

# Fujitsu Software BS2000 ASTI

Version V21.0B

May 2022



## Readme file

---

All rights reserved, including intellectual property rights. Technical data subject to modifications and delivery subject to availability. Any liability that the data and illustrations are complete, actual or correct is excluded. Designations may be trademarks and/or copyrights of the respective manufacturer, the use of which by third parties for their own purposes may infringe the rights of such owner.

Copyright © 2024 Fujitsu

The Fujitsu brand and the Fujitsu logo are registered trademarks of Fujitsu Limited, Japan in Europe and other countries.

<b>1</b>	<b>General</b>	<b>2</b>
<b>2</b>	<b>Operating</b>	<b>3</b>
<b>3</b>	<b>Commands</b>	<b>4</b>
3.1	Command Return Codes	4
3.2	Service Management Commands	4
3.2.1	/START-SERVICE	4
3.2.2	/STOP-SERVICE	8
3.2.3	/SHOW-SERVICE-STATUS	8
3.2.4	/MODIFY-SERVICE-PARAMETER	8
3.3	Client Commands	9
3.3.1	/SHOW-ORDER-STATUS	9
3.3.2	/DELETE-ORDER	10
3.3.3	/REQUEST-ORDER-RESULT	11
3.3.4	/SEND-ORDER	11
3.4	Service Command	12
3.4.1	/PROCESS-ORDER	12
<b>4</b>	<b>Macros</b>	<b>14</b>
<b>5</b>	<b>Procedures</b>	<b>15</b>
5.1	Reset of a Central Lock in ASTI	15
5.2	Example of a Service Procedure	16

# 1 General

The subsystem ASTI is a component that manages the new "Service" interface and performs the communications between one or more client tasks and at least one service task.

ASTI is part of the basic configuration and is initially only released for internal purposes. Since the commands appear at the user interface, they are described in this paper.

\*4 The release level is that of May 2022.

\*4 Changes to release level January 2021 are marked with \*4.

\*3 Changes to release level June 2016 are marked with \*3.

\*2 Changes to release level June 2014 are marked with \*2.

\*1 Changes to release level July 2013 are marked with \*1.

This and other current Readme files are available online at <https://bs2manuals.ts.fujitsu.com/index>.

## 2 Operating

ASTI is not a stand-alone system that directly provides a service to the customer. It is intended for subsystem developers that must realize a type of client/server model. ASTI thereby provides the connection between the client and the server. Connections can be made in all directions (TU-TU, TPR-TU, TU-TPR and TPR-TPR). The client issues an order to the server and the server then processes this. ASTI takes care of the communications between the different tasks.

ASTI takes the order and puts it into a wait queue for the service concerned. All parameters are stored. The service queries ASTI as to whether there is an order pending for it. If there is one for this service, it is forwarded to the service concerned. The service executes the function and reports the result to the client. The processes can be delivered either synchronously, i.e., the client waits for the result, or asynchronously. The client can also control how long the orders must be held in the system before they are processed. An order is stored until it is processed. The client can also control how long the result of an order is stored.

There are three options:

1. The client does not want to receive a result; ASTI rejects the result.
2. The client wants to receive the result during the service session. The return message is held in main memory and deleted after the service session is ended.
3. The client wants to receive the result in all cases. The return message is stored on the disk. The return message is only deleted after it has been successfully delivered.

The subsystem ASTI manages the orders of the services in the files \$TSOS.SYSQUE.ASTI and \$\*. SYS.<tsn>.<service-name>.<order-id>.

\*3 The format of these files has changed in ASTI V2.0B. Such files with orders from ASTI V2.0A cannot be processed in successor versions. Therefore, these files must be removed before migrating from ASTI V2.0A to ASTI V2.0B or a successor version. The subsystem ASTI must not be started

.\$TSOS.SYSQUE.ASTI is created at start of the subsystem ASTI, if the file does not exist.

Following procedure is recommended:

```

/STOP-MAIL-SERVICE
/STOP-LWRESO
*1 /STOP-SUBSYSTEM REWAS
/STOP-SUBSYSTEM ASTI

/DELETE-FILE $TSOS.SYSQUE.ASTI
/DELETE-FILE $*. SYS.<tsn>.<service-name>.<order-id> (if existing)

```

## 3 Commands

### 3.1 Command Return Codes

All commands set subcode 1, subcode 2, and main code. In case of success, the main code gets the value CMD0001. If an error occurs, the main code contains the key to the error message and the spin-off mechanism is triggered.

