1 Preface

PCS (Performance Control System) provides you with a means of controlling performance and workload that is easy to use and ensures optimum functionality. PCS monitors and manipulates the various loads for complex mixed-mode operation with an eye to streamlining performance to suit customer requirements. At short intervals PCS adapts the system parameters such that the prescribed optimization strategy (regarding response time and throughput) and the most effective operating level for this strategy can be maintained at all times.

The more heterogeneous the load, the larger the number of simultaneous tasks and the more varied the resource requirements of the individual loads, the more efficient is PCS. Response times and throughput are noticeably better and more stable when using PCS. Predefined default parameter sets make it easy to use.

1.1 Brief description of the product

PCS enables you:

– to choose an optimization strategy (response time/throughput)

– to assign categories to the various loads according to the response time and throughput

– to define threshold values for the system capacity required per transaction in these categories

– to define service quotas and dilation values for the distribution of computing power over these categories.

Within this framework, **PCS** can:

– independently recognize situations in which performance deviates from the prescribed criteria

– automatically take appropriate action to restore performance to the customary high level.

PCS is operated is interactive mode using a screen-driven user interface.
In batch and procedure modes, commands can be given in SDF format.
PCS V2.7A can only be operated in conjunction with BS2000/OSD-BC V6.0.

1.2 Target group

This manual is intended for BS2000 systems support who want to optimize the setup and operation of a computer system using PCS.

Readers should have a thorough knowledge of BS2000 and be familiar with all its components. You will find more detailed information in the manual "Introductory Guide to Systems Support" [5].

You should also be familiar with the BS2000 command language SDF (System Dialog Facility). For more detailed information, refer to the manual "Introductory Guide to the SDF Dialog Interface" [7].

1.3 Structure of the manual

The manual covers the following topics:

Chapter 1, "Preface", contains a short description of PCS V2.7A and describes the changes made since PCS V2.3A.

Chapter 2, "Basic principles of PCS", tells you how to use PCS. The concepts and strategies embodied by the product are explained here. Recommendations and useful information on the application and operation of PCS are also provided.

Chapter 3, "Basic concepts", lists and explains the key terms, in particular work, performance, capacity, dilation, load unit and category.

Chapter 4, "Introduction to the PCS parameter concept", describes the effects of the category-specific parameters, the global parameters, the task priorities and service allocation.

Chapter 5, "Parameter sets (OPTIONs)", deals with the standard OPTIONs. Also described are the way in which standard OPTIONs are changed and which requirements must be fulfilled before using the installation-specific and load-specific OPTIONs.

In chapter 6, "PCS definition file (PPF)", the structure of the PPF file is presented. The PCS parameter set (OPTION) and the category parameter set (CATEGORY) are also described.

Chapter 7, "PCSDEFINE utility routine", contains an overview of this utility routine, describes how it is started and terminated, and explains which menus and commands are used during a PCSDEFINE run.

Chapter 8, "PCS administration", contains an overview of PCS, describes how it is installed, started and terminated, and explains which commands are used during a PCS run.

Chapter 9, "Messages", lists all the PCS messages, together with their meanings and any actions that may need to be taken.

Chapter 10, "Appendix", contains explanations of service units and of the REQUEST-DELAY-MAX, DURATION and SERVICE-QUOTA-MAX parameters. Also covered in the appendix are the determination of monitored variables, the effect of the task priority, the procedure for CREATE-PCS-OPTION, and a short description of the commands relevant to PCS and of the PCSDEFINE statements. Finally, the SDF syntax used is presented.

The full title of every publication which is referenced in the text by a number is printed next to the corresponding number in the chapter entitled "Related publications". Information on how to order these publications is included at the end of this chapter.

The index will enable you to quickly locate those pages in the manual dealing with particular problems and issues.

1.4 README file

Information on functional changes and additions to the current product version described in this manual can be found in the product-specific README file.
You will find it under the file name `SYSRME.PCS.027.E`.
The user ID under which the README file is cataloged can be obtained from systems support. With IMON you can also find out the file name using the following command:

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

You can view the README file using the `/SHOW-FILE` command or an editor, and print it out on a standard printer using the following command:

```
/PRINT-DOCUMENT $userid.SYSRME.PCS.027.E,LINE-SPACING=*BY-EBCDIC-CONTR
```

1.5 Changes since the last edition

This manual is the successor to the PCS V2.3A manual and contains the following changes:

The introduction of the new command MOVE-TASK-TO-CATEGORY in OSD/BC V6.0 enables systems support to initiate a manual change of category, for example if enhanced support of this task is required.

This option is described in section “Category” on page 15 and in section “Automatic category changeover DURATION/NEXT-CATEGORY” on page 26“.

2 Basic principles of PCS

PCS (Performance Control System) for the BS2000/OSD-BC operating system assists systems support staff in fine-tuning the computer system and optimizing its performance. It enables the capacity of a computer system to be shared among individual task categories and tasks according to user requirements.

2.1 Function of PCS

It is not necessary to be completely familiar with all PCS functions and parameters in order to make use of PCS, i.e. less experienced users can learn to use it gradually at their own speed.

Using control values specified by systems support, PCS is able to carry out performance optimization measures largely automatically during system operation.

Systems support achieves this by

- selecting an optimization strategy (response time/throughput)
- assigning important load components to the appropriate categories in accordance with response time and throughput requirements
- defining threshold values for the service used per transaction in these categories
- specifying service quotas and dilation values for the allocation of computing power to these categories.

On the basis of these specifications, **PCS** can

– recognize situations where performance deviates from specified criteria

– automatically take appropriate action to restore system performance to the customary high level.

It is important to note that **PCS**

– can only function optimally with an appropriate standard set of parameters or appropriate parameter settings defined by systems support

– cannot compensate for an inadequate hardware configuration relative to the load to be processed; it is only able to displace parts of the load by preempting or disallowing them.

In order to achieve effective response times, the system's resources must not be too heavily burdened. This means that throughput will not be optimal.

Conversely, outright throughput optimization can increase response times significantly. For this reason PCS offers a range of performance control capabilities to allow selection of various levels of performance between these two extremes.

The range of control facilities offered by PCS comprises a large number of "soft" settings that are especially well-suited to variable load compositions, and several "hard" settings that systematically optimize response times or throughput; the latter should be used with care (the effectiveness of a setting should be checked by taking measurements).

2.2 Concepts and strategies

PCS works together with the job management and task management (PRIOR) systems as a load and utilization regulator. It monitors the following during system operation:

– the service quota used by tasks in the categories under BS2000/OSD-BC (the "load")

– the utilization of the CPU, main memory and I/O system.

If these values do not correspond to the nominal values specified by systems support (PCS parameters), PCS corrects the amount of capacity allocated by changing adjustment values in BS2000 (e.g. task priority, MIN-MPL and MAX-MPL values for the categories).

PCS calculates the mean of its control values over intervals of approximately 10 seconds and alters the adjustment values according to the load measured over that period of time.

The determination of the service quotas is based mainly on the calculation of category-specific dilation as compared to the specified target values.

Two measures are available here to improve efficiency:

1. Determination of synchronous wait times for the job submitter, even though the job is executed asynchronously by the server. The resultant dilation value is correspondingly greater. This value could in turn lead (at the category level) to a larger service quota. In this context, PCS supplies a programming interface, which should be used by the instances in question (at the moment this refers to DAB).

2. Determination of the average category input/output time through the simultaneous use of *open*SM2.

   Usually, an input/output time of 20 milliseconds is currently assumed and used for the 'dilation calculation' category.

   If the actual input/output time lies (far) below this assumed value, this indicates that this category is being inadequately serviced, possibly very much so.

   When *open*SM2 is being run together with a system statistics facility for all disk devices, this value will be obtained periodically by PCS and added to the calculated dilation value, thereby facilitating more efficient service planning.

   The systems support can include or exclude *open*SM2 at any time. Where appropriate, PCS will use the default value. See the /SHOW-PCS-OPTION command for more information on this subject.

Response time optimization

PCS offers graduated control of response time using nominal values that have either a qualitative or quantitative effect on system performance.

Here are a few examples:

– Processing steps that do not require a great deal of system resources (e.g. edit commands) can be given preference.

– Processing steps that require a large amount of system resources (e.g. compilation) can be given a low priority.

– Response time optimization can be carried out globally for the system as a whole or for individual categories.

Throughput optimization

– Like response time optimization, throughput optimization can be carried out globally for the whole system or for specific categories.

– Where global throughput optimization is specified, PCS influences the composition of the active tasks so that maximum overall performance can be achieved, i.e. the utilization of resources is as equitable as possible.

In both operation modes (response time-oriented or throughput-oriented), PCS tries to meet all user requirements by

– allocating computing power according to specifications (e.g. category-specific service quotas: BATCH 20%, DIALOG 50%)

– overload control:

  only tasks that meet the desired response time requirements or which allow optimum system performance will be allowed.

2.3 Recommended uses of PCS

PCS is particularly useful for the following:

- Controlling a load whose composition shows considerable variation over time (e.g. a mixed BATCH/DIALOG load with a variable DIALOG proportion).
- Separating load components within a category that have very different resource requirements (e.g. real dialogs and batch-type dialogs in a DIALOG category).
- Extensive optimization of well-known loads or of parts thereof in terms of response time or throughput.

2.4 Notes on use

- PCS V2.7A can only ever be used in conjunction with BS2000/OSD-BC V6.0.
- PCS V2.7A and PCSDEFINE V2.7A can process (i.e. read or modify) option parameter files created under PCS V2.3A/PCSDEFINE V2.3A. It is thus possible to process NK2 and NK4 files.
  New parameter files created under PCSDEFINE V2.7A are created as NK4 files and can reside on any pubset or private disk supported by BS2000/OSD-BC V6.0.

2.5 Operating procedures

PCS provides **systems support** with an interface for defining the individual parameters of the PCS parameter set.
This interface allows the user to proceed in steps when specifying control parameters. It comprises the PCSDEFINE utility routine for creating, modifying and cataloging PCS parameter sets (the "offline" part) and several system commands for controlling and monitoring the active PCS (the "online" part).

The categories used for PCS operation must be matched with those defined for the job management program.

The **end user** can differentiate his performance needs by specifying the job class and priority as long as systems support has authorized him to do so. He cannot influence PCS directly.

PCS is operated by carrying out the following steps (see “Installing PCS” on page 120):

– Systems support defines a set of PCS parameters with the help of the PCSDEFINE utility routine (if the standard parameter set supplied does not already meet his needs).

– The operator starts PCS as a subsystem and assigns it a PCS parameter set.

– Systems support or the operator can display the current PCS parameters and the most important monitored variables on the console at any time using the /SHOW-PCS-OPTION command.

– Systems support or the operator can modify certain PCS parameters interactively during a system session, switch the parameter set or terminate PCS.

– Once PCS has been terminated, the normal BS2000/OSD-BC control functions take over.

– To determine the average input/output time under *open*SM2, the monitoring program for system statistics (for all disk devices) can be started or terminated at any time.

– With the help of the /SHOW-PCS-OPTION command, you can determine whether use of *open*SM2 is necessary, i.e. you can determine how much and how often the input/output times deviate from the default value.

– Installation information: see chapter “PCS administration” on page 119.

3 Basic concepts

When making an assessment of the work or the performance of a computer system, it is important at which level the system is being looked at:

– (end) user level

– system software level

– hardware level

At the **user level**, the terms transaction (TP and interactive applications) and job (batch) are used to denote a work unit.

The transaction rate (transactions per unit of time) and the throughput rate (number of jobs per unit of time) are used to describe performance. In addition, response time (TP and interactive applications) and turnaround time (batch) are also taken into account.

The work (throughput) and the ability to react (response time) of the resources of a computer system must be evaluated quantitatively before PCS can fulfill its function as a load regulator at the **system software level**.

Response time is evaluated using a factor called

DILATION

Work is evaluated using

SERVICE UNITs (SU),

which are a new feature introduced with PCS.

Service units provide the functional basis of PCS. It is not, however, necessary for systems support staff to get involved with the mechanics of precisely how PCS calculates service units in order to find a solution to simple or moderately complex tasks (see examples).

3.1 Work (SERVICE UNITs)

The unit of measurement for work is the SERVICE UNIT.

The work done by a computer system is realized by the utilization of the system's resources.

Service as PCS knows it is the weighted sum of the work carried out by the resources: processor (CPU), input/output (I/O) and main memory (MEMORY). This work is measured in service units:

SERVICE UNITs = SERVICE UNITs (CPU)
\+ SERVICE UNITs (I/O)
\+ SERVICE UNITs (MEMORY)

Short form: SU = CPU-SU + IO-SU + MEMORY-SU

The three SU components refer, in effect, to the amount of time each hardware resource is used and, in the case of MEMORY-SU, also to the amount of memory used.

In order to obtain the same service value for the same amount of work done on configurations with differing CPU speeds, a configuration-specific weighting factor is used.

The appendix contains a more detailed definition of service units and an explanation of how they are calculated, see section "Definition of service units (SU)" on page 149.

3.2 Performance (SERVICE RATE)

Performance is defined as the work carried out per unit of time.

The unit of measurement PCS uses to measure performance is the

SERVICE RATE.

Service rate is measured in service units per second. The service rate is calculated not only for the system as a whole but also for individual categories or tasks. What is important for PCS in its function as a load regulator is the proportion of service allocated to a category or task relative to other categories or tasks.

3.3 Capacity

The maximum service rate that can be achieved on a system is referred to as its capacity.

The real capacity depends, however, on how the system load is able to utilize the capacity of the CPU, I/O and the memory.

An inadequate configuration exists when

– response times in one or more response time-oriented categories are too long - not only in the short term but *continually* - even though batch-type load components are preempted by PCS.

– batch throughput cannot be handled efficiently over a *continuous* period.

3.4 Dilation (REQUEST-DELAY)

The concept of a "dilation factor" or "dilation" is of particular importance to response time optimization. This can best be illustrated by a couple of examples:

– If there is only a single task running on a system, this task will be given optimum service, i.e. no delays (dilations) occur due to wait states caused by resources being used by other tasks (except for the setting up of the working set in the address space). Runtime under optimum conditions is referred to as "single runtime". In this particular case it is identical with the "real" runtime.

  Generally speaking, the following holds true:

$$\text{dilation} = \frac{\text{real runtime}}{\text{single runtime}}$$

  In the case of a single task run, dilation = 1.

– Let us assume that there are two tasks running on the system, both of which use only the CPU and do not carry out any I/O operations. If both tasks are given equal service, the real runtime for each task will be double their single runtimes.

Dilation does not, however, necessarily increase proportionately with the increasing number of tasks. The decisive factor is the resource requirements of the competing tasks.

Dilation (REQUEST-DELAY) therefore represents a **measurement for the increase in response time or runtime in a multiprogramming mode**. Because the relation between runtimes cannot be measured, PCS uses an approximate value as the dilation factor. This value is given by the following relation:

$$\frac{\text{Time needed to receive a number of service units}}{\text{Time equivalent of the number of service units}}$$

Because of this approximation, a dilation factor < 1 can be found in certain cases (e.g. when the system has a very light workload). This has no effect on the way PCS functions.

REQUEST-DELAY is a parameter especially tailored to the regulatory function of PCS. It is therefore not exactly the same as the monitored value DILATION supplied by *open*SM2.

As there are no time equivalents for MEMORY-SUs, they are not included in the calculation of the dilation factor.

3.5 Load unit (REQUEST)

A load unit, referred to in the following as a request, is determined by the object making the request (task) and the length of time the request takes. The latter is defined in the normal way for TP and DIALOG tasks as the period of time from the input of a message until the corresponding output, and for batch tasks as the period of time from the start of a task until its termination.

From now on the common term "transaction" will also be used for TP and DIALOG tasks instead of "request" if there is no need for a clear distinction between the two.

3.6 Category

Experience shows that the desired distribution of an installation's capacity for a load is best achieved by dividing the requests and thus the tasks into different performance classes (categories).

Different service requirements may exist for each of the resulting categories of requests. A certain throughput might be desired for one category while a certain maximum response time is desired for a different category.

All tasks that have similar service requirements at a certain point in time will therefore be grouped together as a category.

This does not mean that a task has to belong to the same category for the total period of its existence (or processing of a request). Depending on its dynamic behavior, a changeover to a different category can take place (cf. section "Automatic category changeover DURATION/NEXT-CATEGORY" on page 26). This is especially the case when system behavior is being impaired by resources being monopolized (e.g. by long-running tasks) and a changeover to a category with lower service requirements is indicated.

The JOB-CLASS parameter allows the user to specify the category (if authorized to do so by systems support) in which the processing of each request is to begin. Additional categories specified by systems support for automatic category changeover cannot be accessed directly by the user.

The MOVE-TASK-TO-CATEGORY command enables systems support to modify the assignment of a task to a category if, for example, different (better) support of this task or a reduction of the load on a category is required - with or without using PCS. For more details please refer to the manuals "Introductory Guide to Systems Support" [5] and "Commands, Volumes 1 - 5" [4].

4 Introduction to the PCS parameter concept

A set of PCS parameters for controlling a PCS session is combined in an OPTION. More than one option may be set up, but only one option may be effective at one time.

All options are stored in a special file provided for this purpose. This file is created and the PCS parameters are defined by means of the PCSDEFINE utility (see chapter “PCS definition file (PPF)” on page 59“, and chapter “PCSDEFINE utility routine” on page 67).

A standard file, PPF (name: SYSSSI.PCS.027), is supplied to each user with PCS. This file contains three predefined options (parameter sets): STD#TP, STD#DIA, STD#BAT. Chapter “Parameter sets (OPTIONs)” on page 37“, contains information on when these options should best be used and which parameters they contain.

PCS can be started simply by entering the DSSM command

```
/START-SUBSYSTEM SUBSYSTEM-NAME=PCS,
                 SUBSYSTEM-PARAMETER=C'OPTION-NAME=name,
                 FILE-NAME=name'
```

Default value for OPTION-NAME=STDOPT
Default value for FILE-NAME=SYSSSI.PCS.027.

The FILE-NAME operand may be omitted as the file SYSSSI.PCS.027 is the default file name.

In order to make it easier for systems support to divide the workload into various categories and allocate performance accordingly when planning how to use PCS, the way in which the most important parameters work will be described in more detail.

The parameter set required for a PCS session contains

– global system PCS parameters (see section “Global parameters” on page 30, and section “Description of the PCS parameter set (OPTION)” on page 60).

– category-specific PCS parameters.

4.1 Category-specific parameters

The system contains the following:

– the four standard categories SYS, BATCH, DIALOG and TP

– additional categories (max. 12) defined by systems support as needed.

The parameters that may be used for these categories are explained in the following sections by means of a few examples.

4.1.1 Allocating system capacity SERVICE-QUOTA

SERVICE-QUOTA-MAX	This parameter defines the maximum percentage of the system's capacity that is to be reserved for the category.
Value range	(0, 100)
Default value	100
SERVICE-QUOTA-MIN	This parameter defines the minimum percentage of the system's capacity that is to be reserved for the category.
Value range	(0, SERVICE-QUOTA-MAX - 1)
Default value	0

Notes

– The SERVICE-QUOTA-MIN parameter has no effect for categories without a dilation range (see next section).

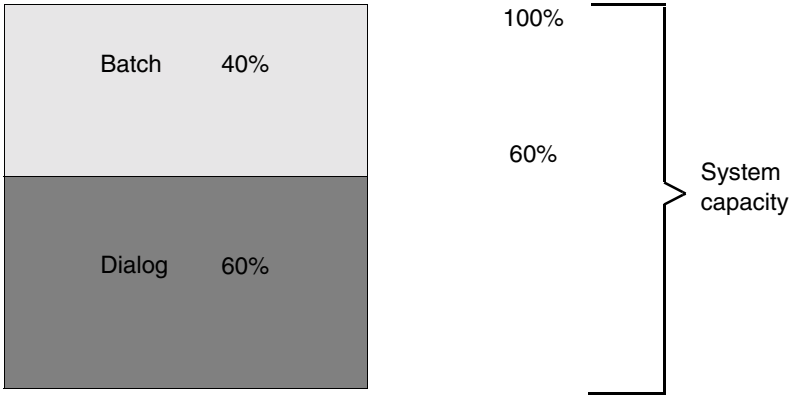
– If the sum of the SERVICE-QUOTA values exceeds 100%, PCS standardizes the corresponding ratios to 100%.

Example 1

Proportional allocation of a computer system's capacity for a dialog/batch application.

DIALOG category: SERVICE-QUOTA-MAX = 60

BATCH category: SERVICE-QUOTA-MAX = 40



This is the simplest example of a PCS option: the proportional allocation of system capacity to the standard categories (here only DIALOG and BATCH) without any additional constraints.

PCS performance control sets this ratio when both categories are functioning at full capacity or both categories taken individually need more service than their service quota entitles them to. A category is considered to be functioning at full capacity when it is fully utilizing its quota of the computer system's capacity. If a category is not using its complete quota, the remaining capacity is available to other categories for use. This ensures that no system capacity remains unused. When, for example, the DIALOG application temporarily has no requests, but BATCH tasks are waiting for resources, the BATCH category receives the share of capacity not being used by the DIALOG application.

4.1.2 Controlling response time behavior REQUEST-DELAY

The proportional allocation of capacity without further constraints is in many cases not enough. Load fluctuations in the course of a working day, a not uncommon feature of DIALOG and TP applications, may lead to unacceptable increases in response time. In this kind of situation, a flexible system response is required, such as temporary preemption of less important applications (e.g. BATCH-category tasks).

PCS achieves this via a PCS-specific dilation factor (request delay, see section “Dilation (REQUEST-DELAY)” on page 13). By monitoring this value and allocating corresponding SERVICE-QUOTA values as adjustment values, response time optimization can be achieved for a category.

REQUEST-DELAY-MAX	Defines the maximum delay for requests in the category and, along with SERVICE-QUOTA-MAX, defines the maximum share of the system’s capacity for the category.
Value range	(1, 100)
Default value	No monitoring of the upper limit for dilation (corresponds to the value 0)

The sum of the SERVICE-QUOTA-MAX values for categories with dilation parameters may exceed 100. This allows individual dilation categories to receive the required service quota if the category has a heavy workload and if other dilation categories are not utilizing maximum capacity.

Note

REQUEST-DELAY is a parameter specially matched to the regulatory function of PCS. It is therefore not directly comparable to the monitored value "DILATION" supplied by *open*SM2.

REQUEST-DELAY-MIN	Defines the minimum delay for requests in the category and together with SERVICE-QUOTA-MIN defines the minimum share of the system’s capacity the category should be allocated.
Value range	(1, REQUEST-DELAY-MAX)
Default value	1 if REQUEST-DELAY-MAX is specified; otherwise, no monitoring of the lower limit of the dilation factor (corresponds to the value 0).

Notes

– If REQUEST-DELAY-MAX is not specified, REQUEST-DELAY-MIN will be ignored.

– REQUEST-DELAY-MIN should always be less than REQUEST-DELAY-MAX. It is not advisable to specify REQUEST-DELAY-MIN = REQUEST-DELAY-MAX (unstable regulation).

Example 2

Response time optimization for a dialog/batch application.

The aim is to optimize response time by specifying a maximum dilation for dialog tasks so that there is no undesired increase in response time when the workload reaches a peak. The additional capacity needed to do this will be taken from the BATCH category. In order to do this, it is a good idea to distinguish between two operation situations:

– **Normal load:**

  This is when the ratio system load to system capacity is balanced. In the example, the dilation factor in the DIALOG category is to be less than 4 for normal loads.

– **Overload:**

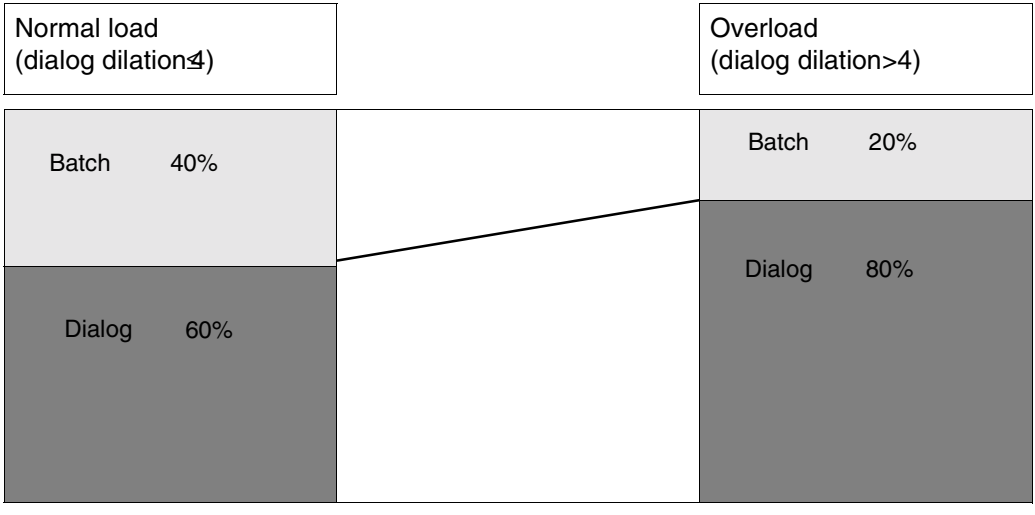
  This is when dilation in the DIALOG category is greater than 4 (due to peak dialog loads with the batch load remaining constant).

PCS parameter settings:

DIALOG category: SERVICE-QUOTA-MAX=80

REQUEST-DELAY-MAX=4

BATCH category: SERVICE-QUOTA-MAX=40



In the case of a **normal load**, capacity is allocated according to the current dialog dilation value (dilation range: 1 - 4).

For example: DIALOG category 60%

BATCH category 40%

In the case of an **overload** - when the dilation threshold value 4 is exceeded - PCS will reduce the service quota for the BATCH category while increasing the service quota for the DIALOG category to SERVICE-QUOTA-MAX.

The dialog quota may therefore increase to 80%. Given this choice of parameters, no further increase is possible even if the dialog dilation factor were to remain at a value greater than 4.

4.1.3 Parameter interaction SERVICE-QUOTA/REQUEST-DELAY

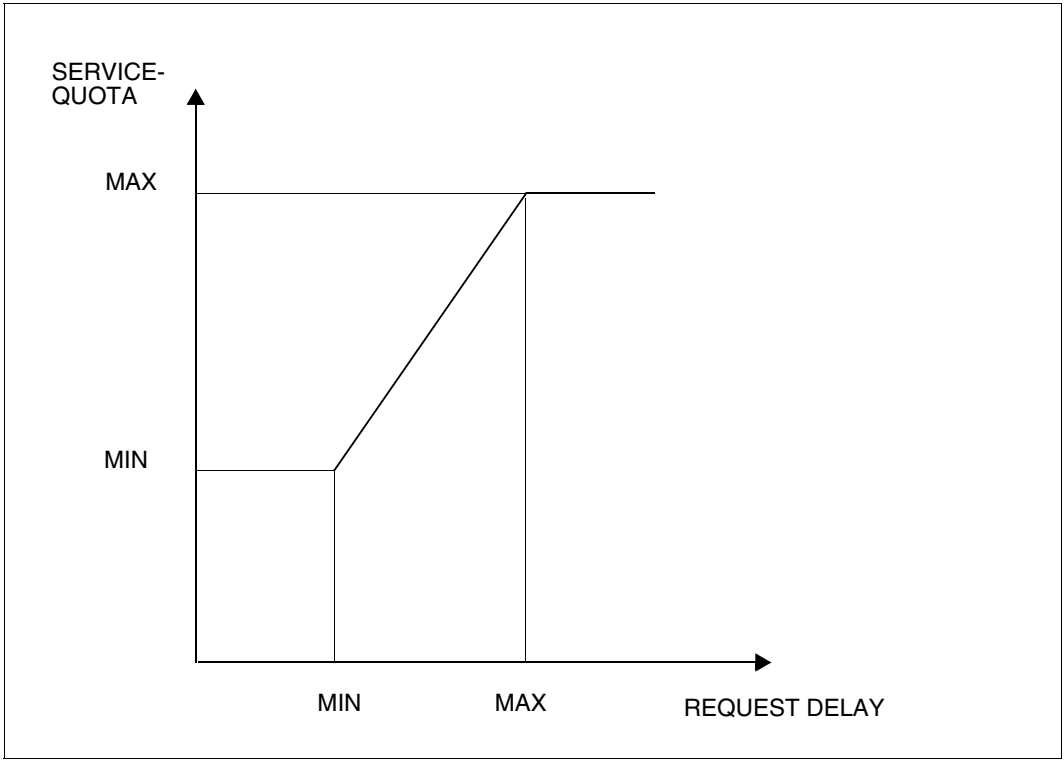
The allocation of service to categories with and without a dilation range is performed in the following order:

1. Categories for which a dilation range (using REQUEST-DELAY-MAX) has been specified will be allocated first.
2. Capacity remaining after categories *with* a dilation range have been allocated service is distributed among the categories *without* a dilation range according to the SERVICE-QUOTA-MAX value. SERVICE-QUOTA-MIN is ignored for these categories.

In order to take full advantage of PCS control mechanisms, there should always be at least one category without a dilation range.

Interpolation procedure:

Categories for which a dilation range has been specified are thus allocated a share of the system's capacity that varies with the load (as expressed by the dilation factor):



The actual dilation value determines the share of the system's capacity in the following way:

1. If the actual dilation factor is smaller than REQUEST-DELAY-MIN, the category is allocated the SERVICE-QUOTA-MIN value.
2. If the actual dilation factor is between REQUEST-DELAY-MIN and REQUEST-DELAY-MAX, allocation is carried out according to a linear interpolated value between SERVICE-QUOTA-MIN and SERVICE-QUOTA-MAX.
3. If the actual dilation factor is greater than REQUEST-DELAY-MAX, the category is allocated to the SERVICE-QUOTA-MAX value.

Example 3

Optimization of a dialog/batch application with load fluctuations.

In example 2 (cf. section "Controlling response time behavior REQUEST-DELAY" on page 20), response times were optimized by a stepwise reduction of the service allocated to the BATCH category when the dilation factor in the DIALOG category exceeded 4.

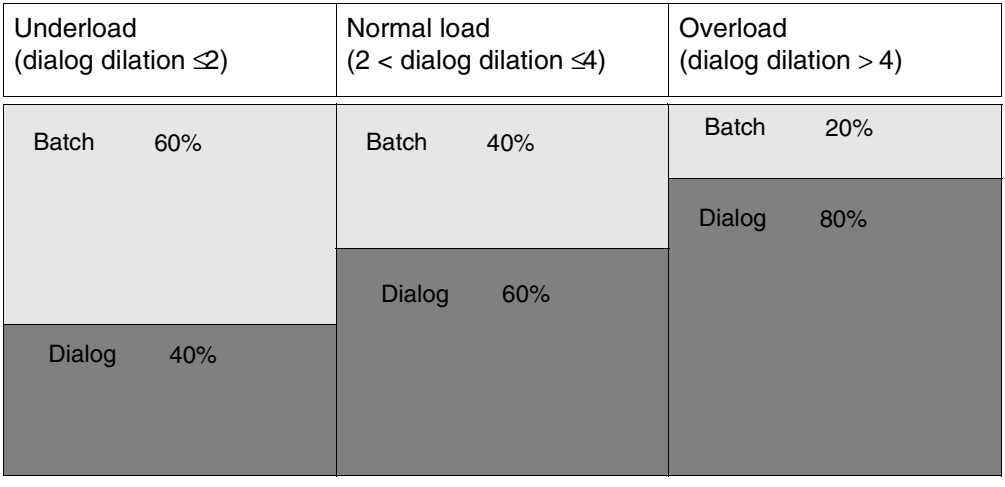
To prevent the dialog application being given better service than necessary at the expense of the BATCH application, it is good practice to introduce a minimum dilation value (REQUEST-DELAY-MIN) for the DIALOG category. If the dilation factor falls below this value, PCS automatically reduces the service allocated to the DIALOG category to the specified SERVICE-QUOTA-MIN value.

