks can have extremely unpleasant effects such as the abortion of current file processing when secondary allocations fail, it is important that systems support staff prevent these from happening or deal with them quickly should they arise. In order to carry out the measures required efficiently, it is important that the detection of bottlenecks and initiation of countermeasures can be automated.

The MODIFY-SPACE-SATURATION-LEVELS command allows the systems support staff to set saturation thresholds for the individual volume sets of the SM pubset. The highest exceeded saturation threshold describes the current saturation situation. The thresholds set can be displayed by means of the SHOW-PUBSET-PARAMETERS command, and the current saturation level can be displayed by means of the SHOW-PUBSET-SPACE-ALLOCATION command. HSMS job variables in which information on the saturation level of an SF pubset or of volume sets is stored are suitable for the automatic detection of storage bottlenecks. They can be queried in procedures and programs. In addition, the systems support staff can indicate that thresholds have been exceeded in programs or command procedures by means of job variable events. Command procedures or programs can thus be used that automatically initiate countermeasures when storage bottlenecks occur. If the systems support staff want more detailed information on volume set occupation, they can use the SHOW-PUBSET-SPACE-ALLOCATION command.

Measures to prevent and eliminate storage bottlenecks

Storage bottlenecks are prevented or eliminated by creating sufficient free space on the affected volume sets again. This is done by means of different types of measures:

– release of previously occupied space by moving or deleting files

– provision of additional capacities

Options for moving files

1. Gaining space through migration to background levels of files that can be migrated for a long period

   One of the first measures for gaining space is to migrate to the background levels the files at the processing level that meet the criteria agreed between users and systems support for background migration. If HSMS management classes are used to specify differentiated conditions for migration, the systems support staff can start a migration request for each HSMS management class. The files that meet the specified criteria for migration are then taken.

2. Gaining space on a volume set through redistribution of files at the S0 level

   Files can be redistributed between the different volume sets of the processing level. This is done by migrating to a background level the files of the volume sets on which there are storage bottlenecks (VOLUME-SET-ID operand in the HSMS statement MIGRATE-FILES) and then bringing them back to the processing level by means of a recall request. Users may experience short-term problems with file access during redistribution.

3. Gaining space through migration of files that cannot be migrated for a long period

   In saturation situations, it may also be necessary for a certain length of time to migrate files to background levels that do not meet the criteria for migration for a longer period (e.g. if the time since the last access is shorter than the period defined in the HSMS management class or if a file has the soft migration lock MIGRATE=*INHIBITED). The migration of files that do not meet the criteria for migratability over a longer period means a poorer service for users, since access can involve long waiting times. The systems support staff should therefore only take this action in exceptional situations. In addition, files that are only to be migrated to background levels for shorter bottlenecks should automatically be brought back to the processing level by the systems support staff as soon as the situation is relieved.

Provision of additional storage capacity

Additional storage capacity can be made available for an SM pubset by adding further volume sets or enlarging existing volume sets. Dynamic pubset reconfiguration permits this action to be taken during pubset operation.

A particularly critical situation arises when there is not enough space on the volume set for files that grow considerably during processing, and no more space can be gained by moving files around. Since open files cannot be moved to other volume sets, the volume set can only be enlarged by adding further volumes. Measures should be taken in advance to counter critical bottlenecks such as this (e.g. by reserving space for secondary allocations). The SM pubsets provide a variety of options here. For example, a volume set can be assigned the lock NEW-FILE-ALLOCATION= *PHYSICAL-ONLY when a saturation threshold is exceeded. If the value goes under the threshold again, the lock can be canceled again. The setting and resetting of the threshold can be automated by means of command procedures with the HSMS job variables for displaying saturation situations. When the lock is set, the system uses the other volume sets of the SM pubsets to create new files. The space that is still free on the volume set is thus used exclusively for enlarging existing files.

10.3 Quality of storage services

Although each SM pubset provides optimal support for certain services in accordance with its volume set configuration, it also permits the creation of files whose file attributes are possible but cannot be optimally catered for. For example, files with the attribute PERFORMANCE=*VERY-HIGH can also be created in an SM pubset whose volume sets all have the volume set attribute PERFORMANCE=*STD. The SHOW-PUBSET-FILE-SERVICES command provides information about the storage services which are possible in a pubset and the level at which these services can be provided.

In the case of services which are optimally supported, users should be able to rely on the fact that files with the corresponding attributes benefit from the required characteristics. In the case of possible but not optimally supported services, it is necessary to accept limitations in terms of quality. The same applies to special storage services which are provided by means of volume set lists. Users can access special storage services via storage classes and systems support describes the service in the comment section of the class. Users should be able to assume that when files are allocated to such storage classes, they will benefit from physical characteristics which correspond to the description of the storage class.

In order for the pubset to be able to provide storage services of sufficient quality, i.e. to ensure that the file attributes specified by users corresponds to the real characteristics of the files, systems support must consider the following questions:

- Are the files located in volume sets with attribute profiles which optimally match the file attributes specified by the user? In cases where storage services are provided using volume set lists, it is also important that files are located in a volume set belonging to the corresponding volume set list.
- Do the volume set attributes for performance and availability specified by systems support correspond to the real physical characteristics of the volume sets? If this is not the case, volume set selection is based on false criteria. A similar problem exists for special storage services which are provided by means of volume set lists. In this case it is necessary to ensure that the characteristics of the volume sets in the corresponding volume set lists correspond to the storage class description created by systems support.

These two points are discussed in detail below.

10.3.1 Optimum file storage location

The suitability of the individual volume sets of an SM pubset as the storage location for a given file can be evaluated by considering the attribute profiles of the volume sets and volume set lists, the allocation locks and configuration status. A volume set is considered not to be the optimum storage location for a file if a different volume set in the SM pubset receives a better evaluation.

In the case of services which are optimally supported by the SM pubset, at least one volume set is present which fully supports the corresponding file attributes. In the case of the possible but not optimally supported services, this is not so. Here the volume set which comes closest to supporting the required file attributes is considered to be optimal. For example, when considering performance, a volume set with PERFORMANCE=*STD is considered to be the optimum storage location for a file with the file attribute PERFORMANCE=*VERY-HIGH if the SM pubset does not contain any volume set with PERFORMANCE=*HIGH or PERFORMANCE=*VERY-HIGH.

Causes for the non-optimum location of files

Files may not be located in an optimally suited volume set in an SM pubset for the following reasons:

- The pubset does not provide sufficient resources. If the volume set which is evaluated as optimal does not contain sufficient space for all the files for which this service is requested then the system also considers volume sets which have not been evaluated as optimal when determining the volume set in which the file is to be located. Thus storage bottlenecks may result in non-optimal allocations.
- If the volume set configuration of a pubset changes, e.g. through the addition of new volume sets, the removal of allocation locks, modifications to the performance and availability profiles etc., volume sets which were previously optimal may no longer be so since better alternatives may be available.
- If a user modifies the required storage service for a file, e.g. by allocating a different storage class to it, the previous storage location is retained if it is compatible with the new attributes, even though it may not be optimally suited.