The service and the client commands also set SDF-P variables. These variables are only set in case of success.

```
SVTVAR-ORDERID
SVTVAR-SERVICE
SVTVAR-DATA
```

### 3.2 Service Management Commands

#### 3.2.1 /START-SERVICE

This command starts a service. A service is a batch job that is generated by ASTI and starts just one program. The program is defined by a symbol and a library. When the program terminates, the job also terminates.

```
/START-SERVICE
```

```
SERVICE-NAME = <text_4..16>
```

```
,FROM-FILE = *LIBRARY-ELEMENT(..)
    (LIBRARY = <full-filename_1..54>
    ,SYMBOL = <composed-name_1..32_with_underscore>
    | <c-string_1..32_with_lower-case>)
| *BY-IMON(..)
    (LOGICAL-ID = SYSLNK | <composed-name_1..30>
    ,INSTALLATION-UNIT = <composed-name_1..30>
    ,VERSION = *STD | <version 3..7>
    ,DEFAULT-LIBRARY = <full-filename_1..54>
    ,SYMBOL = <composed-name 1..32 with underscore>)
| *NONE(TSN = <text_4..4>)
| *PROCEDURE(..)
    (<full-filename_1..54> | *LIBRARY-ELEMENT(..)
    (LIBRARY = <full-filename_1..54>
    ,ELEMENT = <composed-name 1..32_with_underscore>))

,PROCESSING-ADMISSION = *FROM-ORDER
| *FROM-START
| *EXPLICIT(..)
    (USERID = <name_1..8>
    ,ACCOUNT = <text_1..8>
    ,PASSWORD = *NONE | <secure1..8>)
```

```

,ALLOWED-USER = *ALL
    | *NONE
    | *START-USER-IDENTIFICATION(..)
*2      (ALTERNATE-ADMIN = *NONE | list_poss(8): <name_1..8>)
    | *BY-GUARD(NAME = <full-filename_1..24>)

,ORDER-RECOVERY = *PARAMETER(..)
    (ALLOWED = *NO | *SESSION-WIDE | *PERMANENT
    ,DEFAULT = *NO | *SESSION-WIDE | *PERMANENT)

,ORDER-LIMIT = 255 | <integer_1..32000>

,NUMBER-OF-TASKS = 1 | <integer_1..16>
    (MINIMAL = 0 | <integer_0..16>
    ,MAXIMAL = 16 | <integer_1..16>)

,START-PARAMETER = *NONE | <c-string_1..1800> | <x-string_1..3600>

,NOT-ALLOWED-CMD = *NONE
    | list-poss(5): *STOP-SERVICE | *MODIFY-SERVICE-PARAM |
    *SEND-ORDER | *DELETE-ORDER |
    *REQUEST-ORDER-RESULT

,NOT-ALLOWED-SVC = *NONE |
    | list-poss(5): *SVTSTOP | *SVTMODI |
    *SVTSORD | *SVTDORD | *SVTRORD

,SYSTEM-OUTPUT = *ALL | *NONE | *STDOUT | *PARAMETER(..)
    (SYSLSL-OUTPUT = *STDOUT | *NONE | *PRINTER | *MAIL
    ,SYSOUT-OUTPUT = *STDOUT | *NONE | *PRINTER | *MAIL )
*2 ,MONJV = *NONE | <full-filename_1..24>

```

SERVICE-NAME	A name that is unique for this system. Services beginning with \$ can only started by TSOS.
FROM-FILE	This parameter supplies the load library and the symbol to be loaded. Either direct input *LIBRARY-ELEMENT or indirect input via the installation monitor *BY-IMON are possible. The option *NONE can be used to make a running dialog task be a service task.
*LIBRARY-ELEMENT	
LIBRARY	Name of the module library
SYMBOL	Name of a CSECT or an ENTRY with which a module is loaded and started.
*BY-IMON	
LOGICAL-ID	Logical name of the module library
INSTALL-UNIT	Name of a product
VERSION	Version of a product
	*STD is the highest installed version.
DEFAULT-LIB	Default library name if the product is not installed
SYMBOL	Name of a CSECT or an ENTRY with which a module is loaded and started.
*NONE	
TSN	The TSN of a dialog task can be input here.