This means that the allocation of capacity is to be made dependent on the system load status in a more controlled manner. The load status at a given point in time is recorded by the DIALOG dilation factor (REQUEST-DELAY):

– **Underload:** dilation factor < 2
– **Normal load:** dilation factor 2 - 4
– **Overload:** dilation factor > 4

PCS parameter settings:

DIALOG category:	SERVICE-QUOTA: MIN = 40,	MAX = 80
	REQUEST-DELAY: MIN = 2,	MAX = 4
BATCH category:	SERVICE-QUOTA:	MAX = 60



In an **underload** situation, characterized by the actual dilation factor falling below the value specified for REQUEST-DELAY-MIN (2), service will be allocated to the DIALOG category according to the value specified for the SERVICE-QUOTA-MIN parameter (40). This means that PCS can meet response time requirements (assuming that resource requirements per request are not too high) while guaranteeing good batch throughput at the same time.

In a **normal load** situation, the DIALOG category will be assigned an interpolated value that lies between SERVICE-QUOTA-MIN and SERVICE-QUOTA-MAX, depending on the current dilation factor (dilation range: 2 - 4). This results in a reduced batch throughput.

In an **overload** situation, i.e. when the value for REQUEST-DELAY-MAX (4) is exceeded, batch throughput is reduced to 20%.

4.1.4 Automatic category changeover DURATION/NEXT-CATEGORY

The mechanism for automatically changing categories makes it possible to give requests with low resource requirements (e.g. edit commands) preference over requests which require a great deal of the system's resources (e.g. executing programs).

DURATION	The number of service units per request after which a category changeover takes place.
Value range	(1, 100000)
Default value	No automatic category changeover
NEXT-CATEGORY	Name of the successor category for automatic category changeover after <DURATION> service units. This parameter is only significant when used in conjunction with the DURATION parameter.

Rule of thumb for calculating the DURATION parameter:

DURATION = 2 * a * CPU-TIME (per request)

Examples of how to calculate the DURATION parameter can be found in the appendix under "Calculating REQUEST-DELAY-MAX and DURATION"; see section "Empirical values for the DURATION parameter" on page 153 for empirical values, and section "Definition of service units (SU)" on page 149 for the values for the factor "a".

Interworking with the manual change of category by the MOVE-TASK-TO-CATEGORY command:

MOVE-TASK-TO-CATEGORY always assigns tasks only to those target categories which are JMS categories and not auccessor categories.
After a task has been reassigned to a target category and when PCS is used, the task is again subject to the automatic change of category by PCS which is defined for the target category in the PCS parameters.

Note

It is not advisable to specify automatic category changeover for TP loads, because the interdependency that usually exists between the tasks means that even lengthy transactions have to be processed speedily in order to avoid LOCK situations.

Example 4

Response time optimization in a dialog/batch application, whereby the dialog requests have greatly varying resource requirements.

The following conditions are to be met:

– Interactive program development:

  a) Edit commands and other similar commands are to have response times < 3 seconds ("normal" dialog)

  b) There are to be **no** response time requirements for command procedure calls (/CALL-PROCEDURE...) and program execution (/START-PROGRAM...) ("long-running" dialogs).

– Batch mode:
  This part of the load, together with the dialog transactions listed under b), is to be allocated the capacity not utilized by the dialog transactions listed under a).

In order to assign these 3 load components to categories, another category in addition to the existing standard categories DIALOG and BATCH must be defined. This results in the following:

Load component	⟷	Category	
Normal dialog	⟷	DIALOG	(standard)
Long-running dialog	⟷	DIALOG1	(to be defined)
Batch mode	⟷	BATCH	(standard)

Additional categories may be given any name; it is not possible to change the names of the standard categories.

For this example, **automatic category changeover** should be used as follows:

To begin with, the standard category DIALOG accepts all incoming dialog transactions. If a certain predefined number of service units (DURATION parameter) is not sufficient for a specific transaction, it changes over to the category DIALOG1, where the number of service units is unlimited but the service rate is lower than in the original DIALOG category.

Automatic category changeover can be used to

– give short transactions that require few resources preference over other transactions

– push long-running transactions that require a great amount of resources into the background.

Multi-level category changeover is possible according to load classification. As a rule, a maximum of two successor categories is enough. In order to achieve the desired effect, each successor category must have a smaller service quota than the preceding category but a greater value for the DURATION parameter (the last successor category naturally has no DURATION).

An estimate of response times using the DURATION and REQUEST-DELAY-MAX parameters, and the resulting calculation of the DURATION parameter for the DIALOG category, can be found in the section “Calculating REQUEST-DELAY-MAX and DURATION (example)” on page 151. An estimate of the performance requirements for the DIALOG category is given in the section “Calculating the SERVICE-QUOTA-MAX parameter” on page 154.

PCS parameter settings:

Category	SERVICE-QUOTA		REQUEST-DELAY		DURATION	NEXT-CATEGORY
	MIN	MAX	MIN	MAX		
DIALOG	40	80	2	4	300	DIALOG1
DIALOG1	0	30	–	–	–	–
BATCH	0	30	–	–	–	–

This results in the following system behavior:

Underload (dialog dilation < 2)	Normal load (2 ≤dialog dilation ≤4)	Overload (dialog dilation > 4)
Batch 30%	Batch 20%	Batch 10%
		Dialog1 10%
Dialog1 30%	Dialog1 20%	Dialog 80%
Dialog 40%	Dialog 60%	

4.1.5 Response time/throughput optimization THROUGHPUT-QUOTA

In certain categories (e.g. BATCH) the primary consideration is maximization of throughput with good utilization of resources, rather than response time or runtime optimization.

THROUGHPUT-QUOTA This parameter specifies a percentage that determines the relation between response time and throughput optimization for the category.
The effect this has on the operational behavior of the category ranges from full response time optimization through to full throughput optimization:

Response time optimization full		Throughput optimization full
0%	50%	100%

Value range (0, 100)

Default value 0 (full response time optimization)

4.2 Global parameters

The global parameters define global load control (response time or throughput optimization, dilation limits). The more specific requirements of individual load classes can be met by using the category-specific parameters.

4.2.1 Global response time optimization REQUEST-DELAY-MAX

REQUEST-DELAY-MAX	specifies a threshold value for the maximum dilation for all of the tasks admitted by PCS that belong to a category with a dilation range. If the dilation calculated for these tasks exceeds the threshold value, the number of active tasks is reduced.
Value range	(1, 100)
Default value	5 + (THROUGHPUT-QUOTA)/20 (=6)

4.2.2 Global response time/throughput optimization THROUGHPUT-QUOTA

THROUGHPUT-QUOTA specifies a percentage value that determines the relation between response time and throughput optimization for the system.

The effect of the parameter is as follows:

– Either it optimizes the response time, which under certain conditions means that the utilization of resources is not optimal.

– Or it optimizes throughput, which under certain conditions means that response times and job runtimes can vary greatly.

Response time optimization			Throughput optimization	
hard	full		full	hard
0%	20%	50%	80%	100%

Default value 20 (full response time optimization)

Notes

– THROUGHPUT-QUOTA = 0 commits PCS to systematic response time optimization. If, for example, categories with a dilation range are not able to use the planned capacity, this capacity will **not** be made available to categories without a dilation range; this could result in an IDLE state.

– THROUGHPUT-QUOTA = 100 should only be used for pure BATCH loads that are to run at maximum throughput, but where time is not a critical factor for the individual jobs.

4.3 Task priorities

So far we have only looked at mechanisms for global and category-specific service allocation to categories. These take precedence under PCS. Task priorities have no effect outside the boundaries of the given category (exception: fixed priorities). However, the requests within a category can be graded by specifying a priority.

Although the same format is used for specifying priorities in PRIOR and PCS (RUN-PRIORITY or PRIORITY operand in the LOGON or ENTER command; authorization entry in the job class, see "Introductory Guide to Systems Support“ [3]), the effect differs in certain respects.

Variable priorities (priority range: 128-255)

By assigning variable priorities, the category service (defined by the SERVICE-QUOTA parameter) can be allocated according to the urgency of the individual tasks in the category. Thus it is possible, for example, to give better service to database tasks (UDS, SESAM) within the TP category than to data communication tasks (UTM).

In order to be effective in terms of the allocation of the category service, there must be a difference of at least 20 between priorities. As the value of the category-specific parameter THROUGHPUT-QUOTA increases, the impact of differences in priority is lessened (cf. section “Effect of task priorities” on page 157).

The entry THROUGHPUT-QUOTA = 100 is intended to achieve a throughput-oriented mode of operation. In this case external priority specifications are ignored both for global and for category-specific parameters.

Fixed priorities (priority range: 30-127)

The effect is the same as for PRIOR. Specifying fixed priorities greatly restricts PCS’s scope for making decisions and should only be used in special cases (e.g. for benchmark drivers), because tasks with fixed priorities are allocated computer capacity before all other categories.

4.4 Service allocation

In this section the mechanism for allocating service to categories will be explained in more detail and summarized.

In the previous sections, simple examples were used to illustrate that the allocation of service among competing categories is determined by the category parameters SERVICE-QUOTA and REQUEST-DELAY. Categories with the REQUEST-DELAY parameter take precedence over categories without the REQUEST-DELAY parameter.

In order to give preference to important applications (e.g. TP applications), other categories can also be prioritized. This is achieved by using **SERVICE-QUOTA-MAX = 100** in conjunction with REQUEST-DELAY-MAX.

This category is allocated service before all other categories; the amount depends on the dilation and can be up to 100%. A precondition for a total allocation of service is that the category is capable of handling it, i.e. that there are no memory or I/O bottlenecks, no LOCK situations, etc. within the category.

By specifying SERVICE-QUOTA-MAX = 100 in a category for which dilation has been specified, the PCS user is saying that if necessary (e.g. in the case of temporary TP peak loads), the application in all other categories is to be completely displaced.

In other words, PCS allocates service among different categories in the following order:

1. Categories with SERVICE-QUOTA-MAX = 100 and dilation range
2. Categories with SERVICE-QUOTA-MAX < 100 and dilation range
3. Categories without dilation range.

It is recommended that no more than one category with a dilation range and SERVICE-QUOTA = 100 be used per parameter set.

It should again be pointed out that the sum of service requests from tasks in the categories with dilation ranges should not exceed the capacity of the system even at peak load times. Otherwise, PCS is forced to reject any further requests. PCS cannot compensate for missing hardware resources.

Within this scheme, the categories compete for the available service as described in section “Allocating system capacity SERVICE-QUOTA” on page 18 and in section “Parameter interaction SERVICE-QUOTA/REQUEST-DELAY” on page 23. The service quota allocated exclusively to the categories must of course be reduced in line with the amount of service still available.

Special case "dialog main load"

If the main load is from dialog operations, you should avoid switching the working set procedure from "System" to "Selective" in order to prevent a large amount of paging activity.

PCS users can control this using the SERVICE-QUOTA-MAX values in the current PCS option.

A "dialog main load" situation arises if, *for all categories with dilation*, the total SERVICE-QUOTA-MAX values for the dialog categories are greater than the total SERVICE-QUOTA-MAX values for the other categories.

Example 5

Category	SERVICE-QUOTA		REQUEST-DELAY		DURATION	NEXT-CATEGORY
	MIN	MAX	MIN	MAX		
TP	50	100	2	4	–	–
DIALOG	20	40	2	4	500	DIALOG1
DIALOG1	10	30	2	5	2000	BATCH
BATCH	0	20	–	–	–	–

In the last control interval, PCS allocated 65% of the available capacity to the TP category (normal TP load with dilation significantly less than 4); the dialog categories were allocated the remaining capacity. For the current interval PCS again plans to allocate approximately 65% to TP, with DIALOG and DIALOG1 sharing the remaining 35% in proportion to their planned values. No capacity is planned for BATCH.

If DIALOG reached REQUEST-DELAY-MAX in the last interval, and category DIALOG1 did not, more capacity will be made available to DIALOG in the current interval at the expense of DIALOG1. If both DIALOG categories reached REQUEST-DELAY-MAX, the remaining 35% of the system's performance capacity will be divided between DIALOG and DIALOG1 in proportion to their SERVICE-QUOTA-MAX values (40:30, i.e. 20% and 15% of total performance respectively).

On another occasion, the load, and consequently the dilation, increases in the TP category. PCS therefore allocates service to this category at the expense of the DIALOGs until the ratio TP request delay to TP service quota reaches a value corresponding to the linear interpolation (cf. section "Parameter interaction SERVICE-QUOTA/REQUEST-DELAY" on page 23).

If REQEST-DELAY-MAX is still exceeded, TP is allocated all the available capacity if necessary, and none is planned for DIALOG and DIALOG1.

PCS controls allocations and withdrawals of capacity surpluses and shortages for all categories according to the specifications made. The allocation criteria are summarized in the following table:

Category	**Allocation of SERVICE quotas**
Categories with dilation range and S-Q-MAX=100	These categories are allocated service before all other categories ("planned values"); the amount depends on the current dilation factor (cf. section "Parameter interaction SERVICE-QUOTA/REQUEST-DELAY" on page 23). Insufficient capacity: If there is more than one such category (not recommended) and the sum of their planned values exceeds 100, then the planned values will be reduced proportionally until their total equals 100. Surplus capacity: If the sum of the planned values is less than 100, the remaining capacity is allocated to categories with and without a dilation range. If the tasks in the categories with a dilation range and S-Q-MAX=100 cannot utilize the planned service, the remaining capacity will be passed on to the categories with and without dilation ranges (*exception*: global THROUGHPUT-QUOTA = 0, "hard" response time optimization).
Categories with dilation range and S-Q-MAX<100	Service not required by categories with a dilation range and S-Q-MAX=100 is allocated to categories with a dilation range before categories without a dilation range; the amount is determined by the current dilation value (cf. section "Parameter interaction SERVICE-QUOTA/REQUEST-DELAY" on page 23). Insufficient capacity: If there is more than one category with a dilation range and the sum of their planned values exceeds the available capacity, the planned values will be reduced proportionately until their total corresponds to the available capacity. Surplus capacity: If the sum of the planned values is less than the available capacity, the remainder is allocated to a category without a dilation range. If the tasks in the categories with a dilation range cannot make full use of the allocated capacity, the remainder will be made available to categories without a dilation range (*exception*: global THROUGHPUT-QUOTA=0).

Category	Allocation of SERVICE quotas
Categories without a dilation range	Categories without a dilation range are allocated service left over by the categories with a dilation range. This remainder is distributed among categories without a dilation range in proportion to the SERVICE-QUOTA-MAX values. The SERVICE-QUOTA-MIN values have no effect in respect of categories without a dilation range.

In order to make an option easier to interpret, the sum of SERVICE-QUOTA-MAX values for all categories without a dilation range and the sum of SERVICE-QUOTA-MIN values for all categories with a dilation range should equal 100.

If the service currently being used by a category is less than the planned amount, PCS improves the priorities of the tasks in this category in order to achieve the planned values. Although the (current) remainder capacities do not go unused (see above), the tasks in this category will be given preference over tasks in comparable categories that are making better use of their capacity. This may be seen as undesirable by systems support; if so, then it is recommended that the SERVICE-QUOTA values be reduced to a level where the capacity being used corresponds approximately to the planned amount.

However, systems support can also make deliberate use of this effect to give preference to tasks in a certain category. For this purpose, they should specify very generous SERVICE-QUOTA values, i.e. exceeding actual requirements.

Possible reasons why tasks in a category are not able to utilize their service allocation are:

1. Due to insufficient information about the load, the SERVICE-QUOTA values selected for the category are too big and/or the REQUEST-DELAY values too small.
2. A category with a dilation range contains only very short transactions, and these, because of their high initial dilation, are allocated a large amount of service which they cannot fully use.
3. There is no proper balance between the required and the available hardware resources for the category.
4. Deadlock between cooperating tasks in the category (e.g. serialization problems).

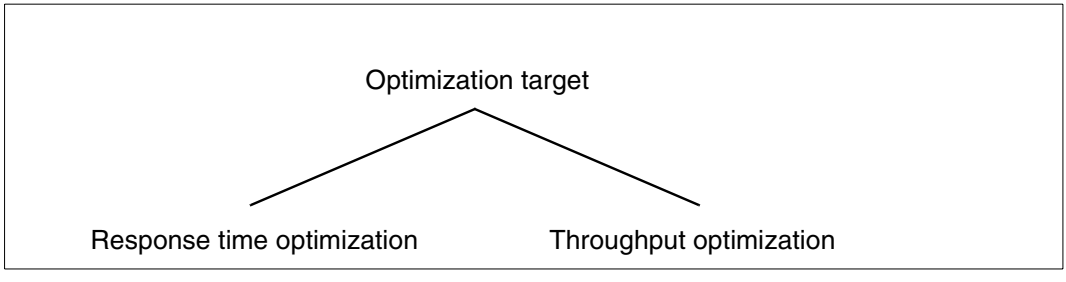
It should also be noted that the REQUEST-DELAY values measured by PCS may reach a very high level not only because of contention in full load or overload situations but also because of paging activity, especially in the initial phase of a transaction (initial dilation).

5 Parameter sets (OPTIONs)

5.1 Standard options

Precise knowledge of the load is necessary in order to ensure optimum parameter settings. A good way to start is to divide up the load roughly according to the operating modes TP, dialog and batch, and define an **optimization target**:

Response time optimization or throughput optimization?



This distinction is important because opposite trends are involved:

In order to achieve high throughput, resources must be utilized as efficiently as possible. In order to achieve good response times, resources must not be so heavily loaded that the waiting times become unacceptable.

These interdependencies must be taken into account when selecting the focal point of the application.

From the innumerable permutations of mode of operation, optimization target and focal point of the application, those combinations which occur most frequently in practice have been selected and furnished with a standard parameter set:

	Main application	Background application
Response time optimization:	- TP - Dialog	Dialog, batch TP, batch
Throughput optimization:	- Batch	Dialog

The TP application is to be considered the main application when the number of TP transactions is

≥ 50% of the total number

of transactions.

The standard options have been set up in such a way that they give preference to the main application and control as broad a range of the load components as possible at a suboptimal level. By adjusting a standard option to suit a specific load composition at an installation, it is usually possible to achieve an improvement in certain performance characteristics (see section “Modifying standard options” on page 51).

5.1.1 Main application: TP (option STD#TP)

Load requirements

TP application:	Service requirements for normal load: 60-70% In order to cater for temporary peak loads (approx. 15 minutes to max. 30 minutes), it should be possible to tap up to 100% of the available service (equivalent to total preemption of the background load if necessary).
Dialog application:	Preference given to dialog transactions with low resource requirements ("normal" dialog: response times < 3 sec) over dialog transactions with high resource requirements ("long-running" dialog).
Batch application:	Utilization of remaining capacity ("normal" batch) Execution of very important batch tasks in exceptional cases ("fast" batch); this must not lead to any degradation of TP mode.

Mapping the load to categories

In order to satisfy these requirements, categories for long-running dialogs (DIALOG1) and "fast" batch (BATCH FAST) must be defined in addition to the standard TP, DIALOG and BATCH categories.

Load type		Category
TP	⟷	TP
Normal dialog	⟷	DIALOG
Long-running dialog	⟷	DIALOG 1 (successor category)
Normal batch	⟷	BATCH
Fast batch	⟷	BATCHF (newly defined)

Note

In order to ensure that TP tasks are put into the TP category at the very start of the TP application, the following parameters of the **DEFINE-JOB-CLASS** statement must be supplied with values (see "Introductory Guide to Systems Support" [3]):

JOB-TYPE = { BATCH / DIALOG },

TP-ALLOWED = YES (CATEGORY = TP),

START-ATTR = TP

Examples of how to estimate the REQUEST-DELAY-MAX and DURATION parameters and the SERVICE-QUOTA-MAX parameter can be found in the section “Calculating REQUEST-DELAY-MAX and DURATION (example)” on page 151. An estimate of the performance requirements for the DIALOG category is given in the section “Calculating the SERVICE-QUOTA-MAX parameter” on page 154.

The value for the category changeover from DIALOG to DIALOG1 was selected such as to allow normal dialogs and the BS2000 functions needed for these dialogs to be executed in the DIALOG category (cf. section “Empirical values for the DURATION parameter” on page 153). An ample value was allowed for REQUEST-DELAY-MAX for DIALOG because DIALOG is not the main application in this option.

Parameter settings (STD#TP)

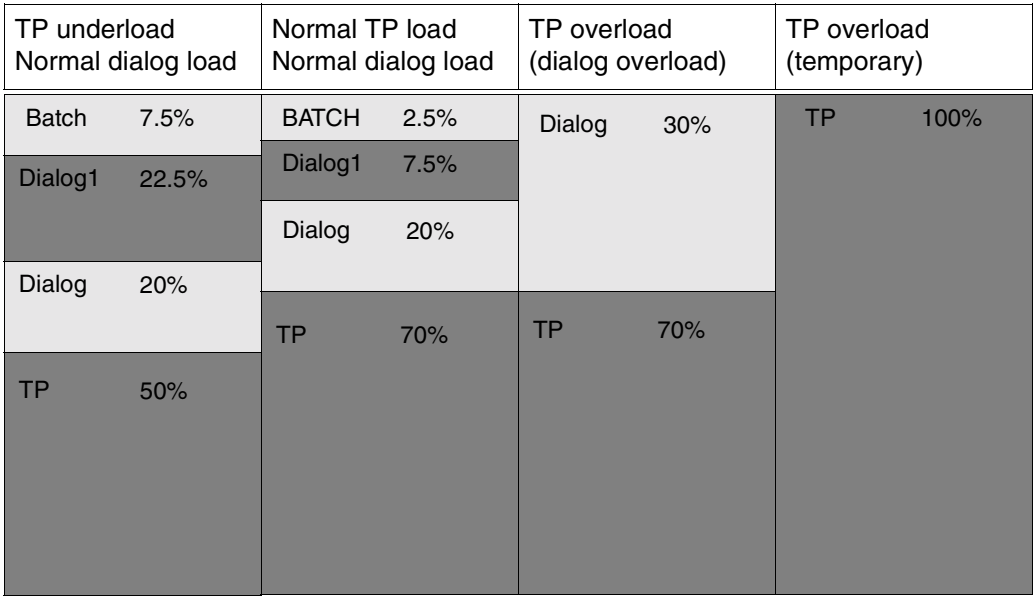
Global parameters:

REQUEST-DELAY-MAX: 6
THROUGHPUT-QUOTA: 20%

Category-specific parameters:

Category	SERVICE-QUOTA		REQUEST-DELAY-		DURATION	NEXT-CATEGORY	THROUGHPUT-QUOTA
	MIN	MAX	MIN	MAX			
TP	50	100	1	3	–	–	–
DIALOG	10	30	2	6	500	DIALOG1	–
DIALOG1	0	30	–	–	–	–	10
BATCH	0	10	–	–	–	–	70
BATCHF	0	20	1	3	–	–	–

This results in the following **system behavior** (initially **without** "fast" batching, which is only to be used in special cases):



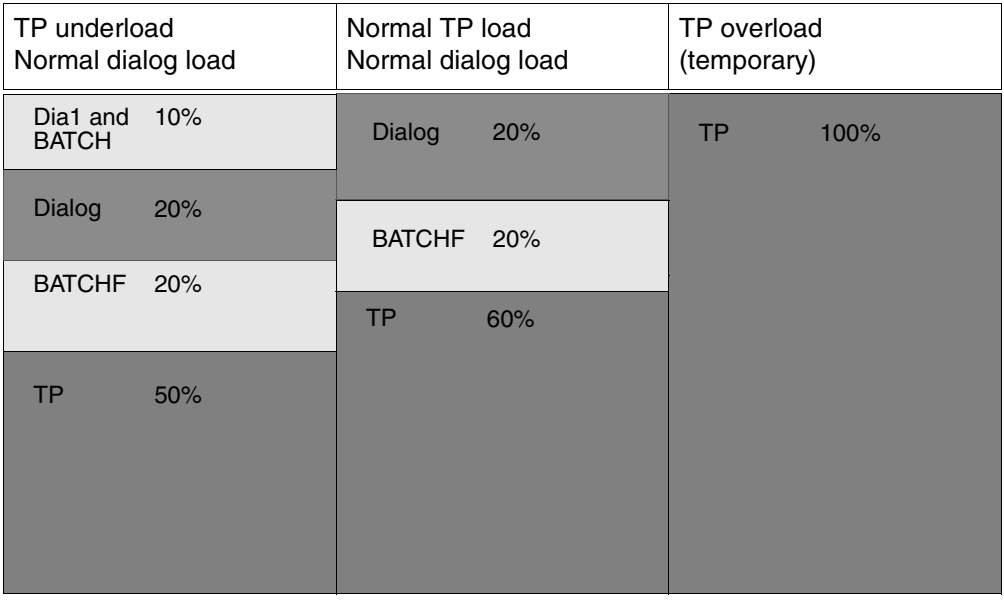
In the case of **TP underload** (50%), remaining capacity will be allocated to the DIALOG, DIALOG1 and BATCH categories. Once the service requirements for the DIALOG category have been met (e.g. 20%, providing the current dilation remains within the specified range), the categories DIALOG1 and BATCH can make use of the remaining service in proportion to their SQ-MAX values.

In the case of **normal TP load** (70%) and **normal dialog load** (20%), only 10% of the system's capacity remain for use by long-running dialog transactions and "normal" batch jobs.

In the case of **normal TP load** (70%) and **dialog overload** (30%), long-running dialog transactions and "normal" batch jobs will suffer total preemption.

Temporary peaks in TP load (**TP overload**) are catered for by total preemption of the background load.

System behavior when important batch jobs ("fast" batch: BATCHF) are taken into account:



If 100% service is needed for the TP category, the background load suffers total preemption.

5.1.2 Main application: dialog (option STD#DIA)

Load requirements

Dialog application:	Short edit commands and other similar commands are to have response times < 1 sec ("short" dialog).
	Commands requiring more resources are to be served within 3 seconds ("normal" dialog).
	Program executions (/START-SUBSYSTEM...) and procedure calls (/CALL-PROCEDURE...) with high resource requirements are to be handled in a similar manner to batch requests but are still to be given preference over batch (long-running dialog).
TP application:	The fact that the TP application is not the main application does not necessarily mean that its capacity requirements are low, but rather that the number of TP transactions < 70% of the total number of transactions. (This is usually the case when TP applications are first introduced.)
	Service requirements: 10%-30%
Batch application:	Utilizes remaining capacity ("normal" batch)
	Execution of very important batch jobs in special cases ("fast" batch).

Mapping the load to categories

As well as the standard categories, it is necessary to define additional categories for "normal" dialog ("short" dialog is placed in the standard category DIALOG), for "long-running" dialog and for "fast" batch.

Load type		Category
TP	⟷	TP
Short dialog	⟷	DIALOG
Normal dialog	⟷	DIALOG1 (successor category)
Long-running dialog	⟷	DIALOG2 (successor category)
Normal batch	⟷	BATCH
Fast batch	⟷	BATCHF (newly defined)

An explanation of the steps necessary for assigning TP tasks in the TP category is given in section “Main application: TP (option STD#TP)” on page 39.

Examples of how to estimate the REQUEST-DELAY-MAX and DURATION parameters and the SERVICE-QUOTA-MAX parameter can be found in the section “Calculating REQUEST-DELAY-MAX and DURATION (example)” on page 151 and in the section “Calculating the SERVICE-QUOTA-MAX parameter” on page 154.

The service requirement for "short" dialogs ranges between 5 and 200 service units, and between 200 and 800 SUs for somewhat more resource-intensive file operations (READ, GET, etc.) (cf. section “Empirical values for the DURATION parameter” on page 153).

To prevent dialog transactions with high resource requirements disrupting "short" and "normal" dialogs, the following values could be used for the category changeover:

DIALOG → DIALOG1: DURATION=100

DIALOG1 → DIALOG2: DURATION=500

In order to provide the best possible general-purpose STD#DIA option ("soft" setting for response time/throughput optimization), the DURATION values have been increased (500, 2000).

This means that "short" dialogs and the BS2000 functions needed for these dialogs can be executed in the DIALOG category. This leads to an improvement in total throughput without any marked negative effect on response time behavior.

Parameter settings (STD#DIA)

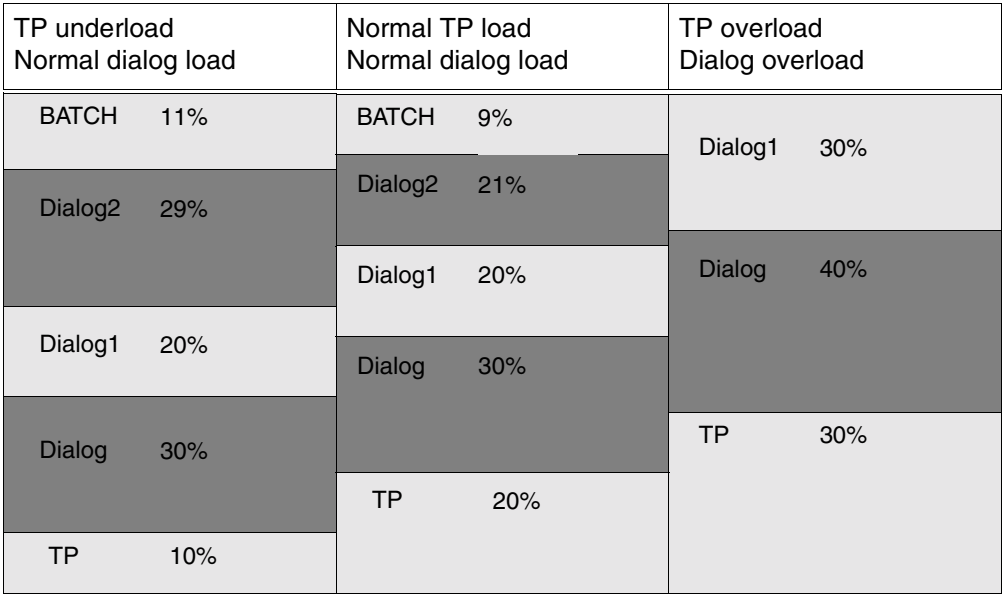
Global parameters:

REQUEST-DELAY-MAX: 6
THROUGHPUT-QUOTA: 20%

Category-specific parameters:

Category	SERVICE-QUOTA		REQUEST-DELAY-		DURATION	NEXT-CATEGORY	THROUGHPUT-QUOTA
	MIN	MAX	MIN	MAX			
TP	0	30	1	3	–	–	–
DIALOG	20	40	2	4	500	DIALOG1	–
DIALOG1	10	30	2	5	2000	DIALOG2	10
DIALOG2	0	50	–	–	–	–	20
BATCH	0	20	–	–	–	–	70
BATCHF	0	20	1	3	–	–	–

System behavior (initially **without** taking account of "fast" batch, which is only to be executed in special cases):



In the case of **TP underload** (10%) and **normal dialog load** (e.g. 50% for DIALOG and DIALOG1 if the current dilation for the categories DIALOG and DIALOG1 is within the specified range), the categories DIALOG2 and BATCH can make use of the remaining capacity in proportion to their SQ-MAX values.

