Thomas Rinneberg
BW Data Staging

Job Scheduling BW - Certification Scenarios

Interface-description Process Chain API's

21. September 2004

Distribution: SAP Partners

Contents:

1 Scenarios

There are three certifiable scenarios on BW Scheduling:

- Minimal scenario for Process Chains
- Full scenario for Process Chains
- Scheduling scenario for InfoPackages

In the following documentation, only the interfaces for the Process Chain scenarios are described in detail. For the InfoPackage-Scenarios, only the corresponding BAPI functions are listed.

Whereas since BW 3.x, only the process chains should be used to schedule jobs in BW, they are not available for BW 2.x, so in BW 2.x partners can address only the InfoPackages. With BW 3.x the InfoPackage BAPI's are still fully functional, but this does not necessarily hold true for future releases of BW.

In system CB4 and as transport request in attached ZIP-file, there exists a demo program `Z_PROCESS_CHAIN_CERTIFICATION` with transaction code `ZPCD`, showing how the various API's can be adressed and demonstrating below mentioned scenarios.

Demo_program.zip

1.1 Minimal scenario for Process Chains

1.1.1 Scenario

- Get a list of process chains in the BW system, supporting all offered selection options
- Start a process chain
- Monitor the execution state of the process chain run
- Display all available logs to the user, including the process chain log and the logs of the single processes

1.1.2 Used (B)API's

• RSPC_API_GET_CHAINS	Process chains in the system
• RSPC_API_CHAIN_START	Start process chain
• RSPC_API_CHAIN_GET_STATUS	Status of the chain run
• RSPC_API_CHAIN_GET_LOG	Log of the process chain
• RSPC_API_CHAIN_GET_PROCESSES	Processes of the chain
• RSPC_API_PROCESS_GET_LOG	Logs of the process
• BAPI_MESSAGE_GETDETAIL	Read short and long text of error message

1.2 Full scenario for Process Chains

1.2.1 Scenario

- Get a list of process chains in the BW system, supporting all offered selection options
- Display the start condition of the chain to the user or allow the user to filter the chain list according to the start conditions

- Start a process chain
- Monitor the execution state of the process chain run
- Display the processes of the chain to the user at least as a list, including descriptions and corresponding batch jobs
- Display all available logs to the user, including the process chain log and the process and batch logs of the single processes, also including message long texts, if available
- Offer the possibility to start the SAP-GUI to display a certain process chain
- Offer the possibility to restart a broken process chain
- Offer the possibility to remove a process chain from schedule

1.2.2 Used (B)API's

• RSPC_API_GET_CHAINS	Process chains in the system
• RSPC_API_CHAIN_GET_STARTCOND	Get start conditions of a process chain
• RSPC_API_CHAIN_START	Start process chain
• RSPC_API_CHAIN_GET_STATUS	Status of the chain run
• RSPC_API_CHAIN_GET_LOG	Log of the process chain
• RSPC_API_CHAIN_GET_PROCESSES	Processes of the chain
• RSPC_API_PROCESS_GET_INFO	Information about a process
• RSPC_API_PROCESS_GET_LOG	Logs of the process
• BAPI_MESSAGE_GETDETAIL	Read short and long text of error message
• RSPC_API_CHAIN_MAINTAIN	Start process chain maintainance in SAP-GUI
• RSPC_API_CHAIN_RESTART	Restart a broken process chain
• RSPC_API_CHAIN_INTERRUPT	Remove the processes of a chain from schedule

1.3 Scheduling scenario for InfoPackages

1.3.1 Scenario

- Get a list of InfoPackages in the system, supporting all offered selection options
- Start an InfoPackage in the BW system
- Monitor the state of the InfoPackage
- Display the log of the request to the user

1.3.2 Used BAPI's

• BAPI_IPAK_GETLIST	Get a list of InfoPackages
• BAPI_IPAK_START	Start an InfoPackage / create request
• BAPI_ISREQUEST_GETSTATUS	Get status and log of the created request

1.3.3 Other BAPI's

There are more BAPI's available for InfoPackages than mentioned in this scenario. However they are intended rather for ETL-Tools (Extraction, Transformation, Loading) than for plain scheduling tools. You may also use the following BAPI's, but they are not part of the certification scenario:

• BAPI_IPAK_GETDETAIL	Get the detail information about an InfoPackage
• BAPI_IPAK_CHANGE	Change the InfoPackage, e.g. data selection conditions
• BAPI_IPAK_STOP	Remove an InfoPackage from schedule

2 Process Chain API's

The function modules used to address the process chains are API's, which means SAP tries to change the function modules only downward-compatible but they are not technically released in the function library (e.g. new fields may be added also in return structures). All the function modules can be found in function group RSPC_API in a BW system.

