

Scheduler

Contents

INTRODUCTION	1
CHAPTER 1. SCHEDULER PURPOSE AND STRUCTURE	2
Basic Definitions	2
Purpose of Scheduler	2
Examples of Most Common Scheduler Jobs	3
Structure of Scheduler	3
Using Scheduler	4
Job Sequences	5
CHAPTER 2. INSTALLING AND CONFIGURING SCHEDULER	6
Registering Scheduler Instances	6
Registering Scheduler Officers	6
Scheduler Installation	7
Scheduler Configuration	8
File scdrunora.cmd	8
File login.cfg	9
File OpenVar.ini	10
File ShadowVar.ini	11
File dirs.ini	11
File replo.cfg	12
CHAPTER 3. WORKING WITH SCHEDULER	13
Starting and Closing Scheduler	13
Configuring Scheduler in DB Manager	13
Creating and Editing Jobs	14
Creating and Editing Tasks	17
Immediate Task Execution	22
Result Logging and Job Execution Log	22
APPENDIX: PARAMETER ENCRYPTION	25
APPENDIX: DATABASE CONFIGURATION DATA	26
Table SC_SCHEDULER	26
Table SC_TASK	27
Table SC_TASK_CALL	28
Table SC_TASK_PARAM	28

Introduction



WAY4 Scheduler is a tool used to execute various jobs such as programs or scripts, following pre-determined rules on workstations with access to the database.

This document is intended for WAY4 system administrators, bank or processing centre employees responsible for the daily operation of the system. It describes operations involved in Scheduler installation and configuration.

While working with this document, it is recommended that users refer to the following reference material from OpenWay's documentation series:

- DB Manager Manual
- Menu Editor
- WAY4™ Security

The following conventions are used throughout this document:

- Field labels in screen form are typed in *italics*.
- Button labels used in screen forms are placed in square brackets, such as [Approve].
- Menu selection sequences are shown with the use of arrows like, for instance, Full → Issuing → Contracts Input & Update.
- Key combinations used while working with DB Manager are shown in angular brackets such as <Ctrl>+<F3>.
- The names of directories and/or files that vary for each local instance of the program are also displayed in angular brackets, like <OWS_HOME>.
- Warnings of possible erroneous actions are marked with  sign.
- Messages marked with  sign contain information about important features, additional facilities, or the optimal use of certain functions of the system.

Chapter 1. Scheduler Purpose and Structure

Basic Definitions

An instance is a copy of Scheduler with an ID unique within the database and its own set of configuration files. The ability of a Scheduler instance to access the database is determined by user privileges specified in its configuration file.

A job is a preconfigured set of tasks that Scheduler executes at a particular time. Tasks that belong to the same job may be executed by different Scheduler instances run on different workstations. When configuring a job, indicate the Scheduler instance that is supposed to check its starting conditions and execute the job if the check is successful.

A task is a step in the execution of a job, for example, executing a procedure or selecting a DB Manager user menu item. By default, when a task is configured, it inherits the ID of the Scheduler instance that will handle it from its job parameters. When needed, users may reconfigure the Scheduler ID. This is appropriate because some tasks can only be executed on a workstation with many processors or a higher frequency, others require high-bandwidth connection to the database, and still others require an extremely secure workstation.



In the current version, a single Scheduler instance cannot execute more than one task at a time.

Purpose of Scheduler

Scheduler executes jobs according to preconfigured rules. It authenticates its users by matching them to the data stored in special configuration files and the database.

Scheduler has the following capabilities:

- Runs processes that require large amounts of calculations on special Scheduler workstations,
- Assures additional security of various jobs by running them on workstations excluded from general use and accessible only to privileged users. Moreover, jobs and tasks may be run on behalf of other users and by other Scheduler instances.
- Executes routine or continuous operations according to preset schedules, the results being controlled through the Process Log. In critical situations, the process may include sending alert notifications by e-mail or SMS.
- Can execute various tasks of a single job on different software/hardware platforms.

- Allows jobs to interact by transferring control from one job to another together with instance-specific data sets, including cases when jobs are transferred to another Scheduler instance.
- Makes job configuration easier by using the standard DB Manager menu editor. Scheduler supports all standard user menu items.

The current version of Scheduler also supports the following:

- Use of substitute data stored in the database
- Local constants of jobs being executed
- Use of local constants as substitute values, with the exception of dates
- IBM iSeries platform

Examples of Most Common Scheduler Jobs

The following are examples of the most common Scheduler jobs:

- Periodic generation of risk management reports
- Periodic generation, encryption and e-mailing of client account statements
- Periodic import of external files, such as banking system files
- Daily procedures

Structure of Scheduler

Scheduler comprises the following three components:

1. Java Scheduler daemon installed on a Scheduler workstation. Its functions are as follows:
 - Scanning jobs as to their readiness for execution
 - Analysing user menu items, compiling command lines and executing jobs
 - Tracking job execution results and registering them in the appropriate system logs (see "Result Logging and Job Execution Log")

These tasks are executed using data contained in configuration files.
2. PL/SQL or Java (formal compile) database tables and procedures. The basic functions of this component are as follows:
 - Storing job-related information
 - Providing access to data and verifying data input from various workstations
3. DB Manager user menu items and forms. This component performs the following:
 - Monitoring Scheduler functioning
 - Managing, editing and creating jobs
 - Registering users with Scheduler access privileges

The interaction of these components is shown in Fig. 1.

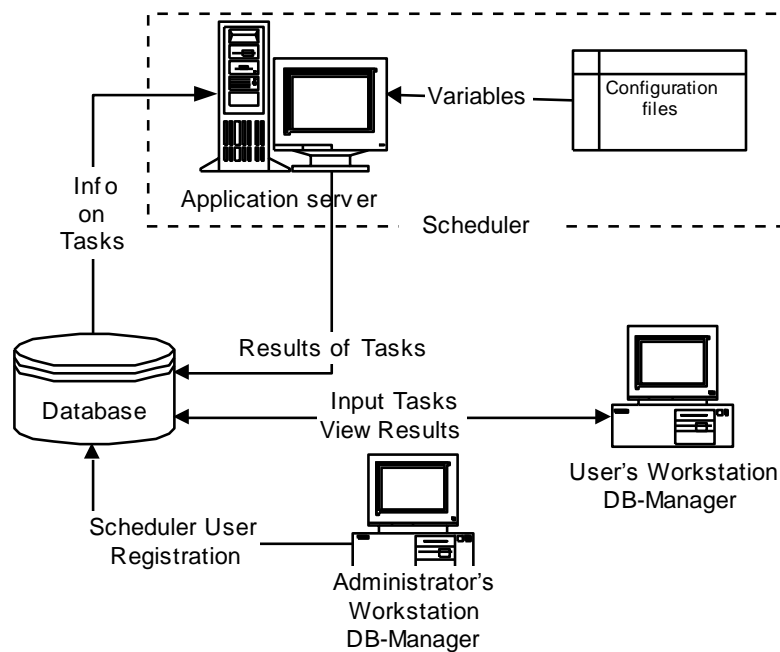


Fig. 1. Structure of Scheduler components interaction in the system

Using Scheduler

Before running Scheduler to execute a job, users must configure it using DB Manager (see "Configuring Scheduler in DB Manager"), indicate the Scheduler instance to execute the job and the time of execution. Scheduler is run on the indicated workstation (see "Starting and Closing Scheduler").