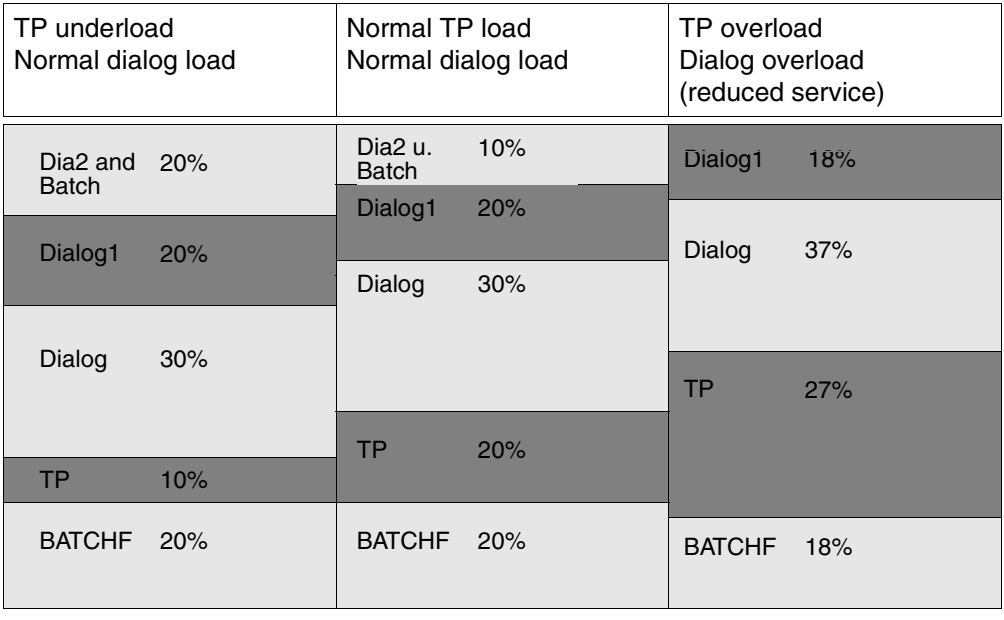
In the case of **normal TP load** (20%) and **normal dialog load** (50%), 30% of the system's capacity is available for long-running dialog transactions and "normal" batch jobs.

In the case of **TP overload** (30%) and **dialog overload** (70%), service for the DIALOG2 and BATCH categories is reduced to zero.

Note

This option assumes that TP, DIALOG and DIALOG1 may be overloaded at the same time (i.e. the sum of the SERVICE-QUOTA-MAX values for these categories is 100.

System behavior when important batch jobs ("fast" batch: BATCHF) are taken into account:



By specifying a dilation parameter for the BATCHF category, this can be given preference over the DIALOG2 and BATCH categories; it can also take precedence over the DIALOG and DIALOG1 categories by specifying smaller dilation values.

In the case of **TP overload** and **dialog overload**, the sum of specified SERVICE-QUOTA-MAX values for categories with a dilation range > 100. Service is therefore allocated in proportion to the planned SERVICE-QUOTA values.

5.1.3 Main application: batch (option STD#BAT)

Load requirements

Batch application: Service requirements: 70%-80%

Short batch jobs ("normal" batch) are given preference over long-running jobs. The desired service allocation ratio is approximately 2:1.

Execution of very important batch jobs in special cases ("fast" batch).

Dialog application: Service requirements for normal load: 20%-30%

Dialog transactions with low resource requirements ("normal" dialog, response times < 3 sec) are given preference over dialog transactions with high resource requirements (long-running dialog).

"Normal" dialog is to take precedence over the batch application.

Operation of the overall system is not to be directed toward response time optimization because throughput is important for the BATCH main application.

Mapping the load to categories

As well as the standard categories, additional categories for long-running batch jobs (long-running batch) and very important batch jobs ("fast" batch) are needed. Long-running dialog transactions will be treated in the same way as batch requests.

Load type		Category
TP	⟷	TP
Normal dialog	⟷	DIALOG
Long-running dialog	⟷	BATCH
Normal batch	⟷	BATCH
Long-running batch	⟷	BATCH1 (successor category)
Fast batch	⟷	BATCHF (newly defined)

Examples of how to estimate the REQUEST-DELAY-MAX and DURATION parameters and the SERVICE-QUOTA-MAX parameter can be found in the section “Calculating REQUEST-DELAY-MAX and DURATION (example)” on page 151 and in the section “Calculating the SERVICE-QUOTA-MAX parameter” on page 154.

The value for the category changeover from DIALOG to BATCH has been selected in order to allow "normal" dialogs and the BS2000 functions required for them to be executed in the DIALOG category.

The standard TP category is allocated a SERVICE-QUOTA-MAX value of 30% in order to make provision for existing TP tasks (e.g. TDADM).

Parameter settings (STD#BAT)

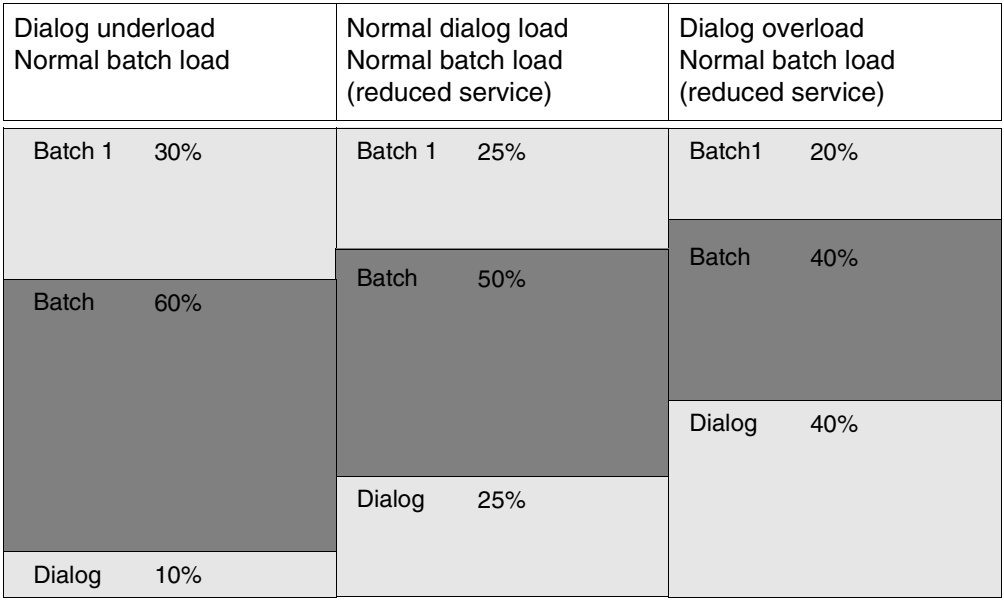
Global parameters:

REQUEST-DELAY-MAX: 6
THROUGHPUT-QUOTA: 50%

Category-specific parameters:

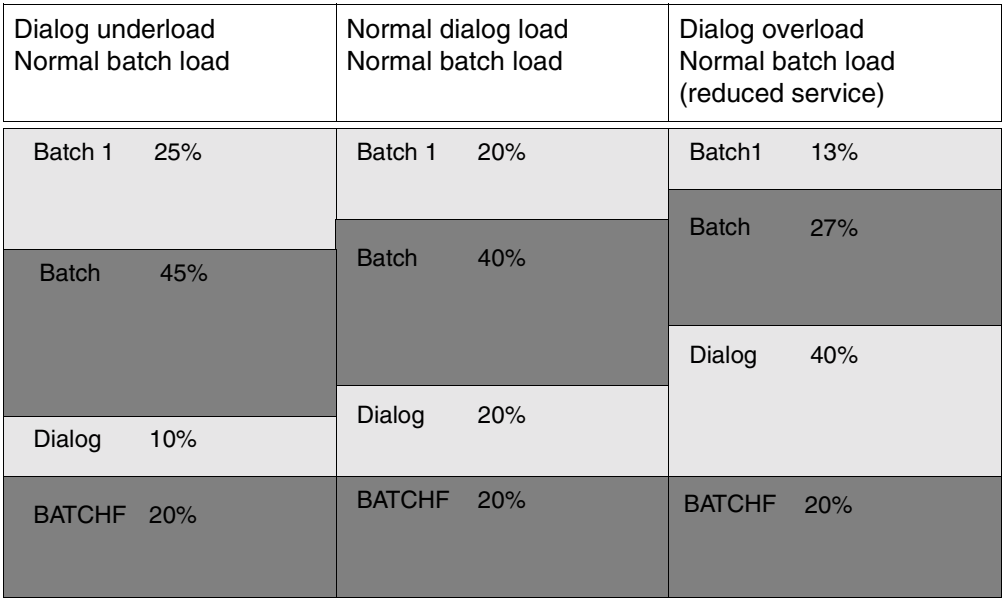
Category	SERVICE-QUOTA		REQUEST-DELAY-		DURATION	NEXT-CATEGORY	THROUGHPUT-QUOTA
	MIN	MAX	MIN	MAX			
TP	0	30	1	3	–	–	–
DIALOG	0	40	1	5	500	BATCH	–
BATCH	0	65	–	–	50000	BATCH1	70
BATCH1	0	35	–	–	–	–	1000
BATCHF	0	20	1	3	–	–	–

System behavior without taking account of "fast" batch, which is only to be executed in special cases:



In the case of **dialog underload**, the categories BATCH and BATCH1 can make full use of their service allocation. In the case of **dialog overload**, the DIALOG category is given preference, thereby reducing the amount of service available to BATCH applications. The TP category is not represented because the amount of service tapped off by TP tasks such as TDADM (see above) is so small.

System behavior when important batch jobs ("fast" batch: BATCHF) are taken into account:



By specifying a dilation parameter for the BATCHF category, this is given preference over the BATCH and BATCH1 categories. The value of 3 defined for REQUEST-DELAY-MAX results in BATCHF being given preference over DIALOG if their current dilation is comparable.

5.2 Modifying standard options

The standard options are designed to give preference to the main application and to provide optimal control options for this application for as much of the load composition as possible. By adjusting a standard option to match the load composition of the particular installation, it is usually possible to achieve an improvement in certain performance characteristics. This section gives some advice on how to perform the type of tuning involved. Information on estimating or measuring the required values is summarized in section “Determining monitored variables” on page 155.

5.2.1 Load composition

The first step is to check whether the composition of the load corresponds to the categories of the selected standard option in terms of type and proportions of the load components (TP, DIALOG, etc.):

1. Are all the categories needed?

   A category that is not needed will cause no problems provided its SERVICE-QUOTA-MIN value = 0. If its SERVICE-QUOTA-MIN value ≠ 0, it should be set to 0 or the category should be removed from the option (cf. chapter “PCS definition file (PPF)” on page 59" and chapter “PCSDEFINE utility routine” on page 67).

2. Is a special category missing for an important load component?

   The missing category should be set up and supplied with suitable parameters. How this is done is explained in section “Creating a special option” on page 54.

3. Do the resource requirements (in SERVICE-UNITs) of the most important of the higher-priority dialog processing steps (REQUESTs) match the category changeover threshold values (DURATION) of the dialog categories? The same test should be applied to the batch category in the case of STD#BAT.

   If they do not match, the DURATION values should be adjusted.

   The DURATION values for successive categories should differ from each other by at least a factor of 4.

4. Does the service actually used (SQACT or SRACT) in the categories during the PCS session roughly match the SERVICE-QUOTA range (SQPLN or SRPLN) of the corresponding category?

   If there is too great a difference between the SQPLN or SQACT values over a long period of time (greater than factor 2 for full load), the corresponding SERVICE-QUOTA limit values should be adjusted accordingly.

5.2.2 Response times

If the response times in one or more dilation categories do not come up to expectations, the following should be checked:

Case 1: The category is handling the load well (the amount of service used is close to the planned value):

First, the SERVICE-QUOTA-MAX value for the category should be increased in order to prioritize the tasks better. If this is not enough (or is not possible, e.g. for TP in STDTTP), the values for REQUEST-DELAY-MIN and REQUEST-DELAY-MAX should be reduced, if necessary to just above the observed actual value for REQUEST-DELAY in the category. This results in the category being allocated more service (which is, of course, taken away from other categories).

If this does not improve the response times, a check should be made to determine whether the category is prevented from utilizing extra service due to internal locks, I/O bottlenecks etc.

If response times improve but not sufficiently, 'hard' response time optimization can be selected by means of the global parameter THROUGHPUT-QUOTA=0 (cf. section "Description of the PCS parameter set (OPTION)" on page 60).

Warning

This PCS parameter has a drastic effect. Its impact on system behavior should be closely monitored.

Response time as a whole can be optimized by reducing the global parameter REQUEST-DELAY-MAX. This improvement is, however, usually gained at the expense of reduced throughput, especially for batch categories. This parameter should be modified carefully in small steps and its effect closely monitored (e.g. in case of excessive IDLE).

Case 2: The category cannot utilize the planned amount of service. (The actual service used is less than the planned quota.)

In this case, the bottlenecks are presumably within the categories (e.g. check the communications network).

An improvement in response times is probably not possible; a check should be made to determine whether the SERVICE-QUOTA values should be adjusted to match the amount of service actually consumed by the category in section "Load composition" on page 51).

5.2.3 Using task priorities

Task priorities may be used to give preference to certain load components within a category. The difference between priorities must be at least 20.

The effect of priorities depends on the category parameter THROUGHPUT-QUOTA (cf. section “Effect of task priorities” on page 157). This, in combination with the highest priority allowed in the category, enables systems support to control the amount of service tapped off by the tasks in the category.

Examples

1. In a TP category, the DB tasks are to have a higher priority than the TP tasks.
2. In a batch category used for preempted dialogs, it may make sense to give the preempted dialogs a higher priority than batch tasks (depending on the particular system performance objectives which systems support is aiming at).

5.3 Creating a special option

The creation of an installation- and load-specific option requires the following:

– adequate knowledge of the PCS parameters to be used (see chapter “PCS definition file (PPF)” on page 59 for a summary)

– evaluation of the statistical data output by PCS (see the description of SHOW-PCS-OPTION on page 119)

– knowledge of how to handle the PCSDEFINE utility routine (see chapter “PCSDEFINE utility routine” on page 67)

– use of a software monitor, e.g. *open*SM2.

Inexperienced users of PCS are advised to start with simple options and refine them as necessary.

An option is created in a series of steps:

1. Defining the required categories
2. Examining the service requirements of the categories
3. Defining and entering the option
4. Using the option: checking its operational behavior
5. Repetition of steps 2 to 4, if necessary.

Information about estimating or measuring the values required can be found in section “Determining monitored variables” on page 155.

5.3.1 Defining the categories

First, it is necessary to determine which parts of the load should be grouped together in a category on the basis of response time and throughput. The categories for the standard options are examples of this (see section “Standard options” on page 37).

For example, it is often better if true dialogs and batch-type processes (/START-SUBSYSTEM ...) with high resource requirements that occur together in the standard DIALOG category are placed into two different categories. Either a separate category can be set up for the batch-type processes or they can be included in an existing batch category. Category changeover by means of the DURATION parameter is a useful way of handling the types of load component which differ only in respect of the resources required for their requests.

Altogether, the three standard categories TP, DIALOG and BATCH, as well as 12 other freely selectable categories, are available. It is advisable not to use too many categories at first, but rather to refine the allocation of service as familiarity with the use of PCS increases.

The PCSDEFINE utility routine may (and should) be used to define the categories in such a way that a wide variety of load compositions can be controlled with a *single* option. A change of option may become necessary if there is a fundamental change in the type of load; for example, from a dialog-oriented mode of operation during the day to pure batch processing at night. Checking and adjusting the options is usually only necessary at major intervals or when the installation undergoes modification.

Job classes and PCS options must be coordinated. All the categories used in a job class must be contained in the option, and all other categories must be accessible via a sequence of DURATION/NEXT-CATEGORY changeovers. Options with categories that cannot be accessed directly by the job scheduler or by PCS via category changeover will be rejected by PCS when the option is started.

5.3.2 Examining service requirements

First of all, the threshold values for category changeover (DURATION) should be defined. Initially, empirical values or rough estimates of the CPU time required can be used as a basis (see section “Empirical values for the DURATION parameter” on page 153); from this an approximate value for DURATION can be calculated using the following formula:

DURATION = 2 * a * CPU time (for information on the constant "a", see section “Definition of service units (SU)” on page 149)

After this, the SERVICE-QUOTA range and, if applicable, dilation windows should be defined. To this end, it is best to obtain an overview of the service requirements of all planned categories, or at least of the response time-oriented categories, by taking simple measurements with PCS or *open*SM2. The expected maximum service requirements for each category should match the SERVICE-QUOTA-MAX value for the category. The sum of SERVICE-QUOTA-MAX values for all categories will usually be greater than 100.

If the examination of the load is done using PCS, the use of a simple measuring option can be helpful:

– SERVICE-QUOTA-MAX = 100 for all time-critical categories (TP, DIALOG, etc.; it is assumed that the sum of the actual service requirements of these categories at any given time does not exceed the total allocatable capacity (=100%)). A better alternative still is to start with estimated approximations for the SERVICE-QUOTA-MAX value.

– SERVICE-QUOTA-MAX = 0 for batch-type categories.

– default values for all other parameters.

This option is switched on for approximately 20 minutes during periods of time-critical peak load and (after a transient phase of max. 10 minutes) causes the batch-type load to be preempted (except for the remainder portion of capacity not needed by the time-critical categories). By repeated use of the SHOW-PCS-OPTION command it is then possible to obtain measuring points for the current amount of service used (SQACT) and the current dilation (RDACT) for the time-critical categories.

The service requirement and dilation values determined in this way can be used directly as SERVICE-QUOTA-MAX or REQUEST-DELAY-MAX values, possibly increased by a small amount to provide some reserve capacity or similar.

The default values for SERVICE-QUOTA-MIN and REQUEST-DELAY-MIN are usually sufficient. If an important category (e.g. TP) requires service very quickly after a period without load (e.g. after the lunch break), SERVICE-QUOTA-MIN should be increased to a suitable value and/or REQUEST-DELAY-MIN decreased. Measurements are useful in this case too.

If the service requirements of a category with a dilation range do not fall below a certain value during the whole time the option is used, then this value should be used as SERVICE-QUOTA-MIN.
The SERVICE-QUOTA-MAX values for categories without a dilation range should be set according to the ratio of expected service quotas for the period during which the planned option is in use. Of course, the service that these categories receive in total may at best equal only what the time-critical categories do not use. The SERVICE-QUOTA-MIN values for the categories without a dilation range do not influence the allocation of service; the default value of 0 for these categories should not be changed.

Generally, the category-specific THROUGHPUT-QUOTA parameter should be set to the default value in the case of time-critical categories, while batch-type categories should be assigned a THROUGHPUT-QUOTA value of 70%.

5.3.3 Entering and using the option

The option is input using PCSDEFINE as described in chapter “PCS definition file (PPF)” on page 59, and used as described in chapter “PCS administration” on page 119.

When a new option is first used, and at later times also, a check should be made to determine whether the desired operational behavior has been achieved. The SHOW command of PCS or the *open*SM2 software monitor is suitable for this.

If the operational behavior does not match up to the planned objectives, the option can be modified according to the recommendations made in this section and in section “Modifying standard options” on page 51.

5.3.4 Possible error sources

This section summarizes

– what PCS **cannot** do

– where due caution should be exercised

– where possible sources of error are in the parameter settings.

PCS cannot

- supply missing hardware if the installation is underconfigured or if the installed hardware cannot cope with the average resource requirements of the load (e.g. I/O bottleneck). In these cases, PCS has to displace parts of the load by preemption at peak times or give preference to load components whose resource requirements can be satisfied by the installed hardware.
- release internal locks set by large program systems. In such cases, increasing the service allocation will not improve response time, but merely impede other categories.

The following parameter settings should be used with care:

- System THROUGHPUT-QUOTA = 100 (cf. section “Description of the PCS parameter set (OPTION)” on page 60)

  Only for non-time-critical (batch) loads for which maximum throughput is to be achieved. Task priorities will be ignored; runtime for jobs requiring scarce hardware resources may increase considerably.

- System THROUGHPUT-QUOTA = 0 (cf. section “Description of the PCS parameter set (OPTION)” on page 60)

  Only for outright response time optimization, even at the expense of throughput. Capacity remaining unused by response-time-oriented categories will not be made available to other categories; IDLE may result.

- Category THROUGHPUT-QUOTA = 100 (cf. section “Description of the category parameter set (CATEGORY)” on page 63)

  Same as system THROUGHPUT-QUOTA = 100, except that here the resulting effect is restricted to the category.

- SERVICE-QUOTA-MAX = 100 for a category with REQUEST-DELAY-MAX > 0 (cf. section “Service allocation” on page 33).

  Only for giving absolute preference to an especially time-critical category (e.g. TP). The category is allocated service on account of its dilation **ahead** of all other dilation categories with SERVICE-QUOTA-MAX < 100 (e.g. DIALOG).

When setting parameters, the following point should be borne in mind:

- Modifying parameters has little or no effect if the corresponding actual value is outside the range of modification (e.g. reducing REQUEST-DELAY-MAX from 5 to 4 will have no effect if the current dilation in the category is already 7).

6 PCS definition file (PPF)

The file PPF is created, supplied with values and modified using the PCSDEFINE utility routine (see chapter “PCSDEFINE utility routine” on page 67). The next few sections explain the structure and content of PPF. The effect of the PCS parameters is also dealt with. The default name for the PPF file is "SYSSSI.PCS.027".

6.1 Structure of PPF records

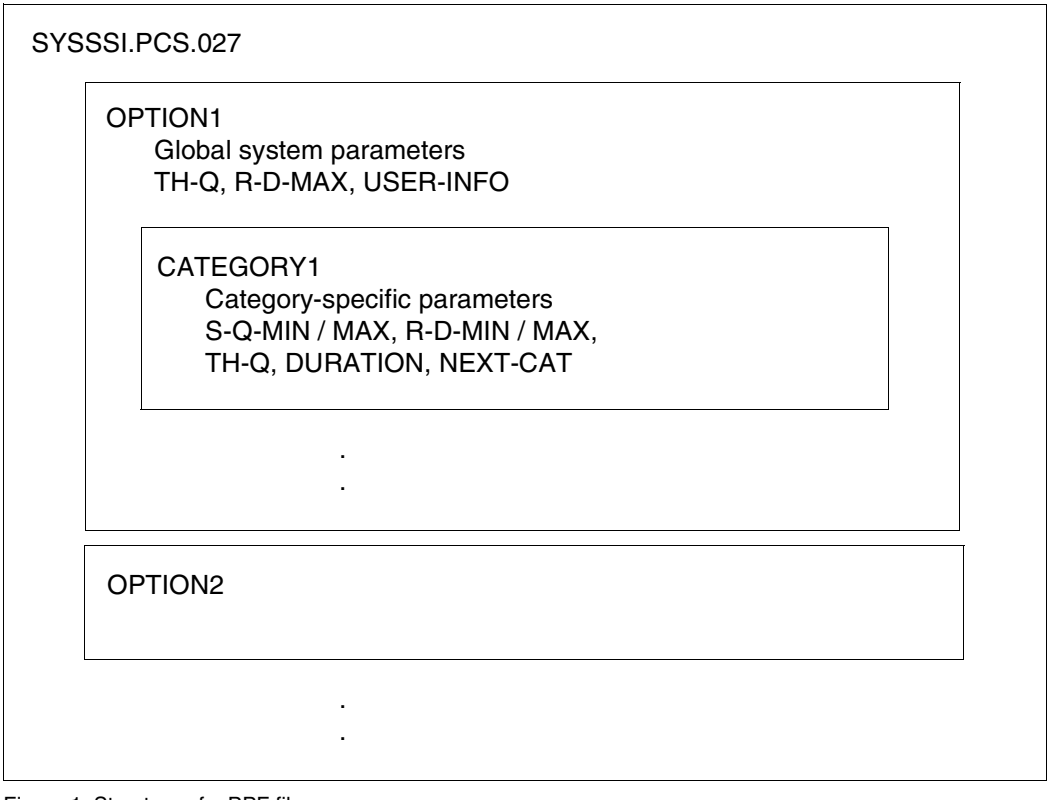


Figure 1: Structure of a PPF file

6.2 Description of the PCS parameter set (OPTION)

In addition to the name of the parameter set, an option also contains the names of all associated category sets, specifications for controlling PCS operation, and the global PCS parameters needed for the selected operating mode (response time-oriented, throughput-oriented).

1. Identification of the entry

OPTION-NAME Name of the PCS option to which the category specified in the CATEGORY-NAME parameter belongs. This parameter is necessary as it ensures that each category is assigned to an option.
The name is referenced by the command /START-SUBSYSTEM PCS... .

Unit Character string (max. 8 characters)

Note

The option name STDOPT is accepted by PCS as the default option name. This means that it is not necessary to specify the option name when starting PCS with the STDOPT option.

2. Global PCS parameters (SYSTEM-PARAMETER)

REQUEST-DELAY-MAX
This parameter specifies a threshold value for the maximum dilation for all tasks within the admission space that belong to a category with a dilation range.

Value range (1, 100)

Default value 5 + (THROUGHPUT-QUOTA) / 20 (=6)

Mode of operation If the mean dilation computed for requests from categories with a dilation range exceeds the threshold value, the PCS load control function intervenes and reduces the number of active tasks.

THROUGHPUT-QUOTA

This specifies a percentage that determines the relation between response time optimization and throughput optimization.
The parameter optimizes either response time (in which case utilization of resources may not be optimal) or throughput (in which case response times and job runtimes may show very considerable divergence).

Response time optimization			Throughput optimization	
hard	full		full	hard
0%	20%	50%	80%	100%

Value range (1, 100)

Default value 20 (full response time optimization)

Mode of operation This parameter influences the global REQUEST-DELAY-MAX value, the initiation priority of the tasks, allocation of remaining capacity, etc.

Notes

- THROUGHPUT-QUOTA = 0 optimizes response time; if, for example, categories with a dilation range are not able to make full use of the planned capacity, this capacity will not be passed on to categories without a dilation range and IDLE may result.
- THROUGHPUT-QUOTA = 100 should only be used for pure BATCH loads that are to run at maximum throughput, but where time is not a critical factor for the individual jobs.

3. Controlling PCS operation

USER-INFORMATION This specifies whether information about PCS is also to be output to nonprivileged users (/SHOW-PCS-OPTION...).

Value range (YES, NO)

Default value NO

CATEGORY List of additionally defined CATEGORY entries belonging to this option. This list does not include the standard categories SYS, DIALOG, BATCH and TP. PCSDEFINE supplies default values for these categories.

Additional categories are required if they are also contained in the 'JOB CLASS' definition (SJMSFILE).
They are useful in cases where the PCS function "automatic category changeover" is to be used.

6.3 Description of the category parameter set (CATEGORY)

1. Identification of the category and associated option

   CATEGORY-NAME This specifies the name of a standard category (see above) or special category defined by systems support. Categories of this latter type are used, for example, as successor categories for automatic category changeover (cf. NEXT-CATEGORY and DURATION parameters).

   Unit Character string (max. 7 characters).

2. Specification of capacity allocation: SERVICE-QUOTA

   SERVICE-QUOTA-MAX

   This parameter defines the maximum percentage of capacity that is to be reserved for the category.

   Value range (0, 100)

   Default value 100

   SERVICE-QUOTA-MIN

   This parameter defines the minimum percentage of capacity that is to be reserved for the category.

   Value range (0, SERVICE-QUOTA-MAX - 1)

   Default value 0

   Mode of operation The effect of the parameters SERVICE-QUOTA-MIN and SERVICE-QUOTA-MAX in conjunction with REQUEST-DELAY is described in section “Parameter interaction SERVICE-QUOTA/REQUEST-DELAY” on page 23.

   Note

   The SERVICE-QUOTA-MIN parameter is effective only for categories with a dilation factor.

3. Specification of dilation range: REQUEST-DELAY

REQUEST-DELAY-MAX

This specifies the maximum delay for requests in the category and, in combination with SERVICE-QUOTA-MAX, defines the maximum share of capacity allocated to the category.

Value range: (1, 100) - with one decimal point.

Default value: No monitoring of the upper limit of the dilation range (corresponds to the value 0).

Notes

– The sum of the SERVICE-QUOTA-MAX values for categories with dilation parameters may exceed 100. This allows individual dilation categories to receive the required service quota when the category has a heavy workload. This can only be done, however, if other dilation categories are not requesting their maximum service quota at the same time.

– REQUEST-DELAY is a parameter specially matched to the control capability of PCS. It is therefore not directly comparable with the "DILATION" statistic supplied by *open*SM2.

REQUEST-DELAY-MIN

Specifies the minimum delay for REQUESTs in the category and, together with SERVICE-QUOTA-MIN, defines the minimum share of capacity that the category is to be given in preference to other categories.

Value range: (1, REQUEST-DELAY-MAX) - with one decimal point.

Default value: 1 if REQUEST-DELAY-MAX is specified; otherwise no monitoring of the lower limit for dilation (value 0).

Mode of operation: The effect of this parameter combined with SERVICE-QUOTA is described in section “Parameter interaction SERVICE-QUOTA/REQUEST-DELAY” on page 23.

Notes

– If REQUEST-DELAY-MAX is not specified, REQUEST-DELAY-MIN is ignored.

– REQUEST-DELAY-MIN should always be less than REQUEST-DELAY-MAX. A setting where REQUEST-DELAY-MIN = REQUEST-DELAY-MAX is **not** recommended (unstable regulation).

4. Throughput parameters

THROUGHPUT-QUOTA

This parameter specifies a percentage that determines the relation between response time and throughput optimization for the category.

Its effect on the operational behavior of the category ranges from full response time optimization through to full throughput optimization (its effect at settings of 0 and 100 is not as "hard" as the global parameter of the same name):

Response time optimization		Throughput optimization
full		full
0%	50%	100%

Value range (0, 100)

Default value 0 (full response time optimization)

Mode of operation This parameter influences the activation priority of tasks and the effect of increases in priority assigned by the capacity allocation and resource utilization function.

5. Automatic category changeover

DURATION Number of service units per request after which an automatic category changeover is to take place (see below).

Value range (1, 100000)

Default value (UNLIMITED): no automatic category changeover.

NEXT-CATEGORY Name of the successor category for automatic category changeover after <DURATION> service units. This parameter only makes sense when used with the DURATION parameter. Its validity can be checked using the CHECK function of the PCSDEFINE utility routine.

Mode of operation As soon as a request uses more service than the DURATION value allows, a changeover is made to the successor category. If the request exceeds the threshold value there too, it changes to a further successor category if this is possible. On completion of the current request, the next request of the task starts in the original category.