All parameters follow the naming convention, that parameters, which can or must be passed to the function (importing parameters) start with I_, all parameters, which are given back by the function (exporting parameters) start with E_.

If an API has exceptions, you should be able to handle them (display it to the user).

2.1 RSPC_API_GET_CHAINS

2.1.1 Functionality

With this module you are able to obtain a list of the process chains in the system. A dialogue is processed if requested. You are able to select chains by indicator and by name.

2.1.2 Parameters

2.1.2.1 I_WITH_DIALOG

A dialog is processed, from which a chain can be selected.

Setting parameter I_WITH_DIALOG to 'X' displays the full list in SAP GUI. You should leave the parameter empty, retrieve the list in E_T_CHAINS, then display it yourself

2.1.2.2 I_SEL_CHAIN

Using the wildcard * you are able to select the name of the chain you are looking for. Wildcard ? is not supported.

2.1.2.3 I_SEL_TXTLG

Using the wildcard * you are able to select the description of the chain you are looking for. Wildcard ? is not supported.

2.1.2.4 E_T_CHAINS

List of chains with descriptions.

2.2 RSPC_API_CHAIN_GET_STARTCOND

2.2.1 Functionality

This module delivers the start conditions for the process chain. This is simultaneously the process variant for the process chain start processes (TRIGGER).

Even if the start conditions claim otherwise, RSPC_API_CHAIN_START starts the chain immediately. The function in question is, therefore, only useful for information purposes.

You can use it e.g. to filter out all chains not meeting the condition, that e_s_trigger-meta is set. This flag indicates, that the user has intended the chain to be started via 3'd party. You can use this API also to give an overview over the start conditions of the chains in the system.

2.2.2 Parameters

2.2.2.1 I_CHAIN

Process Chain

2.2.2.2 E_S_TRIGGER

Chain start condition. The structure contains the usual schedule options for batch management.

2.2.3 Exceptions

2.2.3.1 FAILED

Operation failed

2.3 RSPC_API_CHAIN_START

2.3.1 Functionality

With this module you are able to start a process chain. The process chain will start, even if the start conditions entered in process chain maintenance claim differently.

2.3.2 Parameters

2.3.2.1 I_CHAIN

Process Chain

2.3.2.2 I_T_VARIABLES

The optional parameter I_T_VARIABLES will be passed on to the processes in the chain. At the moment (BW 3.x), except internal usage in SAP-MDM, no process type supports these parameters. This might change with later releases, so you then can offer the feature to run the chain pre-customized by your tool. Thus until then, RSPC_API_CHAIN_GET_VARIABLES will not return any values and therefore is not part of any certification scenario.

2.3.2.3 I_SYNCHRONOUS

The chain is executed directly, not using batchjobs. Also the processes in the chain need to execute synchronous.

This mode for executing processes primarily enables a quick and efficient (real time) execution without administration overhead. Normally only small amounts of data need to be processed so that parallel processing within a processtype can be waived.

In each case the logs need to be minimized. However, a correct status specification also must remain afterwards (especially with process types that implement IF_RSPC_GET_STATUS).

Processes that nevertheless work with HOLD (asynchronous processes) are left in status ‘active’ and the processing of the chain is suspended. It will be synchronously restarted when the process informs the chain about it’s ending.

For instance for the process chain, the execution in batch is waived, and the processes are processed in a series. In spite of this, a log line is writted for each process included, so that a display of the chain run can remain using the normal transaction.

2.3.2.4 I_SIMULATE

The results of the execution of the processes are not persistent. No persistent log is written either. Currently there is no process type that supports this feature.

2.3.2.5 E_LOGID

The log-ID is determined during the runtime (by the start-process). It holds the chain run together.

2.3.3 Exceptions

2.3.3.1 FAILED

Operation failed

2.4 RSPC_API_CHAIN_GET_STATUS

2.4.1 Functionality

This function can be used to poll the status of the chain in suitable intervals. Note that the module also calculates the status of all processes contained in the chain. Depending on the complexity of the chain, this might put load on the system and take some time. Also, the chain is locked during the function executes, so execution of the chain will wait until the function is done. You need to consider this impact when thinking about an auto-run-interval. Also note, that the function also updates the calculated status on the SAP database, which is relevant especially if a process of the chain aborts.

2.4.2 Parameters

2.4.2.1 I_CHAIN

Process Chain

2.4.2.2 I_LOGID

The log-ID is determined during the runtime (by the start-process). It holds the chain run together.