If systems support allocates a different volume set list to a storage class or modifies the allocated volume set lists, this has no effect on file storage locations.

Effects of non-optimum storage locations

If files are located in volume sets which are not optimally suited to them, this has the following effects:

- If files are located in volume sets which do not adequately support the file attributes, the service quality provided to users is impaired. This is, for example, the case if a file with the file attribute PERFORMANCE=*HIGH is located in a volume set with the volume set attribute PERFORMANCE=*STD, despite the fact that a better alternative is available.
- If files are located in a volume set which exceed users' service requirements, this results in the poor utilization of the resources available in the pubset. This situation may occur, for example, if a file with the file attribute AVAILABILITY=*STD is sent to a volume set with the volume set attribute AVAILABILITY=*HIGH.

Assistance in identifying files with poor storage locations

Systems support can use the HSMS statement SELECT-FILE-NAMES (operand ALLOCATION-QUALITY=*BEST-VOLUME SET or *NOT-BEST-VOLUME-SET) to obtain information about files with optimum or non-optimum storage locations.

Pubset reorganization

If the resources of an SM pubset are insufficient to meet users' requirements, systems support must provide further resources (e.g. by adding new volume sets or by enlarging existing volume sets). If the resources are available, but the files are poorly distributed within the SM pubset, the pubset must be reorganized. The procedure is the same as that used to reorganize volume sets with fragmented storage and the two operations can be performed simultaneously. Files which are not optimally located are migrated to background levels. The operand DATA-LOCATION=*NOT-BEST-VOLUME-SET in the MIGRATE statement can be used to identify the files affected. Alternatively, the HSMS statement SELECT-FILE-NAMES can be used to generate a list during the migration job. The files are then recalled into the processing level and their storage location is redetermined.

10.3.2 Defective volume set quality

It is very important that the behavior expected from a volume set (on the basis of the volume set attribute profile or the description in the storage classes) matches its real behavior. Long-term discrepancies between the described and actual behavior mean that the system uses false premises when selecting volume sets and therefore maps user requirements to the physical configuration incorrectly. This reduces the quality of the storage services provided by the pubset.

The possibilities for checking whether a volume set genuinely possesses the characteristics which are expected of it depends to a very great extent on its particular physical configuration. We therefore refer readers to the information functions associated with RAID, DRV, etc., in which these functions are presented in greater detail. If systems support recognizes that a volume set does not meet the definitions in its attribute profile, it is necessary to attempt to eliminate the discrepancy. The required measures are again specific to the storage medium in question. If it is not possible to eliminate the discrepancy, the attribute profiles must be corrected. For example, if a volume set with the volume set attribute AVAILABILITY=*HIGH does not provide the required level of reliability, it should be reclassified with AVAILABILITY=*STD. In this event, the files with high availability requirements must be relocated to other, more reliable volume sets.

10.4 Actions ensuring an SM pubset can be restored

If a volume of an SM pubset fails, the effects depend on the volume set to which it belongs. If it is not the control volume set, only the relevant volume set itself is affected. The files it contains are lost, and it can be removed from the SM pubset as a defective volume set. A list of the files that have been destroyed is created. This list can be used to restore the files from backups. The operation of the rest of the SM pubset does not have to be interrupted, and its remaining functionality is not restricted, since the central pubset metadata is on the control volume set. If the failed volume belongs to the control volume set, however, orderly pubset operation is no longer possible. We examine in detail below how the SM pubset can be restored with as few losses as possible in the latter case.

Because of the serious effects associated with the failure of the control volume set, it is advisable to use particularly failsafe volumes for it (e.g. mirrored disks). In addition, its disk configuration should be kept as compact as possible. The smaller the number of disks, the smaller is the risk of the volume set failing. The failure of the control volume set can thus be made extremely unlikely, but the possibility cannot be ruled out. In order to limit the damage (i.e. the loss of user files) as far as possible when it does occur, special functions are provided for restoring the SM pubset. To enable the SM pubset to be restored as quickly as possible, to allow it to be used as it was before the failure and to minimize the downtime, the systems support staff have to take precautions in normal pubset operation in case there is a failure.

The basic functionality for pubset restoration is made available by a conversion function when the pubset is put into service and by the SMPGEN utility. Both of these together allow the SM pubset to be reconstructed in a rudimentary state from the remaining volume sets that have not failed. The user files are preserved with their current status at the time of the failure. The reconstructed pubset is subject to certain restrictions initially. These affect, for example, access to files that have been migrated to background levels, the use of file backups created before the failure of the pubset, resource protection by means of GUARDS profiles, etc. In order to lift these restrictions and subsequently resume normal service with the SM pubset, further restoration steps are necessary.

The precautions that should be taken by the systems support staff in normal pubset operation in the event of the failure of the control volume set can best be illustrated by the steps required to restore the SM pubset. These show best the loss of functionality and the difficulties that result if the precautions are not taken or if the precautions taken are inadequate. We examine the steps required for restoration by taking the example of an SM pubset whose control volume set contains pubset metadata exclusively (the recommended scenario particularly for large SM pubsets). The user data is located on data volume sets unless it has been migrated to background levels. The SM pubset should also contain an S1 volume set that is used both for file migration and backup.

When the control volume set fails, the data it contains is destroyed, and the other volume sets and their files are still available but cannot be used directly (status: free but not empty volume sets). The volume sets also contain the catalog entries for their files together with the file attributes that are relevant for their usage in SM pubsets, such as availability, performance, work file, assigned storage class, assigned management class, etc. Each volume set also contains rudimentary information about the pubset users whose files it contains.

Restoration of the rudimentary SM pubset with the user files at the S0 level

In order to restore the SM pubset, the systems support staff must first form SF pubsets in an intermediate step. For the future control volume set it is advisable to use SIR to create a new SF pubset. It receives the other SF pubsets by converting the volume sets of the failed SM pubsets to SF pubsets in a special pubset startup procedure. The files they contain are preserved. File attributes that are only relevant for SM pubsets (such as the storage class assigned to a file) are not lost either, but they remain only latent on the SF pubsets. The user catalogs are reconstructed from the user information stored on the volume sets. Although the SF pubsets can then be put into service individually, it is generally advisable not to do this. They should be viewed primarily as intermediate products for the restoration of the entire SM pubset.

In the next step, the systems support staff have to group the SF pubsets to form an SM pubset with the help of SMPGEN. The SF pubset derived from the S1 volume sets must be made known to SMPGEN so that it can receive the same role again. The result is a functioning SM pubset containing all the user files with the status they had before the pubset failure, but which is still missing that part of the previous pubset infrastructure that was stored on the destroyed control volume set. A glance at the system files created on the control volume set shows us which data is affected by this.