*PROCEDURE	
FILENAME	Name of a procedure file
LIBRARY	Name of a module library
ELEMENT	Name of a j-element in a library

## PROCESSING-ADMISSION

regulates which user environment the service works in.

*FROM-ORDER	The service takes over the user environment of the task that has sent the order. This function is not allowed for FROM-FILE=*NONE.
*FROM-START	The service gets a user ID that is not changed during operation.
*EXPLICIT	Specification of the user ID under which the service will be started
USERID	The specified user ID is taken.
ACCOUNT	Account number of the User-ID
PASSWORD	*NONE           No password assigned <secure_1..8>    Password of the User-ID

ALLOWED-USER           controls access to the service

## \*START-USER-IDENTIFICATION(...)

Only the same user ID as with START-SERVICE is allowed, or one of the alternate user IDs from the list.

\*2

## ALTERNATE-ADMIN

A list of up to eight (different) user IDs which also can manage the service.

*BY GUARD	The user IDs are permitted via a guard. The guard is in the start user ID. The length of the guard is (1-8 character(s))
*NONE	The service uses only the service management.
*ALL	All tasks have access.

## ORDER-RECOVERY=\*PARAMETER(...)

The value range of the parameter ORDER-RECOVERY in the command /SEND-ORDER can be influenced with this parameter.

ALLOWED           The value range of the parameter ORDER-RECOVERY in the command /SEND-ORDER can be limited with this parameter.

\*NO                The ORDER and the RESULT are only held in main memory. If a task or the service dies, the orders concerned are acknowledged negatively. If the order task is missing, the results are deleted.

\*SESSION-WIDE    The ORDER is held in memory until the order has been processed. Unprocessed orders and results are held in main memory and only deleted after SESSION-END (/SHUTDOWN or /STOP-SUB ASTI).

\*PERMANENT       The ORDER and the RESULT are stored on the disk. Each part is only deleted after the respective action is completed. With this function, incorrect programming can create order cadavers that remain in the order user ID in the form of files.

DEFAULT           The ORDER-RECOVERY default value is set with this.

ORDER-LIMIT       This defines the maximum number of ORDERS being processed at the same time.

NUMBER-OF-TASKS   1-16  
Number of service tasks that work for this service

MINIMAL	0-16 The minimal number of service tasks, which would be changed by /MODIFY-SERVICE-PARAMETER.
MAXIMAL	1-16 The maximal number of service tasks, which would be changed by /MODIFY-SERVICE-PARAMETER.
START-PARAMETER	This parameter must be queried by the service after the service task has been started. The record is used for initializing the service.
NOT-ALLOWED-CMD	Specific commands can be blocked for a service with this parameter. <ul style="list-style-type: none"> <li>*STOP-SERVICE            This service cannot be stopped with this command.</li> <li>*MODIFY-SERVICE-PARAM    This service cannot be modified.</li> <li>*SEND-ORDER              No order can be placed to this service.</li> <li>*DELETE-ORDER            No orders can be deleted.</li> <li>*REQUEST-ORDER-RESULT   No results can be fetched.</li> </ul>
NOT-ALLOWED-SVC	Program calls for a service can be blocked with this parameter. <ul style="list-style-type: none"> <li>*SVTSTOP                 This service cannot be stopped by the program.</li> <li>*SVTMODI                 This service cannot be modified by the program.</li> <li>*SVTSORD                 No order can be placed by the program.</li> <li>*SVTDORD                 No orders can be deleted by the program.</li> <li>*SVTRORD                 No results can be fetched by the program.</li> </ul>
Note: The TPR interface is not subject to any of these restrictions.	
SYSTEM-OUTPUT	setting parameter of EXIT-JOB <ul style="list-style-type: none"> <li>*ALL                      SYSLST und SYSOUT are printed.</li> <li>*NONE                     No output</li> <li>*STDOUT                  Behavior specified by CLASS-II-Option "SSMOUT"</li> <li>*PARAMETER <ul style="list-style-type: none"> <li>SYSLST-OUTPUT        Parameter for SYSLST</li> <li>SYSOUT-OUTPUT        Parameter for SYSOUT</li> <li>*STDOUT                Class-II-Option "SSMOUT"</li> <li>*NONE                  No output of selected file</li> <li>*PRINTER                Print of selected file</li> <li>*MAIL                    Sending file as mail</li> </ul> </li> </ul>
*2	MONJV                     The name of a JV to monitor the task. <ul style="list-style-type: none"> <li>*NONE                    MONJV is not used here.</li> </ul>

Note: there can be only one task being monitored by a MONJV.