At start-up, Scheduler reads the following data from configuration files:

- Scheduler instance ID
- Names of directories containing configuration files
- Login information
- Frequency of DB polling
- Requests and programs for registering on the server and determining the current task

Having logged onto the database, Scheduler uses DBInterface interface functions to poll the DB. In the current version, this interface is implemented in the "DBIntImpl" class. Another implementation may be defined by changing the value of the following parameter in the [DIRS] section of the "dirs.ini" file:

```
DBInterfaceClassName= com.openwaygroup.ScdClient.NewClient.DBIntImpl
```

This class uses the "SCD" class from the "com.openwaygroup.timetabler" package. When the next task is executed (see "Creating and Editing Jobs"), the package function returns data needed to compile the command line.

The command line is compiled and interpreted by the translator implemented in the "com.openwaygroup.subitemparser" Java package. If needed, Scheduler

analyses DB Manager user menu item data so that it is transformed into a set of either operating system commands or operator calls and SQL commands.

The resulting lines are processed by context-dependent substitution software, which first substitutes the current variable values with those from "ShadowVar.ini" and "OpenVar.ini" files (see "File OpenVar.ini" and "File ShadowVar.ini"), then with values from the LOCAL_CONSTANTS database table, and finally, with values from the SC_TASK_PARAM table. These variable values used by pipes or processes activated by DB Manager menu items may also be entered in the configuration window for the DB Manager session, invoked to the screen through the "Database => Configure" system menu item. Scheduler executes the resulting command line or PL/SQL script and stores the completion code in the process log. Based on this code, Scheduler selects the next task, and if it is configured, generates data required for its execution.

Scheduler checks if conditions are correct for the execution of a job at the time intervals preset in configuration files (see "File login.cfg").

Scheduler analyses data contained in the database and decides what job to execute and when it should be executed (see "Job Sequences"). Any JDBC-compatible database may be used. The current version of Scheduler has been tested with Oracle DB and DB2.

A job may include individual tasks whose precedence must be defined. The sequence of tasks may be conditionally detailed depending on the results of the execution of the last executed task.

A command line may include identifiers whose values may be substituted by Scheduler as the execution of a job progresses.

✓ Jobs and individual tasks may be executed by different Scheduler instances run on different workstations. This includes jobs that start DB Manager opening a user interactive window.

Job Sequences

The next job execution time is specified in the job's configuration (see "Creating and Editing Jobs"). When Scheduler is run, it selects an arbitrary job among those with an execution time earlier or equal to the local machine system date. Scheduler executes the first task of the selected job, then analyses all jobs (including the one being executed) and again selects an arbitrary job among those whose execution time has come. If the job being executed is selected, its second task is executed. If no task in the selected job has been executed, its first job is executed. When a job has been completed, its execution time changes, and Scheduler will not be able to select it for execution before the next execution time.

Chapter 2. Installing and Configuring Scheduler

Registering Scheduler Instances

Scheduler instances are registered in the "OpenWay Stations" grid form. To open it, select the "Full → DB Administrator Utilities → Users & Grants → OpenWay Stations" menu item (see Fig. 2).

Station Name	Network Address	CS Code
TEST_INSTANCE	Test_Instance	111

Fig. 2. Making an entry for a Scheduler instance run on a workstation

This form contains the following fields:

- *CS Code* – unique 3-digit identifier (code) of the Scheduler instance
- *NetWork Address* – name of the Scheduler instance used by the system (any abstract word may be used for this name). It should be noted that this field is case-sensitive. The field may not contain more than 32 characters.



This name must be also shown in the corresponding parameter contained in the "login.cfg" configuration file (see "File login.cfg").

- *Station Name* – name of the Scheduler instance assigned to the system user (it may be the same as the value in the *NetWork Address* field)

Registering Scheduler Officers

Users who run Scheduler under their own login names (see "File login.cfg") must have certain DB access privileges.

To create WAY4 users with the required privileges, use the "Scheduler Engine Group" user group (see the "WAY4 System Users" section in the WAY4™ Security Administrator Manual).

If this user group has not been created yet, create it through the "DB Manager Users and Groups" form (Full → DB Administrator Utilities → Users & Grants → User Groups and Users – Edit). In the *Menu Tree* field of the form, specify the "Scheduler → Scheduler Engine Rights" user menu path.

When the user group has been created, users with the privilege of running Scheduler may be registered in it.

To register Scheduler officers, i.e. users of a particular Scheduler instance, use the "Scheduler Officers" grid form. It is opened through the "Scheduler → Scheduler Officers" menu item (see Fig. 3).

Officer	Instance
Test_User	TEST_INSTANCE

Fig. 3. Creating an officer for the Scheduler instance run on a particular workstation

In the *Officer* field of the form, specify a user name that will be used to register results of executing tasks executed by the Scheduler instance specified in the *Instance* field.

Scheduler Installation

The Scheduler installation set includes the following:

- Java class set for the Scheduler daemon, compatible with any platform that supports Java VM 1.4 and later versions
- Java VM version 1.4 for MS Windows
- Set of DB Manager forms used to edit jobs
- Test sets of configuration files with initial settings to work with Oracle DB and DB2 databases
- JDBC drivers for Oracle DB and DB2 to work on the IBM iSeries platform

The Scheduler installation procedure is as follows:

- Create a "<Sc_Work>" work directory for every Scheduler instance.
- If working with Oracle DB, edit the "<Sc_Work>\scdrunora.cmd" file through a text editor (see "File scdrunora.cmd"). If working with DB2, edit the "<Sc_Work>\scdrundb2" file in the same way.
- If working with Oracle DB:
 - Create a "\Config.ora" configuration subdirectory in the "<Sc_Work>" work directory.
 - Copy the "scdrunora.cmd" file from the "<OW_Home>\Client\Scd" directory into the "<Sc_Work>" work directory.
 - Copy the contents of the "<OW_Home>\Client\Scd\Config.ora" directory into the "<Sc_Work>\Config.ora" configuration subdirectory.
- If working with DB2:
 - Create a "\Config.db2" configuration subdirectory in the "<Sc_Work>" directory.
 - Copy the "scdrundb2.cmd" file from the "<OW_Home>\Client\Scd" directory into the "<Sc_Work>" directory.