2.4.2.3 E_STATUS

The state has values

- 'R' - red
- 'G' - green
- 'X' - aborted
- 'A' - active

other values listed in the domain values of domain RSPC_STATE are not used by the process chain itself. They might appear for single processes, however.

2.5 RSPC_API_CHAIN_GET_LOG

2.5.1 Functionality

With this API you are able to obtain a process chain log as a list of T100 messages. You can put a T100 Message into short, or if necessary, longtext using the BAPI_MESSAGE_GETDETAIL module.

The log contains one message per process of the chain

2.5.2 Parameters

2.5.2.1 I_CHAIN

Process Chain

2.5.2.2 I_LOGID

The log-ID is determined during the runtime (by the start-process). It holds the chain run together.

2.5.2.3 E_T_LOG

List of messages in T100 format (msgno, msgid, msgty, msgv*).

You can put a T100 message into short and, if necessary, long text using the BAPI_MESSAGE_GETDETAIL module.

2.6 RSPC_API_CHAIN_GET_PROCESSES

2.6.1 Functionality

This module delivers the list of contained processes

- when indicating the log ID for a process chain run,
- when this is not indicated for the planned version of the chain.

2.6.2 Parameters

2.6.2.1 I_CHAIN

Process Chain

2.6.2.2 I_LOGID

The log-ID is determined during the runtime (by the start-process). It holds the chain run together.

The parameter I_LOGID is optional, so you can use this API also to read the chain before it runs to show to the user, what it will do.

2.6.2.3 E_T_PROCESSLIST

List of processes of the chain.

You can find information on the fields in the structure in ABAP Dictionary . Here you will find documentation on the data elements of the respective fields and the short description of the fields in the structure.

The return structure of the API is the internally used structure, so it contains all information at all known about a process. Also this means it will be subject to changes (e.g. additional fields).

Note, that in case of I_LOGID is given, those processes, which are already run, will have the flag PREDECESSOR filled, and this flag indicates, that the fields of include-structure RSPC_S_INSTANCE as well as the timestamps will be filled, whereas if the process did not yet run, the include-structure RSPC_S_PLANNED will be filled and the PREDECESSOR-flag will be empty.

The fields EVENT_* and EVENTP_* can be used to derive the layout (predecessor and successor information) of the chain.

The field STATE contains the state, which the process returned at runtime. The field ACTUAL_STATE contains the state, which was retrieved by executing the API. These states may differ, as e.g. a loading process may turn to red by manual action long after the load was finished. Watch the domain values of RSPC_STATE for the valid values of the state-fields.

The jobcount can be used to determine the corresponding batchjob of the process. However note that many processes do not end in this batchjob (called asynchronous processes). The jobname is determined by internal function module RSPC_JOBNAME, currently it is 'BI_PROCESS_<type>'. But as the module is subject to changes, you cannot rely on this convention. The jobname as well as the joblog of the corresponding job can be retrieved by function RSPC_API_PROCESS_GET_LOG when you pass the jobcount to it.

2.6.3 Exceptions

2.6.3.1 FAILED

Operation failed

2.7 RSPC_API_PROCESS_GET_INFO

2.7.1 Functionality

Currently, this function returns only the description text of a process variant.

2.7.2 Parameters

2.7.2.1 I_TYPE

Process Type. This determines, among other things, which tasks the process has and which properties it has in maintenance. The process type is maintained in the table RSPROCESSTYPES.

2.7.2.2 I_VARIANT

Process Variant (Name of the process). In the context of process chains a variant is the static configuration of a process of a specific type. A variant is only uniquely defined in connection with the type.

This parameter is optional, you can retrieve the description of the process type only by leaving parameter I_VARIANT empty.

2.7.2.3 E_TYPE_TEXT

Description of the process type

2.7.2.4 E_VARIANT_TEXT

Description of the process variant

2.7.3 Exceptions

2.7.3.1 NOT_AVAILABLE

Process type or variant does not exist

2.8 RSPC_API_PROCESS_GET_LOG

2.8.1 Functionality

With this module you can obtain the process log and the batch log for a process in the chain. You first have to determine the process with module RSPC_API_CHAIN_GET_PROCESSES.

Note, that if the chain was executed synchronously, no job exists, so jobname, jobcount and joblog will be empty. If the specific process has no own log, the process log will be empty.

2.8.2 Parameters

2.8.2.1 I_LOGID

The log-ID is determined during the runtime (by the start-process). It holds the chain run together.

2.8.2.2 I_TYPE

Process Type. This determines, among other things, which tasks the process has and which properties it has in maintenance. The process type is maintained in the table RSPROCESSTYPES.