Notes

– Special categories as well as suitable primary categories can be used as NEXT-CATEGORY.

– In order to achieve the desired effect, the value of the DURATION parameter for each successor category must be greater than the value for the preceding category (the final successor category, of course, has no DURATION value).

7 PCSDEFINE utility routine

- The utility routine PCSDEFINE is used to manage all of the PCS parameters that can be defined externally by systems support. PCSDEFINE recognizes two user interfaces: a statement-oriented interface (section “PCSDEFINE in statement mode” on page 86) and a menu-oriented interface (section “PCSDEFINE in menu mode” on page 73).
- If systems support works exclusively with the predefined standard options, he does not need to use this utility routine. However, he may still choose to use it to output the parameter settings for the standard options in edited form.
- Matching categories to those used by JMS (Job Management System).

  The entire set of categories (category names) recognized by JMS must be identical to the entire set of categories (category names) set in the current PCS option for the system insofar as the categories are not successor categories.

  This means that unmatched categories must either be added to the option or declared to JMS using the "JMU" utility (see the manual "Utility Routines" [9]).
  Successor categories are so-called "duration runout categories" and are only valid for the scope covered by PCS.
  Users can, however, check these successor categories using the "/STATUS CATEGORY" command or, in particular, using the PCS command "/SHOW-PCS-OPTION".

7.1 Starting and terminating PCSDEFINE

The PCSDEFINE program can be started in either of the following ways:

1. by using the /START-PCSDEFINE command
2. by using the abbreviated command /PCSDEFINE.

By calling one of these two commands, the PCSDEFINE utility routine is started in statement mode. PCSDEFINE can be switched from statement mode to menu mode by pressing the F2 function key.

The /START-PCSDEFINE command

Domain: SYSTEM-TUNING, UTILITIES

Privileges: TSOS, OPERATING, STD-PROCESSING

START-PCSDEFINE

VERSION = *** STD** / <product version 6..10> / product version 4..8 without-corr> /
<product version 3..7 without man>
,**MONJV** = ***NONE** / <filename 1..54 without-gen-vers>
,**CPU-LIMIT** = ***JOB-REST** / <integer 1..32767 *seconds*>

Operands

VERSION =
The selected PCSDEFINE version will be used.

VERSION = *STD
The version set using the /SELECT-PRODUCT-VERSION command is loaded. If no version has been set, then the highest (most recent) PCSDEFINE version available will be loaded.

VERSION = <product-version 6..10>
Specifies the version.

VERSION = <product-version 4..8 without-corr>
Version designation without the correction (update) status.

VERSION = <product-version 3..7 without-man>
Version designation without the correction (update) and release status.

MONJV = *NONE / <filename 1..54 without-gen-vers>
Defines a job variable that is to monitor the program execution.

CPU-LIMIT = *JOB-REST / <integer 1..32767 *seconds*>
Defines the maximum CPU time that PCSDEFINE is allowed to run. If no time is entered, PCSDEFINE will execute until the task is finished.

The /PCSDEFINE command

PCSDEFINE
VERSION = *** STD** / <product-version 6..10> / product-version 4..8 without-corr> / <product-version 3..7 without-man> ,**MONJV** = ***NONE** / <filename 1..54 without-gen-vers> ,**CPU-LIMIT** = ***JOB-REST** / <integer 1..32767 *seconds*>

If PCS has been installed under IMON, then the program is stored in a file hidden from the user, otherwise it is stored as an LLM object "$PCSDEFN" in the library $.SYSLNK.PCSDEFINE.027.

Terminating the PCSDEFINE utility routine

PCSDEFINE is terminated with the END statement.

7.2 CREATE-PCS-OPTION
Create and modify parameter files

Domain: SYSTEM-TUNING, UTILITIES

Privileges: TSOS, OPERATING, STD-PROCESSING

The /CREATE-PCS-OPTION command allows you to modify one of the OPTIONs in the default parameter file or to create a new parameter file with individual OPTIONs and to store it under any name.

The /CREATE-PCS-OPTION command is especially suited for automatically creating an OPTION parameter file as well as for preparing to install or use a new, higher version.

In this way, any number of prepared OPTIONs can be stored in a procedure file (which is used as a 'container'), and from these options a specific parameter file can be generated and used in the desired PPF with the aid of the CREATE-PCS-OPTION command.

Processing this command leads to the execution of this procedure. The procedure, which is stored in a library, is part of standard delivery. It contains lines of control commands which PCSDEFINE can use to generate OPTIONs.

Depending on which of the three predefined OPTION names STD#DIA (default value), STD#BAT or STD#TP is specified, execution of the procedure branches accordingly. After calling the PCSDEFINE utility routine, a set of commands is read in which generates a corresponding OPTION that is stored in the specified parameter file.

Furthermore, any number of additional OPTIONs with individual names can be stored. The OPTION "USER#01" serves this purpose and can be copied under a different name - ranging from (USER#02 .. USER#99) - as often as required.

Note

These commands may be modified before execution in order to create individual OPTIONs with default names. This should be done **only** after a backup copy has been made.
The name of the library element ($PCSCREO) in the library SYSSPR.PCSDEFINE.<version> must not be changed.
The file must be known to IMON in the version <version>, otherwise you must store it under the name $.SYSSPR.PCSDEFINE.027.

The version of the procedure supplied is linked to the PCSDEFINE version, while the version of the OPTION parameter file is linked to the PCS version. Accordingly, the default parameters for VERSION (PCSDEFINE) and the PPF version (OPTION file) are predefined by *STD).

Modified copies with other versions can, of course, also be selected after informing IMON using the appropriate operands.

Format

CREATE-PCS-OPTION

VERSION = ***STD** / <product-version>

,**MONJV** = ***NONE** / <filename 1..54>

,**CPU-LIMIT** = ***JOB-REST** / <integer 1..32767 *seconds*>

,**PCS-PARAM**ETER-FILE = ***STD**(...) / <filename 1..54>

***STD**(...)

PPF-VERSION = ***STD** / <product-version>

,**OPTION-NAME** = ***STD** / **STD#DIA** / **STD#BAT** / **STD#TP** / <name 1..7>

Operands

VERSION =
Product version of the parameter file. If no value has been entered, the delivery default version identical to that of PCSDEFINE is used.

VERSION = *STD
The version set using the /SELECT-PRODUCT-VERSION command is loaded. If no version has been set, then the highest (most recent) version of PCSDEFINE available is used.

VERSION = <product-version>
Specifies the version.

MONJV = *NONE / <filename 1..54>
Name of the monitoring job variable used to monitor the PCSDEFINE routine.

CPU-LIMIT = *JOB-REST / <integer 1..32767 *seconds*>
Defines the maximum CPU time that PCSDEFINE is allowed to run. If no time is entered, PCSDEFINE will execute until the task has terminated.

PCS-PARAMETER-FILE =
Name of the OPTION parameter file. If no value has been entered, then *STD is used.

PCS-PARAMETER-FILE = *STD(...)
An attempt is made internally through IMON to obtain the actual (i.e. the physical) file name. The search is performed using the search criteria SYSSSI (logical ID) and PCS (installation unit). The current PCS version is also evaluated. If these are not available, then *STD is used.

PPF-VERSION =
The assigned version is selected or the corresponding parameter file is searched for.

PPF-VERSION = *STD
The file entered under *STD for IMON is opened. If an explicit version is entered, then IMON must know beforehand of its existence.

PPF-VERSION = <product-version>
The exact designation of the PPF file is entered, for example V2.7. The version number entered must begin with a 'V'. The corresponding parameter file is searched for internally for the product version entered. If the specified version is unknown to IMON, then the PCS parameter file is assumed to be stored under the name $.SYSSSI.PCS.027.

PCS-PARAMETER-FILE = <filename 1..54>
Name of the OPTION parameter file. A file with this name must physically exist.

OPTION-NAME =
OPTION name of the PCS parameter file. This name indicates the set of statements from which a corresponding option will be generated. The parameter settings correspond to those of the default parameter file supplied with delivery.

OPTION-NAME = *STD
The default OPTION is stored in the parameter file.

OPTION-NAME = STD#DIA
An OPTION with the name STD#DIA is stored in the parameter file.

OPTION-NAME = STD#BAT
An OPTION with the name STD#BAT is stored in the parameter file.

OPTION-NAME = STD#TP
An OPTION with the name STD#TP is stored in the parameter file.

OPTION-NAME =<name 1..7>
OPTIONs with any name can be stored in the parameter file. For this purpose, the procedure file supplied with delivery contains an OPTION "USER#01" that can be copied as required (using a different name each time) in order to make individual parameter settings.
To change the settings, modify the name and parameter values in the appropriate sequences between the "IF..." and "END" keywords.

7.3 PCSDEFINE in menu mode

Switching the PCSDEFINE utility routine to menu mode makes managing the PCS parameter set easier.

Two function screens are available to the user for processing PPF files. Using these screens, the user can create, modify and manage options in an interactive dialog. In addition, two help screens containing instructions on how to use the routine can be invoked.

The menu-oriented PCSDEFINE supports the entire range of functions provided by statement mode, offering the user simpler and more efficient operation.

7.3.1 Functional description

Before we go into more detail concerning the use of the menu-driven PCSDEFINE program, we will present an overview of the available screen masks and their functions below.

Overview of the screen masks

Management of the PCS parameter sets is handled interactively, using the following screen masks:

Mask	Function
START MASK	To create and manage OPTIONs
HELP-START MASK	To help users complete the START screen
OPTION MASK	To modify an OPTION
HELP-OPTION MASK	To help users complete the OPTION screen

Note

The exchange of messages between PCSDEFINE and the terminal in menu mode is handled by FHS (Format Handling System).

7.3.2 START screen

When the user exits statement mode (=line mode), the menu-oriented PCSDEFINE displays the START screen, which controls management of the PCS parameter sets.

```
 1  PCSDEFINE 02.7A00                                                START MASK
 2  ------------------------------------------------------------------------------
 3  PPF-filename :      *STD(PPF-VERSION=*STD)
 4
 5  create/modify/show OPTION ........
 6  print              OPTION ........  ........  ........  ........  ........
 7  delete             OPTION ........  ........  ........  ........  ........
 8  check              OPTION ........  ........  ........  ........  ........
 9  copy            TO-OPTION ........
10                FROM-OPTION ........
 :                  FROM-FILE ..................................................
 :                TO-CATEGORY .......
 :              FROM-CATEGORY .......
 :
 :  OPTIONs  :      ........  ........  ........  ........  ........  ........
 :                 ........  ........  ........  ........  ........  ........
 :                 ........  ........  ........  ........  ........  ........
 :                 ........  ........  ........  ........  ........  ........
18
19
20  select function:   type  OPTION / FILE / CATEGORY  name(s)     and    press  DUE
21  continue OPTION list: DUE    help: F1   change to LINE MODE: F2    terminate: F3
22  ------------------------------------------------------------------------------
23
24
25  LTG                                                               TAST
```

Figure 2: START screen

1. The header line contains the name of the utility routine (including its version number) in the top left-hand corner and the name of the screen mask in the right-hand corner.
2. Lines 3 to 20 offer a menu from which a function may be selected or in which data may be entered.
3. Line 3 contains the name and version of the PPF file.
   The default value for the file name is `PPF-filename`:
   `*STD(PPF-VERSION=*STD)`. This means that the default file name known to IMON is assumed, but in this case the PCS version is evaluated.
   The default value for the version is `PPF-VERSION = *STD`. This version is the version which was made known to IMON at the time of installation.
4. Lines 21 and 22 indicate the functions which may be invoked by the function keys.
5. Line 24 is the message line for PCSDEFINE and contains information and messages for the user.

Range of functions

The actual **PPF file** used is specified either by entering the name of the file required - the physical file name - in line 3 or, if *STD is set, by IMON, which evaluates the version number designation and selects the relevant parameter file.

If no explicit version information has been entered and *STD has been specified under IMON, the file with the currently declared default version will be selected.

Notes

The default version number *STD can be overwritten by an explicit version number beginning with "V" (e.g. V02.7A00 or V2.7) and which ends with ")".

"PPF-VERSION" can be abbreviated using "PPF-VERS" or "P-V".

If the default file name (*STD) is specified, the parameter file is expected under $.SYSSSI.PCS.027 if the product was not installed under IMON or if the version number entered is not known to IMON.

If IMON knows the product version number entered and an explicit but invalid version is specified, a corrected entry is expected.

Caution:

The default value *STD for the OPTION parameter file is only valid for the performance administrator and the system administrator, who are authorized to access the files (of different versions), as these persons are the main users of the program and are the most sensible choice of persons to be allowed to modify the parameter file.

Nonprivileged users must enter an explicit file name of a new or existing parameter file. If this file is to be used later in the session, then the administrator for these files must define this file as the new default file for IMON.

By pressing the DUE function key, the file is opened or created. At the same time, PCSDEFINE displays a list of the **OPTIONs** contained in the file in the START window. If the file contains more than 24 PCS parameter sets, you can page through the list using the DUE function key.

The START screen appears as shown below for the PPF file supplied with delivery if the IMON entry is missing:

```
PCSDEFINE 02.7A00                                                    START MASK
------------------------------------------------------------------------------
PPF-filename :      $.SYSSSI.PCS.027

create/modify/show OPTION ........
print              OPTION ........  ........  ........  ........  ........
delete             OPTION ........  ........  ........  ........  ........
check              OPTION ........  ........  ........  ........  ........
copy            TO-OPTION ........
              FROM-OPTION ........
                FROM-FILE ..................................................
             TO-CATEGORY .......
           FROM-CATEGORY .......

OPTIONs  :      STD#BAT   STD#DIA   STD#TP..  ........  ........  ........
                ........  ........  ........  ........  ........  ........
                ........  ........  ........  ........  ........  ........
                ........  ........  ........  ........  ........  ........


select function:   type  OPTION / FILE / CATEGORY  name(s)     and    press  DUE
continue OPTION list: DUE    help: F1   change to LINE MODE: F2    terminate: F3
------------------------------------------------------------------------------


LTG                                                                  TAST
```

Figure 3: START screen

Closing the PPF file

A PPF file is closed either by ending the PCSDEFINE utility routine (implicitly) or by opening a new file (overwriting the file name with a new name).

Notes

– Because PCSDEFINE automatically switches to a display of the OPTION screen when the CREATE, MODIFY or SHOW statement line is selected, only one option name may be entered on this line.

– As in the case of the PCSDEFINE statements, one or more option names may be specified using *ALL is entered. This allows all the options contained in the file to be specified for consistency checking, deletion or printing.

– In one dialog step, only one PCSDEFINE statement may be initiated in addition to changing the PPF file name.

Function key assignments (START screen)

DUE

Pressing the DUE key causes the utility routine to start the selected function, changing to a different screen display if necessary.

After the START screen has appeared, the default value *STD(PPF-VERSION=*STD) is displayed. This data can be overwritten with the current file name.

After pressing the DUE function key, the name of the file to be opened appears, or the file name "$.SYSSSI.PCS.027" appears, flashing and accompanied by a message.

After pressing the DUE function key, the names of the existing OPTIONs are displayed (starting with line 16). If the PPF file opened contains more than 24 PCS parameter lines and no OPTION, FILE or CATEGORY name is entered, then more OPTION names will be listed in the START screen by pressing the DUE function key.

F1

By pressing key F1 the user invokes the HELP START screen, which contains information on how to fill in the START screen.

F2

By pressing F2 the user changes to statement mode.

F3

Pressing F3 terminates the PCSDEFINE utility routine.

7.3.3 OPTION screen

PCS parameters within an option can be modified using the OPTION screen shown in Figure 4.

```
1  PCSDEFINE 02.7A00    PPF : SYSSSI.PCS.027                          OPTION MASK
2  ------------------------------------------------------------------------------
3  STD#DIA       R-D-MAX  6.0     TH-Q  20                 USER-INFO NO    CHECKED
4
5   CAT-NAME ! S-Q-MIN ! S-Q-MAX ! R-D-MIN ! R-D-MAX ! TH-Q ! DURATION ! NEXT-CAT
6  ----------+---------+---------+---------+---------+------+----------+----------
7  DIALOG    !      20 !      40 !       2 !       4 !    0 !      500 ! DIALOG1
8  BATCH     !       0 !      70 !       - !       - !   70 !        - !
9  TP        !       0 !      30 !       1 !       3 !    0 !        - !
10 DIALOG1   !      10 !      30 !       2 !       5 !    0 !     2000 ! BATCH
:  BATCHF    !       0 !      20 !       1 !       3 !    0 !        - !
:  .......   !         !         !         !         !      !          !
:  .......   !         !         !         !         !      !          !
:  .......   !         !         !         !         !      !          !
:  .......   !         !         !         !         !      !          !
:  .......   !         !         !         !         !      !          !
:  .......   !         !         !         !         !      !          !
:  .......   !         !         !         !         !      !          !
18 .......   !         !         !         !         !      !          !
19 .......   !         !         !         !         !      !          !
20 .......   !         !         !         !         !      !          !
21 edit: DUE     help: F1   write and check OPTION: F2     return to START MASK: F3
22
23
24
25 ------------------------------------------------------------------------------
   LTG                                                                  TAST
```

Figure 4: OPTION screen

1. The header line contains the name of the utility routine (including the version number) in the top left-hand corner, in the middle of the line is the name of the PPF file, shortened to a maximum of 40 characters, and the name of the screen mask appears in the right-hand corner.
2. PCS parameter values may be modified in lines 3 to 21 by overwriting the values displayed.
3. The functions invoked by the function keys are listed in line 22.
4. Line 24 is the message line for PCSDEFINE and contains information and messages for the user.

The number of categories within an option can be changed as follows:

1. A new category is created by entering the new name in an empty CAT-NAME field (= '.....').
2. A category is removed from an option by overwriting the name with '.....'.
3. The name of a category is changed by overwriting the old name with a new name.

Function key assignments (OPTION screen)

DUE

Pressing the DUE key causes the screen input to be edited, i.e. the new categories are subsequently output with the default values. The values are not, however, saved in the PCS parameter set.

F1

Pressing function key F1 invokes the HELP OPTION screen, which contains information for the user about the OPTION screen.

F2

Pressing function key F2 causes the PCS parameter set displayed on the screen to be checked and then the values written to the option.

This allows the user to save an option that is not yet complete and continue working on it at a later time.
If input errors are found during the consistency check, the relevant fields flash and the option status remains UNCHECKED.

Otherwise, the option status is set to CHECKED.

F3

By pressing function key F3 the user switches back to the START screen. Note that the parameter values entered on the current screen are not saved automatically. If the current PCS parameter set has not been saved, the message OPTION <optionname> NOT YET SAVED is issued and the new screen will not be displayed until F3 is pressed again.
In all other cases, the START screen will be displayed without a message being issued.

7.3.4 HELP screens

PCSDEFINE provides a help function for the START and OPTION screens. The help functions are called by pressing the [F1] function key. The HELP screen then appears, in which the user can request further information on the current screen.

HELP START screen

The HELP START screen displays more detailed information on the PPF file entry.

```
PCSDEFINE 02.7A00    PPF : SYSSSI.PCS.027                              HELP START
--------------------------------------------------------------------------------

functional description
----------------------
  specify PPF-name    : write filename into line 3 and press  DUE

  select functions    : insert values into '........' fields and press  DUE
                        ( NOTE : with '*ALL' you select all OPTIONs )
:
:
:
:
:
:
:
:



                                                        return to START MASK: F3
--------------------------------------------------------------------------------



LTG                                                           TAST
```

Figure 5: HELP START screen

HELP OPTION screen

The HELP OPTION screen displays more detailed information on the OPTION parameter entries.

```
1  PCSDEFINE 02.7A00    PPF : SYSSSI.PCS.027                          HELP OPTION
2  ------------------------------------------------------------------------------
3
4  functional description
5  ----------------------
6
7  modify OPTION-parameters : write values into parameter fields
8
9  add category    : write category name into empty CAT-NAME field ( ='.......')
10 rename category : write new category name into CAT-NAME field
:  delete category : write '.......' into CAT-NAME field
:
:
:  function key description
:  ------------------------
:
:    DUE : edit values
:    F1  : go to HELP-OPTION MASK
18   F2  : write and check OPTION
19   F3  : return to START MASK without saving the OPTION
20
21                                                    return to OPTION MASK: F3
22 ------------------------------------------------------------------------------
23
24
25 ____________________________________________________________________________
   LTG                                                              TAST
```

Figure 6: HELP OPTION screen

7.3.5 Application example in menu mode

Problem

Create an option DIA#1 with the following parameter settings in a PPF file with the name PPF.DIA:

Global parameters:

R-D-MAX 6

TH-Q 20

Category-specific parameters:

Category	S-Q-		R-D-		TH-Q	DURATION	NEXT-CAT
	MIN	MAX	MIN	MAX			
DIALOG	20	40	2	4	–	500	DIALOG1
BATCH	0	70	–	–	70	–	
TP	0	30	1	3	–	–	
DIALOG1	10	30	2	5	–	2000	BATCH
BATCHF	0	20	1	3	–	–	

Solution

1. Call the PCSDEFINE utility routine

   PSCDEFINE can be called using the /START-PCSDEFINE command.

2. Switch to menu mode

   By pressing the F2 function key, PCSDEFINE switches from statement mode to menu-driven mode. The START screen appears

3. Enter the name of the PPF file and the option name

   In the START screen, the file name `PPF.DIA` is entered for the `PPF-filename` parameter, and `DIA#1`is entered for the OPTION `create/modify/show`. After confirming this with the DUE function key, PCSDEFINE switches implicitly to the OPTION screen.

```
PCSDEFINE 02.7A00                                                    START MASK
------------------------------------------------------------------------------

PPF-filename :      PPF.DIA

create/modify/show OPTION DIA#1...
print              OPTION ........  ........  ........  ........  ........
delete             OPTION ........  ........  ........  ........  ........
check              OPTION ........  ........  ........  ........  ........
copy            TO-OPTION ........
              FROM-OPTION ........
                FROM-FILE ..................................................
              TO-CATEGORY .......
            FROM-CATEGORY .......

OPTIONs  :      ........  ........  ........  ........  ........  ........
                ........  ........  ........  ........  ........  ........
                ........  ........  ........  ........  ........  ........
                ........  ........  ........  ........  ........  ........


select function:   type  OPTION / FILE / CATEGORY  name(s)     and    press  DUE
continue OPTION list: DUE    help: F1   change to LINE MODE: F2    terminate: F3
------------------------------------------------------------------------------


LTG                                                                  TAST
```

Figure 7: START screen

4. Enter the categories in the OPTION screen
   The category names DIALOG1 and BATCHF are also entered in this screen and are confirmed by pressing the F2 function key.

```
PCSDEFINE 02.7A00    PPF : PPF.DIA                                 OPTION MASK
------------------------------------------------------------------------------
DIA#1       R-D-MAX  6.0     TH-Q  20                 USER-INFO NO    CHECKED

 CAT-NAME ! S-Q-MIN ! S-Q-MAX ! R-D-MIN ! R-D-MAX ! TH-Q ! DURATION ! NEXT-CAT
----------+---------+---------+---------+---------+------+----------+----------
DIALOG    !       0 !     100 !     0.0 !     0.0 !    0 !        - !
BATCH     !       0 !     100 !     0.0 !     0.0 !    0 !        - !
TP        !       0 !     100 !     0.0 !     0.0 !    0 !        - !
DIALOG1   !         !         !         !         !      !          !
BATCHF    !         !         !         !         !      !          !
.......   !         !         !         !         !      !          !
.......   !         !         !         !         !      !          !
.......   !         !         !         !         !      !          !
.......   !         !         !         !         !      !          !
.......   !         !         !         !         !      !          !
.......   !         !         !         !         !      !          !
.......   !         !         !         !         !      !          !
.......   !         !         !         !         !      !          !
.......   !         !         !         !         !      !          !
.......   !         !         !         !         !      !          !
edit: DUE     help: F1   write and check OPTION: F2     return to START MASK: F3
------------------------------------------------------------------------------


LTG                                                                  TAST
```

Figure 8: OPTION screen

5. Enter the PCS parameter values in the OPTION screen

   Additional PCS parameter values for the DIALOG, BATCH, TP, DIALOG1 and BATCHF parameters are entered in this screen.

```
PCSDEFINE 02.7A00    PPF : PPF.DIA                                  OPTION MASK
------------------------------------------------------------------------------
DIA#1       R-D-MAX  6.0     TH-Q  20                 USER-INFO NO    CHECKED

 CAT-NAME ! S-Q-MIN ! S-Q-MAX ! R-D-MIN ! R-D-MAX ! TH-Q ! DURATION ! NEXT-CAT
----------+---------+---------+---------+---------+------+----------+----------
DIALOG    !      20 !      40 !       2 !       4 !    0 !      500 ! DIALOG1
BATCH     !       0 !      70 !       - !       - !   70 !        - !
TP        !       0 !      30 !       1 !       3 !    0 !        - !
DIALOG1   !      10 !      30 !       2 !       5 !    0 !     2000 ! BATCH
BATCHF    !       0 !      20 !       1 !       3 !    0 !        - !
.......   !         !         !         !         !      !          !
.......   !         !         !         !         !      !          !
.......   !         !         !         !         !      !          !
.......   !         !         !         !         !      !          !
.......   !         !         !         !         !      !          !
.......   !         !         !         !         !      !          !
.......   !         !         !         !         !      !          !
.......   !         !         !         !         !      !          !
.......   !         !         !         !         !      !          !
edit: DUE     help: F1   write and check OPTION: F2     return to START MASK: F3


LTG                                                              TAST
```

Figure 9: OPTION screen

Note

Steps 4 and 5 may be combined.

6. Save the entered parameter values

   The values entered in the OPTION screen may then be saved by pressing the F2 function key.

7. Return to the START screen

   Pressing the F3 function key causes PCSDEFINE to return to the START screen.

```
PCSDEFINE 02.7A00                                                    START MASK
------------------------------------------------------------------------------
PPF-filename :      PPF.DIA

create/modify/show OPTION ........
print              OPTION ........  ........  ........  ........  ........
delete             OPTION ........  ........  ........  ........  ........
check              OPTION ........  ........  ........  ........  ........
copy            TO-OPTION ........
              FROM-OPTION ........
                FROM-FILE ..................................................
              TO-CATEGORY .......
            FROM-CATEGORY .......

OPTIONs  :     DIA#1........  ........  ........  ........  ........  ........
                 ........  ........  ........  ........  ........  ........
                 ........  ........  ........  ........  ........  ........
                 ........  ........  ........  ........  ........  ........


select function:   type  OPTION / FILE / CATEGORY  name(s)     and    press  DUE
continue OPTION list: DUE    help: F1   change to LINE MODE: F2    terminate: F3
------------------------------------------------------------------------------


LTG                                                                  TAST
```

Figure 10: START screen

8. Terminate PCSDEFINE

   The program is terminated by pressing the F3 function key.

7.4 PCSDEFINE in statement mode

After PCSDEFINE has been started, the utility routine is in statement mode (see section “Starting and terminating PCSDEFINE” on page 68).

7.4.1 Functional description

Execution of the routine is controlled by statements read in from SYSDTA. These statements are described in the following sections. They can be divided into four groups:

1. statements for carrying out file operations (create, open, close, information function, program termination: section “File operations” on page 88)
2. statements for processing OPTION entries (create, delete, modify, output, check: section “Statements for processing PCS parameter sets (options)” on page 89)
3. statements for processing CATEGORY entries (add, delete, modify, output, check: section “Statements for processing PCS parameter sets (category)” on page 90)
4. statements for copying OPTION and CATEGORY entries (section “Copy statements and use of SOURCEFILEs” on page 93)

Files used by PCSDEFINE:

1. M A I N F I L E is the actual work file on which the above-mentioned functions, such as create, delete, modify, check entries, are carried out. MAINFILE must always be opened explicitly by means of the OPEN statement before OPTION or CATEGORY entries can be processed. It is automatically assigned the file link name MAINLINK via an implicit call of the FILE macro. This means that there is no need for a /SET-FILE-LINK command before the PCSDEFINE routine is executed.
2. The S O U R C E F I L E file is only required when entries from a different PPF are to be copied to the MAINFILE (cf. copy statements in section “Copy statements and use of SOURCEFILEs” on page 93). In this case, one (and only one) SOURCEFILE may be opened in addition; this is automatically assigned the file link name SRCELINK. At the start, or for applications which are not too complex, systems support can easily get by without a SOURCEFILE. The functional scope of PCS will not be affected in any way.

7.4.2 Statement structure and syntax conventions

The representation of the PCSDEFINE statements follows the guidelines for SDF commands/statements. However, the statements cannot be checked using the SDF functionality: there are no overviews, no guided dialog, no correction dialog via SDF; when the statements are entered they are not preceded by a double slash.

Input is program-driven, even in statement mode. In other words, an asterisk (*) is supplied whenever a statement is entered.

Notes on statement entry:

1. All operands of a statement are keyword operands and may be input in any order when the keywords are used.
2. Operands that are omitted are assigned a default value. In the statement syntax descriptions, the default values are underlined (cf. also the example under point 4 below: USER-INFORMATION).
3. All statement names, keyword operands, and operand values representing keywords (e.g. *STD, UNCHANGED, *NONE) may be abbreviated as desired as long as they remain unambiguous. This also applies to hyphen-separated substrings in statement or parameter names.
4. Statements must not be longer than one line; continuation is not possible. When lists are output, it may be necessary to enter several such statements, depending on the length of the names of the list elements. This situation is not taken into account in the specification of the maximum number of list elements in the syntax descriptions.

   Example

   COPY-OPTION FROM-FILE-NAME = <filename>,...

   Valid abbreviations for the above statement include:

   CO-OF-F=<filename>,...(minimal form)

   COPY-OF-F-N=<filename>,...

   CO-OPTF-FILE=<filename>,...

   The abbreviation C-O for the statement name is ambiguous and will be rejected because there is no way of telling whether it refers to the CREATE-OPTION or CHECK-OPTION statement.

5. All keyword operands can be left out and the operand values entered as positional operands. This may only be done if they are entered in the correct order (i.e. in the order in which the operands appear in the syntax descriptions). A missing operand must be represented by a comma.

   Example

   CR OPTION#1,(,50),,,(BATCH1,DIALOG1)

   corresponds to

   ```
   CREATE-OPTION     OPTION-NAME          = OPTION#1,
                     SYSTEM-PARAMETER     = (R-D-MAX    = STD,
                                             T-QUOTA     = 50),
                     USER-INFORMATION     = NO,
                     CATEGORY             = (BATCH1, DIALOG1)
   ```

6. Keyword and positional operands may be entered in any mix. Note that the order of the positional operands is significant, whereas keyword operands may be input in random order.

   It follows that the example in point 5 above may also be written as:

   CR OPTION#1, C = (BATCH1,DIALOG1), S = (T=50, R=STD)

7.4.3 File operations

Opening a PCS parameter file (PPF)

The OPEN-FILE statement opens a file in PPF format (other file types will be rejected).

Closing a PCS parameter file (PPF)

The CLOSE-FILE statement closes a MAINFILE and/or a SOURCEFILE.

Requesting information

The HELP statement (without operands) displays a list of all available PCSDEFINE statements and the corresponding minimal abbreviations.