### 3.2.2 /STOP-SERVICE

This command stops a service. The service is set to termination mode. The separate tasks must stop themselves. New orders are no longer accepted.

Remaining orders are handled according to the settings, which have been made at their sending by the parameter ORDER-RECOVERY.

```
/STOP-SERVICE SERVICE-NAME= <text_4..16>
```

SERVICE-NAME            Name of the service

### 3.2.3 /SHOW-SERVICE-STATUS

The status of a single service or of all services are displayed with this command. This information can also be output to OPS-variables.

```
/SHOW-SERVICE-STATUS SERVICE-NAME = *ALL | <text_4..16>
```

#### Output SERVICE-NAME=\*ALL

SID	SERVICE-NAME	USERID	STATUS	TASK	ALLOWED-USER	PRO-ADM
@006	SERVICE3	ASTITEST	RUNNING	0001	ALL	FROM-START
@005	SERVICE2	TSOS	RUNNING	0001	START-USERID	FROM-ORDER
@004	SERVICE1	TSOS	RUNNING	0001	ALL	FROM-ORDER
@003	MAILCLNT	TSOS	RUNNING	0001	ALL	FROM-START
@002	LWRES	TSOS	RUNNING	0001	START-USERID	FROM-START
@001	\$ASTIQUE	TSOS	RUNNING	0001	ALL	FROM-START

#### Output SERVICE-NAME=MAILCLNT

```
SID SERVICE-NAME STATUS ADMINSTR ALLOWED-USER PROCESSING MODE
@003 MAILCLNT RUNNING TSOS ALL FROM-START SERVICE
ORDER-RECOVERY: ALLOWED = PERMANENT DEFAULT = PERMANENT
SERVICE-TASKS : 01 LIMITS (MIN=00,MAX=16)
SYSTEM-OUTPUT : SYSOUT = *NONE SYSLST = *NONE
SYMBOL : MSSRVC
LIBRARY : :SQN6:$TSOS.SYSLNK.MAIL.032.BACKEND
GUARD : *NONE
TASK: 9UKX ACT-ORDER: 000 TERMINATION: NONE START-PARAM: NONE
% SVTS000 Service MAILCLNT: Command executed
```

### 3.2.4 /MODIFY-SERVICE-PARAMETER

```
/MODIFY-SERVICE-PARAMETER
```

```
SERVICE-NAME = <text_4..16>
```

```
,ALLOWED-USER = *UNCHANGED
| *ALL
| *START-USER-IDENTIFICATION
| *BY-GUARD(NAME=<full-filename_1..24>)
```

```
,NUMBER-OF-TASKS = *UNCHANGED | *PAST-VALUE | <integer_0..16>
```

```
,ORDER-LIMIT = *UNCHANGED | <integer_1..32000>
```

```
,START-PARAMETER = *UNCHANGED | *NONE
| <c-string_1..1800> | <x-string_1..3600>
```

SERVICE-NAME	Name of the service
ALLOWED-USER	controls who can use the service.
NUMBER-OF-TASKS	Number of service tasks that can work with the same service name
0-16	The number of tasks can be reduced to 0. In this case, the service is still present and accepts orders, but does not execute them. This parameter is ignored for a dialog service.
*PAST-VALUE	sets the number to the value before the last /MODIFY-SERVICE-PARAMETER.
ORDER-LIMIT	The number of orders for this service that are in the system can be limited with this parameter.
START-PARAMETER	The initialization parameters can be changed with this parameter. The next /PROCESS-ORDER receives the RC "new parameter". If a /PROCESS-ORDER is in the wait queue when this parameter is modified, RC x'00040010' is only set after the queue timeout.

### 3.3 Client Commands

These commands connect the client to the service.