- Copy the contents of the "<OW_Home>\Client\Scd\Config.db2" directory into the "<Sc_Work>\Config.db2" configuration subdirectory.
- Create a "\log" subdirectory in the "<Sc_Work>" directory. It will contain a configuration file containing the parameters for registering the results of job execution.

When Scheduler is first run, the "deflog.properties" file is automatically created in this directory. This file contains the default values of the parameters defining how job execution results are logged. These values may be changed when the file is edited.



If Scheduler needs to be reinstalled and the "deflog.properties" file has been modified, it is recommended that users save the file in some other directory, and when Scheduler is started after the reinstallation, copy the custom configurations to the automatically created "deflog.properties" file.

- Using a text editor, edit the "<Sc_Work>\<configuration subdirectory name>\login.cfg" file (see "File login.cfg").
- Using a text editor, edit the "<Sc_Work>\<configuration subdirectory name>\replo.cfg" file (see "File OpenVar.ini").
- Using a text editor, edit the "<Sc_Work>\<configuration subdirectory name>\OpenVars.ini" and "<Sc_Work>\<configuration subdirectory name>\ShadowVars.ini" files (see "File OpenVar.ini" and "File ShadowVar.ini").

Scheduler Configuration

To configure Scheduler, use the configuration files located in "<Sc_Work>\<configuration subdirectory name>".

File scdrunora.cmd

The "scdrunora.cmd" file is a command file located in Scheduler's home directory. It is used to start the application. The file contains the definitions of environmental variables such as the paths to Scheduler's home directory and Java classes, the database type, or the owner of the database scheme.

The following is an example of the "scdrunora.cmd" file:

```
set OWS_HOME=L:\OW_HOME\  
set BWX_WORK=L:\OW_WORK\  
set BWX_LOG=L:\LOG\  
set WAY4SCDHOME=%OWS_HOME%\client\scd\  
set WAY4SCWORK=%BWX_WORK%  
set BWX_DBMS=Oracle  
set USE_SYNONYMS=NO  
set OWS_OWNER=OWS  
set BWX_HOME=%OWS_HOME%
```

```
call %OWS_HOME%client\jre\set_recent_jre.bat %OWS_HOME%client\jre
call %OWS_HOME%db\dbms\oracle\set_jdbc.bat
cd %WAY4SCWORK%
%JRE_PATH%\bin\java -Duser.region=US -Duser.language=en -cp
%OWS_HOME%client\shared\javalib\launcher.jar
com.openwaygroup.launcher.MainLauncher
%OWS_HOME%client\shared\javalib;%WAY4SCDHOME%;%OWS_HOME%\client\dbm
com.openwaygroup.ScdClient.ScdClient %WAY4SCWORK%\config.ora\dirs.ini
1>%WAY4SCWORK%\log.txt 2>%WAY4SCWORK%\err.txt
```

The Scheduler command line may include the names of files for job progress information. In this example, they are "log.txt", where complete information on job progress is logged, and "err.txt", the destination of the last execution error message.

File login.cfg

The "login.cfg" file contains seven lines describing the parameters of access to database management data:

- Name of the database JDBC driver (either Oracle or DB2)
 - Database connection parameters. The format is as follows: "jdbc:oracle:thin:@<server IP address>:<port number>:<SID>". The necessary values may be copied from the "db.ini" file, the [soft] section, line JDBC_CONNECT).
 - Name of the user who has access to Scheduler control data. Scheduler is run under this user.
 - User password
- ✓ User passwords may be encrypted (see "Parameter Encryption"). To specify encrypted passwords in the "login.cfg" file, use the following format:

```
~<encrypted password>
```

- Scanning interval in milliseconds. This value does not need to be changed during installation.
- Name of the Scheduler instance. This is an ID unique for the database, the same as the ID found in the *NetWork Address* field of the "Open Way Stations" table (see "Registering Scheduler Instances").
- Encryption key of the password transferred between DB Manager and the database



The password encryption key specified in the "login.cfg" file must be the same as the value of the "PWD_ENCRYPTION" parameter in the "db.ini" file (see section "Appendix. Data Access Restriction through User Password Encryption in the WAY4™ Security Administrator Manual).

An example of the "login.cfg" file with an encrypted password:

```
oracle.jdbc.driver.OracleDriver
jdbc:oracle:thin:@172.16.1.44:1521:WORK
```

```
Test_User
~AA69DE264E71552BBAE5927645835896F479843D70EAED56CED3
1000
Test_Instance
123456
```

File OpenVar.ini

The "OpenVar.ini" file contains the definitions of open (generally accessible) context-dependent substitutions. The file consists of lines in the "<name>=<value>" format.

Examples of "OpenVar.ini" file fragments:

The substitute path to the "cmd.exe" system file:

```
C:\WINNT\SYSTEM32\CMD.EXE=C:\WINDOWS\SYSTEM32\CMD.EXE
```

The substitute path to the "rwrunc60.exe" or "rwrunc.exe" file:

```
@REPRUN@=C:\ORACLE\920\BIN\RWRUN60.EXE
```

The substitute connection string used by pipes to connect to the ODBC source in order to access the database:

```
@CONNECT_PIPE_STR@="DSN=ORACLE920;UID=@USER@;PWD=@PASSWORD@"
```

The substitute database name used by pipes:

```
@DATABASE_NAME@="NEW DB2"
```

The substitute paths to the WAY4 <OWS_HOME> and <OWS_WORK> system directories:

```
@OWS_HOME@=L:\OW_HOME\
@OWS_WORK@=L:\OW_WORK\
```

The substitute name of and path to the temporary pipe parameters directory:

```
@PIPE_PARMS_DIR@=c:\temp\parms\
```

The substitute path to DB Manager:

```
@OWEXEPATH@=L:\OW_HOME\client\dbm\
```

The substitute path to the pipes directory:

```
@PIPE_PATH@=L:\OW_WORK\Client\Shared\Pipes\
```

The substitute procedure loading local variables for every connection to the database:

```
@INIPROC_P@=declare p number;begin
std.set_parms('@CONN_ID@','@AMND_OFFICER@');select last_process into p
from v_local_constants; sy_process.attach_process(p,null); end;
@INIPROC@=begin std.set_parms('@CONN_ID@','@AMND_OFFICER@'); end;
```

The substitution for assuring compatibility with DB2:

```
{ts @LOCAL_DATE@}=glob_ldate()
{ts @LOCAL_DATE@}=glob_ldate()
SELECT ID FROM STANDING_ORDER=SELECT ID,PRIORITY FROM STANDING_ORDER
```

The substitute path to the reports directory:

```
@REPORTSPATH@=L:\OW_WORK\Client\Shared\Reports\
```

File ShadowVar.ini

The "ShadowVar.ini" file contains the definitions of closed (restricted access) context-dependent substitutions. As required, the file may be stored on a network or a local disk to restrict data access. This file consists of lines in the "<name>=<value>" format.