Terminating the routine

The END statement (without operands) terminates the PCSDEFINE routine. Before this is done, however, any parameter files still open are closed (by implicitly calling the statement CLOSE-FILE FILE-NAME = *ALL).

The CLOSE statement in turn causes the CHECK function to be carried out on all MAINFILE options that have not yet been checked for consistency.

7.4.4 Statements for processing PCS parameter sets (options)

In the following sections all the statements for processing, outputting and checking options for consistency are described. The first operand of each statement is OPTION-NAME, which specifies the name (<optname>) of the relevant option (in some cases a list of names is permitted). If OPTION-NAME = *ALL is specified, the statement operates on all the MAINFILE options. statements can only be executed after the MAINFILE has been opened.

Conventions for option names

An option name may have up to 8 characters in any combination of letters, numbers and the special characters '@', '$' and '#'. The first character must always be either a letter or the special character '$'.

Creating and modifying options

If the predefined standard options prove to be unsuitable or inadequate for achieving optimal system operation, systems support staff can define their own parameter sets or modify existing ones. The statements CREATE-OPTION and MODIFY-OPTION are available for this purpose. They are used to specify the **global** PCS parameters.

When the list of operands is processed, a check is made to ensure that the individual operands are correctly formatted. A more extensive consistency check of the contents of the entries and operands (operand groups) is not carried out at the time the option is created or modified. If such a check is required, it must be initiated explicitly by way of the CHECK-OPTION statement.

If, however, systems support attempts to close a MAINFILE or terminate the PCSDEFINE routine via the END statement without first having checked all the processed entries by issuing the CHECK-OPTION or CHECK-CATEGORY statement, an implicit CHECK call is executed for the entries involved. This is intended to ensure that every PPF is maintained in a consistent state.

Checking options

The CHECK-OPTION statement enables the format and contents of PCS parameter sets to be checked for consistency and validity. In particular, a check is made to determine whether there is a corresponding CATEGORY entry in the PPF for each category reference in the entry.

Creating options

The CREATE-OPTION statement generates a new OPTION entry in a PPF. The PPF must already have been opened explicitly using the OPEN-FILE statement.

Deleting options

The DELETE-OPTION statement deletes OPTION entries from a MAINFILE. In addition, all the CATEGORY entries associated with the specified option are deleted.

Modifying PCS parameter sets (options)

The MODIFY-OPTION statement is used to modify the parameters of existing options (i.e. options already created by means of CREATE-OPTION).

Displaying options

The SHOW-OPTION statement is used to output OPTION entries from the PPF to SYSOUT in edited form.

7.4.5 Statements for processing PCS parameter sets (category)

In the following sections the statements for processing, outputting and checking categories for consistency are described. The first operand of each statement is CATEGORY-NAME, which specifies the name (<name>) of the relevant category (in some cases a list of names is permitted). If CATEGORY-NAME = *ALL is specified, the statement operates on all categories of the option. Statements can only be executed after the MAINFILE has been opened.

Conventions for category names

A category name may have up to 7 characters in any combination of letters, numbers and the special characters '@', '$' and '#'. The first character must always be a letter or the special character '$'.

Creating and modifying categories

The statements ADD-CATEGORY and MODIFY-CATEGORY are used to add CATEGORY-type parameter sets to an option or to modify existing ones. These statements must be preceded by the OPEN-FILE statement for a MAINFILE and, if necessary, the CREATE-OPTION statement.

The CATEGORY-NAME operand of the ADD/MODIFY-CATEGORY statements specifies the name of the category; this operand is mandatory.

The OPTION-NAME operand of the ADD/MODIFY-CATEGORY statements specifies which option the category parameter set to be created or modified belongs to. The OPTION-NAME operand may be omitted if the operation to be performed is to be carried out on a category parameter set belonging to the same option that has already been specified in a previous statement. All other parameters (DURATION, NEXT-CATEGORY, etc.) are optional.

When the list of operands is processed, a check is made to ensure that the individual operands are correctly formatted. A more extensive consistency check of the contents of the entries and operands (operand groups) is not carried out at the time the category is created or modified. If such a check is required, it must be initiated explicitly with the CHECK-CATEGORY statement. If, however, systems support attempts to close a MAINFILE or terminate the PCSDEFINE routine via the END statement without first having checked all the processed entries by issuing the CHECK-OPTION or CHECK-CATEGORY statement, a consistency check is initiated implicitly.

Adding CATEGORY entries to options

The ADD-CATEGORY statement creates a new CATEGORY entry in a PPF and assigns it to a specific option. The following preconditions must be met in order to do this:

1. The MAINFILE must be open.
2. An option with the name <name> must exist in this MAINFILE.
3. This option must contain a reference to the category <name> that is to be created, i.e. it must have been created by means of

   CREATE-OPTION OPTION-NAME = <name>,
   CATEGORY = (<name>, ...),
   : :

   or modified accordingly using MODIFY-OPTION.

Checking CATEGORY entries

The CHECK-CATEGORY statement checks the individual CATEGORY entries to see whether the parameter values in the parameter groups are consistent and whether the parameters lie within the specified limits.

CHECK-OPTION executes an implicit CHECK-CATEGORY for all the categories defined in the PCS parameter set.

Modifying CATEGORY entries

The MODIFY-CATEGORY statement is used to modify the parameters of existing categories.

OPTION-NAME is optional if an option name has already been specified in a previous statement.

The MODIFY-CATEGORY statement can be used for all predefined standard categories with the exception of SYS, as well as for all categories defined via ADD-CATEGORY. The parameters of the SYS category are preset and must not be changed.

Removing CATEGORY entries

The REMOVE-CATEGORY statement is used to delete CATEGORY parameter sets; the associated reference is also removed from the option. Note, however, that only non-standard categories may be deleted (i.e. only those defined by the user).

Displaying CATEGORY entries

The SHOW-CATEGORY statement is used to output CATEGORY entries to SYSOUT in edited form.

7.4.6 Copy statements and use of SOURCEFILEs

Copying options

An existing PCS parameter set (option) can be used as the basis for creating a new (additional) parameter set. The COPY-OPTION statement described below allows the duplication of an option within a MAINFILE or the copying of an option from the SOURCEFILE into the MAINFILE for this purpose. All CATEGORY entries in the source option are also copied.

Copying CATEGORY entries

In addition to modifying individual PCS parameters via the MODIFY-OPTION statement or modifying the parameters of a CATEGORY entry using the MODIFY-CATEGORY statement, it is also possible to copy complete CATEGORY entries within an option or from one option to another.

Using the COPY-CATEGORY statement, it is possible to create new CATEGORY-type entries by copying existing entries.

7.4.7 PCSDEFINE statements

This chapter describes, in alphabetical order, the statements available for use during a PCSDEFINE run.

Overview of functions

Command	Function
ADD-CATEGORY	Adds entries of the type CATEGORY to an OPTION
CHECK-CATEGORY	Checks CATEGORYs
CHECK-OPTION	Checks OPTIONs
CLOSE-FILE	Closes a PCS parameter file (PPF)
COPY-CATEGORY	Copies CATEGORYs
COPY-OPTION	Copies OPTIONs
CREATE-OPTION	Creates OPTIONs
DELETE-OPTION	Deletes OPTIONs
END	Terminates the PCSDEFINE routine
HELP	Help function
MODIFY-CATEGORY	Modifies entries of the type CATEGORY
MODIFY-OPTION	Modifies PCS parameter sets (OPTIONs)
OPEN-FILE	Opens a PCS parameter file (PPF)
REMOVE-CATEGORY	Deletes CATEGORY entries
SHOW-CATEGORY	Displays CATEGORY entries
SHOW-OPTION	Displays OPTION entries

ADD-CATEGORY
Add CATEGORY entries to option

The ADD-CATEGORY statement creates a new CATEGORY entry in a PPF and assigns it to a specific option. The following preconditions must be met in order to do this:

1. The MAINFILE must be open.
2. An option with the name <name> must exist in this MAINFILE.
3. This option must contain a reference to the category <name> that is to be created, i.e. it must have been created by means of

   CREATE-OPTION OPTION-NAME = <name>,
   CATEGORY = (<name>, ...),
   : :

   or modified accordingly using MODIFY-OPTION.

Format

ADD-CATEGORY

CATEGORY-NAME = <name 1..7>
,**OPTION-NAME** = <name 1..8>
,**DURATION** = **UNLIMITED** / <integer 0..100000>
,**NEXT-CATEGORY** = ***NONE** / <name 1..7>
,**THROUGHPUT-QUOTA** = **STD** / <integer 0..100>
,**RELATIVE** = (...)
(...)
SERVICE-QUOTA = (...)
(...)
MIN = **STD** / <integer 0..99>
,**MAX** = **STD** / <integer 1..100>
,**REQUEST-DELAY** = **UNCONTROLLED** / (...)
(...)
MIN = **STD** / <integer 0..100>
,**MAX** = **STD** / <integer 0..100>

Operands

CATEGORY-NAME = <name 1..7>
Specifies the name of the new category.

OPTION-NAME = <name 1..8>
The newly created category is assigned to the specified option.

DURATION = UNLIMITED
Categories are not switched automatically.

DURATION = <integer 0..100000>
Specifies the number of service units per request after which an automatic category switch takes place.

NEXT-CATEGORY = *NONE
There is no switch to the next category.

NEXT-CATEGORY = <name 1..7>
Specifies the name of the next category, i.e. the category to which the automatic switch is made.

THROUGHPUT-QUOTA = STD / <integer 0..100>
Defines a percentage value which is used to express the relationship between response-time optimization and throughput optimization of the system.
Default value: 20% (full response-time optimization).
The value THROUGHPUT-QUOTA = 100 ensures exclusively throughput-oriented operation; THROUGHPUT-QUOTA = 0 ensures exclusively response time-oriented operation.

RELATIVE = (...)
Comprises the operands which define the capacity of the category as a percentage of the system performance capacity.

SERVICE-QUOTA = (...)
Defines the capacity unit SERVICE-QUOTA.

MIN = STD / <integer 0..99>
Defines the minimum percentage of the system capacity that is to be reserved for the category.
Default value: 0.

MAX = STD / <integer 1..100>
Defines the maximum percentage of the system capacity that is to be reserved for the category.
Default value: 100.

REQUEST-DELAY =
Defines the dilation range REQUEST-DELAY.

REQUEST-DELAY = UNCONTROLLED
The upper limit of the dilation factor is not monitored (corresponds to the value 0).

REQUEST-DELAY = (...)

MIN = STD / <integer 0..100>
Defines the minimum delay for requests in the category.
Default value: 1 if REQUEST-DELAY-MAX has been specified, otherwise the lower limit of the dilation factor is not monitored (corresponds to the value 0).

MAX = STD / <integer 0..100>
Defines the maximum delay for requests in the category.
Default value: the upper limit of the dilation factor is not monitored (corresponds to the value 0).

Example

```
CREATE-OPTION   OPTION-NAME      = SESSION1,
                     :                :
                CATEGORY         = (DIALOG1)
ADD-CATEGORY    CATEGORY-NAME    = DIALOG1,
                OPTION-NAME      = SESSION1,
                     :                :
```

The above statements represent a complete definition of the option with the name SESSION1 and the categories SYS, DIALOG, BATCH, TP and DIALOG1.

As the standard categories are predefined, they do not need to be created by means of the ADD-CATEGORY statement. If the parameters in these categories are to be modified, this can be done using the MODIFY-CATEGORY statement.

CHECK-CATEGORY
Check CATEGORY entries

The CHECK-CATEGORY statement checks the individual CATEGORY entries to see whether the parameter values in the parameter groups are consistent and whether the parameters lie within the specified limits.

CHECK-OPTION executes an implicit CHECK-CATEGORY for all the categories defined in the PCS parameter set.

Format

CHECK-CATEGORY
CATEGORY-NAME = ***ALL** / list-poss(15): <name 1..7> ,**OPTION-NAME** = <name 1..8>

Operands

CATEGORY-NAME = *ALL
All CATEGORY entries are checked.

CATEGORY-NAME = list-poss(15): <name 1..7>
The specified CATEGORY entries are checked.

OPTION-NAME = <name 1..8>
The CATEGORY entries for the option with the specified name are checked.

CHECK-OPTION
Check OPTION entries

The CHECK-OPTION statement enables the format and contents of PCS parameter sets to be checked for consistency and validity. In particular, a check is made to determine whether there is a corresponding CATEGORY entry in the PPF for each category reference in the entry

Format

CHECK-OPTION
OPTION-NAME = ***ALL** / list-poss: <name 1..8>

Operands

OPTION-NAME = *ALL / list-poss: <name 1..8>
Either all options or those with the specified names are checked.

Note

In addition, the CATEGORY entries associated with each of the specified options are checked for consistency. This is done by an implicit call of the statement

CHECK-CATEGORY CATEGORY-NAME = *ALL

for the option <name>, for the options in the list or for all the options (*ALL) in the PPF.

CLOSE-FILE
Close PCS parameter file (PPF)

The CLOSE-FILE statement closes a MAINFILE and/or a SOURCEFILE.

Format

CLOSE-FILE
FILE-NAME = **<u>*ALL</u>** / ***MAIN** / ***SOURCE** / <filename 1..54>

Operands

FILE-NAME =
Specifies the name(s) of the file(s) to be closed.

FILE-NAME = <u>*ALL</u>
All files, including the MAINFILE and the SOURCEFILE, are closed.

FILE-NAME = *MAIN
The MAINFILE is closed.

FILE-NAME = *SOURCE
The SOURCEFILE is closed.

FILE-NAME = <filename 1..54>
The file with the specified name is closed.

Note

Before a MAINFILE is closed, the following statement is called implicitly (see below):

CHECK-OPTION OPTION-NAME = *ALL

If the file contains inconsistent OPTIONs, an error message is issued; the file will still be closed, however.

COPY-CATEGORY
Copy CATEGORY entries

In addition to modifying individual PCS parameters via the MODIFY-OPTION statement or modifying the parameters of a CATEGORY entry using the MODIFY-CATEGORY statement, it is also possible to copy complete CATEGORY entries within an option or from one option to another.

Using the COPY-CATEGORY statement, it is possible to create new CATEGORY-type entries by copying existing entries.

Format

COPY-CATEGORY

TO-CATEGORY-NAME = <name 1..7>

,**FROM-CATEGORY-NAME** = <name 1..7>

,**TO-OPTION-NAME** = <name 1..8>

,**FROM-OPTION-NAME** = <name 1..8>

,**FROM-FILE-NAME** = ***SOURCE** / ***MAIN** / <filename 1..54>

Operands

TO-CATEGORY-NAME = <name 1..7>
Defines the target category; this category is entered in the MAINFILE under the name specified here.

FROM-CATEGORY-NAME = <name 1..7>
Specifies the name of the source category.

TO-OPTION-NAME = <name 1..8>
Defines the target option; this category is entered in the MAINFILE under the name specified here.

FROM-OPTION-NAME = <name 1..8>
Specifies the name of the source option.

FROM-FILE-NAME =
Defines the source file.

FROM-FILE-NAME = *SOURCE
The source entry is to be copied from the SOURCEFILE to the MAINFILE. The SOURCEFILE must already have been opened explicitly by an OPEN statement or by another COPY statement.

FROM-FILE-NAME = *MAIN
The source option is to be duplicated within the MAINFILE.

FROM-FILE-NAME = <filename 1..54>
The file with the specified name is opened as the new SOURCEFILE. If another SOURCEFILE is still open, it will first be closed implicitly.

COPY-OPTION
Copy OPTION entries

An existing PCS parameter set (option) can be used as the basis for creating a new (additional) parameter set. The statement described below allows the duplication of an option within a MAINFILE or the copying of an option from the SOURCEFILE into the MAINFILE for this purpose. All CATEGORY entries in the source option are also copied.

Format

COPY-OPTION
TO-OPTION-NAME = <name 1..8> ,**FROM-OPTION-NAME** = <name 1..8> ,**FROM-FILE-NAME** = ***SOURCE** / ***MAIN** / <filename 1..54>

Operands

TO-OPTION-NAME = <name 1..8>
This operand specifies the target option; it is entered in the MAINFILE under this name.

FROM-OPTION-NAME = <name 1..8>
Specifies the name of the source option.

FROM-FILE-NAME =
Defines the source file.

FROM-FILE-NAME = *SOURCE
The source entry is to be copied from the SOURCEFILE to the MAINFILE. The SOURCEFILE must already have been opened explicitly by an OPEN statement or by another COPY statement.

FROM-FILE-NAME = *MAIN
The source option is to be duplicated within the MAINFILE.

FROM-FILE-NAME = <filename 1..54>
The file with the specified name is opened as the new SOURCEFILE. If another SOURCEFILE is still open, it will first be closed implicitly.

CREATE-OPTION
Create OPTION entries

The CREATE-OPTION statement generates a new OPTION entry in a PPF. The PPF must already have been opened explicitly using the OPEN-FILE statement.

Format

CREATE-OPTION

OPTION-NAME = <name 1..8>

,**SYSTEM-PARAMETER** = (...)

(...)

REQUEST-DELAY-MAX = **STD** / <integer 0..100>,

,**THROUGHPUT-QUOTA** = **STD** / <integer 0..100>

,**USER-INFORMATION** = **NO** / **YES**

,**CATEGORY** = ***STD** / list-poss(12): <name 1..7>

Operands

OPTION-NAME = <name 1..8>
Specifies the name of the option that is to be entered in the PCS parameter file.

SYSTEM-PARAMETER = (...)
When the new option entry is created, the following global system parameters can be set in addition:

REQUEST-DELAY-MAX = STD
Specifies a threshold value for the maximum dilation of all tasks in the permitted range which belong to a category with a dilation area.
The default value depends on the value of the THROUGHPUT-QUOTA operand and is calculated using the following formula:

```
5 + (THROUGHPUT-QUOTA) / 20
```

REQUEST-DELAY-MAX = <integer 0..100>
This operand is used to set the optimal multiprogramming factor.

THROUGHPUT-QUOTA = STD / <integer 0..100>
Defines a percentage value which is used to express the relationship between response-time optimization and throughput optimization of the system.
Default value: 20% (full response-time optimization).
The value THROUGHPUT-QUOTA = 100 ensures exclusively throughput-oriented operation; THROUGHPUT-QUOTA = 0 ensures exclusively response time-oriented operation.

USER-INFORMATION = NO / YES
Specifies whether information on PCS is to be displayed for the user.

CATEGORY = *STD
Standard categories are created.

CATEGORY = list-poss(12): <name 1..7>
In addition to the standard categories other categories with the specified names can be created.

DELETE-OPTION
Delete OPTION entries

The DELETE-OPTION statement deletes OPTION entries from a MAINFILE. In addition, all the CATEGORY entries associated with the specified option are deleted.

Format

DELETE-OPTION
OPTION-NAME = **<u>*NONE</u>** / ***ALL** / list-poss(12): <name 1..8>

Operands

OPTION-NAME = <u>*NONE</u>
No option entries are deleted.
In contrast to other statements, the default value for OPTION-NAME is *NONE. This is to prevent all the OPTION entries being deleted by mistake.

OPTION-NAME = *ALL
All PPF option entries are deleted.

OPTION-NAME = list-poss(12): <name 1..8>
The specified option entries are deleted from the MAINFILE.

END
Terminate PCSDEFINE

The END statement (without operands) terminates the PCSDEFINE routine. Before this is done, however, any parameter file(s) still open is (are) closed (by implicitly calling the statement CLOSE-FILE FILE-NAME = *ALL).

The CLOSE statement in turn causes the CHECK function to be carried out on all MAINFILE OPTIONs that have not yet been checked for consistency.

Format

END

HELP
Help function

The HELP statement (without operands) displays a list of all available PCSDEFINE statements and the corresponding minimal abbreviations.

Format

HELP

MODIFY-CATEGORY
Modify CATEGORY entries

The MODIFY-CATEGORY statement is used to modify the parameters of existing categories.

OPTION-NAME is optional if an option name has already been specified in a previous statement.

The MODIFY-CATEGORY statement can be used for all predefined standard categories with the exception of SYS, as well as for all categories defined via ADD-CATEGORY. The parameters of the SYS category are preset and must not be changed.

Format

MODIFY-CATEGORY

CATEGORY-NAME = <name 1..7>
,**OPTION-NAME** = <name 1..8>
,**DURATION** = **UNCHANGED** / <integer 0..100000>
,**NEXT-CATEGORY** = ***UNCHANGED** / ***NONE** / <name 1..7>
,**THROUGHPUT-QUOTA** = **UNCHANGED** / <integer 0..100>
,**RELATIVE** = **SERVICE-QUOTA**(...) / **REQUEST-DELAY**(...)
SERVICE-QUOTA(...)
MIN = **UNCHANGED** / <integer 1..99>
,**MAX** = **UNCHANGED** / <integer 0..100>
,**REQUEST-DELAY**(...)
MIN = **UNCHANGED** / <integer 0..100>
,**MAX** = **UNCHANGED** / <integer 0..100>

Operands

CATEGORY-NAME = <name 1..7>
Specifies the name of the category whose parameters are to be modified.

OPTION-NAME = <name 1..8>
The category is assigned to the specified option.

DURATION = UNCHANGED
The DURATION value is not changed.

DURATION = <integer 0..100000>
Modifies the number of service units per request after which an automatic category switch takes place.

NEXT-CATEGORY = *UNCHANGED
The specification for NEXT-CATEGORY remains unchanged.

NEXT-CATEGORY = *NONE
No switch is made to the next category.

NEXT-CATEGORY = <name 1..7>
Specifies the name of the next category to which a switch is made when an automatic category switch takes place.

THROUGHPUT-QUOTA = UNCHANGED
The THROUGHPUT-QUOTA value remains unchanged.

THROUGHPUT-QUOTA = <integer 0..100>
Defines a percentage value which is used to express the relationship between response-time optimization and throughput optimization of the system.
The value THROUGHPUT-QUOTA = 100 ensures exclusively throughput-oriented operation; THROUGHPUT-QUOTA = 0 ensures exclusively response time-oriented operation.

RELATIVE = (...)
Comprises the operands which define the capacity of the category as a percentage of the system performance capacity.

SERVICE-QUOTA = (...)
Modifies the capacity unit SERVICE-QUOTA.

MIN = UNCHANGED
The SERVICE-QUOTA-MIN value remains unchanged.

MIN = <integer 0..99>
Defines the minimum percentage of the system capacity that is to be reserved for the category.

MAX = UNCHANGED
The SERVICE-QUOTA-MAX value remains unchanged.

MAX = <integer 1..100>
Defines the maximum percentage of the system capacity that is to be reserved for the category.

REQUEST-DELAY =
Modifies the dilation range REQUEST-DELAY.

MIN = <u>UNCHANGED</u>
The REQUEST-DELAY-MIN value remains unchanged.

MIN = <integer 0..100>
Defines the minimum delay for requests in the category.

MAX = <u>UNCHANGED</u>
The REQUEST-DELAY-MAX value remains unchanged.

MAX = <integer 0..100>
Defines the maximum delay for requests in the category.

MODIFY-OPTION
Modify PCS parameter sets (options)

The MODIFY-OPTION statement is used to modify the parameters of existing options (i.e. options already created by means of CREATE-OPTION).

Format

MODIFY-OPTION

OPTION-NAME = <name 1..8>

,**SYSTEM-PARAMETER** = (...)

(...)

REQUEST-DELAY-MAX = **UNCHANGED** / <integer 0..100>

,**THROUGHPUT-QUOTA** = **UNCHANGED** / <integer 0..100>

,**USER-INFORMATION** = **UNCHANGED** / **NO** / **YES**

,**CATEGORY** = ***UNCHANGED** / ***STD** / list-poss(12): <name 1..7>

Operands

OPTION-NAME = <name 1..8>
Defines the name of the option.
OPTION-NAME is optional if a name has already been defined by a preceding CREATE-OPTION or MODIFY-OPTION statement.

SYSTEM-PARAMETER = (...)
Specifies all the global system parameters that are to be modified.

REQUEST-DELAY-MAX = UNCHANGED
The threshold value for the maximum dilation of all the tasks in the permitted range and which belong to a category with a dilation range remain unchanged.
The default value depends on the value of the THROUGHPUT-QUOTA operand and is calculated using the following formula:

```
5 + (THROUGHPUT-QUOTA) / 20
```

REQUEST-DELAY-MAX = <integer 0..100>
Serves to set the optimum multiprogramming factors.

THROUGHPUT-QUOTA = UNCHANGED
The set percentage value, which expresses the relationship between response-time optimization and throughput optimization of the system, remains unchanged.
Default value: 20%.

THROUGHPUT-QUOTA = <integer 0..100>
Sets a new percentage value.
The value THROUGHPUT-QUOTA = 100 ensures exclusively throughput-oriented operation; THROUGHPUT-QUOTA = 0 ensures exclusively response time-oriented operation.

USER-INFORMATION = UNCHANGED / NO / YES
Specifies whether information on PCS is to be made available to the user.

CATEGORY = *UNCHANGED
The defined categories are not changed.

CATEGORY = *STD
Standard categories are created.

CATEGORY = list-poss(12): <name 1..7>
In addition to the standard categories, other categories with the specified names can be created.

OPEN-FILE
Open PCS parameter file (PPF)

The OPEN-FILE statement opens a file in PPF format (other file types will be rejected)

Format

OPEN-FILE

FILE-NAME = **<u>*STD</u>**(...) / <filename 1..54>

<u>*STD</u>(...)

PPF-VERSION = ***STD**

,**MODE** = **<u>MAIN</u>** / **SOURCE**

,**ACCESS** = **<u>WRITE</u>** / **READ**

Operands

FILE-NAME =
Specifies the name of the PCS parameter file (PPF).

FILE-NAME = <u>*STD</u>(...)
An OPTION parameter file with the default version is opened, upon installation this is the current PCS version.

PPF-VERSION = *STD
The standard PPF with the name SYSSSI.PCS.027 is opened. If a version is explicitly stated, then IMON must know of this version.

FILE-NAME = <filename 1..54>
If a file with the specified name already exists, it is opened. If the file does not exist, a new file is created as a MAINFILE and initialized in PPF format. Any attempt to open a non-existent file as a SOURCEFILE leads to an error message.

MODE =
Defines the processing mode.

MODE = <u>MAIN</u>
The specified file is opened as a MAINFILE. All operations for processing OPTION and CATEGORY entries can now be performed.

MODE = SOURCE
The specified file is opened as a SOURCEFILE. A SOURCEFILE can only be used as a source file for the definition of new PPF records for a MAINFILE (cf. the COPY-OPTION and COPY-CATEGORY statements). The combination MODE=SOURCE, FILE-NAME=*STD is not permitted, as no default name is provided for SOURCEFILE.

ACCESS =
Defines the access mode for PPF.

ACCESS = <u>WRITE</u>
Specifying this default value allows the parameter values to be processed. This only makes sense for MAINFILEs and is only permitted for such files.

ACCESS = READ
PPF is opened for read access. This prevents parameter values from being changed or records from being deleted by mistake.

REMOVE-CATEGORY
Delete CATEGORY entries

The REMOVE-CATEGORY statement is used to delete CATEGORY parameter sets; the associated reference is also removed from the option. Note, however, that only non-standard categories may be deleted (i.e. only those defined by the user).

Format

REMOVE-CATEGORY
CATEGORY-NAME = ***NONE** / ***ALL** / list-poss(15): <name 1..7> ,**OPTION-NAME** = <name 1..8>

Operands

CATEGORY-NAME = *NONE
No CATEGORY parameter sets are deleted.

CATEGORY-NAME = *ALL
All CATEGORY parameter sets are deleted.

CATEGORY-NAME = list-poss(15): <name 1..7>
The CATEGORY parameter sets with the specified names are deleted.

OPTION-NAME = <name 1..8>
Specifies the option from which references to the CATEGORY parameter sets are to be deleted.

SHOW-CATEGORY
Display CATEGORY entries

The SHOW-CATEGORY statement is used to output CATEGORY entries to SYSOUT in edited form.

Format

SHOW-CATEGORY
CATEGORY-NAME = **<u>*ALL</u>** / list-poss: <name 1..7> ,**OPTION-NAME** = <name 1..8>

Operands

CATEGORY-NAME = <u>*ALL</u> / list-poss: <name 1..7>
All CATEGORY entries or the specified CATEGORY entries are output to SYSOUT.

OPTION-NAME = <name 1..8>
The CATEGORY entries are assigned to the specified option.

SHOW-OPTION
Display OPTION entries

The SHOW-OPTION statement is used to output OPTION entries from the PPF to SYSOUT in edited form.

Format

SHOW-OPTION
OPTION-NAME = ***ALL** / list-poss: <name 1..8>

Operands

OPTION-NAME = *ALL
All option entries are output to SYSOUT.

OPTION-NAME = list-poss: <name 1..8>
Defines the OPTION entries that are to be output to SYSOUT.

Note

Examples of such outputs can be found in section “Application examples in statement mode” on page 118.

7.4.8 Application examples in statement mode

The OPTION and CATEGORY entries from the PPF can be output to SYSOUT in edited form using the PCSDEFINE commands SHOW-OPTION and SHOW-CATEGORY.

Example 1

SHOW-OPTION OPTION-NAME=STD#TP

```
OPTIONNAME      CHECKSTATUS      R-D-MAX      THP-QTA          USR-INF  #CAT
STD#TP            CHECKED                 6        20            NO       5
                                C A T E G O R I E S
SYS....   DIALOG   BATCH.   TP...    BATCHF.   .......   ......    ......
.......   ...........   ......   .......  .......   ......    ......
```

Example 2

SHOW-CATEGORY OPTION-NAME=STD#TP, CATEGORY-NAME=*ALL

OPTION: STD#TP			CHECKSTATUS: CHECKED			#CATEGORIES:6	
CATEGORY	DURATION	NEXT-CAT	TH-Q	S-Q-MIN	S-Q-MAX	R-D-MIN	R-D-MAX
SYS	*Unlim*	*NONE*	0	0	0	0.0	0.0
DIALOG	500	DIALOG1	0	10	30	2.0	6.0
BATCH	*Unlim*	*NONE*	70	0	10	0.0	0.0
TP	*Unlim*	*NONE*	0	50	100	1.0	3.0
BATCHF	*Unlim*	*NONE*	0	0	20	1.0	3.0
DIALOG1	*Unlim*	*NONE*	10	0	30	0.0	0 .0

Example 3

A formatted output of the PCS parameter sets to SYSLST is possible using the format-oriented PCSDEFINE (PRINT OPTION .....).

OPT2	R-D-MAX 6		TH-Q 20		USER-INFO	NO UNCHECKED	
CAT-NAME	S-Q-MIN	S-Q-MAX	R-D-MIN	R-D-MAX	TH-Q	DURATION	NEXT-CAT
DIALOG	0	100	0	0	0	-	
BATCH	0	100	0	0	0	-	
TP	0	100	0	0	0	-	
BATCHF	0	100	0	0	0	-	

8 PCS administration

At the simplest level, the duties of system administrators (or operators) once PCS is installed are confined to entering a command to generate and start the PCS subsystem. At a higher level, there are commands for modifying individual PCS operands during a PCS session, for outputting PCS statistics and for terminating PCS.

8.1 PCS installation and preparing for startup

The Performance Control System is installed with the help of a product tape containing all files necessary for PCS and the PCSDEFINE utility routine.