Creation of the HSMS environment for the SM pubset

The control volume set also contains the HSMS metadata files. This is not obligatory for the directories, but it is recommended. Of these, the control file and the directories for migration and backup archives are particularly important. The control file contains the configuration data of the HSMS environment created for the SM pubset (archive definitions, configuration parameters, etc.). As a result of its loss, the HSMS environment must be set up again on the reconstructed SM pubset. In addition, the lost directories must be restored from backups or copies. These should be as up to date as possible. This can be achieved for a backup archive, for example, by always backing up the directory of the backup archive after user files are backed up. Another way to do this is to always keep current copies of the directories on another volume set of the SM pubset. Directories for which no backups exist can, if necessary, also be reconstructed from the save files with the file backups or the migrated files, although this is an extremely laborious process.

Restoration of the background levels

For the files that were migrated to background levels at the failure of the control volume set, the file contents stored on the tapes of the S2 level or on the S1 volume set are still available, but, as a result of the failure of the control volume set, the directory of the migration archive (see above) and the catalog file for migrated files are also lost. The catalog file for migrated files is created again automatically when the SM pubset is formed with SMPGEN, but it is initially empty. It must be filled again by restoring saved catalog entries for the migrated files. The systems support staff must take active precautions in normal pubset operation so that backups that are as up to date as possible can be made available in an emergency.

Restoration of the pubset configuration data

The configuration file contains all the essential configuration parameters for the SM pubset. These include the attribute profiles, usage types, usage restrictions, saturation thresholds for the individual volume sets and pubset-global specifications such as the defaults for the pubset space. These settings are lost in the event of the failure of the control volume set. When the SM pubset is formed with SMPGEN, most of the configuration parameters are assigned default values, but SMPGEN allows some to be specified by the systems support staff. The configuration parameters correspond to the usage scenario wanted by the user. In order to be able to use the SM pubset again in the same way as before it failed, after it is reconstructed the dynamic pubset reconfiguration commands must be used to configure it appropriately. There is no way to back up the configuration file directly. To enable the old settings to be discovered when necessary, it is advisable after changes are made to configuration settings in normal pubset operation to output their current values in S variables using the pubset information functions and store them in a file. It goes without saying that the file must not be on the control volume set. This process can be automated with command procedures. The setting of the usage type HSMS-CONTROLLED requires special consideration. By default, when pubsets are formed with SMPGEN, all volume sets receive the usage type STANDARD. After the pubset is formed, a change is only possible for volume sets with the status “defined only”. If a former S1 volume set is assigned the usage type STANDARD when the pubset is formed, it can no longer be redefined appropriately later without losing its S1 backups and migrated files. It is therefore important to set the usage type HSMS-CONTROLLED for such a volume set when a pubset is formed with SMPGEN.

Pubset user catalog

When the SM pubset is formed, the user catalog is created again, but it does not contain the user quotas, user rights or physical allocation and assignments for default storage classes that were on the previous SM pubset. Since these are specifications that can have an extremely important effect on the use of the SM pubset (by assigning default storage classes, for example, the systems support staff can have an important influence on the allocation strategy in an SM pubset), it should be possible to restore this information from a backup of the pubset user catalog, and the information should be as up to date as possible at the time of the failure.

Creation of the GUARDS catalog

As a result of the failure of the control volume set, the GUARDS catalog is lost. In order to establish file protection again, restoration from backups is necessary.

Storage classes, volume set lists and management classes

The storage classes and volume set lists created on the SM pubset are lost as a result of the failure of the control volume set. If the systems support staff use them in order to implement storage usage strategies on the SM pubset, they must set them up again with the desired statues after the failure of the control volume set. The same applies to the management classes.

11 Controlling resources in SM pubsets

The following functions are available for user-specific resource monitoring in SM pubsets.

- User quotas for space occupied by files
- DMS-TUNING-RESOURCES
- GUARDS protection for storage classes and HSMS management classes
- Physical allocation authorizations
- Maximum numbers for cataloged files and job variable

These functions are discussed in detail in the sections below.

11.1 User quotas for occupied space

The concept of user quotas, which permit systems support to monitor and limit the space occupied by individual users in a pubset, is already present in SF pubsets and continues to be supported in SM pubsets. However, here the quotas are more differentiated in order to take account of the special characteristics of SM pubsets. In the case of SM pubsets for which the differentiated use and monitoring of resources is not important, the user quotas can be administered as for SF pubsets.

In SF pubsets the quotas for the space occupied by permanent and temporary files are allocated separately (PERM-SPACE quota, TEMP-SPACE quota) and their effect is restricted to disk storage space occupied in the processing level. Quota allocation comprises the following (pubset-specific) functions:

- Registration of space for permanent and temporary files occupied by individual users (space used record)
- Facilities for systems support to set upper limits to the space occupied by permanent or temporary files belonging to individual users (space limits)

- Facilities for systems support to specify how the system is to react if individual users exceed the space limits (public space excess). These definitions have identical effects for both the PERM-SPACE and the TEMP-SPACE quota.

The quota concept for SM pubsets has the following enhancements:

- The WORK-SPACE quota type for space occupied by work files (WORK-FILE file attribute)
- The PERM-SPACE, TEMP-SPACE and WORK-SPACE quotas are structured in themselves. By means of subquotas it is taken into account that the occupied space can vary in quality as regards availability and performance.
- The space occupied at the S2 background level is also taken into account with quotas.

As in the case of SF pubsets, it is possible to define user-specific rules to be applied when quotas are exceeded. These operate identically for all the quotas and subquotas of the SM pubset.

11.1.1 Quota structures for permanent files in SM pubsets

The quota structure for permanent files in an SM pubset is based on the idea that the quality of the space in an SM pubset can be characterized by varying levels of detail. These are illustrated in figure 30:

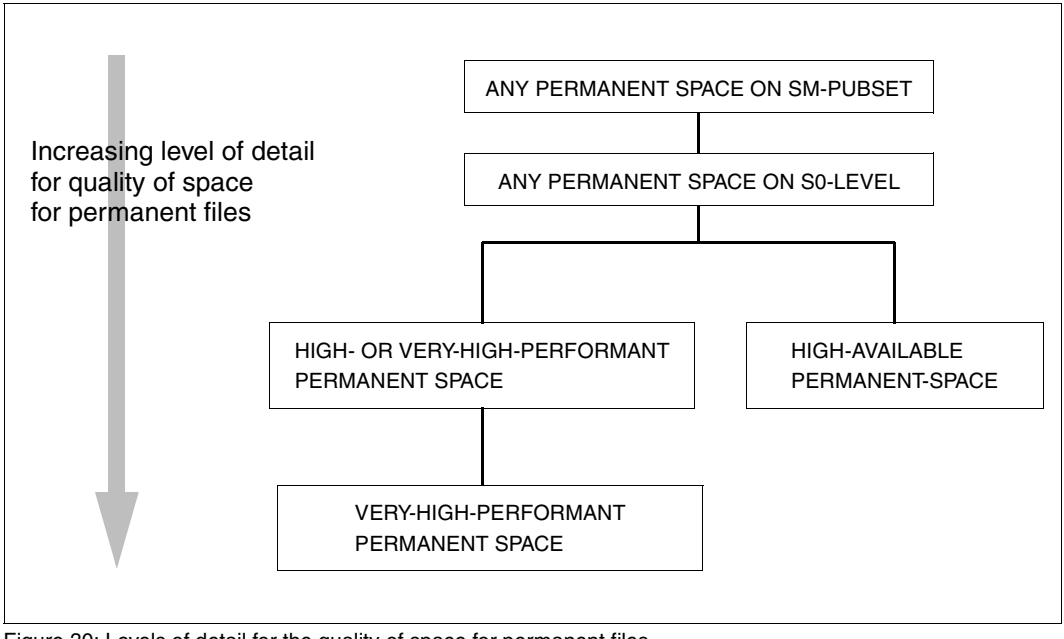


Figure 30: Levels of detail for the quality of space for permanent files

To implement the concept of varying levels of detail for space quality, individual quotas contain subquotas for that proportion of their contents which has more exacting requirements. The individual quotas and the relationships between them are illustrated in figure 31 for the permanent files of an SM pubset:

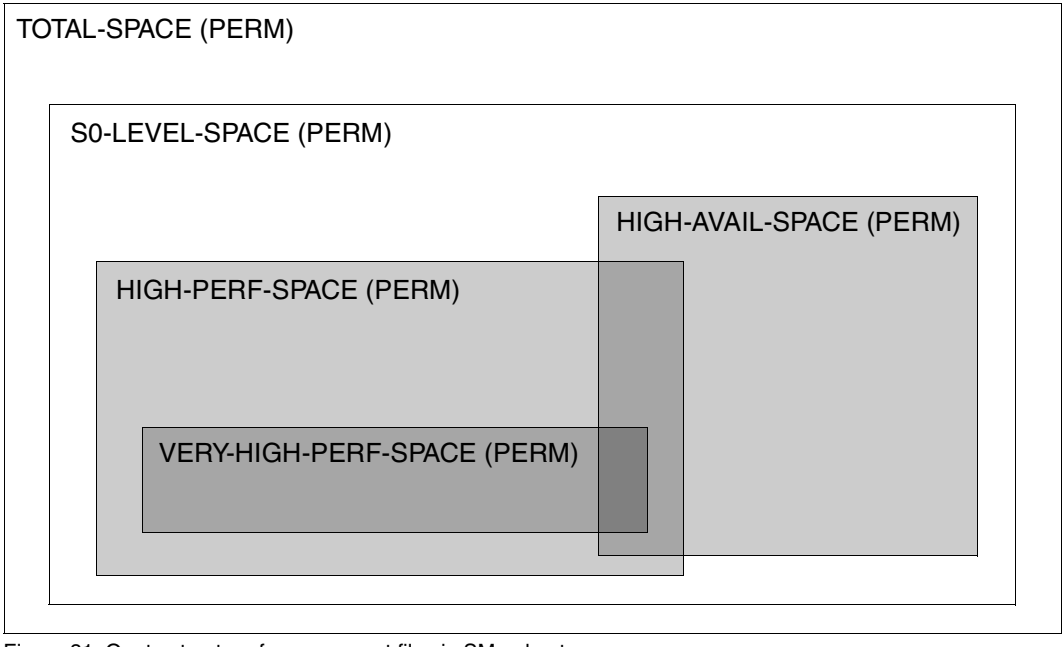


Figure 31: Quota structure for permanent files in SM pubsets

The space allocated to a file is charged to the user quotas. The affected user quotas can be determined from the file attributes. Figure 32 illustrates the relation between file attributes and quotas for permanent files. It should be noted that charging allocated space to a quota also implicitly results in its being charged to all the quotas which include this quota. For example, the creation of a file with the attributes AVAILABILITY=*HIGH and PERFORMANCE=*HIGH results in an increase not just of the occupancy values of the quotas for HIGH-AVAIL-SPACE and HIGH-PERF-SPACE but also for S0-LEVEL-SPACE and TOTAL-SPACE.

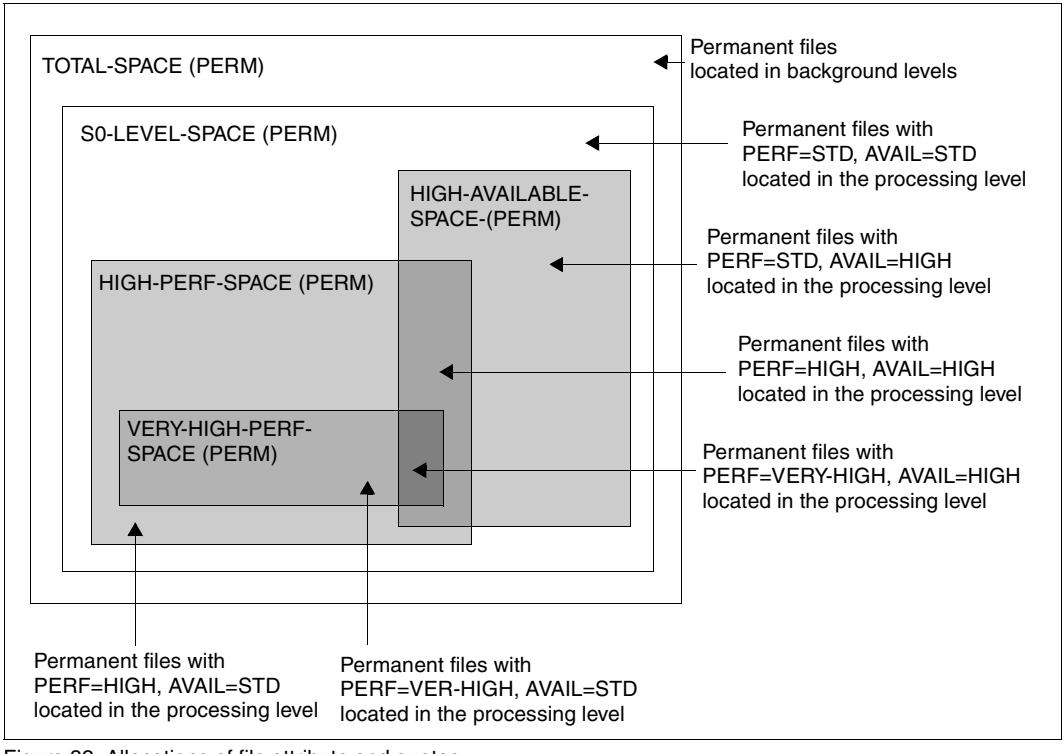


Figure 32: Allocations of file attribute and quotas

Figure 32 corresponds to the definitions for the individual quotas below:

TOTAL-SPACE quota for permanent files

The space occupied by all the permanent files (belonging to one user) which are located in the pubset is allocated to this quota irrespective of whether these files are located in a background level or the S0 level and of the performance and availability attributes which are allocated to them. In the case of a file located in a background level, it is not the space actually occupied in the background level that is allocated to the quota but the space which the file would occupy following a recall to the S0 level.

S0-LEVEL-SPACE quota for permanent files

The proportion of the total space quota to which the space occupied in the S0 level is charged forms the S0-LEVEL-SPACE subquota. The space occupied by all the permanent files (belonging to one user) is charged to this quota, irrespective of the performance and availability attributes which are allocated to them.