The following is an example of the "ShadowVar.ini" file:

```
@USER@=owsadm
@PASSWORD@=OWS
@TNS@=WORK
@JDBCDRIVER@=oracle.jdbc.driver.OracleDriver
@JDBCCONN@=jdbc:oracle:thin:@ 172.16.1.44:1521:WORK
@MAIN_OFFICER@=OWS
```



It should be kept in mind that the "@MAIN_OFFICER@" variable had an incorrect (mistyped) name, "@MAIN_OFFICIER@", in the previous Scheduler versions. This variable name is correct in versions 3.20 and later.

Parameters used in the "ShadowVar.ini" file may be encrypted (see "Parameter Encryption"). To specify encrypted parameter values, use the following format:

```
<~><encrypted parameter value>
```

Example of using an encrypted parameter in the "ShadowVar.ini" file:

```
@PASSWORD@=<~>AA69DE264E71552BBAE5927645835896F479843D70EAED56CED3
```

File dirs.ini

The "dirs.ini" file consists of two sections. Its [Main] section contains a pointer to the Scheduler daemon version.

```
[Main]
SCD_CLASS = com.openwaygroup.ScdClient.NewClient.NewClient
```

The [Dirs] section contains non-standard paths to configuration files, for instance:

```
[Dirs]
Login = .\<configuration directory name>\login.cfg
Replo = .\<configuration directory name>\replo.cfg
ShadowVar = .\<configuration directory name>\ShadowVar.ini
OpenVar = .\<configuration directory name>\OpenVar.ini
DBInterfaceClassName= com.openwaygroup.ScdClient.NewClient.DBIntImpl
```

The DBInterfaceClassName parameter is used to indicate the interface for accessing the database.

File replo.cfg

The "replo.cfg" file contains partial program settings, such as the name and password of the recipient of reports. This information must be protected, that is, stored on a secured workstation. The contents of this file are passed to the "DBInterface" database access interface as the ":login_str" variable.

The file contains a string consisting of two parts. Each part is placed within quotation marks and separated from each other with a space character.

- The first part is an ODBC connection string used to access the database.
- The second part is the database alias, the same as the name found in the "DB.ini" file and the MS Windows registry (HKEY_LOCAL_MACHINE\SOFTWARE\OpenWay Group\DB Manager\Databases\).

Here is an example of the "replo.cfg" file:

```
"DSN=Oracle8 Tables;UID=lasg;PWD=lasgl;SRVR=work" "WORK"
```

✓ When creating the "replo.cfg" file, value substitution variables found in the "ShadowVar.ini" file are used for convenience (see "File ShadowVar.ini").

The following is an example of the "replo.cfg" file containing context-dependent value substitution variables:

```
@CONNECT_PIPE_STR@ @DATABASE_NAME@
```

Chapter 3. Working with Scheduler

Starting and Closing Scheduler

Scheduler is started by running one of the following command files:

- "<Sc_Work>\scdrunora.cmd" – when working with Oracle DB
- "<Sc_Work>\scdrundb2.cmd" – when working with DB2 (see the "Scheduler Installation" section).

If necessary, Scheduler may be started through the MS Windows Startup folder. At start-up, Scheduler connects to the database using data from the "<Sc_Work>\<configuration directory name>\login.cfg" file.

Normally, Scheduler is shut down through the DB Manager active processes window invoked through the "Full → Process Log → Active Process" menu item (see the "DB Manager Processes" section in the DB Manager Administrator Manual). To stop Scheduler, select the process indicating the name of the appropriate Scheduler instance in the *Parameters* field of the window and click the [Stop] button.

For forced Scheduler shutdown, press the <Ctrl>+<C> key combination while in the Scheduler window. After a forced shutdown of a Scheduler instance, the corresponding process must be terminated by clicking the [Stop] button in the active processes window (Full → Process Log → Active Process). Having terminated the process, click the [Clear] button and, in the "Clear for <process name>" window that opens, again click the [Clear] button.

Configuring Scheduler in DB Manager

Scheduler is configured in DB Manager through the "Scheduler" menu group (see Fig. 4).

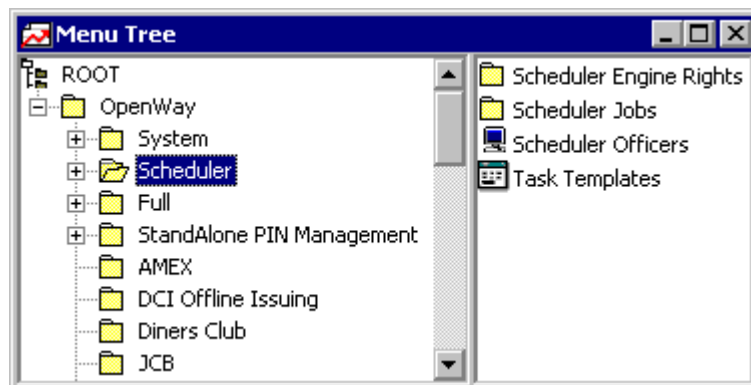


Fig. 4. User menu group for configuring Scheduler

Creating and Editing Jobs

New jobs are created or existing ones edited in the "Scheduler" form (see Fig. 5) invoked by selecting the "Scheduler → Scheduler Jobs → Scheduler" menu item.

To create a new job, click the [Ins] button and enter the necessary data.