In detail, these are the following files:

Files	Description
SIPLIB.PCS.027	contains the PCS "restricted macros"
SYSLNK.PCS.027	contains dynamically loadable components of PCS
SYSRMS.PCS.027	contains PCS loader components
SYSSSI.PCS.027	contains the default options
SYSMES.PCS.027	is a file containing message texts
SYSSDF.PCS.027	is the SDF syntax file for PCS
SYSSMB.PCS.027	is a symbol file for the diagnostic tool DAMP
SYSSSC.PCS.027	contains the PCS subsystem declarations
SYSSII.PCS.027	contains the IMON structure and installation information
SYSFGM.PCS.027.D	release notice in German
SYSFGM.PCS.027.E	release notice in English
PCSDEFINE	utility routine responsible for administration of the PCS OPTION file
SYSFHS.PCSDEFINE.027	FHS module for PCSDEFINE in format mode (format objects)
SYSLNK.PCSDEFINE.027	loadable components of PCS (LLM objects)
SYSRMS.PCSDEFINE.027	PCSDEFINE loader components

Files	Description
SYSSDF.PCSDEFINE.027	SDF syntax file for PCSDEFINE
SYSSII.PCSDEFINE.027	IMON structure and installation information
SYSSPR.PCSDEFINE.027	procedure library for the CREATE-PCS-OPTION command

The SOLIS2 delivery information contains a list of the currently valid file and volume characteristics.

Installing PCS

Standard installation is performed using the product IMON. Standard installation must be performed under TSOS.

When installing PCS without using IMON, please note the following:

– Creating the DSSM catalog
  Make a backup copy of the existing catalog and add the new PCS catalog object with SSCM using the START-CATALOG-MODIFICATION and ADD-CATALOG-ENTRY commands with the declarations from the SYSSSC.PCS.027 file.

– The SDF syntax file SYSSDF.PCSDEFINE.027 must be inserted into the system syntax file.

– The loader SYSREP.PCS.027 must be created from the SYSRMS.PCS.027 loader component set with the help of the utility routine RMS.

– The loader SYSREP.PCSDEFINE.027 must be created from the SYSRMS.PCSDEFINE.027 loader component set with the help of the utility routine RMS.

In either case (with or without IMON), you should note that:

– With regard to the task category known to the system and available in the activated OPTION, a complete, two-way comparison must be carried out unless you are dealing with a follow-up category (target category after duration runout). This means that such categories should, one the one hand, be input into the OPTION and, on the other hand, be made known to Job Management via JMU.

– If necessary, set up BATCHF.
The BATCHF category is defined in each of the standard options STD#DIA, STD#TP and STD#BAT supplied with delivery.

  If PCS is to be operated with one of these options, then this category must be defined in the desired job stream and the corresponding job class with the help of the JMU utility routine, as stated above.

– Choose a suitable PCS parameter set for the PCS application,
see the description of STD#TP, STD#DIA or STD#BAT in section “Standard options” on page 37.

Starting PCS

1. Either: /START-SUBSYSTEM PCS,S-P=C'O-N =<name 1..8>'
2. Or: copy the selected option into STDOPT using the PCSDEFINE utility routine with the help of the COPY-OPTION statement and then enter
/START-SUBSYSTEM PCS,SYNCHRONOUS=Y

Using *open*SM2 to determine the category-specific input/output time

*open*SM2 can be started or stopped at any time while PCS is running.

In order for PCS to receive the current input/output times from *open*SM2, do the following:
– at the console or a terminal: /START-SUBSYSTEM SM2,SYNCHRONOUS=Y
– at a terminal, start the measurement program: /START- SM2
– using "c-a", switch to guided dialog
– SET-SYSTSTAT-PARAMETER, USED-DEVICES=*DISK
– MODIFY-MEASUREMENT-PERIODS, OFFLINE-PERIOD=10
(or other values)
– START-MEASUREMENT-PROGRAM, TYPE=*SYSSTAT

8.2 PCS commands

This chapter describes how PCS is started and terminated, and which commands are available during runtime.

Loading and starting PCS

The following commands are available for creating and starting PCS:
the /START-SUBSYSTEM command and the /RESUME-SUBSYSTEM command.

Stopping and unloading PCS

There are two commands with different effects that can be used to terminate PCS operation: the /HOLD-SUBSYSTEM and /STOP-SUBSYSTEM commands

Modifying PCS parameters

Modifying PCS parameters means changing the activated option. The /MODIFY-PCS-OPTION is available for this purpose.
This does not lead to a change of the option in the corresponding PPF. Permanent changes are only possible using the PCSDEFINE utility routine.

Outputting PCS parameters and monitored variables

This command supplies, on the one hand, information on global and category-specific parameters as well as current performance data (service rates) and control parameter values (dilation) and, on the other hand, information on task-specific data such as accumulated service units, (smoothed) service rates and maximum service rate values in the recent past, the task priority and a comparison of the PCS service units with the accounting service units. The output of the parameters of the activated option and the configuration-dependent constants used for the definition of the service units and of current monitored variable values on the console or to SYSOUT is handled using the /SHOW-PCS-OPTION command.

Overview of functions

/HOLD-SUBSYSTEM	places PCS in the wait state
/MODIFY-PCS-OPTION	modifies PCS parameters
/RESUME-SUBSYSTEM	cancels the wait state status for PCS
/SHOW-PCS-OPTION	outputs PCS parameters and monitored values
/START-SUBSYSTEM	activates PCS
/STOP-SUBSYSTEM	deactivates PCS

HOLD-SUBSYSTEM
Place PCS in wait state

Domain: SYSTEM-MANAGEMENT

Privileges: OPERATING
SUBSYSTEM-MANAGEMENT

Function

The HOLD-SUBSYSTEM command places the PCS subsystem in the wait state. No more new connections to PCS are allowed; the necessary resources (holder task, address space) remain available. With the aid of the FORCED option, it is also possible to wait until all tasks currently executing are finished or to force the immediate abortion of these tasks. After the deinitialization phase has been completed, PCS is in the wait state. The wait state can be exited using the RESUME-SUBSYSTEM command.

Format

HOLD-SUBSYSTEM
SUBSYSTEM-NAME = **PCS** ,**VERSION** = ***STD** / <product-version 6..10> / <product-version 3..7 without-man> ,**SUBSYS**TEM**-PAR**AMETER = ***NONE** / <c-string 1..254> ,**SYNCH**RONOUS = ***NO** / ***Y**ES

Operands

SUBSYSTEM-NAME = PCS
The PCS subsystem is placed in the wait state. PCS is stopped but not unloaded. It can be resumed using the same or a modified parameter set by means of the /RESUME-SUBSYSTEM command.

VERSION =
Specifies the version number.

VERSION = *STD
If there is only **one** version of the PCS subsystem loaded, then that version is selected. If there are **several** appropriate versions, the required version must be specified.

VERSION = <product-version 6..10> / <product-version 3..7 without-man>
Version number of the PCS subsystem. The value entered must agree - including the release and correction (update) status - with the format entered in the definition of the PCS subsystem.

SUBSYSTEM-PARAMETER = *NONE / <c-string 1..254>
States whether special parameters that can only be evaluated by the PCS subsystem are to be processed.

SYNCHRONOUS =
Allows you to choose between synchronous and asynchronous processing.

SYNCHRONOUS = *NO
The command is to be processed asynchronously, i.e. without having to wait for further input to continue processing. After the command syntax has been checked, the calling task receives message ESM0216. Error messages pertaining to the execution of the command are output at the console.

SYNCHRONOUS = *YES
Processing cannot continue until the command has been executed.
Corresponding error messages concerning execution of the command are sent to the task.
In the context of a version switch, this input is only relevant for the new version. Deactivation of the other, 'old' version is always performed asynchronously.

Command return codes

(SC2)	SC1	Maincode	Meaning
	0	CMD0001	No error
1	0	CMD0001	No action necessary
	1	ESM0414	Syntax error: an invalid version was entered
	32	ESM0224	The command was not processed
	32	ESM0228	The command terminated abnormally

MODIFY-PCS-OPTION
Modify PCS parameters

Domain: SYSTEM-TUNING, UTILITIES

Privileges: TSOS, OPERATING

Function

Modifying PCS parameters is taken here to mean modifying the *active* option. The /MODIFY-PCS-OPTION command is used to do this. The option in the associated PPF is not modified. Permanent changes are therefore only possible using the PCSDEFINE utility routine.

As the format of the command indicates, modifications may only be made to global system operands and not to category parameters. This is intended to prevent inconsistencies, as the consistency of category parameters can only be determined from a complete category set (cf. the PCSDEFINE commands CHECK-OPTION and CHECK-CATEGORY).

Format

MODIFY-PCS-OPTION

SYSTEM-**PAR**AMETER = ***PARAM**ETERS(..)

***PARAM**ETERS(...)

REQUEST-**DELAY-MAX** = ***UNCHA**NGED / <integer 1..100>

,**THROUGHPUT-QUOTA** = ***UNCHA**NGED / <integer 0..100>

,**LOG-INTERVAL** = ***UNCHA**NGED / ***NO/** <integer 0..10000>

,USER-INFORMATION = ***UNCHA**NGED / ***Y**ES / ***NO**

Operands

SYSTEM-PARAMETER = *PARAMETERS(...)
Specifies the parameters of the PCS parameter set which are to be modified.

REQUEST-DELAY-MAX = *UNCHANGED
The threshold value for the maximum dilation of all the tasks in the permitted range and which belong to a category with a dilation range remain unchanged.
The default value depends on the value of the THROUGHPUT-QUOTA operand and is calculated using the following formula:

```
5 + (THROUGHPUT-QUOTA) / 20
```

REQUEST-DELAY-MAX = <integer 0..100>
Serves to set the optimum multiprogramming factors.

THROUGHPUT-QUOTA = *UNCHANGED
The set percentage value, which expresses the relationship between response-time optimization and throughput optimization of the system, remains unchanged.
Default value: 20%.

THROUGHPUT-QUOTA = <integer 0..100>
Sets a new percentage value.
The value THROUGHPUT-QUOTA = 100 ensures exclusively throughput-oriented operation; THROUGHPUT-QUOTA = 0 ensures exclusively response time-oriented operation.

LOG-INTERVAL = *UNCHANGED / *YES / *NO
The parameters are no longer supported.

USER-INFORMATION = *UNCHANGED / *YES / *NO
Specifies whether information on PCS is to be made available to the user.

Command return codes

(SC2)	SC1	Maincode	Meaning
	0	CMD0001	No error
	1	CMD0202	Syntax error
	32	CMD0221	Internal system error, command not executed
	64	PCS0016	Privilege error
	130	ETMPC17	Internal lock not available, command not executed
	130	ETMPC20	PCS not started

RESUME-SUBSYSTEM
Cancel wait state status for PCS

Domain: SYSTEM-MANAGEMENT

Privileges: OPERATING
SUBSYSTEM-MANAGEMENT

Function

Using the /RESUME-SUBSYSTEM command, the system administrator can cancel the wait state for PCS. Following successful execution of the command, connections to PCS can be set up again, provided PCS was placed in a defined wait state by means of a /HOLD-SUBSYSTEM command. This guarantees that all necessary resources (holder task, address space) are still available and that the initialization routine can be executed.

Format

RESUME-**SUBSYS**TEM

SUBSYSTEM-NAME = **PCS**

,**VERSION** = **_*STD_** / <product-version 6..10> / <product-version 3..7 without-man>

,**SUBSYS**TEM-**PAR**AMETER = **_*NONE_** / <c-string 1..254> / C'...'

C'...

OPTION-NAME = **_*STD_** / <name 1..8>

,**FILE-NAME** = **_*STD_** / <filename 1..54 without-gen>

,**RESET** = **_*NO_** / ***Y**ES

,**SYNCH**RONOUS = **_*NO_** / ***Y**ES

Operands

SUBSYSTEM-NAME = PCS
PCS is no longer in the wait state.

VERSION =
Specifies the version number.

VERSION = *STD
If there is only **one** version of the PCS subsystem loaded, then that version is selected. If there are **two or more** appropriate versions, the required version must be specified.

VERSION = <product-version 6..10> / <product-version 3..7 without-man>
Version number of the PCS subsystem. The value entered must agree - including the release and correction (update) status - with the format entered in the definition of the PCS subsystem.

SUBSYSTEM-PARAMETER = *NONE / <c-string 1..254> / C'...'
Specifies whether special parameters that can only be evaluated by the PCS subsystem are to be processed.

SUBSYSTEM-PARAMETER = C'...'
For activation of the PCS subsystem the following parameters can additionally be specified as a C string in single quotes:

OPTION-NAME = *STD / <name 1..8>
Specifies the name of the PCS option defined in the PCSDEFINE utility routine. The name of the standard option is STDOPT.

FILE-NAME = *STD
The standard PPF with the name SYSSSI.PCS.027 is expected for activation of the option.

FILE-NAME = <filename 1..54 without-gen>
Specifies the name of the PCS parameter file (PPF) that contains the option.

RESET =
Influences the behavior and the priority of command processing.

RESET = *NO
If the PCS subsystem has not yet reached a defined wait state, the command is rejected until the subsystem has reached a defined wait state.

RESET = *YES
The command is accepted even if a clear operation is still in process and the PCS subsystem or any components are initialized immediately (see the note below also).

SYNCHRONOUS =
Allows you to choose between synchronous and asynchronous processing.

SYNCHRONOUS = *NO
The command is to be processed asynchronously, i.e. without having to wait for further input to continue processing. After the command syntax has been checked, the calling task receives message ESM0216. Error messages pertaining to the execution of the command are output at the console.

SYNCHRONOUS = *YES
Processing cannot continue until the command has been executed.
Corresponding error messages concerning execution of the command are sent to the task.
In the context of a version switch, this input is only relevant for the new version. Deactivation of the other, 'old' version is always performed asynchronously.

Command return codes

(SC2)	SC1	Maincode	Meaning
	0	CMD0001	No error
1	0	CMD0001	No action necessary
	1	ESM0414	Syntax error: an invalid version was entered
	32	ESM0224	The command was not processed
	32	ESM0228	The command terminated abnormally

Notes:

– Defining an automatic start for PCS at system initialization (e.g. with a SYSGEN parameter) is not possible as a PCS start involves file accesses and this would lengthen system initialization time. Consequently, PRIOR is always started when the system is initialized.

– When PCS is started, the PRIOR parameters MIN-MPL, MAX-MPL and WEIGHT are saved and modified during operation according to the load and the specifications made; the subsequent PRIOR parameter values are then derived from the PCS parameter values.

– Executing a /MODIFY-TASK-CATEGORIES command during the PCS session changes the saved values. They do not become effective until PCS has been terminated.

Example

```
/RESUME-SUBSYSTEM SUBSYSTEM-NAME=PCS,
   SUBSYSTEM-PARAMETER=C'OPTION-NAME=STD#TP,FILE-NAME=SYSSSI.PCS.027'
```

SHOW-PCS-OPTION
Output PCS parameters and monitored variables

Domain: SYSTEM-TUNING, UTILITIES

Privileges: *ALL

Function

The /SHOW-PCS-OPTION command displays, on the one hand, information on global and configuration-dependent parameters and the service rates and control values (dilations) and, on the other hand, information on task-specific data such as service units accumulated, (smoothed) service rates and recent maximum service rate values, the task priorities and, finally, a comparison of the PCS service units with the accounting service units.

The output of the parameters of the active option and the configuration-dependent constants needed for defining the service units, as well as current monitored variables, is directed to the console or to SYSOUT.

Format

SHOW-PCS-OPTION
CATEGORY**-NAME** = ***ALL** / list-poss(5): <name 1..7> ,**TSN** = ***NOTSPECIFIED** / ***OWN** / <alphanum-name 1..4>

Operands

CATEGORY-NAME =
Specifies the names of the categories on which information is requested.

CATEGORY-NAME = *ALL
Outputs information on all categories.

CATEGORY-NAME = list-poss(5): <name 1..7>
Specifies the names of the categories on which information is requested. If the specified categories do not exist, an error message is issued.

TSN =
Specifies the task sequence numbers on which information is requested.

TSN = *NOTSPECIFIED
No TSN-specific information is output.

TSN = *OWN
The TSN-specific information on the user's own task is output.

TSN = <alphanum-name 1..4>
The TSN-specific information of the specified task sequence number is output. Nonprivileged users can only specify their own TSN.

Note

You are only allowed to specify one of these two operands.

Command return codes

(SC2)	SC1	Maincode	Meaning
	0	CMD0001	No error
	1	CMD0202	Syntax error
	1	PCS0032	Category not available
	1	PCS0033	Task for specified TSN cannot be accessed
	32	CMD0221	Internal system error, command not executed
	64	PCS0016	Infringement of privileges
	130	ETMPC17	Internal lock not available, command not executed
	130	ETMPC20	PCS not started

Outputting monitored variables with /SHOW-PCS-OPTION

A set of important global and category-specific monitored variables can be output for the current status during the last control interval using the PCS command /SHOW-PCS-OPTION. The output has the following format:

Example 1

```
*** PERFORMANCE CONTROL SUBSYSTEM ***                    STARTED
OPTION=STDOPT, USER-INFO=NO,
FILE=SYSSSI.PCS.027
GLOBAL          SERVICE-RATE-                  REQUEST-DELAY-   THROUGHPUT-
                  ACT     MAX                   ACT   MAX          QUOTA
                14830   14747                   1.8   6.0           20
CATEGORY-             SERVICE-QUOTA            REQUEST-DELAY       SERVICE-
NAME            MIN      MAX      PLN    ACT    MIN   MAX   ACT       RATE
SYS               0        0      0.0    0.6    0.0   0.0   2.8         93
DIALOG           20       40     20.0    0.2    2.0   4.0   2.0         36
BATCH             0       20      9.3    0.0    0.0   0.0   0.0          0
TP                0       30     30.0    0.1    1.0   3.0   3.4         29
BATCHF            0       20      7.5   11.2    1.0   3.0   1.8       1665
DIALOG1          10       30     10.0    0.0    2.0   5.0   0.0          0
DIALOG2           0       50     23.2   87.7    0.0   0.0   5.6  1.300E+04

CATEGORY-         SERVICE-RATE                  TH-Q   CAT-        NEXT-
NAME            CPU      I/O     MEM                   DUR.      CATEGORY
SYS              89        2       1              0       0
DIALOG           34        1       0              0     500      DIALOG1
BATCH             0        0       0             70       0
TP               29        0       0              0       0
BATCHF         1134      479      51              0       0
DIALOG1           0        0       0             10    2000      DIALOG2
DIALOG2   1.214E+04        0     860             20       0
```

Note:

Numbers > 9999 are represented in floating-point format (base 10).

If the **command is entered without operands**, the global parameters and monitored values are output:

OPTION
Name of the active PCS parameter set of the type OPTION.

FILE
Name of the PPF file from which the current parameter set was activated.

USER-INFO,REQUEST-DELAY-MAX,THROUGHPUT-QUOTA
Values of the global PCS parameters for the active option.

SERVICE-RATE-ACT,SERVICE-RATE-MAX
Current or maximum system capacity as determined by PCS.

REQUEST-DELAY-ACT
Current dilation factor for requests, averaged over all categories with a dilation range.

Note
The dilation value output is restricted to a maximum of "999".

The **CATEGORY-NAME operand** of the /SHOW-PCS-OPTION command initiates output of all category parameters and monitored values for the categories specified:

SERVICE-QUOTA-MIN, SERVICE-QUOTA-MAX
Values of the category-specific SERVICE parameters for the active option, SM2R1 report 73.

SERVICE-QUOTA-PLN
The SERVICE-QUOTA value planned by PCS, SM2 report: PCS, SM2R1 report 73.

SERVICE-QUOTA-ACT
The percentage of capacity allocated to the category.

SERVICE-RATE
The capacity currently used by the category (SERVICE-RATE-ACTUAL), SM2 report: PCS, SM2R1 report 77.

SERVICE-RATE-CPU, SERVICE-RATE-IO, SERVICE-RATE-MEM
The service rate used by the category, split up according to the three resources CPU, I/O and main memory.

REQUEST-DELAY-MIN, REQUEST-DELAY-MAX
Values of the category-specific dilation parameters for the active option, SM2R1 report: 74.

REQUEST-DELAY-ACT

Current dilation in the category.

Note

The dilation value output is restricted to a maximum of "999". SM2R1 report: 74.

I/O-DURATION Value of the category-specific I/O time in milliseconds.

This value is needed for the dilation calculation and determines the efficiency of the service planning.

If the I/O time determination with PCS is not being performed at the same time as with *open*SM2 (see the command description), then a fixed value of 20 milliseconds is used; this fact is indicated by an asterisk (*).

The fixed value is also used if no I/O time was determined during the last *open*SM2 observation interval for this category.

THROUGHPUT-QUOTA

Value of the category-specific throughput parameter of the active option.

DURATION, NEXT-CATEGORY

Values of the category-specific parameters for implementing automatic category changeover.

The **TSN operand** in the /SHOW-PCS-OPTION command initiates output of the current category and the original (START) category of the specified task, as well as output of the service units used by the task.

Note:

If USER-INFO = YES applies, then every user can obtain information on all tasks under his or her user ID. From Version 2.3A, the task-specific information supplied by this operand has been considerably expanded.

It is then possible, e.g. by means of regular logging, to observe the load behavior as influenced by a task and the way it is served by the system, and thereby draw conclusions about modified parameter settings

Example 2

```
/SHOW-PCS-OPTION TSN=07B5

PCS-DATA   AT    11.53:13 :

TSN       ACT-CAT   ORG-CAT     SUM-SU‘S   CPU-SU‘S   I/O-SU‘S  MEM-SU‘S
07B5      DIALOG    DIALOG     4.100E+04  3.843E+04       1200      1374

SERVICE-RATE                    SUM-S-R    CPU-S-R    I/O-S-R   MEM-S-R
  SHORT-TIME-MAXIMUM...             265        248          8         9
  ACTIVE (SMOOTHED)...              192        182          6         4

INTERNAL, PRIORITY (0...128=HIGH) :  20

ACCOUNTING-DATA :              SUM-SU‘S   CPU-SU‘S   I/O-SU‘S  MEM-SU‘S
                              4.102E+04  3.843E+04       1220      1374
```

Example 3

Observing a system task:

```
/SHOW-PCS-OPTION TSN=BCAM

PCS-DATA   AT   17:52:56:

TSN       ACT-CAT   ORG-CAT     SUM-SU‘S   CPU-SU‘S   I/O-SU‘S  MEM-SU‘S
BCAM      SYS       SYS        2.188E+06  2.315E+04  2.165E+06        79

SERVICE-RATE                    SUM-S-R    CPU-S-R    I/O-S-R   MEM-S-R
  SHORT-TIME-MAXIMUM...             367          7        360         0
  ACTIVE (SMOOTHED)...              158          1        157         0

INTERNAL, PRIORITY (0...128=HIGH) : FIX

ACCOUNTING-DATA :              SUM-SU‘S   CPU-SU‘S   I/O-SU‘S  MEM-SU‘S
                              2.364E+04  2.315E+04        414        79
```

Note

Numbers > 9999 are represented in floating-point format (base 10).

1. PCS-DATA

ACT-CAT	current category that the task is processing (possibly according to duration)
ORG-CAT	Start category

Current PCS service units. Individually:

SUM-SUs	Sum of the relevant service units used for PCS (with respect to the dilation calculation, only the physical input/output is relevant).
	This is the total of the following values:
CPU-SUs	CPU service units
I/O-SUs	I/O service units
MEM-SUs	Memory service units

Service rate output. A distinction is drawn between the following two cases:

a) SHORT-TIME-MAXIMUM:

Within a short time interval (10 seconds) each (!) of the maximum service rates is determined (from a maximum of 10 current snapshots in the active space). The calculated values are stored for one interval and then recalculated.

Output are:

SUM-S-R	Maximum service rate sum
	P l e a s e n o t e :
	If all 4 maximum values originate from different intervals, then the sum is not necessarily the total of the 3 individual values !
CPU-S-R	Maximum CPU service rate
I/O-S-R	Maximum I/O service rate
MEM-S-R	Maximum memory service rate

b) ACTIVE (SMOOTHED):

The smoothed service rates are output here for which the increase in service units between two consecutive 'snapshots' of the task in the active space (more precisely: tasks in QO...Q5) is placed in relation to the time between snapshots.

The resulting current service rate is the weighted sum of the current and previous rates, the weighting being 3/4 to 1/4. In this way, a (relatively quick) smoothing is carried out. If the same value is measured for 3 consecutive intervals, then the result is within about 1.5% of the actual value.

SUM-S-R Smoothed service rate sum

This is the sum of the following values:

CPU-S-R smoothed CPU service rate

I/O-S-R smoothed I/O service rate

MEM-S-R smoothed memory service rate

INTERNAL PRIORITY This is a so-called initiation priority, which is used to assign processor time to the tasks.

The average value is therefore 64, representing a balance between processor capacity quota assigned and actual capacity used (actual and planned values).

The value displayed is the result of the calculation in the last periodic interval.

This is the actual value for TP tasks that are not to be deactivated, while a historical value is displayed for deactivated tasks.

For tasks that are generally not controlled by PCS, "FIX" is displayed as the priority as these are usually tasks with a fixed priority.

2. ACCOUNTING-DATA

Current accounting service units. These are:

SUM-SUs	Sum of the service units used for accounting.
	This is the total of the following:
CPU-SUs	CPU service units
I/O-SUs	I/O service units (logical input/output)
MEM-SUs	Memory service units

Notes

1. Both the time determination and the determination of the PCS accounting SU values are performed (synchronously) when executing the command, while the service rates are determined periodically at 1-second intervals, but only if the task is then in the admission space (see above).

   This means that while you may be able to observe a continuous increase in the service units for a task every time you enter a /SHOW-PCS-OPTION command, service rate values and task priorities may remain more or less constant over a long period of time if the admission space phases occur often enough 'between' the 1-second interval measurement points.

   This often happens with dialog tasks, while for TP tasks, you may assume that the values are 'fresh', i.e. up-to-date.

2. The interval width can - if necessary - be changed by means of a REP.

Generally, the accounting I/O service units are equal to or larger than the number of PCS I/O service units, as logical I/O operations are evaluated there and not every logical I/O results in a physical I/O for cached media.

If, however, several physical I/Os need to be started for a logical I/O when the file is correspondingly split up (into a number of extents), then the number of PCS service units may - at least for a short time - be larger than the number of accounting service units.

START-SUBSYSTEM
Activate PCS

Domain: SYSTEM-MANAGEMENT

Privileges: OPERATING, SUBSYSTEM-MANAGEMENT

Function

System administration can use this command to activate the PCS subsystem. The following information is needed from the dynamic subsystem catalog to activate PCS:

– data for loading and linking the PCS subsystem
– data for initializing/deinitializing and for terminating the job relationships
– data for creating calls, subcomponents and operational dependencies (see the corresponding SSCM commands)

Format

START-SUBSYSTEM

SUBSYSTEM-NAME = **PCS**

,**VERSION** = **_*STD_** / <product-version 6..10> / <product-version 3..7 without-man>

,**SUBSYS**TEM**-PAR**AMETER = **_*NONE_** / <c-string 1..254> / C'...'

C'...'

OPTION-NAME = **_*STD_** / <name 1..8>

,**FILE-NAME** = **_*STD_** / <filename 1..54 without-gen>

,**RESET** = **_*NO_** / ***Y**ES

,**SYNCH**RONOUS = **_*NO_** / ***Y**ES

,**VERSION-PARALLELISM** = **_*NONE_** / ***EXCHANGE-MODE**(...) / ***COEXIST**ENCE**-MODE**

***EXCHANGE-MODE**(...)

SUBSYSTEM**-PAR**AMETER = **_*NONE_** / <c-string 1..254>

Operands

SUBSYSTEM-NAME = PCS
The PCS subsystem is activated.

VERSION =
Specifies the version number.

VERSION = *STD
If there are several versions of the PCS subsystem available and no version is specified or *STD has been entered explicitly, then the version that has been declared with the start attribute CREATION-TIME=*AT-SUBSYSTEM-CALL is loaded. If this condition is not met, then the lowest PCS version number in the static subsystem catalog is selected.

VERSION = <product-version 6..10> / <product-version 3..7 without-man>
Version number of the PCS subsystem. The value entered must agree - including the release and correction (update) status - with the format entered in the definition of the PCS subsystem.

SUBSYSTEM-PARAMETER = *NONE / <c-string 1..254> / C'...'
Specifies whether special parameters that can only be evaluated by the PCS subsystem are to be processed.

SUBSYSTEM-PARAMETER = C'...'
For activation of the PCS subsystem the following parameters can additionally be specified as a C string in single quotes:

OPTION-NAME = *STD / <name 1..8>
Specifies the name of the PCS option defined in the PCSDEFINE utility routine. The name of the standard option is STDOPT.

FILE-NAME = *STD
The standard PPF with the name SYSSSI.PCS.027 is expected for activation of the option. If installation took place under IMON, the actual file name is known there; otherwise the file is expected under $.SYSSSI.PCS.027.

FILE-NAME = <filename 1..54 without-gen>
Specifies the name of the PCS parameter file (PPF) that contains the option.

RESET =
Influences the behavior and the priority of command processing.

RESET = *NO
If the PCS subsystem has not yet reached a defined wait state, the command is rejected until the subsystem has reached a defined wait state.

RESET = *YES
The command is accepted even if a clear operation is still in process and the PCS subsystem or any components are initialized immediately (see the note below).
The version parameter is mandatory for this operand.

SYNCHRONOUS =
Allows you to choose between synchronous and asynchronous processing.

SYNCHRONOUS = *NO
The command is to be processed asynchronously, i.e. without having to wait for further input to continue processing. After the command syntax has been checked, the calling task receives message ESM0216. Error messages pertaining to the execution of the command are output at the console.

SYNCHRONOUS = *YES
Processing cannot continue until the command has been executed.
Corresponding error messages concerning execution of the command are sent to the task.
In the context of a version switch, this input is only relevant for the new version. Deactivation of the other, 'old' version is always performed asynchronously.

VERSION-PARALLELISM =
Specifies whether or not different versions of the PCS subsystem may be active simultaneously.
Note: Only the value *NONE is allowed for the PCS subsystem.

VERSION-PARALLELISM = *NONE
The coexistence of different versions of the PCS subsystem - independent of the entries in the definition - is not allowed. If the status of a version is not equal to "NOT-CREATED", activation is not performed.

Command return codes

(SC2)	SC1	Maincode	Meaning
	0	CMD0001	No error
1	0	CMD0001	No action necessary
	1	ESM0414	Syntax error: an invalid version was entered
	32	ESM0224	The command was not processed
	32	ESM0228	The command terminated abnormally

Example

```
/START-SUBSYSTEM SUBSYSTEM-NAME=PCS,
   SUBSYSTEM-PARAMETER=C'OPTION-NAME=STD#TP,FILE-NAME=SYSSSI.PCS.027'
```

STOP-SUBSYSTEM
Deactivate PCS

Domain: SYSTEM-MANAGEMENT

Privileges: OPERATING
SUBSYSTEM-MANAGEMENT

Function

System administration can use this command to deactivate the PCS subsystem. The command has the following effects:

1. Access to the PCS subsystem is locked for all new callers.
2. The PCS subsystem is deactivated as soon as all jobs accessing the subsystem are finished. If the operand FORCED=*YES was entered in the STOP-SUBSYSTEM command, then all jobs accessing the PCS subsystem are aborted immediately (the operand FORCED=*YES is only accepted if the STOP-SUBSYSTEM command could not terminate the job using the operand FORCED=*NO).
3. The PCS subsystem is unloaded.
4. All occupied resources are released.