HIGH-PERF-SPACE quota for permanent files

This is a subquota of the S0-LEVEL-SPACE quota. The space for all the permanent files (of one user) which are located in the pubset's S0 level and which have the performance attribute PERFORMANCE=*HIGH or PERFORMANCE=*VERY-HIGH is charged to this subquota.

VERY-HIGH-PERF-SPACE quota for permanent files

This is a subquota of the HIGH-PERF-SPACE quota for permanent files. The space for all the permanent files (of one user) which are located in the pubset's S0 level and which have the performance attribute PERFORMANCE=*VERY-HIGH is charged to this subquota.

HIGH-AVAILABLE-SPACE quota for permanent files

This is a subquota of the S0-LEVEL-SPACE quota. The space for all the permanent files (of one user) which are located in the pubset's S0 level and which have the attribute AVAILABILITY=*HIGH is charged to this subquota.

Each quota is characterized by its maximum value (space limit) and the current occupancy value (space used). The current occupancy value of quotas is calculated from the size of a user's files and the associated file attributes. When files are created or enlarged, the system performs monitoring to ensure that the maximum values are not exceeded unless special conditions (public space excess) which permit this apply.

A user may exhaust a quota which contains a high requirement subquota, even though the subquota itself is not exhausted. Figure 33 makes this clear. In this example, the TOTAL-SPACE quota is completely exhausted but not the S0-LEVEL-SPACE quota which it contains. In such situations, it is not possible to create new files or extend existing files since this would result in the maximum value for the TOTAL-SPACE quota being exceeded. No restrictions apply to the migration of files from the processing level to a background level since in this case the occupancy value for the TOTAL-SPACE quota remains unchanged, the occupancy value of the S0-LEVEL-SPACE quota is reduced and the occupancy values of the subquotas which it contains are either reduced or remain the same. In the situation illustrated in the example, it is also possible to recall files from a background level into the processing level, although this is only possible as log as the maximum values of the S0-LEVEL-SPACE quota and its subquota are not exceeded. For example, the RECALL of a file with the attribute PERFORMANCE=*VERY-HIGH would fail since the VERY-HIGH-PERF-SPACE quota is already exhausted.

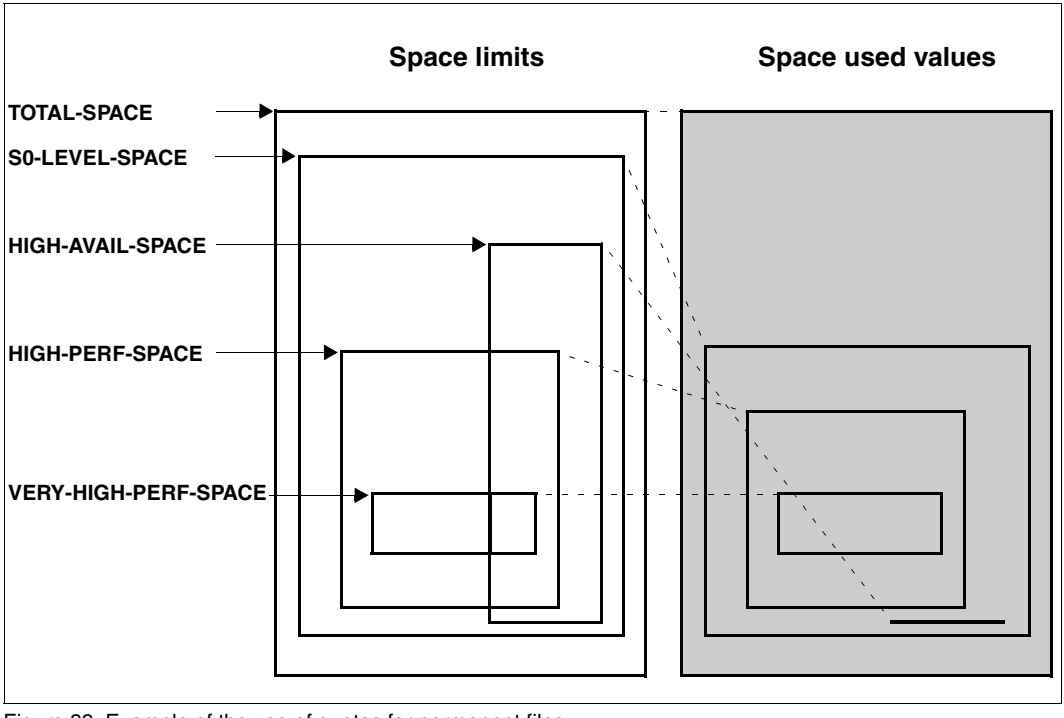
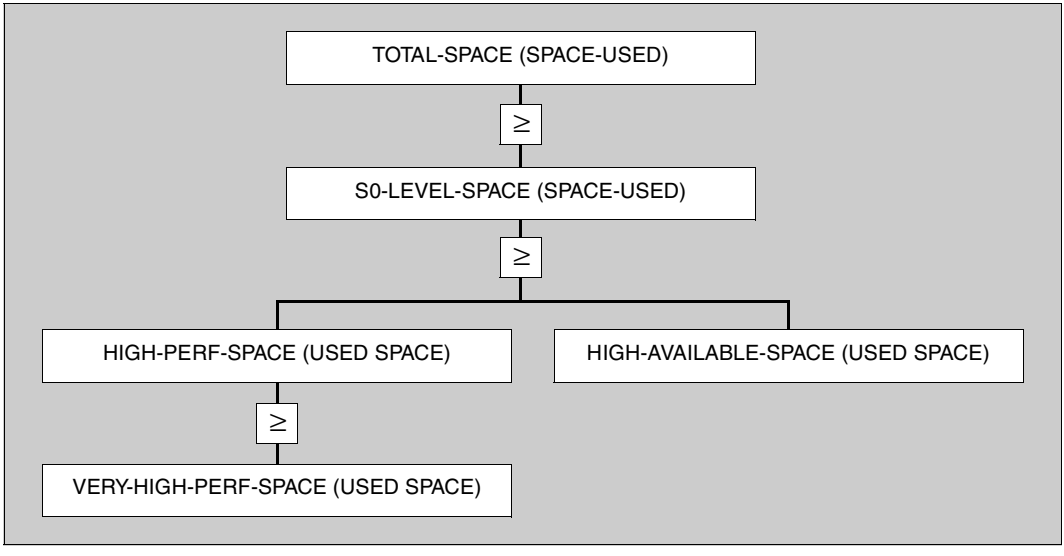
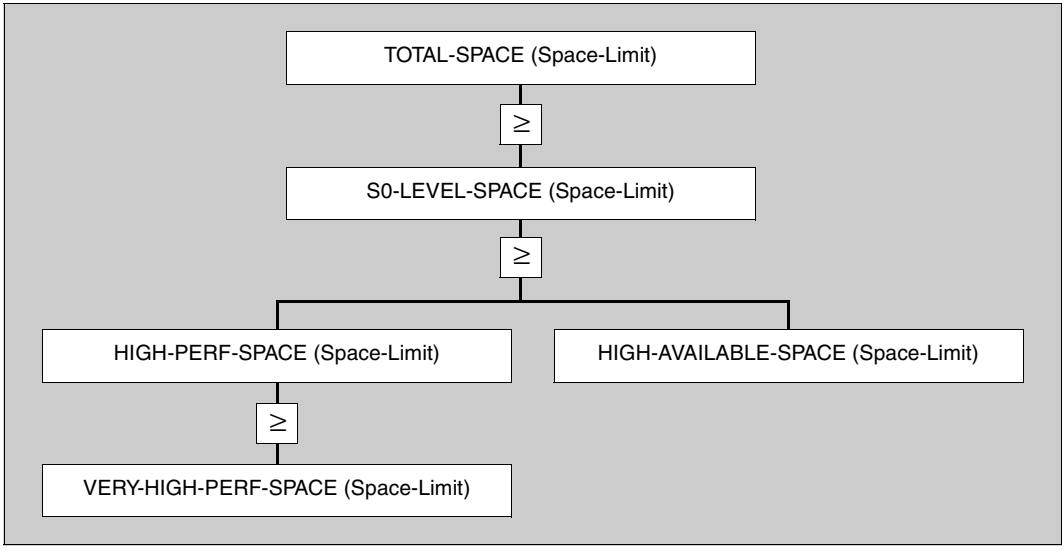