The screenshot shows a window titled "Scheduler" with a toolbar at the top right containing navigation icons and "1 of 1". The form fields are as follows:

- Job Name:** Test_Job
- Week TimeTable:** Daily
- Execute on Instance:** TEST_INSTANCE
- Time From / To:** 12:00 to 15:00
- Job Status:** Waiting
- Time Interval:** 0
- Date From/To:** 12/11/2007 to 19/11/2007
- # Of Calls:** 5
- Estimated Start:** 14/11/07 12:00:00
- First Task:** Test_Task_1
- Today # Of Calls:** 0
- Current Task:** (empty)
- Is Ready:** Ready

At the bottom of the form are buttons: Ins, Del, Query, Approve, Task Calls, Task, and Brief.

Fig. 5. Form for editing jobs

The following fields are in this form:

- *Job Name* is the name of a job.
- *Execute On Instance* is the Scheduler instance that will execute the indicated job will be executed. This value is selected from the list of instances registered in the system (see "Registering Scheduler Instances").
- *Job Status* is the status of the execution of a job. The field can take on one of the following values:
 - "Waiting" means that the job is ready to be executed.
 - "Suspended" means that the job is being executed.
 - "Posted" means that the job has been executed successfully.
 - "Decline" means that the job has terminated due to an error.
 - "InActive" or "Closed" means that the execution of the job has been terminated by the user.
- *Date From/To* is a group of two fields. The earliest date when the job may be executed is specified in the first field. The latest date (inclusive) when the job may be executed is specified in the second one. The fields are mandatory.
 - ✓ For a job to be executed continuously over an indefinite period of time, a long-past date such as 01.01.1900 should be entered in the first field of the *Date From/To* field group and a very remote future date such as 01.01.2100 should be entered in the second field.

- *Estimated Start* is the time when the job will be executed for the next time. The job is next executed if the value in the *Estimated Start*, while preceding or being the same as the local system time, lies between the *Date From/To* and *Time From/To* field group values.
 - *Week Time Table* indicates how often the job is executed. The field can take on one of the following values:
 - "Daily" means that the job is executed daily.
 - "Working Day" means that the job is executed on any day indicated as such in the business calendar (see the "Business Calendar" section of the WAY4™ Dictionaries Administrator Manual).
 - "Non-working Day" means that the job is executed on any day indicated as such in the business calendar.
 - "Monday" means that the job is executed on Mondays.
 - "Tuesday" means that the job is executed on Tuesdays.
 - "Wednesday" means that the job is executed on Wednesdays.
 - "Thursday" means that the job is executed on Thursdays.
 - "Friday" means that the job is executed on Fridays.
 - "Saturday" means that the job is executed on Saturdays.
 - "Sunday" means that the job is executed on Sundays.
 - *Time From/To* is a group of two fields. The earliest time in the day when the job may be executed is specified in the first field. The latest time in the day (inclusive) when the job may be executed is specified in the second field.
 - If the first field of the *Time From/To* group is not filled in or "00:00" is entered in it, the value of the *Estimated Start* field will be calculated by adding the time interval specified in the *Time Interval* field to the time the job was last started.
 - If the value entered in the first field of the *Time From/To* group is anything other than "00:00", the value of the *Estimated Start* field will be calculated according to the following formula:

$$Time_From + Current_Call \cdot Time_Interval,$$
 where *Time From* is the value specified in the first field of the *Time From/To* group, and *Current Call* and *Time Interval* are the values of the *Today # of Calls* and *Time Interval* fields respectively.
 - If the calculated value of the *Estimated Start* field exceeds the value specified in the second field of the *Time From/To* group, the job will be executed on the day selected according to the *Week Time Table* field setup.
- ✓ Note that the first field of the *Time From/To* group may contain a value exceeding the second field value. In this case, task execution may be continued on the following day. In particular, if a task is scheduled for a

certain day of the week, it may continue being executed on the next day of the week; if a task is scheduled for a non-business day, it may continue being executed on the next day after the non-business day.

- *Time Interval* is the time interval (in minutes) between job starts, if the job must be executed more than once.
- *# Of Calls* is the required number of job starts per day.
- *First Task* is the first task of the job.
- *Current Task* is the task being currently executed. If the job is not being executed, the field is left blank.
- *Today # of Calls* is the number of times the job has been started by this moment during the day.
- *Is Ready* indicates whether the job is ready to be executed. The field takes on the "Ready" value if the job is ready to be executed, otherwise, "Not Ready".

The [Task Calls] button in the "Scheduler" form (see Fig. 5) opens the "Task Calls for <name of job>" grid form, which contains job execution results (see "Result Logging and Job Execution Log").

The [Task] button opens the "Task for <name of job>" form. It is used to edit tasks (see "Creating and Editing Tasks").

The [Brief] button opens the "Brief for <name of job>" grid form, which contains a list of all tasks making up a given job (see "Creating and Editing Tasks"). This form can also be used to edit tasks.

After editing a job or a task, users must click on the [Approve] button to confirm changes.

If no errors are detected in the data, the approval of the changes in job parameters is confirmed by a message to this effect (see Fig. 6):



Fig. 6. Message confirming the changes

If errors are detected when approving the changes, the following error message appears (see Fig. 7):

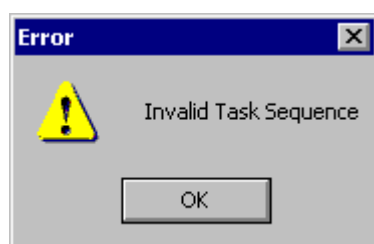


Fig. 7. Example of an error message at the attempt to approve changes made in a job

Creating and Editing Tasks

To access the list of tasks included in the current job, click the [Brief] button in the "Scheduler" form (see Fig. 5). This will invoke the "Brief for <name of job>" grid form (see Fig. 8).

Task Number	Instance	Task Name	Task Type	Menu Item	Next Task	Error Task	Is Ready
1	TEST_INSTANCE	Test_Task_1	Menu Item	OWS.Accept Documents	Test_Task_2		Ready
2	TEST_INSTANCE	Test_Task_2	Menu Item	OWS.VISA.VISA BASE II Outward Processing			Ready

Fig. 8. Grid form listing tasks

The form contains fields for entering individual task parameters. To access full task information, click on one of the following buttons:

- [Task] – to configure a task
- [Task params] – to configure additional task parameters

The [Task] button in the "Brief for <name of job>" form invokes the "Task for <name of job>" form (see Fig. 9). This button works the same way as the [Task] button in the "Scheduler" form.

Fig. 9. Form for configuring tasks

This form contains the following fields:

- *Task #* is the number of the task in the job.
- *Instance* is the name of the Scheduler instance that will execute the task.
- *Task Name* is the name of the task.
- *Menu Item* is a field showing a drop-down list containing DB Manager user menu items. This field is used only if the task executes a menu item.