Format

STOP-**SUBSYS**TEM
SUBSYSTEM-NAME = **PCS** ,**VERSION** = **_*STD_** / <product-version 6..10> / <product-version 3..7 without-man> ,**SUBSYS**TEM-**PAR**AMETER = **_*NONE_** / <c-string 1..254> ,**SYNCH**RONOUS = **_*NO_** / ***Y**ES

Operands

SUBSYSTEM-NAME = PCS
The PCS subsystem is deactivated.

PCS is stopped and unloaded. Storage space used by PCS is returned. PCS can only be started again using the /START-SUBSSYSTEM command. As much of the requested memory as possible will be returned and all resources released.

On termination of PCS (including abnormal termination due to an error), the system is once more controlled via the PRIOR parameters MIN-MPL, MAX-MPL and WEIGHT. The operands for a PRIOR session started in this manner are automatically saved by PCS when the /START-PCS or /RESUME-PCS command is executed. A switch back to PRIOR operation with these parameters will also take place if PCS is found to be in an error state.

VERSION =
Specifies the version number.

VERSION = *STD
If there is only **one** version of the PCS subsystem loaded, then that version is selected. If there are **two or more** appropriate versions, the required version must be specified.

VERSION = <product-version 6..10> / <product-version 3..7 without-man>
Version number of the PCS subsystem. The value entered must agree - including the release and correction (update) status - with the format entered in the definition of the PCS subsystem (see statement SET-SUBSYSTEM-ATTRIBUTES)

SUBSYSTEM-PARAMETER = *NONE / <c-string 1..254>
States whether special parameters that can only be evaluated by the PCS subsystem are to be processed.

SYNCHRONOUS =
Allows you to choose between synchronous and asynchronous processing.

SYNCHRONOUS = *NO
The command is to be processed asynchronously, i.e. without having to wait for further input to continue processing. After the command syntax has been checked, the calling task receives message ESM0216. Error messages pertaining to the execution of the command are output at the console.

SYNCHRONOUS = *YES
Processing cannot continue until the command has been executed.
Corresponding error messages concerning execution of the command are sent to the task.
In the context of a version switch, this input is only relevant for the new version. Deactivation of the other, 'old' version is always performed asynchronously.

Command return codes

(SC2)	SC1	Maincode	Meaning
	0	CMD0001	No error
1	0	CMD0001	No action necessary
	1	ESM0414	Syntax error: an invalid version was entered
	32	ESM0224	The command was not processed
	32	ESM0228	The command terminated abnormally

Example

```
/STOP-SUBSYSTEM SUBSYSTEM-NAME=PCS,
   SUBSYSTEM-PARAMETER=C'OPTION-NAME=STD#TP,FILE-NAME=SYSSSI.PCS.027'
```

9 Messages

ETMPC17 CMD NOT ACCESSIBLE NOW. PLEASE TRY LATER
(B) Routing code: A Weight: 30

ETMPC20 PCS NOT STARTED. CMD IGNORED
(B) Routing code: A Weight: 30

PCSCOPY Copyright (C) '(&00)' '(&01)' All Rights Reserved

PCSLOAD Program '(&00)', Version '(&01)' of '(&02)' loaded from file '(&03)'

PCS0010 INVALID KEYWORD IN SPECIFIED PCS COMMAND. COMMAND IGNORED
(B) Routing code: A Weight: 30

PCS0011 INVALID OPERAND VALUE IN SPECIFIED PCS COMMAND. COMMAND IGNORED
(B) Routing code: A Weight: 30

PCS0016 NO AUTHORIZATION FOR COMMAND. COMMAND IGNORED
(B) Routing code: A Weight: 30

PCS0022 MODIFICATION OF PCS PARAMETERS PROCESSED
(B) Routing code: A Weight: 50

PCS0023 OPERAND '(&00)' OUT OF RANGE. COMMAND IGNORED
(B) Routing code: A Weight: 30

PCS0024 OPERAND '(&00)' NO LONGER SUPPORTED. OPERAND IGNORED
(B) Routing code: A Weight: 30

PCS0032 CATEGORY '(&00)' DOES NOT EXIST
(B) Routing code: A Weight: 70

PCS0033 TASK WITH TSN '(&00)' NOT ACCESSIBLE
(B) Routing code: A Weight: 30

PCS0099 INVALID MIN MPL VALUE (<0) CALCULATED IN PCS. PCS TERMINATED ABNORMALLY
(B) Routing code: A Weight: 99

Meaning
Abnormal program termination to get diagnosis information.

Response
Take dump and send it to the diagnosis team.

PCS0101 PCS INITIALIZED, VERSION '(&00)'
(B) Routing code: A Weight: 99

PCS0102 PCS TERMINATED NORMALLY
(B) Routing code: A Weight: 99

PCS0103 PCS TERMINATED ABNORMALLY
(B) Routing code: A Weight: 99

PCS0104 PCS STARTED. OPTION '(&00)', FILE '(&01)'
(B) Routing code: A Weight: 99

PCS0110 DMS OPEN ERROR '(&01)', FILE '(&00)'. COMMAND IGNORED. IN SYSTEM MODE: /HELP-MSG DMS(&01)
(B) Routing code: A Weight: 30

Meaning
For more detailed information about the DMS error code enter /HELP-MSG in system mode or see the BS2000 manual 'System Messages'.

PCS0111 DMS ERROR '(&01)', FILE '(&00)'. COMMAND IGNORED. IN SYSTEM MODE: /HELP-MSG DMS(&01)
(B) Routing code: A Weight: 30

Meaning
For more detailed information about the DMS error code enter /HELP-MSG in system mode or see the BS2000 manual 'System Messages'.

PCS0112 OPTION '(&00)' IN FILE '(&01)' DOES NOT EXIST
(B) Routing code: A Weight: 30

PCS0113 FILE '(&00)' DOES NOT EXIST. COMMAND IGNORED
(B) Routing code: A Weight: 30

PCS0114 OPTION '(&00)' NOT CONSISTENT. COMMAND IGNORED
(B) Routing code: A Weight: 30

PCS0115 FILE '(&00)' NO PCS PARAMETER FILE. COMMAND IGNORED
(B) Routing code: A Weight: 70

PCS0116 ERROR IN MODULE '(&00)'. PCS NOT TERMINATED
(B) Routing code: A Weight: 99

PCS0117 ERROR IN MODULE '(&00)'. PCS TERMINATED
(B) Routing code: A Weight: 99

PCS0120 ERROR WHEN RELEASING CATEGORY '(&00)'
(B) Routing code: A Weight: 70

PCS0124 NUMBER OF CATEGORIES EXHAUSTED. COMMAND IGNORED
(B) Routing code: A Weight: 70

PCS0125 JS CATEGORY NOT FOUND IN PCS OPTION. COMMAND IGNORED
(B) Routing code: A Weight: 70

PCS0126 CYCLE IN CATEGORY CHAIN DEFINITION. COMMAND IGNORED
(B) Routing code: A Weight: 70

PCS0127 PCS CATEGORY '(&00)' NOT IN CATEGORY CHAIN. COMMAND IGNORED
(B) Routing code: A Weight: 70

PCS0140 WARNING: OPERAND '(&00)' EXCEEDS LIMIT. RESET TO '(&01)'
(B) Routing code: A Weight: 70

PCS0160 TASK SERVICE UNITS < 0 IN MODULE 'EPCPR'. PCS CONTINUES
(B) Routing code: A Weight: 99

Meaning
SNAPSHOT dump produced as diagnosis information.

Response
Send dump to the diagnosis team.

10 Appendix

10.1 Definition of service units (SU)

The short form of the formula for calculating service units (SUs) is:

SU = CPU-SU + IO-SU + MEMORY-SU

Definition of the **CPU-SU**:

In the case of the CPU-SUs, extra time is allowed, in addition to the TU/TPR time, for I/O requirements in the SIH processor state.

CPU-SU = a * CPU-TIME

where:

CPU-TIME — CPU time in seconds used by the object under scrutiny (system, category, task) in the TU and TPR processor states.
The portion for peripheral SVC processing in the SIH processing state is added to this time.

2000 CPU instructions are equivalent to one SU.
(a = CPU speed / 2000)
The coefficient "a" thus represents the SU value of one CPU second.

Definition of **I/O-SUs**:

IO-SU = d * #IO + e * #BLOCK

where:

#IO — Number of disk and tape inputs/output operations started for the object.

One IO is equivalent to 6 SUs (d = 6).

#BLOCK — Number of 2-Kb blocks (PAM pages) transferred for the object. For tapes this is calculated from the number of bytes transferred; for disks the number of standard blocks transferred is used.

The transfer of one 2-Kb block is equivalent to 2 SUs (e = 2).

Definition of **MEMORY-SUs**:

MEMORY-SU = f * WSI

where:

WSI — WORKING-SET integral for the task in units of #PAGES (4KB)* CPU-TIME. The working set integral is a measurement for the number of main memory (MM) pages used per task for a given period (CPU time in seconds).

In order to obtain accurate values for WSI and thus for the memory service units value when the "system working set procedure" is set, the value is dependent on paging operations, i.e. the fewer paging operations, the more volatile the value.

A task is assumed to take one MEMORY-SU for 50 pages in the time taken for 20,000 instructions (f = a/10 * 1/50 = a/500).

Notes

– MEMORY-SUs are recorded solely for paging memory.

– The coefficients "a" through "f" are set when task management is initialized and cannot be changed by systems support.
Their values are selected such that an average hardware configuration with a proper, balanced load uses roughly the same number of service units for CPU and I/O purposes.

– The value of coefficient "a" is dependent on the CPU capacity and is derived from the RPF value of the processor configuration at system initialization.

The value of constant "f" is a/500 and thus depends on CPU capacity. All other constants are independent of the CPU capacity and have the following values:

b = 1

d = 6

e = 2

The server-specific value for a is derived from the relevant RPF value according to the following formula:

a = (RPF * 400) / number of CPUs

Information on the RPF values is provided in the "Performance Handbook“ [1], in den "Performance Guidelines“ [2] and in the product information on the servers.

10.2 Calculating REQUEST-DELAY-MAX and DURATION (example)

The maximum dilation and the maximum number of service units that may be used in a category can be calculated approximately if a maximum response time is specified and precise details of the amount of resources consumed by the transactions in the category are available.

Calculating the REQUEST-DELAY-MAX parameter

For a "normal" dialog load (see example 4 in section “Automatic category changeover DURATION/NEXT-CATEGORY” on page 26) a response time < 1 seconds is required.

Specifying a maximum response time directly as a PCS parameter is not possible because the response time depends on the CPU and disk types. PCS can only set the desired response time behavior via dilation. Systems support therefore has to take account of the **connection** between **response time**, **dilation** and **resource requirements**:

The following should be assumed for **resource requirements** per transaction in the DIALOG category:

CPU time: 0.1 sec (Business Server SX140-10C)

Number of I/O operations: 30 (block size: 4Kb)

This gives the "individual runtime" per transaction (one input/output takes 5 milliseconds):

CPU time: 0.10 sec

I/O time: 0.15 sec (= 30 * 0.005 sec)

———————

0.25 sec

It follows that, in order to achieve a response time < 1 sec, the maximum permissible dilation is:

$$\text{REQUEST-DELAY-MAX} = \frac{1 \text{ sec}}{0.25 \text{ sec}} = 4$$

Calculating the DURATION parameter

In order to guarantee the required response time, it is necessary to determine the maximum number of service units a transaction in the DIALOG category may use before PCS effects a category change to the successor category DIALOG1.

As mentioned in section “Work (SERVICE UNITs)” on page 12, and section “Definition of service units (SU)” on page 149, the number of service units permitted is composed of the weighted sum of CPU-SUs, IO-SUs and MEMORY-SUs.

The resource requirements per transaction in the DIALOG category should be subject to the following restrictions:

CPU:	CPU-TIME 0.1 sec					
I/O:	#IO 20		#BLOCK 60			
Main memory:	#PAGES 60	*	CPU-TIME 0.1	=	WSI 6	

This means that a transaction can only remain in the DIALOG category and thus expect a response time < 1 seconds if it does not need more than 0.1 seconds of CPU time and not more than 20 I/Os, as well as not more than an average of 60 main memory pages = 6 WSI units.

It is, however, possible for a transaction to consume more than 0.1 seconds of CPU time if it uses fewer I/Os. Only the total number of service units used in the DIALOG category is relevant.

The following table illustrates the calculation of the service requirement per transaction using the formulas from section “Definition of service units (SU)” on page 149. The values for a type SX140-10C CPU are used for coefficients a and f.

CPU-SU:	a	*	CPU-TIME					
	30000	*	0,1					~ 3000
IO-SU:	d	*	#SSCH	+	e	*	#BLOCK	
	6	*	30	+	2	*	60	= 300
MEMORY-SU:	f	*	#WSI					
	60	*	6					~ 3600
Total SUs:								~ 3660

Rounded up, this gives a DURATION value of 3600.

10.3 Empirical values for the DURATION parameter

Empirical values can be used as a basis for determining the value for the DURATION parameter of a category if the qualitative load targets are known:

Program step	**Number of service units**
Simple edit commands	up to 600
File handling	up to 3000
Program loading / short program runs	up to 12000
Short compilations	up to 24000

10.4 Calculating the SERVICE-QUOTA-MAX parameter

The SERVICE-QUOTA-MAX value for the DIALOG category is to be estimated for example 4 (see section “Automatic category changeover DURATION/NEXT-CATEGORY” on page 26).

For this, an assumed value for the average service requirements per transaction (’S-TA’) in the category is required; let us assume S-TA = 2500 SU (equivalent to 70% of the DURATION value). In order to simplify subsequent calculations, we will also assume that this category only includes transactions with these resource requirements. Though certainly not a very realistic assumption, it does considerably simplify the calculation. In the next approximation, this value represents the mean performance requirement of the transactions in a category; depending on how the available capacity is distributed, this will be roughly 30% to 70% of the DURATION value for the category. It should be noted that all long transactions run in the DIALOG category until the DURATION value is reached, i.e. they always exploit the maximum possible capacity of the category.

The maximum transaction rate TR-MAX in this category is to be 4 per second.

This means that the maximum service needed within the category is

SERVICE-RATE-MAX = S-TA * TR-MAX

which gives the following for this example:

SERVICE-RATE-MAX = 2500 SU * 4/sec = 10000 SU/sec

The distributable capacity of a Business Server SX140-10C including peripheral devices is approximately 50000 to 60000 SU/sec (the actual value will of course depend on the hardware configuration and the load).

This results in a SERVICE-QUOTA-MAX of about 20%.

10.5 Determining monitored variables

The monitored variables for PCS operation are found in or derived from the following sources:

1. PCS command /SHOW-PCS-OPTION
2. *open*SM2 and SM2R1 reports

System capacity

The capacity of the system depends on the hardware configuration and the resource requirements of the load. It changes with time and is determined dynamically by PCS.

The current capacity values in SU/sec can be obtained via

/SHOW-PCS-OPTION (SERVICE-RATE-MAX value)

Global dilation

The current values for global PCS dilation REQUEST-DELAY, computed as an average for all requests from categories with a dilation range, can be obtained via

1. /SHOW-PCS-OPTION (REQUEST-DELAY-ACT value)
2. *open*SM2 report: ACTIVITY

The global value DILATION (SM2R1 report 57) cannot be used because it also includes the dilation of categories without a dilation range and is calculated differently.

Category service allocation

The current allocation of service to categories, expressed as a percentage of system capacity (PCS planned values), can be obtained via

1. /SHOW-PCS-OPTION (SERVICE-QUOTA-PLN value)
2. *open*SM2 report: PCS

The same values expressed as a function of time can be obtained via

3. SM2R1 report 73

Service used by a category

The amount of service currently used per category, expressed as a percentage or in SU/sec, can be obtained via

1. /SHOW-PCS-OPTION (SERVICE-QUOTA-ACT and SERVICE-RATE values)
2. *open*SM2 report: PCS

The average amount of service used per category in SU/sec can be obtained via

3. SM2R1 report 77 (SERVICE-RATE value)

In addition, the average service used per category can be estimated using CPU time consumption, DMS I/Os per sec and the number of SVC/sec; values from

4. SM2R1 report 62 and 63 and task statistics

Category dilation

The current dilation per category can be obtained via

1. /SHOW-PCS-OPTION (category-specific REQUEST-DELAY-ACT value)
2. *open*SM2 report: PCS

As a function of time, this value can be obtained from

3. SM2R1 report 74.

Service used per request

The category-specific mean value for the amount of service used per request (SU/TA) is the sum of the service units used by the corresponding category (SRACT in SU/sec) divided by the transaction rate (TA/sec) per category and can be obtained from

SM2R1 report 23

A distribution function for the service used per request cannot be measured directly at the present time.

10.6 Effect of task priorities

The table below shows the effect of task priorities as a function of the category-specific parameter THROUGHPUT-QUOTA.

THROUGHPUT-QUOTA	Priority	Performance factor
0	128	11
	129 - 140	10
	141 - 153	9
	154 - 166	8
	167 - 178	7
	179 - 191	6
	192 - 204	5
	205 - 216	4
	217 - 229	3
	230 - 242	2
	243 - 255	1
1-10	128	10
	129 - 142	9
	143 - 156	8
	157 - 170	7
	171 - 184	6
	185 - 198	5
	199 - 212	4
	213 - 226	3
	227 - 240	2
	241 - 255	1
11-20	128	9
	129 - 143	8
	144 - 159	7
	160 - 175	6
	176 - 191	5
	192 - 207	4
	208 - 223	3
	224 - 239	2
	240 - 255	1

Table 1: Effect of task priorities (part 1 of 2)

THROUGHPUT-QUOTA	Priority	Performance factor
21-30	128	8
	129 - 146	7
	147 - 164	6
	165 - 182	5
	183 - 200	4
	201 - 218	3
	219 - 236	2
	237 - 255	1
31-40	128	7
	129 - 149	6
	150 - 170	5
	171 - 191	4
	192 - 212	3
	213 - 233	2
	234 - 255	1
41-50	128	6
	129 - 153	5
	154 - 178	4
	179 - 204	3
	205 - 229	2
	230 - 255	1
51-60	128	5
	129 - 159	4
	160 - 191	3
	192 - 223	2
	224 - 255	1
61-70	128	4
	129 - 170	3
	171 - 212	2
	213 - 255	1
71-80	128	3
	129 - 191	2
	192 - 255	1
81-90	128	2
	129 - 255	1
91-100	128 - 255	1

Table 1: Effect of task priorities (part 2 of 2)

10.7 Procedure for using CREATE-PCS-OPTION

The procedure reproduced below for creating/modifying OPTION parameter files can be found in the LMS library SYSSPR.PCSDEFINE.027 as the element $PCSCREO and can be modified to suit the customer's wishes.

Note

The procedure should only be modified after a backup copy of the LMS library has been made.

After that, the $PCSCREO element in the original library must be copied using a different name ($PCSCREO-ORIGIN, for example).

The name of the currently called library element in the library SYSSPR.PCSDEFINE.<version> is always $PCSCREO, i.e. it denotes the procedure coming up for execution.

In this way, any number of individual procedures can be created in the library.

The file must have been made known to IMON under the version <version>, otherwise it should be stored in:

$.SYSSPR.PCSDEFINE.027.

Implementing individual OPTIONs

The standard options (STD#DIA, STD#BAT, STD#TP) and an additional option USER=01 are contained in the procedure file supplied with delivery.

All options can be modified as required.

The command/statement sequence for the option USER#01 - this option is preset with the parameters of the option STD#BAT - can be copied as many times as needed, with the name "USER#01" being replaced by a unique name each time (USER#02, USER#03,..., for example).

In this manner, you can create an individual set of options from which you should select a suitable set to be used for automatically starting up PCS.

For example:

```
/PROC
/C-PC-O          O-N=USER#02
/START-SUBSY     PCS,S-P='O-N=USER#02'
/ENDP
```

If PCS is started without any operand entries, then the desired option must first be copied into an option named "STDOPT".

Procedure listing:

```
/SET-PROCEDURE-OPTIONS -
/                     LOGGING-ALLOWED   = *YES -
/                    ,INTERRUPT-ALLOWED = *YES -
/                    ,DATA-ESCAPE-CHAR  = *STD
/
/BEGIN-PARAM-DECL
/DECL-PARAM  MONJV                (INIT='*NONE')
/DECL-PARAM  CPU-LIMIT            (INIT='JOB-REST')
/
/DECL-PARAM  PCS-PARAMETER-FILE  (INIT='*STD', TYPE=*STRING, -
/                                   TRANSFER-TYPE=*BY-VALUE)
/DECL-PARAM   OPTION-NAME         (INIT='*STD', TYPE=*STRING, -
/                                   TRANSFER-TYPE=*BY-VALUE)
/END-PARAM-DECL
/
/MODIFY-TERMINAL-OPTION   OVERFLOW-CONTROL = *NO
/
/WRITE-TEXT 'PCS-PARAMETER-FILE=&PCS-PARAMETER-FILE'
/WRITE-TEXT 'OPTION-NAME=&OPTION-NAME'
/
/IF (OPTION-NAME =='*STD')  OR (OPTION-NAME =='STD#DIA') -
/                        OR (OPTION-NAME =='STD#BAT') -/
OR (OPTION-NAME =='STD#TP' ) -
/
/IF (OPTION-NAME =='*STD')  OR (OPTION-NAME =='STD#DIA')
/
/PCSDEFINE       MONJV=&MONJV,CPU-LIMIT=&CPU-LIMIT
OPEN-FILE   FILE-NAME=&PCS-PARAMETER-FILE
DEL-O STD#DIA
CRE-O TD#DIA   ,S=(R-D= 6,T-Q=20,USER=N,CAT=(DIALOG1,DIALOG2,BATCHF)
MOD-C C=DIALOG ,R=(S-Q=(20, 40),R-D=(2,4)),T= 0,N-C=DIALOG1,D= 500
ADD-C C=DIALOG1,R=(S-Q=(10, 30),R-D=(2,5)),T=10,N-C=DIALOG2,D=2000
ADD-C C=DIALOG2,R=(S-Q=( 0, 50),R-D=(0,0)),T=20,N-C=*NONE  ,D=   0
MOD-C C=BATCH  ,R=(S-Q=( 0, 20),R-D=(0,0)),T=70,N-C=*NONE  ,D=   0
MOD-C C=TP     ,R=(S-Q=( 0, 30),R-D=(1,3)),T= 0,N-C=*NONE  ,D=   0
ADD-C C=BATCHF ,R=(S-Q=( 0, 20),R-D=1(,3)),T= 0,N-C=*NONE  ,D=   0
SH-O  STD#DIA
SH-C
END
/WRITE-TEXT   'GEN_PCS_OPT:  OPTION  ''&OPTION-NAME''   CREATED/MODIFIED'
/END-IF
/
/IF (OPTION-NAME =='*STD#BAT')
/
/PCSDEFINE       MONJV=&MONJV,CPU-LIMIT=&CPU-LIMIT
OPEN-FILE   FILE-NAME=&PCS-PARAMETER-FILE
```

```
DEL-O STD#BAT
CRE-O STD#BAT  ,S=(R-D=  7,T-Q=50,USER=N,CAT=(BATCH1,BATCHF)
MOD-C C=DIALOG ,R=(S-Q=( 0, 40),R-D=(1,5)),T=  0,N-C=BATCH ,D= 500
MOD-C C=BATCH  ,R=(S-Q=( 0, 65),R-D=(0,0)),T= 70,N-C=BATCH1,D=5000
ADD-C C=BATCH1 ,R=(S-Q=( 0, 35),R-D=(0,0)),T=100,N-C=NONE  ,D=   0
MOD-C C=TP     ,R=(S-Q=( 0, 30),R-D=(1,3)),T=  0,N-C=*NONE ,D=   0
ADD-C C=BATCHF ,R=(S-Q=( 0, 20),R-D=1(,3)),T=  0,N-C=*NONE ,D=   0
SH-O  STD#BAT
SH-C
END

/WRITE-TEXT   'GEN_PCS_OPT:  OPTION  ''&OPTION-NAME''   CREATED/MODIFIED'
/END-IF
/
/IF (OPTION-NAME ==’*STD#TP’)
/
/PCSDEFINE       MONJV=&MONJV,CPU-LIMIT=&CPU-LIMIT
OPEN-FILE   FILE-NAME=&PCS-PARAMETER-FILE
DEL-O STD#TP
CRE-O STD#TP   ,S=(R-D=  6,T-Q=20,USER=N,CAT=(BATCHF,DIALOG1)
MOD-C C=DIALOG ,R=(S-Q=(10, 30),R-D=(2,6)),T=  0,N-C=DIALOG1 ,D= 500
ADD-C C=DIALOG1,R=(S-Q=( 0, 30),R-D=(0,0)),T= 10,N-C=*NONE   ,D=   0
MOD-C C=BATCH  ,R=(S-Q=( 0, 10),R-D=(0,0)),T= 70,N-C=*NONE   ,D=   0
MOD-C C=TP     ,R=(S-Q=(50,100),R-D=(1,3)),T=  0,N-C=*NONE   ,D=   0
ADD-C C=BATCHF ,R=(S-Q=( 0, 20),R-D=1(,3)),T=  0,N-C=*NONE   ,D=   0
SH-O  STD#TP
SH-C
END

/WRITE-TEXT   'GEN_PCS_OPT:  OPTION  ''&OPTION-NAME''   CREATED/MODIFIED'
/END-IF
/
/ELSE
/
/"================================================================="
/"EXPANSION FOR U S E R - I N D I V I D U E L L   O P T I O N S"
/
/IF   (OPTION-NAME = =’USER#01’
/
/PCSDEFINE       MONJV=&MONJV,CPU-LIMIT=&CPU-LIMIT
OPEN-FILE   FILE-NAME=&PCS-PARAMETER-FILE
DEL-O OPTION-NAME
CRE-O OPTION-NAME  ,S=(R-D=  7,T-Q=50,USER=N,CAT=(BATCH1,BATCHF)
MOD-C C=DIALOG ,R=(S-Q=( 0, 40),R-D=(1,5)),T=  0,N-C=BATCH ,D= 500
MOD-C C=BATCH  ,R=(S-Q=( 0, 65),R-D=(0,0)),T= 70,N-C=BATCH1,D=5000
ADD-C C=BATCH1 ,R=(S-Q=( 0, 35),R-D=(0,0)),T=100,N-C=NONE  ,D=   0
```

```
MOD-C C=TP     ,R=(S-Q=( 0, 30),R-D=(1,3)),T=  0,N-C=*NONE ,D=   0
ADD-C C=BATCHF ,R=(S-Q=( 0, 20),R-D=1(,3)),T=  0,N-C=*NONE ,D=   0
SH-O  OPTION-NAME
SH-C
END

/WRITE-TEXT   'GEN_PCS_OPT:  OPTION  ''&OPTION-NAME''   CREATED/MODIFIED'
/
/END OF   U S E R - I N D I V I D U E L L    O P T I O N S
"================================================================="
/
/ELSE
/WRITE-TEXT   'GEN_PCS_OPT:  PARAMETER  ''&OPTION-NAME''   NOT SUPPORTED'
/
/END-IF
/MODIFY-TERMINAL-OPTION OVERFLOW-CONTROL=*USER-ACK
/
/EXIT-PROCEDURE
```

10.8 Checking PCS settings/performance data

– /SHOW-PCS-OPTION command
– /STATUS-CATEGORY command
– *open*SM2 reports (73, 74, 75, 76, 77)

10.9 Short descriptions of commands relevant to PCS

The PCS and DSSM commands listed in this section are arranged in alphabetical order.

Commands used to start PCSDEFINE

START-PCSDEFINE
VERSION = *** STD** / <product-version 6..10> / product-version 4..8 without-corr> / <product-version 3..7 without-man> ,**MONJV** = ***NONE** / <filename 1..54 without-gen-vers> ,**CPU-LIMIT** = ***JOB-REST** / <integer 1..32767 *seconds*>

or abbreviated to:

PCSDEFINE

PCS commands

MODIFY-PCS-OPTION - Modify PCS parameters

SYSTEM-**PAR**AMETER = (...)

(...)

REQUEST-**DELAY-MAX** = ***UNCHA**NGED / <integer 1..100>

,**THROUGHPUT-QUOTA** = ***UNCHA**NGED / <integer 0..100>

,**USER-INF**ORMATION = ***UNCHA**NGED / ***Y**ES / ***NO**

SHOW-PCS-OPTION - Output PCS parameters

CATEGORY-**NAME** = ***ALL** / list-poss(5): <name 1..7>

,**TSN** = ***NOTSPECIFIED** / ***OWN** / <alphanum-name 1..4>

DSSM commands

HOLD-SUBSYSTEM - **Place PCS in the wait state**

SUBSYSTEM-NAME = **PCS**

,**VERSION** = ***STD** / <product-version 6..10> / <product-version 3..7 without-man>

,**SUBSYS**TEM-**PAR**AMETER = ***NONE** / <c-string 1..254>

,**SYNCH**RONOUS = ***NO** / ***Y**ES

RESUME-**SUBSYS**TEM - **Cancel wait state status for PCS**

SUBSYSTEM-NAME = **PCS**

,**VERSION** = ***STD** / <product-version 6..10> / <product-version 3..7 without-man>

,**SUBSYS**TEM-**PAR**AMETER = ***NONE** / <c-string 1..254> / C'...'

C'...

OPTION-NAME = ***STD** / <name 1..8>

,**FILE-NAME** = ***STD** / <filename 1..54 without-gen>

,**RESET** = ***NO** / ***Y**ES

,**SYNCH**RONOUS = ***NO** / ***Y**ES

START-SUBSYSTEM - **Activate PCS**

SUBSYSTEM-NAME = **PCS**
,**VERSION** = ***STD** / <product-version 6..10> / <product-version 3..7 without-man>
,**SUBSYS**TEM**-PAR**AMETER = ***NONE** / <c-string 1..254> / C'...'
C'...'
OPTION-NAME = ***STD** / <name 1..8>
,**FILE-NAME** = ***STD** / <filename 1..54 without-gen>
,**RESET** = ***NO** / ***Y**ES
,**SYNCH**RONOUS = ***NO** / ***Y**ES
,**VERSION-PARALLELISM** = ***NONE** / ***EXCHANGE-MODE**(...) / ***COEXIST**ENCE**-MODE**
***EXCHANGE-MODE**(...)
SUBSYSTEM**-PAR**AMETER = ***NONE** / <c-string 1..254>

STOP**-SUBSYS**TEM - **Deactivate PCS**

SUBSYSTEM-NAME = **PCS**
,**VERSION** = ***STD** / <product-version 6..10> / <product-version 3..7 without-man>
,**SUBSYS**TEM**-PAR**AMETER = ***NONE** / <c-string 1..254>
,**SYNCH**RONOUS = ***NO** / ***Y**ES

10.10 Short descriptions of the CREATE-PCS-OPTION command and the PCSDEFINE statements

The CREATE-PCS-OPTION command is a component of PCSDEFINE.