Figure 33: Example of the use of quotas for permanent files

The relations between quotas and subquotas are expressed in terms of ‘greater than/less than or equal to’ relationships which reflect the space occupied by a user:



The maximum values for quotas are only of use if they represent similar relationships:



When setting the maximum values for the quotas, systems support must pay attention to these relationships. The system monitors whether they are respected. The command interface supports the systems support staff when setting maximum values. Adherence to the rules is facilitated by procedures that determine the default values.

11.1.2 Quotas for temporary files and work files in SM pubsets

Temporary files and work files cannot be migrated to background levels and must not have the file attribute AVAILABILITY=*HIGH. This means that there is no need to distinguish between quotas for TOTAL-SPACE and S0-LEVEL-SPACE and the HIGH-AVAILABLE-SPACE quota is not required. This means that the quota structure for temporary files and WORK files is simpler than that of permanent files.

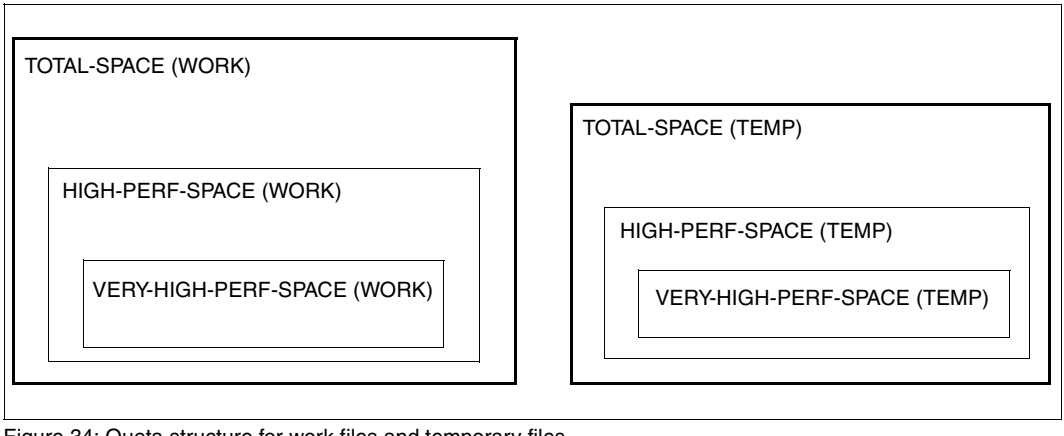


Figure 34: Quota structure for work files and temporary files

The maximum values for quotas for temporary files and work files must conform to the following relationship:

TOTAL-SPACE ≥ HIGH-PERF-SPACE ≥ VERY-HIGH-PERF-SPACE

11.1.3 Command interface for quota administration

The ADD-USER/MODIFY-USER-ATTRIBUTES commands used with SF pubsets to specify space limits and conditions can also be used for SM pubsets as long as there is no need for differentiated administration of the user quotas. Differentiation going beyond this is enabled by the MODIFY-USER-PUBSET-ATTRIBUTES command. This can also be used for SF pubsets, but only for the quotas defined for it. The ADD-USER/MODIFY-ATTRIBUTES and MODIFY-USER-PUBSET-ATTRIBUTES commands thus offer alternative ways of managing the pubset quotas for both SF and SM pubsets. They can also be used alternatively. This generally does not make sense, but it can be useful during a change from SF pubsets to SM pubsets. Different terms can be used for the different quotas in these commands. They can be mapped to each other as follows:

S0-LEVEL-SPACE for permanent files	in **MODIFY-USER-PUBSET-ATTRIBUTES**
refers to the same quota as	
PUBLIC-SPACE	in **ADD-USER/MODIFY-USER-ATTRIBUTES**

TOTAL-SPACE for temporary files in	in **MODIFY-USER-PUBSET-ATTRIBUTES**
refers to the same quota as	
TEMP-SPACE	in **ADD-USER/MODIFY-USER-ATTRIBUTES**

11.1.4 Allocating pubset space quota at user group level

It is possible to define user groups both in SM and SF pubsets. The administrator of a group defined in a pubset is authorized to define the space limits for the members of this group in the pubset. The limit values which a group administrator may allocate to individual group members are themselves restricted by higher-order limitations at group level. The space limits at group level are set by systems support during group definition using the ADD-USER-GROUP command. These limits can later be modified using the MODIFY-USER-GROUP command. The SHOW-USER-GROUP command displays the values that have been set.

11.2 DMS-TUNING-RESOURCES and user quotas

Both user quotas and DMS-TUNING-RESOURCES authorization provide the systems support staff with ways of limiting the creation of high-performance files by individual users (i.e. there are overlaps between the two functions). However, DMS-TUNING-RESOURCES offers options (automatic modification of user specifications in the assignment of the static performance attribute for files, inclusion in the determination of dynamic performance) for user quotas offer no equivalent. On the other hand, only user quotas permit differentiated gradations for the space limits of high-performance files. Because of the different functionalities and in order to preserve compatibility when there is a change from SF pubsets to SM pubsets, both mechanisms are supported at the same time on SM pubsets. To ensure that they complement each other well, however, the following recommendations should be taken into account by systems support staff:

1. If systems support staff have used DMS-TUNING-RESOURCES up to now for SF pubsets as a means of resource monitoring, and if they regard the scope offered as adequate, they can completely ignore the space limits for HIGH-PERF-SPACE and VERY-HIGH-PERF-SPACE when changing from SF pubsets to SM pubsets.
2. If space limits are used for HIGH-PERF-SPACE and VERY-HIGH-PERF-SPACE in order to achieve better differentiation, the following should be taken into account:
   a) Explicit specification of a HIGH-PERF-SPACE limit for work files, temporary files or permanent files generally only makes sense when the following also applies: DMS-TUNING-RESOURCES =*CONCURRENT-USE or *EXCLUSIVE-USE. Otherwise, the limit settings cannot be used.
   b) In the same way, explicit specification of a VERY-HIGH-PERF-SPACE limit for work files, temporary files or permanent files generally only makes sense when DMS-TUNING-RESOURCES =* EXCLUSIVE-USE is specified at the same time.