#### 3.3.1 /SHOW-ORDER-STATUS

Information about orders is output with this command. The command works asynchronously to normal operation and therefore the consistency of the number of orders and the separate sub-queues is not assured.

This information can also be output into OPS-variables.

/SHOW-ORDER-STATUS

```
INFORMATION = *SUMMARY ( SERVICE-NAME = *ALL | <text_4..16> )
              *ORDER-LIST ( SERVICE-NAME = *ALL | <text_4..16> )
              *ALL ( ORDER-ID = <x-text_1..16 > )
```

INFORMATION

*SUMMARY	Number of orders per subqueue
*ORDER-LIST	All relevant order IDs, including status
*ALL	Information about the specified order (Administrator: about all orders, normal user: about his own)

SERVICE-NAME	Name of the service
--------------	---------------------

ORDER-ID	Order identification
----------	----------------------

During the same subsystem session, the last eight bytes are sufficient, for orders of other sessions all sixteen bytes are required.

Output INFORMATION=\*SUMMARY

SERVICE	ALL-Q	RDY-Q	ACT-Q	RES-Q	WAI-Q	NRR-Q	IAC-Q	DEQ-R
SERVICE4	3	1	1	1	0	0	0	0

ALL-Q all orders of this service  
 RDY-Q all orders that are waiting for execution by the service  
 ACT-Q all orders that are currently being executed  
 RES-Q all results that have not been fetched yet  
 WAI-Q all orders that have been deferred  
 NRR-Q all results that have not been requested  
 AC-Q orders from stopped service (session-wide)  
 DEQ-R results whose tasks no longer exist

Output INFORMATION=\*ORDER-LIST

ORDER-ID	SERVICE	QUEUE	TASK
1266CAAA00000007	SERVICE4	READY-QUEUE	0BA4
1266CAAA00000006	SERVICE4	ACTIVE-QUEUE	0BA4
1266CAAA00000005	SERVICE4	RESULT-QUEUE	0BA4

% SVTS000 Service SERVICE4: Command executed

Output INFORMATION=\*ALL

ORDER	SERVICE	CURRENT QUEUE	
1266CAAA00000007	SERVICE4	READY-QUEUE	
USERID	ACCOUNT	TSN	USER-STATE
TSOS	ADMINSTR	0BA4	RESULT-REQUESTED

Comment:

USERID user Id of the sender  
 ACCOUNT account number of the sender  
 TSN TSN of the sender task  
 USER-STATE state of result request

**3.3.2 /DELETE-ORDER**

Either one order or all orders of a service can be deleted with this command. A normal user can only delete orders of his own user ID, whereas the administrator can delete all orders of the service concerned.

/DELETE-ORDER

ORDER-ID = \*TASK(SERVICE-NAME = <text\_4..16>  
 |\*USER(SERVICE-NAME = <text\_4..16>  
 |\*ALL(SERVICE-NAME = <text\_4..16>  
 |<x-text\_1..16>

ORDER-ID	Order identification During the same subsystem session, the last eight bytes are sufficient, for orders of other sessions all sixteen bytes are required. Only orders from the same task are deleted. Exception: permanent orders, whose callers no longer exist.
*TASK	All orders that this task sent to the service are deleted, including permanent orders.
*USER	All orders that were sent to the service by this user ID are deleted, including permanent orders
*ALL	All orders that were sent to the service by this user ID are deleted; with the administrator, all orders are deleted.
SERVICE-NAME	Name of the service

Note:

All orders concerned are searched for and marked with "order to delete". Orders that are currently being processed by the service are completed by the service. If the client is waiting, the result is delivered. If the client is not waiting, the result is deleted. Orders in the ready queue or in the wait queue are deleted. If the client is waiting for the result, ASTI delivers a negative result (key: SVTS903).

**3.3.3 /REQUEST-ORDER-RESULT**

The result of an asynchronous order is fetched with this command. The result is thereby deleted in the ASTI management. Results of permanent orders whose caller task no longer exists can also be fetched with this command.