Note that Scheduler correctly executes menu items only if these rules are followed:

- If the type of a menu subitem is "Pipe", specify value "Silent" in the "Dialogue Type" field in the window for entering subitem parameters (see section "Pipe Type" in the Menu Editor Administrator Manual).
- Use a "Special" subitem of the "Execute SQL" type instead of a menu subitem of the "Form" type with the "Name" parameter set to

"Date From - To". In the "SQL" field of the window for entering subitem parameters, specify an expression setting fields "DATE_FROM" and "DATE_TO" of the "LOCAL_CONSTANTS" table to the necessary values (see section "Special (PBD) Type" in the Menu Editor Administrator Manual).

- *Job to call* is a drop-down list of registered Scheduler jobs. This field is only filled in if the task executes a job.
- *Task Type* is the type of the task. The field can take on one of the following values:
 - "EXE File" runs an executable or a batch file and waits for the completion of the run (see "Starting Executable Files").
 - "EXE Process" runs an executable or a batch file without waiting for completion. This value is reserved for future use.
 - "SQL Block" runs a PL/SQL script stored in an external file and waits for the completion of the run (see "Starting PL/SQL Scripts").
 - "SQL Process" runs a PL/SQL script stored in an external file without waiting for its completion. This value is reserved for future use.
 - "Menu Item" executes a DB Manager user menu item. When this type of task is selected, the menu item to be executed must be specified in the *Menu Item* field.



It should be kept in mind that when a menu item is executed by Scheduler, all subitems of the "Special (Execute SQL)" type are executed with the following parameters: File Type = "Tab-Separated", With Headers = "Yes". This is regardless of the values of these parameters specified in the menu subitem.

- "Set Parameter" assigns a parameter to a task (see Assigning Parameters to Tasks).
- "Call Scd Job" executes another Scheduler job. If this type is selected, specify a Scheduler task name in the *Job to call* field.
- "Set Flag", "Wait for Flag", "Check Flag" are values used to synchronise tasks execution (see Synchronising Tasks Execution).
- *Next Task* is the name of the task to be executed next if the current task is completed successfully.
- *Error Task* is the name of the task to be executed next if the current task fails.
- *Task Strings* is a group of 6 fields that are portions of the command line used to:
 - Start an executable file, if the task type is "EXE File"; for more details, see "Starting Executable Files".
 - Start a PL/SQL script, if the task type is "SQL Block"; for more details, see "Starting PL/SQL Scripts".

- Start a menu item for parallel run, if the task type is "Menu Item"; for more details, see "Executing Menu Items in Parallel").
- Assign a parameter to a task, if the task type is "Set Parameter"; for more details, see "Assigning Parameters to Tasks".
- Synchronise tasks execution, if the task type is "Set Flag", "Wait for Flag", or "Check Flag"; for more details, see "Synchronising Tasks Execution".
- Start the task with the use of the "%WAITFOR%" operator; for more details, see "Using %WAITFOR% Operator with Files" and "Using %WAITFOR% Operator with PL/SQL Scripts".

The [Run] button in the "Task for <name of job>" form (see Fig. 9) is used to execute the selected task and all the following tasks using the current values of the local variables for the workstation from which the task is started.

The [Task Calls] button opens the "Task Calls for <name of job>" grid form, which contains the job execution results (see "Result Logging and Job Execution Log").

The [Params] button opens the "Task params for <name of task>" grid form (see Fig. 10). The form is intended to configure additional task parameters. This button works the same way as the [Task params] button in the "Brief for <name of job>" form.

Officer	Instance	Parameter Section	Parameter Code	Value	Comment
Test_User	TEST_INSTANCE				

Fig. 10. Form for configuring additional task parameters

This form contains the following fields:

- *Officer* is the name of the user for whom the configured parameter is used. In different situations, this field may have different meanings:
 - When Scheduler is started, this is the name of the user who starts Scheduler.
 - When Scheduler executes a task automatically, this is the name of the user specified in the configuration file (see "File login.cfg").
 - When a task is started through DB Manager (see "Immediate Task Execution"), this is the name of the user working with DB Manager.

If the defined parameter should be accessible for all users, this field should be left blank.

- *Instance* is the name of the Scheduler instance for which the given parameter is defined.
- *Parameter Section* is the name of a parameter group. This field allows parameters to be structured and divided into groups.

- *Parameter Code* is the parameter code unique for the task.
- *Value* is the value of the parameter.
- *Comment* is the field for entering comments.

After a task has been edited, the changes must be confirmed by clicking the [Approve] button in the form used to edit jobs (see "Creating and Editing Tasks").

Starting Executable Files

If the "EXE File" value (starting an executable file) is selected in the *Task Type* field of the "Task for <name of job>" form (see Fig. 9), the *Task Strings* field group of the abovementioned form must include the name of the file to be started and starting parameters, like "cmd.exe /c dir c:*. * >c:\dir.txt". The command line will be generated as the concatenation of all string information from the *Task Strings* field group.

Starting PL/SQL Scripts

If the "SQL Block" value (starting a PL/SQL script) is selected in the *Task Type* field of the "Task for <name of job>" form (see Fig. 9), the body of the PL/SQL script must be included in the *Task Strings*, for instance:

Begin
Std.PROCESS_MESSAGE('E', 'Error while executing task');
:p := 0;
End;

The ":p" operator is used to pass a value returned by the expression contained in the right-hand part of the assignment operator to the PL/SQL script invocation point. If the returned value equals "0", it means that the script has been executed successfully and the next task may be started. Any other value means switching to the error task.



It should be noted that all strings of a PL/SQL script are consecutively concatenated and sent for execution as a single string. Therefore, a space character must be added at the beginning of every string in the body of a script.

Using %WAITFOR% Operator with Files

If the "EXE File" value is selected in the *Task Type* field of the "Task for <name of job>" form (see Fig. 9), the use of the "%WAITFOR%" operator means that Scheduler will expect the appearance of the indicated file in the specified directory. If the expected file is found in the specified directory, Scheduler switches to the next task. If the waiting time is exceeded, the error task will be executed.

To implement this scheme, the first field in the *Task String* field group must contain the "%WAITFOR%" operator. The second field then shows the path to the directory where the appearance of the file is expected. The path must be absolute and end with a "\" symbol. The third field contains the regular expression (mask) of the file name and the fourth – the maximum waiting time in minutes.