DMS-TUNING-RESOURCES authorization can be set for SF pubsets and SM pubsets by means of the ADD-USER and MODIFY-USER-ATTRIBUTES commands or the MODIFY-USER-PUBSET-ATTRIBUTES command. The value set is displayed by the SHOW-USER-ATTRIBUTES command.

11.3 Authorization to perform physical allocation

Users authorized to perform physical allocation are able to assert considerable influence on allocation strategies within an SM pubset. It must therefore be possible to restrict the ability to use physical allocation. The TSOS privilege generally authorizes users to perform physical allocation irrespective of their user ID. In order to allow normal users to perform physical allocations in SM pubsets, systems support must use the ADD-USER and MODIFY-USER-PUBSET-ATTRIBUTES commands to grant the corresponding user IDs the pubset-specific right to physically allocate files under this ID. Physical allocation is generally permitted for private disks and the disks of work volume sets. When a user ID is set up in a pubset, the authorization to perform physical allocations is initialized with the value ‘not authorized’. Users can obtain information about their authorization to perform physical allocation by means of the SHOW-USER-ATTRIBUTES command.

Physically allocated files are also subject to quota monitoring. In order to prevent users from evading the requirements of quota monitoring by physically allocating files in high value volume sets without, however, allocating them the corresponding file attributes (example: file with AVAILABILITY=*STD is physically allocated to a volume set with AVAILABILITY=*HIGH and therefore implicitly benefits from high availability), the file attributes are raised if necessary (in our example: AVAILABILITY=*HIGH). This problem is discussed in greater detail in section “Physical allocation” on page 193.

11.4 GUARDS profiles for storage classes and HSMS management classes

Storage classes and HSMS management classes can be protected using GUARDS profiles. In storage classes to which no volume set lists have been allocated this is not a purposeful method of resource protection since users can create files with the corresponding individual attributes at any time provided that they possess adequate user quotas.

Storage classes which are associated with volume set lists represent the principal area of application for GUARDS protection for storage classes. Such storage classes make it possible to reserve the use of certain volume sets to users with specific GUARDS authorizations except in special situations (e.g. when storage bottlenecks occur). To do this, these volume sets must be entered in volume set lists which are associated with storage classes which are protected by GUARDS profiles. To simplify matters in such cases, we speak of GUARDS-protected volume sets.

However, it should be noted that the volume set configuration of an SM pubset must meet certain requirements before GUARDS protection can be effective for volume sets. If a user uses direct attribution to specify a combination of file attributes which are supported at the generally accessible volume sets (i.e. volume sets which do not belong to any volume set list and for which no allocation locks are set) then volume sets which belong to volume set lists are also used by the system when meeting the user's requirements. Thus users who do not possess the corresponding GUARDS authorization can also create files in GUARDS-protected volume sets. GUARDS protection for volume sets can therefore only be guaranteed if every combination of location-relevant file attributes, which can be supported by a volume set which belongs to a volume set list is also be supported by a generally accessible volume set. Systems support can use the SHOW-PUBSET-FILE-SERVICES command to display whether or not an SM pubset meets this condition. If this is not the case, systems support must assume that users will behave correctly.

Related publications

The manuals are available as online manuals, see *http://manuals.fujitsu-siemens.com*, or in printed form which must be paid and ordered separately at *http://FSC-manualshop.com*.

[1] **BS2000/OSD-BC V6.0**
Utility Routines
User Guide

Target group
The manual addresses both nonprivileged users and systems support.
Contents
The manual describes the utilities:
DPAGE V15.0A, INIT V15.0A, JMP V2.0A, JMU V14.0A, LMSCONV V3.3B, PAMCONV V12.0A, PASSWORD V15.0A, PVSREN V2.0A, RMS V7.1E, SCDM V6.0A, SMPGEN V15.0A, SPCCNTRL V15.0A, TPCOMP2 V15.0A, VOLIN V15.0A.
Order number
U4303-J-Z125-8-76

[2] **SECOS V5.0** (BS2000/OSD)
Security Control System
User Guide

Target group
– BS2000 system administrators
– BS2000 users working with extended access protection for files
Contents
Capabilities and application of the functional units:
– SRPM (System Resources and Privileges Management)
– SECOS-KRB (Authentication with Kerberos)
– SRPMSSO (Single Sign On)
– GUARDS (Generally Usable Access Control Administration System)
– GUARDDEF (Default Protection)
– GUARDCOO (Co-owner Protection)
– SAT (Security Audit Trail).
Order number
U5605-J-Z125-8-76

[3] **BS2000/OSD-BC V6.0**
System Installation
User Guide

Target group
This manual is intended for BS2000/OSD system administration.
Contents
The manual describes the generation of the hardware configuration with IOGEN and the following installation services: disk organization in pubsets, the installation of volumes using the SIR utility routine, and the IOCFCOPY subsystem.
Order number
U2505-J-Z125-16-76

[4] **HSMS V7.0A / HSMS-SV V7.0A** (BS2000/OSD)
Hierarchical Storage Management System
Volume 1: Functions, Management and Installation
User Guide

Target group
– BS2000/OSD users
– BS2000/OSD system administrators
– HSMS administrators
Contents
– Description of the data saving, archival, migration and data transfer functions
– HSMS management, invocation, execution and installation
– HSMS messages
Order number
U6043-J-Z125-11-76

[5] **BS2000/OSD-BC V6.0**
Introductory Guide to Systems Support
User Guide

Target group
This manual is addressed to BS2000/OSD systems support staff and operators.
Contents
The manual covers the following topics relating to the management and monitoring of the BS2000/OSD basic configuration: system initialization, parameter service, job and task control, assignment of privileges, accounting, memory/device/system time/user/file/pubset management and operator functions.
Order number
U2417-J-Z125-15-76

[6] **DRV V3.0A** (BS2000/OSD)
Dual Recording by Volume
User Guide

Target group
This manual is intended for systems support, operators and nonprivileged users.
Contents
The manual describes the recording method DRV (Dual Recording by Volume) which allows data to be stored in duplicate on two disks. The use of DRV in the computer center increases the availability of the data stored on disk. The manual contains descriptions of all the procedures, commands and macro extensions required for the installation, use, control and monitoring of DRV. The use of DRV to migrate disks is also described in detail.
Order number
U6515-J-Z125-4-76