/REQUEST-ORDER-RESULT

```
ORDER-ID= <x-text_1..16>
          | *TASK(SERVICE-NAME=<text_1..16>)
          | *USER(SERVICE-NAME=<text_1..16>)
*4 ,WAIT-FOR-RESULT = *YES (TIME-LIMIT= *NO | <integer_1..2147483647>)
          | *NO
```

ORDER-ID	Order identification
	During the same subsystem session, the last eight bytes are sufficient, for orders of other sessions all sixteen bytes are required.
*TASK	The next order of this task is requested.
*USER	The next order of this user ID is requested.
SERVICE-NAME	Selects the service.
WAIT-FOR-RESULT	specifies, whether the client wants to wait for the result.
	TIME-LIMIT specifies the time for waiting.

\*4 The parameter value \*USER for ORDER-ID also allows to request the result of an order whose request permission has been set to \*USER at /SEND-ORDER or which has been sent by a task which is no longer present.

**3.3.4 /SEND-ORDER**

An order is placed with this command.

/SEND-ORDER

```
SERVICE-NAME = <text_4..16>
*4 ,WAIT-FOR-RESULT = *YES (TIME-LIMIT = *NO | <integer_1..2147483647>)
*4                   | *NO (RESULT = *NO | *YES(...))
*4                   REQUEST-PERMISSION = *STD | *USER
,ORDER-RECOVERY = *STD
                   | *NONE
                   | *SESSION-WIDE
                   | *PERMANENT
,DATA = *NO | <c-string 1...1800> | <x-string 1..3600>
```

SERVICE-NAME                      Name of the service

WAIT-FOR-RESULT

\*YES The order is accepted, and the task waits until the order is completed. The result is displayed.  
 TIME-LIMIT TIME-LIMIT restricts the waiting time.  
 \*NO The order is accepted, and the task waits until the order.  
 RESULT=\*NO No result requested  
 RESULT=\*YES A result is requested, that must be fetched later with /REQUEST-ORDER-RESULT.

\*4 The permission for requesting the result is restricted.

\*4 REQUEST-PERMISSION controls the permission for requesting the result.  
 \*STD The standard method is used. Only the sending client task has the permission to request the result. If the client task has been terminated in the meantime and the result is still existing due to the setting of ORDER-RECOVERY, client tasks with the same user ID as the sending task have the permission to request the result.  
 \*USER Each client task running under the same user ID as the sending task can collect the result even if the sending task is still alive.

#### ORDER-RECOVERY

\*STD The default value of /START-SERVICE is taken.  
 \*NONE No special measures are met regarding transaction security.  
 \*SESSION-WIDE The order is retained until acknowledged by the service and the result is also retained until delivery to the client. If these events do not occur, the order or the result is deleted when the session ends.  
 \*PERMANENT In contrast to \*SESSION-WIDE, neither orders nor results are deleted at termination of the service or the subsystem ASTI. Results can be fetched at next start of ASTI session; orders are delivered at start of the service concerned.

DATA The contents of the parameter represent the order to the service task.

### 3.4 Service Command

#### 3.4.1 /PROCESS-ORDER

The service fetches orders from ASTI and returns results with this command.

/PROCESS-ORDER

ACTION= \*GET-ORDER (WAIT-FOR-ORDER =  
 \*YES(TIME-LIMIT = \*NO  
 | <integer\_1..2147483647>)  
 | \*NO

\*4

\*SEND-ACK (ORDER-ID=<x-text\_1..16>  
 ,RETURN-DATA = \*NONE  
 | <c-string\_1..1800>  
 | <x-string\_1..3600>)

\*SEND-NAK (ORDER-ID = <x-text\_1..16>  
 ,RETURN-KEY = <text\_7..7>)  
 ,RETURN-DATA = \*NONE  
 | <c-string\_1..1800>  
 | <x-string\_1..3600>)

```
*SEND-RETRY(ORDER-ID = <x-text_1..16>
             ,WAIT-TIME = <integer_1..2000>
             ,RETURN-DATA = *NONE
                       | <c-string_1..1800>
                       | <x-string_1..3600>)
```

```
*GET-START-PARAMETER
```