Using %WAITFOR% Operator with PL/SQL Scripts

If the "SQL Block" value is selected in the *Task Type* field of the "Task for <name of job>" form (see Fig. 9), the use of the "%WAITFOR%" operator means that Scheduler will execute the indicated PL/SQL script within the preset time interval. If the script is executed successfully, Scheduler switches to the next task, if not, to the error task.

To implement this scheme, the first field in the *Task String* field group must contain the "%WAITFOR%" operator. The second field contains the maximum script execution time in minutes. The next four fields contain the PL/SQL script in question. After its execution, the script returns a numeric value. The query contained in the script is executed within the time interval indicated in the second field, reiterating after the period set in the "login.cfg" configuration file (see "File login.cfg"). As long as "1" is returned, the execution is repeated. If "0" is returned, Scheduler switches to the next task. In case of any other value, Scheduler switches to the error task. If the value specified in the second field is exceeded, Scheduler also switches to the error task.

Executing Menu Items in Parallel

If the "Menu Item" value is selected in the *Task Type* field of the "Task for <name of job>" form (see Fig. 9) and either "DB Procedure Cycle" or "SQL Cycle" is specified in the *Menu Item* field, the menu items may be executed as parallel Oracle processes. For this, the first field in the *Task String* field group must indicate the number of Oracle processes to be performed in parallel.



Note that if a menu item is already running in parallel, the number of Oracle processes cannot be changed.

Assigning Parameters to Tasks

If the "Set Parameter" value is selected in the *Task Type* field of the "Task for <name of job>" form (see Fig. 9), it is necessary to specify a parameter name in the first field of the *Task Strings* group and its value in the second field. For the next executed task, this parameter will be added to the "SC_TASK_PARAM" table and automatically matched with the parameter group "JOB VARS" (see the description of the *Parameter Section* field of the "Task params for <name of task>" form in section "Creating and Editing Tasks").

Synchronising Tasks Execution

To synchronise tasks execution, use the "Set Flag", "Wait for Flag" and "Check Flag" task types.

The "Set Flag" tasks type sets a special parameter (flag) to a specified value. For this, specify a parameter name in the first field of the *Task Strings* group in the "Task for <name of job>" form (see Fig. 9) and its value in the second field. For the next executed task, this parameter (flag) will be added to the "SC_TASK_PARAM" table and automatically matched with the parameter group "FLAG" (see the description of the *Parameter Section* field of the "Task params for <name of task>" form in section "Creating and Editing Tasks").

When executing a task of the "Wait for Flag" type, Scheduler will switch to wait mode and remain in it till the value of the flag whose name is specified in the first field of the *Task Strings* group of the "Task for <name of job>" form is equal to the value specified in the second field of the group. Then, Scheduler will execute the task specified in the *Next Task* field.

When executing a task of the "Check Flag" type, Scheduler will check the flag whose name is specified in the first field of the *Task Strings* group of the "Task for <name of job>" form. If the value of the flag is equal to the value specified in the second field of the group, Scheduler will execute the task specified in the *Next Task* field. Otherwise, Scheduler will execute the task specified in the *Error Task* field.


Immediate Task Execution

To execute a task immediately, use the "Scheduler Task Executor" form (see Fig. 11) invoked by selecting the "Scheduler → Scheduler Jobs → Scheduler Task Executor" user menu item.

Job Name	Status	Is Ready	Estimated Start	Today Calls	Current Task
Test_Job	Waiting	Ready	14/11/07 12:00:00	0	

Fig. 11. Form for immediate task execution

In this form, click on the [Execute] button. As a result, the selected task will be executed at once, without waiting for the scheduled starting time.

 If a value exceeding "1" is indicated in the *# Of Calls* field for a task, clicking the [Execute] button will start the task without waiting for its scheduled starting time. If the task has already been executed the number of times indicated in the *# Of Calls* field during the course of the day, clicking the [Execute] button will execute the task again for the number of times indicated in the field. That is, if the value in the *# Of Calls* field is five and all five calls have been made for the day, clicking the [Execute] button will result in the execution of the task five more times.

Result Logging and Job Execution Log

Job execution results are logged in the system Process Log accessed through the "Full → Process Log → Process Log" menu item and the "Task Calls for <name of job>" table (see Fig. 12). The latter is accessed through the [Task Calls] button in the forms for configuring jobs and tasks (see Fig. 5 and Fig. 9 respectively).

Started	Finished	Status	Job	Task
15/01/2003 15:15:30	15/01/2003 15:15:37	Decline	test2	test task
15/01/2003 15:12:04	15/01/2003 15:14:30	Decline	test2	test task
15/01/2003 12:05:32	15/01/2003 12:05:33	Posted	test2	test task
15/01/2003 11:55:11	15/01/2003 11:55:12	Posted	test2	test task
15/01/2003 09:17:16	15/01/2003 09:17:17	Posted	test2	test task
14/01/2003 21:10:32	14/01/2003 21:10:33	Posted	test2	test task
14/01/2003 20:52:49	00/00/0000 00:00:00	Suspended	test2	test task

Fig. 12. Form containing job execution results

This form contains the following fields:

- *Started* – the actual time when the task was started
- *Finished* – the actual time when the task execution was finished
- *Status* – the task execution status. The field can take on one of the following values:
 - "Suspended" means that the task is being executed.
 - "Posted" means that the task has been executed successfully.
 - "Decline" means that the task has terminated due to an error.
- *Job* – the name of the job to which the task belongs
- *Task* – the name of the task

Execution results are logged according to the configurations set up in the "deflog.properties" file (see "Scheduler Installation"). Here is an example of the "deflog.properties" file:

```
com.openwaygroup.ScdClient.log.DefaultVerbose=YNN
com.openwaygroup.ScdClient.log.DBDedicat=N
com.openwaygroup.ScdClient.log.DBReq=N
com.openwaygroup.ScdClient.log.FileDest=.\log\SCD.log
com.openwaygroup.ScdClient.log.LogImplClass=com.openwaygroup.ScdClient.log.LogImpl
com.openwaygroup.ScdClient.log.LogFileVolumeLimit=10000
com.openwaygroup.ScdClient.log.LogFileCountLimit=1000
com.openwaygroup.ScdClient.log.LogFileCounter=.\log\lfcoun.properties
```

The possible values of the parameters contained in the "deflog.properties" file are shown in the table below Table 1.