CREATE-PCS-OPTION - Create and modify parameter files

VERSION = ***STD** / <product-version>
,**MONJV** = ***NONE** / <filename 1..54>
,**CPU-LIMIT** = ***JOB-REST** / <integer 1..32767 seconds>
,**PCS-PARAM**ETER-FILE = ***STD**(...) / <filename 1..54>
***STD**(...)
PPF-VERSION = ***STD** / <product-version>
,**OPTION-NAME** = ***STD** / **STD#DIA** / **STD#BAT** / **STD#TP** / <name 1..7>

The other PCSDEFINE functions are initiated via statements. These statements are described in this section in alphabetical order.

ADD-CATEGORY - Add CATEGORY entries to an option

CATEGORY-NAME = <name 1..7>
,**OPTION-NAME** = <name 1..8>
,**DURATION** = **UNLIMITED** / <integer 0..100000>
,**NEXT-CATEGORY** = ***NONE** / <name 1..7>
,**THROUGHPUT-QUOTA** = **STD** / <integer 0..100>
,**RELATIVE** = (...)
(...)
SERVICE-QUOTA = (...)
(...)
MIN = **STD** / <integer 0..99>
,**MAX** = **STD** / <integer 1..100>
,**REQUEST-DELAY** = **UNCONTROLLED** / (...)
(...)
MIN = **STD** / <integer 0..100>
,**MAX** = **STD** / <integer 0..100>

CHECK-CATEGORY - Check individual CATEGORY entries

CATEGORY-NAME = ***ALL** / list-poss(15): <name 1..7>

,**OPTION-NAME** = <name 1..8>

CHECK-OPTION - Perform consistency and validity checks on form and content of the PCS parameter sets

OPTION-NAME = ***ALL** / list-poss: <name 1..8>

CLOSE-FILE - Close a PCS parameter file (PPF)

FILE-NAME = ***ALL** / ***MAIN** / ***SOURCE** / <filename 1..54>

COPY-CATEGORY - Copy a CATEGORY entry

TO-CATEGORY-NAME = <name 1..7>

,**FROM-CATEGORY-NAME** = <name 1..7>

,**TO-OPTION-NAME** = <name 1..8>

,**FROM-OPTION-NAME** = <name 1..8>

,**FROM-FILE-NAME** = ***SOURCE** / ***MAIN** / <filename 1..54>

COPY-OPTION - Copy OPTION entries

TO-OPTION-NAME = <name 1..8>

,**FROM-OPTION-NAME** = <name 1..8>

,**FROM-FILE-NAME** = ***SOURCE** / ***MAIN** / <filename 1..54>

CREATE-OPTION - Create a new OPTION entry

OPTION-NAME = <name 1..8>

,**SYSTEM-PARAMETER** = (...)

(...)

REQUEST-DELAY-MAX = **STD** / <integer 0..100>,

,**THROUGHPUT-QUOTA** = **STD** / <integer 0..100>

,**USER-INFORMATION** = **NO** / **YES**

,**CATEGORY** = ***STD** / list-poss(12): <name 1..7>

DELETE-OPTION - Delete OPTION entries

OPTION-NAME = ***NONE** / ***ALL** / list-poss(12): <name 1..8>

END - Terminate the PCSDEFINE routine

HELP - List the PCSDEFINE statements

MODIFY-CATEGORY - Modify PCS parameters within a category that has already been created

CATEGORY-NAME = <name 1..7>
,**OPTION-NAME** = <name 1..8>
,**DURATION** = **UNCHANGED** / <integer 0..100000>
,**NEXT-CATEGORY** = ***UNCHANGED** / ***NONE** / <name 1..7>
,**THROUGHPUT-QUOTA** = **UNCHANGED** / <integer 0..100>
,**RELATIVE** = **SERVICE-QUOTA**(...) / **REQUEST-DELAY**(...)
SERVICE-QUOTA(...)
MIN = **UNCHANGED** / <integer 1..99>
,**MAX** = **UNCHANGED** / <integer 0..100>
,**REQUEST-DELAY**(...)
MIN = **UNCHANGED** / <integer 0..100>
,**MAX** = **UNCHANGED** / <integer 0..100>

MODIFY-OPTION - Modify PCS parameter sets (OPTIONs)

OPTION-NAME = <name 1..8>
,**SYSTEM-PARAMETER** = (...)
(...)
REQUEST-DELAY-MAX = **UNCHANGED** / <integer 0..100>
,**THROUGHPUT-QUOTA** = **UNCHANGED** / <integer 0..100>
,**USER-INFORMATION** = **UNCHANGED** / **NO** / **YES**
,**CATEGORY** = ***UNCHANGED** / ***STD** / list-poss(12): <name 1..7>

OPEN-FILE - Open a PCS parameter file (PPF)

FILE-NAME = ***STD**(...) / <filename 1..54>

***STD**(...)

PPF-VERSION = ***STD**

,**MODE** = **MAIN** / **SOURCE**

,**ACCESS** = **WRITE** / **READ**

REMOVE-CATEGORY - Delete CATEGORY parameter sets

CATEGORY-NAME = ***NONE** / ***ALL** / list-poss(15): <name 1..7>

,**OPTION-NAME** = <name 1..8>

SHOW-CATEGORY - Output CATEGORY entries

CATEGORY-NAME = ***ALL** / list-poss: <name 1..7>

,**OPTION-NAME** = <name 1..8>

SHOW-OPTION - Output OPTION entries

OPTION-NAME = ***ALL** / list-poss: <name 1..8>

10.11 SDF syntax representation

The following example shows the representation of the syntax of a command in a manual. The command format consists of a field with the command name. All operands with their legal values are then listed. Operand values which introduce structures and the operands dependent on these operands are listed separately.

HELP-SDF Alias: **HPSD**

GUIDANCE**-MODE** = ***NO** / ***Y**ES

,**SDF-COM**MANDS = ***NO** / ***Y**ES

,**ABBR**EVIATION**-RULES** = ***NO** / *****YES

,**GUID**ED**-DIA**LOG = ***Y**ES (...)

***Y**ES(...)

SCREEN-STEPS = ***NO** / ***Y**ES

,**SPEC**IAL**-FUNC**TIONS = ***NO** / ***Y**ES

,**FUNC**TION**-KEYS** = ***NO** / ***Y**ES

,**NEXT-FIELD** = ***NO** / ***Y**ES

,**UNGUID**ED**-DIA**LOG = ***Y**ES (...) / ***NO**

***Y**ES(...)

SPECIAL**-FUNC**TIONS = ***NO** / ***Y**ES

,**FUNC**TION**-KEYS** = ***NO** / ***Y**ES

This syntax description is valid for SDF V4.5A.The syntax of the SDF command/statement language is explained in the following three tables.

Table 2: Notational conventions

The meanings of the special characters and the notation used to describe command and statement formats are explained in table 2.

Table 3: Data types

Variable operand values are represented in SDF by data types. Each data type represents a specific set of values. The number of data types is limited to those described in table 3.

The description of the data types is valid for the entire set of commands/statements. Therefore only deviations (if any) from the attributes described here are explained in the relevant operand descriptions.

Table 4: Suffixes for data types

Data type suffixes define additional rules for data type input. They contain a length or interval specification and can be used to limit the set of values (suffix begins with *without*), extend it (suffix begins with *with*), or declare a particular task mandatory (suffix begins with *mandatory*). The following short forms are used in this manual for data type suffixes:

cat-id	cat
completion	compl
correction-state	corr
generation	gen
lower-case	low
manual-release	man
odd-possible	odd
path-completion	path-compl
separators	sep
temporary-file	temp-file
underscore	under
user-id	user
version	vers
wildcard-constr	wild-constr
wildcards	wild

The description of the ‘integer’ data type in table 4 contains a number of items in italics; the italics are not part of the syntax and are only used to make the table easier to read.
For special data types that are checked by the implementation, table 4 contains suffixes printed in italics (see the *special* suffix) which are not part of the syntax.

The description of the data type suffixes is valid for the entire set of commands/statements. Therefore only deviations (if any) from the attributes described here are explained in the relevant operand descriptions.

Metasyntax

Representation	Meaning	Examples
UPPERCASE LETTERS	Uppercase letters denote keywords (command, statement or operand names, keyword values) and constant operand values. Keyword values begin with *.	**HELP-SDF** **SCREEN-STEPS** = ***NO**
UPPERCASE LETTERS in boldface	Uppercase letters printed in boldface denote guaranteed or suggested abbreviations of keywords.	**GUID**ANCE**-MODE** = ***Y**ES
=	The equals sign connects an operand name with the associated operand values.	**GUID**ANCE**-MODE** = ***NO**
< >	Angle brackets denote variables whose range of values is described by data types and suffixes (see tables 3 and 4).	**SYNTAX-F**ILE = <filename 1..54>
Underscoring	Underscoring denotes the default value of an operand.	**GUID**ANCE**-MODE** = ***NO**
/	A slash serves to separate alternative operand values.	**NEXT-FIELD** = ***NO** / ***Y**ES
(…)	Parentheses denote operand values that initiate a structure.	,**UNGUID**ED**-DIA**LOG = ***Y**ES (...) / ***NO**
[ ]	Square brackets denote operand values which introduce a structure and are optional. The subsequent structure can be specified without the initiating operand value.	**SELECT** = [***BY-ATTR**IBUTES](...)
Indentation	Indentation indicates that the operand is dependent on a higher-ranking operand.	,**GUID**ED**-DIA**LOG = ***Y**ES (...) ***Y**ES(...) **SCREEN-STEPS** = ***NO** / ***Y**ES

Table 2: Metasyntax (part 1 of 2)

Representation	Meaning	Examples
|	A vertical bar identifies related operands within a structure. Its length marks the beginning and end of a structure. A structure may contain further structures. The number of vertical bars preceding an operand corresponds to the depth of the structure.	**SUP**PORT = ***TAPE**(...) ***TAPE(...)** | **VOL**UME = **_*ANY_**(...) | **_*ANY_**(...) | | ...
,	A comma precedes further operands at the same structure level.	**GUID**ANCE**-MODE** = **_*NO_** / ***Y**ES ,**SDF-COM**MANDS = **_*NO_** / ***Y**ES
list-poss(n):	The entry “list-poss” signifies that a list of operand values can be given at this point. If (n) is present, it means that the list must not have more than n elements. A list of more than one element must be enclosed in parentheses.	list-poss: ***SAM** / ***ISAM** list-poss(40): <structured-name 1..30> list-poss(256): ***OMF** / ***SYSLST**(...) / <filename 1..54>
Alias:	The name that follows represents a guaranteed alias (abbreviation) for the command or statement name.	**HELP-SDF** Alias: **HPSDF**

Table 2: Metasyntax (part 2 of 2)

Data types

Data type	Character set	Special rules
alphanum-name	A…Z 0…9 $, #, @	
cat-id	A…Z 0…9	Not more than 4 characters; must not begin with the string PUB
command-rest	freely selectable	
composed-name	A…Z 0…9 $, #, @ hyphen period catalog ID	Alphanumeric string that can be split into multiple substrings by means of a period or hyphen. If a file name can also be specified, the string may begin with a catalog ID in the form :cat: (see data type filename).
c-string	EBCDIC character	Must be enclosed within single quotes; the letter C may be prefixed; any single quotes occurring within the string must be entered twice.
date	0…9 Structure identifier: hyphen	Input format: yyyy-mm-dd jjjj: year; optionally 2 or 4 digits mm: month tt: day
device	A…Z 0…9 hyphen	Character string, max. 8 characters in length, corresponding to a device available in the system. In guided dialog, SDF displays the valid operand values. For notes on possible devices, see the relevant operand description.
fixed	+, - 0…9 period	Input format: [sign][digits].[digits] [sign]: + oder - [digits]: 0...9 must contain at least one digit, but may contain up to 10 characters (0...9, period) apart from the sign.

Table 3: Data types (part 1 of 6)

Data type	Character set	Special rules
filename	A…Z 0…9 $, #, @ hyphen period	Input format: [:cat:][$user.] { file | file(no) | group | group { (*abs) | (+rel) | (-rel) } } :cat: optional entry of the catalog identifier; character set limited to A...Z and 0...9; maximum of 4 characters; must be enclosed in colons; default value is the catalog identifier assigned to the user ID, as specified in the user catalog. $user. optional entry of the user ID; character set is A…Z, 0…9, $, #, @; maximum of 8 characters; first character cannot be a digit; $ and period are mandatory; default value is the user's own ID. $. (special case) system default ID file file or job variable name; may be split into a number of partial names using a period as a delimiter: $name_1$[.$name_2$[...]] $name_i$ does not contain a period and must not begin or end with a hyphen; file can have a maximum length of 41 characters; it must not begin with a $ and must include at least one character from the range A...Z.

Table 3: Data types (part 2 of 6)

Data type	Character set	Special rules
filename (contd.)		#file (special case) @file (special case) # or @ used as the first character indicates temporary files or job variables, depending on system generation. file(no) tape file name no: version number; character set is A...Z, 0...9, $, #, @. Parentheses must be specified. group name of a file generation group (character set: as for “file”) group{(*abs) (+rel) (-rel)} (*abs) absolute generation number (1-9999); * and parentheses must be specified. (+rel) (-rel) relative generation number (0-99); sign and parentheses must be specified.
integer	0…9, +, -	+ or -, if specified, must be the first character.
name	A…Z 0…9 $, #, @	Must not begin with 0...9.

Table 3: Data types (part 3 of 6)

Data type	Character set	Special rules
partial-filename	A…Z 0…9 $, #, @ hyphen period	Input format: [:cat:][$user.][partname.] :cat: see filename $user. see filename partname optional entry of the initial part of a name common to a number of files or file generation groups in the form: $name_1$.[$name_2$.[...]] $name_i$ (see filename). The final character of “partname” must be a period. At least one of the parts :cat:, $user. or partname must be specified.
posix-filename	A...Z 0...9 special characters	String with a length of up to 255 characters; consists of either one or two periods or of alpha-numeric characters and special characters. The special characters must be escaped with a preceding \ (backslash); the / is not allowed. Must be enclosed within single quotes if alter-native data types are permitted, separators are used, or the first character is a ?, ! or ^. A distinction is made between uppercase and lowercase.
posix-pathname	A...Z 0...9 special characters structure identifier: slash	Input format: [/]$part_1$/.../$part_n$ where $part_i$ is a posix-filename; max. 1023 characters; must be enclosed within single quotes if alter-native data types are permitted, separators are used, or the first character is a ?, ! or ^.

Table 3: Data types (part 4 of 6)

Data type	Character set	Special rules
product-version	A…Z 0…9 period single quote	Input format: [[C]']][V][m]m.naso['] correction status release status where m, n, s and o are all digits and a is a letter. Whether the release and/or correction status may/must be specified depends on the suffixes to the data type (see suffixes without-corr, without-man, mandatory-man and mandatory-corr in Table 4). product-version may be enclosed within single quotes (possibly with a preceding C). The specification of the version may begin with the letter V.
structured-name	A…Z 0…9 $, #, @ hyphen	Alphanumeric string which may comprise a number of substrings separated by a hyphen. First character: A...Z or $, #, @
text	freely selectable	For the input format, see the relevant operand descriptions.
time	0…9 structure identifier: colon	Time-of-day entry: Input format: { hh:mm:ss / hh:mm / hh } hh: hours mm: minutes ss: seconds } Leading zeros may be omitted
vsn	a) A…Z 0…9 b) A…Z 0…9 $, #, @	a) Input format: pvsid.sequence-no max. 6 characters pvsid: 2-4 characters; PUB must not be entered sequence-no: 1-3 characters b) Max. 6 characters; PUB may be prefixed, but must not be followed by $, #, @.

Table 3: Data types (part 5 of 6)

Data type	Character set	Special rules
x-string	Hexadecimal: 00…FF	Must be enclosed in single quotes; must be prefixed by the letter X. There may be an odd number of characters.
x-text	Hexadecimal: 00…FF	Must not be enclosed in single quotes; the letter X must not be prefixed. There may be an odd number of characters.

Table 3: Data types (part 6 of 6)

Suffixes for data types

Suffix	Meaning
x..y *unit*	With data type “integer”: interval specification x minimum value permitted for “integer”. x is an (optionally signed) integer. y maximum value permitted for “integer”. y is an (optionally signed) integer. *unit* with “integer” only: additional units. The following units may be specified: *days* *byte* *hours* *2Kbyte* *minutes* *4Kbyte* *seconds* *Mbyte* *milliseconds*
x..y *special*	With the other data types: length specification For data types catid, date, device, product-version, time and vsn the length specification is not displayed. x minimum length for the operand value; x is an integer. y maximum length for the operand value; y is an integer. x=y the length of the operand value must be precisely x. *special* Specification of a suffix for describing a special data type that is checked by the implementation. “special” can be preceded by other suffixes. The following specifications are used: *arithm-expr* arithmetic expression (SDF-P) *bool-expr* logical expression (SDF-P) *string-expr* string expression (SDF-P) *expr* freely selectable expression (SDF-P) *cond-expr* conditional expression (JV) *symbol* CSECT or entry name (BLS)
with	Extends the specification options for a data type.
-compl	When specifying the data type “date”, SDF expands two-digit year specifications in the form yy-mm-dd to: 20jj-mm-tt if jj < 60 19jj-mm-tt if jj ≥ 60
-low	Uppercase and lowercase letters are differentiated.
-path-compl	For specifications for the data type “filename”, SDF adds the catalog and/or user ID if these have not been specified.

Table 4: Data type suffixes (part 1 of 7)

<table>
<tr><th>Suffix</th><th>Meaning</th></tr>
<tr><td>with (contd.)</td><td></td></tr>
<tr><td>-under</td><td>Permits underscores (_) for the data type “name”.</td></tr>
<tr><td>-wild(n)</td><td>Parts of names may be replaced by the following wildcards.
n denotes the maximum input length when using wildcards.
Due to the introduction of the data types posix-filename and posix-pathname, SDF now accepts wildcards from the UNIX world (referred to below as POSIX wildcards) in addition to the usual BS2000 wildcards. However, as not all commands support POSIX wildcards, their use for data types other than posix-filename and posix-pathname can lead to semantic errors.
Only POSIX wildcards or only BS2000 wildcards should be used within a search pattern. Only POSIX wildcards are allowed for the data types posix-filename and posix-pathname. If a pattern can be matched more than once in a string, the first match is used.
<table>
<tr><th>BS2000 wildcards</th><th>Meaning</th></tr>
<tr><td>*</td><td>Replaces an arbitrary (even empty) character string. If the string concerned starts with *, then the * must be entered twice in succession if it is followed by other characters and if the character string entered does not contain at least one other wildcard.</td></tr>
<tr><td>Termina-ting period</td><td>Partially-qualified entry of a name.
Corresponds implicitly to the string “./*”, i.e. at least one other character follows the period.</td></tr>
<tr><td>/</td><td>Replaces any single character.</td></tr>
<tr><td><s_x:s_y></td><td>Replaces a string that meets the following conditions:
– It is at least as long as the shortest string (s_x or s_y)
– It is not longer than the longest string (s_x or s_y)
– It lies between s_x and s_y in the alphabetic collating sequence; numbers are sorted after letters (A...Z, 0...9)
– s_x can also be an empty string (which is in the first position in the alphabetic collating sequence)
– s_y can also be an empty string, which in this position stands for the string with the highest possible code (contains only the characters X’FF’)</td></tr>
<tr><td><s_1,…></td><td>Replaces all strings that match any of the character combinations specified by s. s may also be an empty string. Any such string may also be a range specification “s_x:s_y” (see above).</td></tr>
</table>
</td></tr>
</table>

Table 4: Data type suffixes (part 2 of 7)

Suffix	**Meaning**	
with-wild(n) (contd.)	-s	Replaces all strings that do not match the specified string s. The minus sign may only appear at the beginning of string s. Within the data types filename or partial-filename the negated string -s can be used exactly once, i.e. -s can replace one of the three name components: cat, user or file.
	Wildcards are not permitted in generation and version specifications for file names. Only system administration may use wildcards in user IDs. Wildcards cannot be used to replace the delimiters in name components cat (colon) and user ($ and period).	
	POSIX wildcards	Meaning
	*	Replaces any single string (including an empty string). An * appearing at the first position must be duplicated if it is followed by other characters and if the entered string does not include at least one further wildcard.
	?	Replaces any single character; not permitted as the first character outside single quotes.
	$[c_x\text{-}c_y]$	Replaces any single character from the range defined by c_x and c_y, including the limits of the range. c_x and c_y must be normal characters.
	[s]	Replaces exactly one character from string s. The expressions $[c_x\text{-}c_y]$ and [s] can be combined into $[s_1c_x\text{-}c_ys_2]$.
	$[!c_x\text{-}c_y]$	Replaces exactly one character not in the range defined by c_x and c_y, including the limits of the range. c_x and c_y must be normal characters. The expressions $[!c_x\text{-}c_y]$ and [!s] can be combined into $[!s_1c_x\text{-}c_ys_2]$.
	[!s]	Replaces exactly one character not contained in string s. The expressions [!s] and $[!c_x\text{-}c_y]$ can be combined into $[!s_1c_x\text{-}c_ys_2]$.

Table 4: Data type suffixes (part 3 of 7)

<table>
<tr><th>Suffix</th><th>Meaning</th></tr>
<tr><td>with (contd.)

wild-
constr(n)</td><td>Specification of a constructor (string) that defines how new names are to be constructed from a previously specified selector (i.e. a selection string with wildcards). See also with-wild. n denotes the maximum input length when using wildcards.
The constructor may consist of constant strings and patterns. A pattern (character) is replaced by the string that was selected by the corresponding pattern in the selector.
The following wildcards may be used in constructors:
<table>
<tr><th>Wildcard</th><th>Meaning</th></tr>
<tr><td>*</td><td>Corresponds to the string selected by the wildcard * in the selector.</td></tr>
<tr><td>Termina-
ting period</td><td>Corresponds to the partially-qualified specification of a name in the selector;
corresponds to the string selected by the terminating period in the selector.</td></tr>
<tr><td>/ or ?</td><td>Corresponds to the character selected by the / or ? wildcard in the selector.</td></tr>
<tr><td><n></td><td>Corresponds to the string selected by the n-th wildcard in the selector, where n is an integer.</td></tr>
</table>
Allocation of wildcards to corresponding wildcards in the selector:
All wildcards in the selector are numbered from left to right in ascending order (global index).
Identical wildcards in the selector are additionally numbered from left to right in ascending order (wildcard-specific index).
Wildcards can be specified in the constructor by one of two mutually exclusive methods:
1. Wildcards can be specified via the global index: <n>
2. The same wildcard may be specified as in the selector; substitution occurs on the basis of the wildcard-specific index. For example:
the second “/” corresponds to the string selected by the second “/” in the selector</td></tr>
</table>

Table 4: Data type suffixes (part 4 of 7)

Suffix	Meaning
with-wild-constr(n) (contd.)	The following rules must be observed when specifying a constructor: – The constructor can only contain wildcards of the selector. – If the string selected by the wildcard <...> or [...] is to be used in the constructor, the index notation must be selected. – The index notation must be selected if the string identified by a wildcard in the selector is to be used more than once in the constructor. For example: if the selector “A/” is specified, the constructor “A<n><n>” must be specified instead of “A//”. – The wildcard * can also be an empty string. Note that if multiple asterisks appear in sequence (even with further wildcards), only the last asterisk can be a non-empty string, e.g. for “****” or “*//*”. – Valid names must be produced by the constructor. This must be taken into account when specifying both the constructor and the selector. – Depending on the constructor, identical names may be constructed from different names selected by the selector. For example: “A/*” selects the names “A1” and “A2”; the constructor “B*” generates the same new name “B” in both cases. To prevent this from occurring, all wildcards of the selector should be used at least once in the constructor. – If the constructor ends with a period, the selector must also end with a period. The string selected by the period at the end of the selector cannot be specified by the global index in the constructor specification.

Table 4: Data type suffixes (part 5 of 7)

<table>
<tr><th>Suffix</th><th>Meaning</th></tr>
<tr><td>with-wild-
constr(n)
(contd.)</td><td>Examples:
<table>
<tr><th>Selector</th><th>Selection</th><th>Constructor</th><th>New name</th></tr>
<tr><td>A//*</td><td>AB1
AB2
A.B.C</td><td>D<3><2></td><td>D1
D2
D.CB</td></tr>
<tr><td>C.<A:C>/<D,F></td><td>C.AAD
C.ABD
C.BAF
C.BBF</td><td>G.<1>.<3>.XY<2></td><td>G.A.D.XYA
G.A.D.XYB
G.B.F.XYA
G.B.F.XYB</td></tr>
<tr><td>C.<A:C>/<D,F></td><td>C.AAD
C.ABD
C.BAF
C.BBF</td><td>G.<1>.<2>.XY<2></td><td>G.A.A.XYA
G.A.B.XYB
G.B.A.XYA
G.B.B.XYB</td></tr>
<tr><td>A//B</td><td>ACDB
ACEB
AC.B
A.CB</td><td>G/XY/</td><td>GCXYD
GCXYE
GCXY. [1]
G.XYC</td></tr>
</table>
[1] The period at the end of the name may violate naming conventions (e.g. for fully-qualified file names).</td></tr>
<tr><td>without</td><td>Restricts the specification options for a data type.</td></tr>
<tr><td>-cat</td><td>Specification of a catalog ID is not permitted.</td></tr>
<tr><td>-corr</td><td>Input format: [[C]’][V][m]m.na[’]
Specifications for the data type product-version must not include the correction status.</td></tr>
<tr><td>-gen</td><td>Specification of a file generation or file generation group is not permitted.</td></tr>
<tr><td>-man</td><td>Input format: [[C]’][V][m]m.n[’]
Specifications for the data type product-version must not include either release or correction status.</td></tr>
<tr><td>-odd</td><td>The data type x-text permits only an even number of characters.</td></tr>
<tr><td>-sep</td><td>With the data type “text”, specification of the following separators is not permitted: ; = ( ) < > ␣ (i.e. semicolon, equals sign, left and right parentheses, greater than, less than, and blank).</td></tr>
<tr><td>-temp-
file</td><td>Specification of a temporary file is not permitted (see #file or @file under filename).</td></tr>
</table>

Table 4: Data type suffixes (part 6 of 7)

Suffix	Meaning
without (contd.)	
-user	Specification of a user ID is not permitted.
-vers	Specification of the version (see “file(no)”) is not permitted for tape files.
-wild	The file types posix-filename and posix-pathname must not contain a pattern (character).
mandatory	Certain specifications are necessary for a data type.
-corr	Input format: [[C]’][V][m]m.naso[’] Specifications for the data type product-version must include the correction status and therefore also the release status.
-man	Input format: [[C]’][V][m]m.na[so][’] Specifications for the data type product-version must include the release status. Specification of the correction status is optional if this is not prohibited by the use of the suffix without-corr.
-quotes	Specifications for the data types posix-filename and posix-pathname must be enclosed in single quotes.

Table 4: Data type suffixes (part 7 of 7)

Abbreviations

DB	DataBase
DMS	Data Management System
DSSM	Dynamic SubSystem Management
EXCP	Execute Channel Program
FHS	Format Handling System
JMU	Job Management Utility
MM	Main Memory
MPL	Multiprogramming Level
PCS	Performance Control System
PPF	PCS Parameter File
R-D	Request Delay
S-Q	Service Quota
SSCH	Start SubCHannel
SIH	System Interrupt Handling
SU	Service Unit
SVC	SuperVisor Call
TA	TransAction
T-Q	Throughput Quota
TP	Transaction Processing
TPR	Task PRivileged
TR	Transaction Rate
TSA	Task Scheduling Attribute
TSN	Task Sequence Number
TU	Task Unprivileged

UDS	Universal Database System
UTM	Universal Transaction Monitor
WSI	Working Set Integral

Related publications

The manuals are available as online manuals, see *http://manuals.fujitsu-siemens.com*, or in printed form which must be payed and ordered separately at *http://FSC-manualshop.com*.

[1] **BS2000/OSD-BC V6.0**
Performance Handbook

Target group
Computer center and system support staff
Contents
The manual helps system users to evaluate the performance of their dp system and points out how to use hardware and software cost-effectively and how to improve system performance. Diagrams, formulas and examples explain the processes in the system and their influence on overall performance.
Order number
U1794-J-Z125-11-76

[2] **OSD/XC**
Performance Guidelines
User Guide

Target group
Computer center and system support staff.
Contents
This manual focuses on the principles and measurements involved in evaluating the performance of BS2000/OSD applications which run on SX systems with SPARC64 architecture. The performance-related characteristics of the SX server architecture are described together with the underlying operating sequences. Extensive information on tuning the configuration and the software make it possible to optimize the economic efficiency of OSD/XC operation.

[3] ***open*SM2 V6.0A** (BS2000/OSD)
Software Monitor Volumes 1 and 2
User Guide

Target group
This manual is addressed to users and systems support staff.
Contents
*open*SM2 V6.0A (BS2000/OSD) supplies users with statistical data on the performance of their BS2000/OSD and on resource utilization.
Volume 1 of the manual describes the operation of the SM2 monitor, the SM2 monitoring programs and the SM2 screen reports.
Volume 2 of the manual describes the SM2U1 utility routine for editing and administering the SM2 output files, and the analysis routines SM2R1, ANALYZER, INSPECTOR and SM2-PA.
Order numbers
U3585-J-Z125-10-76 Volume 1: Administration and Operation
U41078-J-Z125-3-76 Volume 2: Analysis and Display of SM2 Monitored Data

[4] **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

[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] **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

For a description of the IOGEN utility which replaces the UGEN utility for hardware generation as of BS2000/OSD-BC V5.0B, see Readme file SYSRME.BS2CP.140.E.

[7] **SDF V4.5A** (BS2000/OSD)
Introductory Guide to the SDF Dialog Interface
User Guide

Target group
BS2000/OSD users
Contents
This manual describes the interactive input of commands and statements in SDF format. A Getting Started chapter with easy-to-understand examples and further comprehensive examples facilitates use of SDF. SDF syntax files are discussed.
Order number
U2339-J-Z125-8-76

[8] **DSSM V4.0/SSCM V2.3**
Subsystem Management in BS2000/OSD
User Guide

Target group
This manual addresses systems support staff and software consultants of BS2000/OSD.
Contents
The following are described: BS2000/OSD subsystem concept, dynamic subsystem management (DSSM) V4.0, subsystem catalog management (SSCM) V2.3 and the associated commands and statements.
DSSM supports the option of creating and managing user-specific subsystem configurations on a task-local basis.
Order number
U23166-J-Z125-3-76

[9] **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

Index

X

Contents

PCS V2.7A (BS2000/OSD)

Performance Control Subsystem

User Guide

Target group

This manual is addressed to systems support staff.

Contents

The manual describes how the Performance Control Subsystems (PCS) can be used to optimize the performance of a computer system in accordance with the task category concept. An introduction to the basic principles of PCS is followed by a description of PCS operation. The possible values for the PCS parameters are described in detail. Important information on the various settings are presented in a number of tables.

Edition: December 2004

File: pcs.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 PCS V2.7A
Performance Control System

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.