## ACTION

*GET-ORDER	Requests a new order.
WAIT-FOR-ORDER	
*YES	The service task waits until a new order arrives.
*NO	The service task checks whether a new order is pending and continues working immediately.
*SEND-ACK	The order with the specified ID was executed correctly.
ORDER-ID	Order identification
RETURN-DATA	Return parameter
*NONE	No value is returned.
*SEND-NAK	The order with the specified ID was aborted with an error.
ORDER-ID	Order identification
RETURN-KEY	Specifies the error.
RETURN-DATA	Return parameter
*NONE	No value is returned.
*SEND-RETRY	The order with the specified ID can temporarily not be executed.
ORDER-ID	Order identification
WAIT-TIME	Time in seconds
	Order should be restarted in n seconds. The wait time may extend up to the queue timeout (30 seconds).
RETURN-DATA	Return parameter
*NONE	No value is returned.
*GET-START-PARAMETER	Initialization parameters are queried after the service start or after a request via RC.

## **4 Macros**

There are also macros available for the interfaces.

## 5 Procedures

### 5.1 Reset of a Central Lock in ASTI

A deadlock situation occurs, if a task, which holds a lock on central data of ASTI, terminates abnormally, so that the lock is not released. If this lock is not released, ASTI does not accept new orders or perform existing orders.

\*3 If there is no collaboration between task termination processing and ASTI, this problem cannot be avoided. To remove the deadlock situation and to avoid the reboot of the system, the procedure **SYSPRC.ASTI.210.RESET-LOCK** is delivered.

The procedure requires the TSOS privilege.

It should be called when a deadlock situation is suspected.

The procedure displays a corresponding message, but it does not perform any modifications.

\*3 SYSPRC.ASTI.210.RESET-LOCK performs the following operations:

- Get the content of the lock area in the central lock table of ASTI
- Depending on the lock content, one of the following messages is displayed
  1. No deadlock situation: X'00000000' or X'FFFFFFFF', nothing must be done  
returned message: "ACTIVE-Q NOT LOCKED AT THE MOMENT"
  2. With any other content, the following messages are displayed:  
"ACTIVE-Q LOCKED BY TSN <TSN>"  
"RESET LOCK FOR EXISTING TSN <tsn>? (Y/N)" or  
"RESET LOCK FOR UNKNOWN TSN <tsn>? (Y/N)"
- If "Y" given, the program HELGA is called to reset the lock.

## 5.2 Example of a Service Procedure

```
/SET-PROCEDURE-OPTION
/BEGIN-PARAMETER-DECLARATION
/END-PARAMETER-DECLARATION
/*
/* DO.SERVICE
/* THIS PROCEDURE REALIZES A SERVICE
/*
/ PROCESS-ORDER ACTION=*GET-ORDER(WAIT-FOR-ORDER=*YES)
/ IF-CMD-ERROR;END-IF
/ IF (MAINCODE() NE 'SVTS016')
/ IF-BLOCK-ERROR;END-IF
/ IMPORT-VARIABLE VARIABLE-NAME=SVTVAR-ORDERID
/ IMPORT-VARIABLE VARIABLE-NAME=SVTVAR-SERVICE
/ IMPORT-VARIABLE VARIABLE-NAME=SVTVAR-DATA
/LOOP: REPEAT
/ SHOW-VARIABLE VARIABLE-NAME=SVTVAR-SERVICE
/ IF (MAINCODE() EQ 'CMD0001')
/ PROCESS-ORDER ACTION=*SEND-ACK( -
/ ORDER-ID=&(SVTVAR-ORDERID), -
/ RET-DATA = 'OKAY')
/ IF-BLOCK-ERROR;END-IF
/ ELSE-IF (MAINCODE() EQ 'SVTS011')
/ PROCESS-ORDER ACTION=*GET-START-PARAMETER
/ IF-BLOCK-ERROR;END-IF
/ WRITE-TEXT TEXT='START-PAR: SVTVAR-DATA'
/ ELSE-IF (MAINCODE() NE 'SVTS016')
/ WRITE-TEXT TEXT='ERROR MAINCODE = &(MAINCODE())'
/ EXIT-BLOCK BLOCK=LOOP
/ END-IF
/ PROCESS-ORDER ACTION=*GET-ORDER(WAIT-FOR-ORDER=*YES)
/ IF-CMD-ERROR;END-IF
/ UNTIL (MAINCODE() EQ 'SVTS016')
/ END-IF
[/EXIT-PROC]
/* must be replaced to avoid CMD2001 during task termination
/EXIT-JOB
```