[7] **BS2000/OSD-BC V5.0**
Files and Volumes Larger than 32 GB
User Guide

Target group
This manual is intended for
– programmers in BS2000/OSD
– BS2000/OSD systems support staff
Contents
The manual is designed to simplify migration to configurations with large objects (files and volumes larger than or equal to 32 GB). It describes:
– the theoretical principles for the introduction of large objects
– the role of systems support staff in the introduction and maintenance of large objects (areas of activity: system initialization, parameter service, file and pubset management), including the command interfaces
– the extensions to the nonprivileged command interfaces and the Assembler macro interfaces with regard to large files
– the theoretical sequence of steps involved in converting Assembler programs and programs in higher programming languages

The manual contains a list showing the executability of the programs of SWK BS2000/OSD V5.0 with regard to large files.
Order number
U41253-J-Z125-1-76

[8] **SHC-OSD V4.0A** (BS2000/OSD)
Symmetrix Host Component
User Guide

Target group
This manual is intended for systems support staff and service technicians.
Contents
The SHC-OSD software product is the BS2000 host component for Symmetrix systems. It provides information services and commands for controlling the Symmetrix functions SRDF (Symmetrix Remote Data Facility) and TimeFinder (Symmetrix Multi Mirror Facility).
Brief overview of contents:
– Symmetrix in BS2000/OSD
– Software component SHC-OSD
– SRDF: downtime scenarios and measures for ensuring continued operation
– TimeFinder: working with multi-mirror pairs
– commands (listed in alphabetical order)
– Assembler and C program interface
Order number
U41000-J-Z125-4-76

[9] **SPACEOPT V3.0** (BS2000/OSD)
Disk Optimization and Reorganization
User Guide

Target group
This manual is addressed to systems support staff.
Contents
The SPACEOPT subsystem is used to optimize reorganization of the volumes on a pubset. Reorganization makes available large areas of contiguous free storage space and reduces the number of file extents. Reorganization can take place on a volume- or file-related basis. With the aid of SPACEOPT, users can assess the occupancy and fragmentation status of the volumes of a pubset as well as monitor the SPACEOPT jobs. For ISAM files it is also possible to determine the number of free pages that can be produced during processing.
Overview of contents:
– SPACEOPT jobs and job options
– assessment of the volume status
– installing, starting, terminating, managing
– commands in alphabetical order
Order number
U41073-J-Z125-3-76

[10] **BS2000/OSD-BC V6.0**
Commands, Volumes 1 - 5
User Guide

Target group
This manual is addressed to nonprivileged users and systems support staff.
Contents
Volumes 1 through 5 contain the BS2000/OSD commands ADD-... to WRITE-... (basic configuration and selected products) with the functionality for all privileges. The command and operand functions are described in detail, supported by examples to aid understanding. An introductory overview provides information on all the commands described in Volumes 1 through 5.
The Appendix of Volume 1 includes information on command input, conditional job variable expressions, system files, job switches, and device and volume types.
The Appendix of Volumes 4 and 5 contains an overview of the output columns of the SHOW commands of the component NDM. The Appendix of Volume 5 contains additionally an overview of all START commands.
There is a comprehensive index covering all entries for Volumes 1 through 5.
Order numbers
U2338-J-Z125-16-76 Commands, Volume 1, A – C
U41074-J-Z125-3-76 Commands, Volume 2, D – MOD-JO
U21070-J-Z125-6-76 Commands, Volume 3, MOD-JV – R
U41075-J-Z125-3-76 Commands, Volume 4, S – SH-PRI
U23164-J-Z125-5-76 Commands, Volume 5, SH-PUB – Z

[11] **BS2000/OSD-BC V6.0**
Commands, Volume 6, Output in S Variables and SDF-P-BASYS
User Guide

Target group
This manual is addressed to programmers and users who write procedures.
Contents
Volume 6 contains tables of all S variables that are supplied with values by the SHOW commands in conjunction with structured output. Further chapters deal with:
– introduction to working with S variables
– SDF-P-BASYS V2.3A
Order number
U23165-J-Z125-5-76

[12] **BS2000/OSD**
Softbooks English
CD-ROM

Target group
BS2000/OSD users
Contents
The CD-ROM "BS2000/OSD SoftBooks English" contains almost all of the English manuals and README files for the BS2000 system software of the latest BS2000/OSD version and also of the previous versions, including the manuals listed here.
These Softbooks can also be found in the Internet on our manual server. You can browse in any of these manuals or download the entire manual.
Order number
U26175-J8-Z125-1-76
Internet address
http://manuals.fujitsu-siemens.com

Index

Contents

BS2000/OSD-BC V6.0
System-Managed Storage

User Guide

Target group

This manual is intended for systems support staff.

Contents

The manual provides an overview of the functions available in BS2000/OSD and the products of the basic configuration for supporting system-managed storage. The most important of these is the SM pubset, whose structure, characteristics and application options are described in the manual. A variety of scenarios are presented, showing how the SMS functionality can best be used and explaining the transition from SF pubsets to SM pubsets.

Edition: December 2004

File: sms.pdf

This manual was produced by
cognitas. Gesellschaft für Technik-Dokumentation mbH
www.cognitas.de

Fujitsu Siemens computers GmbH
User Documentation
81730 Munich
Germany

Fax: (++49) 700 / 372 00000

e-mail: manuals@fujitsu-siemens.com
http://manuals.fujitsu-siemens.com

Comments
Suggestions
Corrections

Submitted by

Comments on BS2000/OSD-BC V6.0
System Managed Storage

Information on this document

On April 1, 2009, Fujitsu became the sole owner of Fujitsu Siemens Computers. This new subsidiary of Fujitsu has been renamed Fujitsu Technology Solutions.

This document from the document archive refers to a product version which was released a considerable time ago or which is no longer marketed.

Please note that all company references and copyrights in this document have been legally transferred to Fujitsu Technology Solutions.

Contact and support addresses will now be offered by Fujitsu Technology Solutions and have the format *...@ts.fujitsu.com*.

The Internet pages of Fujitsu Technology Solutions are available at *http://ts.fujitsu.com/*...
and the user documentation at *http://manuals.ts.fujitsu.com*.

Hinweise zum vorliegenden Dokument

Zum 1. April 2009 ist Fujitsu Siemens Computers in den alleinigen Besitz von Fujitsu übergegangen. Diese neue Tochtergesellschaft von Fujitsu trägt seitdem den Namen Fujitsu Technology Solutions.

Das vorliegende Dokument aus dem Dokumentenarchiv bezieht sich auf eine bereits vor längerer Zeit freigegebene oder nicht mehr im Vertrieb befindliche Produktversion.

Bitte beachten Sie, dass alle Firmenbezüge und Copyrights im vorliegenden Dokument rechtlich auf Fujitsu Technology Solutions übergegangen sind.

Kontakt- und Supportadressen werden nun von Fujitsu Technology Solutions angeboten und haben die Form *...@ts.fujitsu.com*.

Die Internetseiten von Fujitsu Technology Solutions finden Sie unter *http://de.ts.fujitsu.com/*..., und unter *http://manuals.ts.fujitsu.com* finden Sie die Benutzerdokumentation.