Table 1. Parameter values contained in the "deflog.properties" file

Parameter and Default Value	Possible Values
-----------------------------	-----------------

Parameter and Default Value	Possible Values
DefaultVerbose=YYNN	<p>Determines whether or not information is logged. The value is a four-character code, consisting of these values:</p> <p>1st position (left to right) – fatal error messages</p> <p>2nd position – all error messages</p> <p>3rd position – information messages</p> <p>4th position – debugging messages</p> <p>The following symbols may be used in these code positions:</p> <p>"Y" – all appropriate messages are logged in the file defined by the "FileDest" parameter</p> <p>"N" – no messages are logged</p>
DBDedicat=N	The value of this parameter must not be edited.
DBReq=N	The value of this parameter must not be edited.
FileDest=.\log\SCD.log	<p>The path to and name of the log file. The value of this parameter may be edited by the user. Log files make up an archive whose name is as follows: <file name defined by the "FileDest" parameter>.<sequential number defined by the LogFileVolumeLimit and LogFileCountLimit parameters>.</p>
LogImplClass=com.openwaygroup.ScdClient.log.LogImpl	System parameter that defines class implementation.
LogFileVolumeLimit=10000	The maximum log file size in bytes. The actual size of a file may exceed this value by the size of the last record.
LogFileCountLimit=1000	The maximum number of log files in an archive file. When this value is exceeded, the system will periodically overwrite the existing archive files.
LogFileCounter=.\log\lfcoun.properties	The name of the file containing the sequential number of the last archive file. This file must not be edited.

Appendix: Parameter Encryption

The password specified in the "login.cfg" file and the parameters specified in the "ShadowVar.ini" file may be encrypted using the "nscipher" utility, which is included in NetServer's distribution kit. To encrypt data, enter the following command in the command line:

```
nscipher ows_application > <name of resulting file>
```

As a result, a prompt to enter a parameter value to be encrypted will be displayed:

```
Please enter plain data :
```

After the data is entered, a prompt to reenter the data will be displayed:

```
Please repeat plain data:
```

After the data is reentered, the encrypted parameter value will be saved to the file with the specified name.

Appendix: Database Configuration Data

Scheduler uses the following database tables:

Table SC_SCHEDULER

This table contains job parameters:

- ID is a unique job identifier.
- JOB_NAME is the name of a job.
- DATE_FROM is the earliest date on which a job may be executed.
- DATE_TO is the latest date on which a job still may be executed.
- WEEK_TIMETABLE is the periodicity of job execution (daily, weekly, etc.).
- TIME_FROM is the earliest time of the day when a job may be executed in "HHMM" format.
- TIME_TO is the latest time of the day when a job may be executed in the "HHMM" format.
- TIME_INTERVAL is the interval in minutes between the iterations of a job if it is to be executed repeatedly.
- NUMBER_OF_CALLS is the necessary number of daily iterations of a job. The interval between the iterations is determined by the value of TIME_INTERVAL.
- CURR_CALLS is the number of times a job has been performed on the current calendar day.
- ESTIMATED_START is the next scheduled execution of a job.
- CURRENT_TASK is the task currently being executed.
- FIRST_TASK is the first task of a job.
- STATUS is the status of the last executed job:
 - "Waiting" means that the job is ready to be executed.
 - "Suspended" means that the job is being executed.
 - "Posted" means that the job has been executed successfully.
 - "Closed" or "Inactive" means that the job has been terminated by the user.
 - "Decline" means that the job has terminated due to an error.
- IS_READY is the approval status of a job:
 - "Y" means that the changes have been approved.

- "N" means that the changes have not been approved.
- STATION_CODE is the name of a Scheduler instance.

Table SC_TASK

This table contains task parameters:

- ID is a unique task identifier.
- SC_SCHEDULER__OID is the job to which a task belongs.
- TASK_NUMBER is the number of a task in a job.
- TASK_NAME is the name of a task.
- TASK_TYPE is the type of a task.
 - "E" means running an executable file.
 - "S" means running a PL/SQL script.
 - "I" means executing a DB Manager user menu item.
 - "P" means assigning a parameter to a task.
 - "C" means executing another Scheduler job.
 - "F" means assigning a special parameter (flag) to a task.
 - "W" means waiting.
 - "L" means flag checking.
- TASK_STRING_1 is a partial command line.
- TASK_STRING_2 is a partial command line.
- TASK_STRING_3 is a partial command line.
- TASK_STRING_4 is a partial command line.
- TASK_STRING_5 is a partial command line.
- TASK_STRING_6 is a partial command line.
- NEXT_TASK is the task that must be executed next if the execution of the current task is a success.
- ERROR_TASK is the task that must be next executed if the execution of the current task results in an error.
- MENU_ITEM__ID is the ID of a DB Manager user menu item.
- IS_READY is the approval status of a task:
 - "Y" means that the changes have been approved.
 - "N" means that the changes have not been approved.

Table SC_TASK_CALL

This is a log table for registering tasks being executed and the results of their execution.

- ID is the unique identifier of a job execution record.
- SC_SCHEDULER__ID is a reference to a job description.
- SC_TASK__OID is a reference to a task description.
- STARTED is the actual time when task execution began.
- FINISHED is the actual time when task execution was completed.
- STATUS is the task execution status:
 - "P" is "Posted", the task was successfully completed.
 - "D" is "Decline", the task terminated due to an error.
 - "S" is "Suspended", the task is being executed.
- PROCESS_LOG__ID is the ID of a record in the Process Log.
- CONNECTION__ID is the ID of a row in the LOGIN_HISTORY and LOCAL_CONSTANTS tables. This field is used to pass external parameters to a job. The parameters may be those of a DB Manager session, Scheduler instance or another job or task.

Table SC_TASK_PARAM

This table is used for context-dependent substitutions, much like certain configuration files (see "File OpenVar.ini" and "File ShadowVar.ini").

- PARAM_CODE is the code of a parameter unique within the task.
- PARAM_VALUE is the parameter value.
- PARAM_NAME is the parameter name.
- CONNECTION_ID is the ID of a row in the LOGIN_HISTORY and LOCAL_CONSTANTS tables. This field is used to pass external parameters such as those of DB Manager to a job. It allows different values to be assigned to the same parameters when tasks are executed by different Scheduler instances.
- TASK_ID is the ID of the task for which the parameter is being defined.
- OFFICER_ID is the ID of the user for whom this parameter is used. In different situations, this field may have different meanings:
 - When Scheduler is started, this is the name of the user who starts Scheduler.
 - When Scheduler automatically executes a job, this is the User ID specified in the configuration file (see "File login.cfg").

- When a task is executed through DB Manager (see "Immediate Task Execution"), this is the ID of the user working with DB Manager.
- WORKSTATION is the ID of the Scheduler instance for which the parameter is being defined.
- PARAM_SECTION is the name of a parameter section. This field allows parameters to be structured by grouping them in sections.