2.8.2.3 I_VARIANT

Process Variant (Name of the process). In the context of process chains a variant is the static configuration of a process of a specific type. A variant is only uniquely defined in connection with the type.

2.8.2.4 I_INSTANCE

The instance was defined by the EXECUTE-Methods (consistent for all systems and times) for example as GUID with the help of the module RSSM_UNIQUE_ID, Return Parameter E_UNI_IDC25) After the process ended, it was transferred to process chain management where it was then saved. Each process uses the instance in the key to save its log information.

For example: The instance is the same as the request number during the loading process.

2.8.2.5 I_JOBCOUNT

The ID number of the job with which the process was executed.

This number serves to identify the process if the process instance is not available because the process was terminated.

2.8.2.6 E_JOBNAME

The name of the background job with which the process was executed. The jobname is determined by internal function module RSPC_JOBNAME, currently it is 'BI_PROCESS_<type>'. But as the module is subject to changes, you cannot rely on this convention.

2.8.2.7 E_JOBCOUNT

The ID number of the job with which the process was executed.

An internal ID number assigned to each batch job. The number is based upon the time at which a job was scheduled. For programming, the job number is needed, together with the job name, to identify a particular job. The combination of number and name uniquely identifies a job. The number is returned by the background processing system when a job is created by a program.

2.8.2.8 E_STATUS

The state of the process. Except ‘S’ (skipped) in principle all status listed in the domain values of domain RSPC_STATE can occur here:

State	Description	Map to
R	Ended with errors	error
G	Successfully completed	success
F	Completed	success
A	Active	active
X	Canceled	error
P	Planned	active
Q	Released	active
Y	Ready	active
<blank>	Undefined	active

2.8.2.9 E_T_PROCESS_LOG

List of messages in T100 format (msgno, msgid, msgty, msgv*).

You can convert these messages into short, and if necessary, long text using the BAPI_MESSAGE_GETDETAIL module.

2.8.2.10 E_T_JOB_LOG

Job Log of the corresponding batch job.

Messages are specified both in T100 format and as short text here. Also time information is given in this log.

2.8.3 Exceptions

2.8.3.1 FAILED

Operation failed

2.9 RSPC_API_CHAIN_MAINTAIN

2.9.1 Functionality

This module starts either the process chain WIN-GUI maintenance or, if you specify the Log ID, the log view.

2.9.2 Parameters

2.9.2.1 I_CHAIN

Process chain

2.9.2.2 I_LOGID

The log-ID is determined during the runtime (by the start-process). It holds the chain run together.

The parameter I_LOGID is optional, so you can use this API also to call up the maintainance of the process chain. If you pass the logid, then the corresponding log will be displayed

2.9.3 Exceptions

2.9.3.1 ERROR

Operation failed

2.10 RSPC_API_CHAIN_RESTART

2.10.1 Functionality

This module restarts any process for which the entry restart is showing in the context menu log view. These are mostly cancelled processes or processes with errors.

This function does only change something, if there is a broken process (with status R or X) in the chain. So you do not need to offer this function in case that no such process is in the current chain run. Please also check notes in CSN, as there have been some bugs with that function

Not all processes can be restarted. The load process in particular cannot be restarted. In that case, each process that follows the load process in the chain will be started instead.

2.10.2 Parameters

2.10.2.1 I_CHAIN

Process chain

2.10.2.2 I_LOGID

The log-ID is determined during the runtime (by the start-process). It holds the chain run together.

2.10.3 Exceptions

2.10.3.1 FAILED

Operation failed

2.11 RSPC_API_CHAIN_INTERRUPT

2.11.1 Functionality

With this module the chain is removed from the schedule.

This function will only remove the jobs from schedule, which did not yet run. It will not kill running jobs, as this may leave the system in an inconsistent state, moreover killing processes is possible for synchronous processes only anyhow. This means, that there will be no errors with interrupted chains, which on the other hand means an interrupted chain is not restartable. So for a running chain this function is intended to be an emergency break only. You can use it, however, to remove a scheduled chain from schedule, such that it does not start except by your product.

2.11.2 Parameters

2.11.2.1 I_CHAIN

Process chain

2.11.3 Exceptions

2.11.3.1 FAILED

Operation failed

3 InfoPackage BAPI’s

3.1 BAPI_IPAK_GETLIST

3.1.1 Functionality

Generates a List of InfoPackages

This method delivers a table with InfoPackages that correspond to the selections made.

3.1.2 Parameters

The importing parameter JOB_STATUS as well as the selection tables SELTEXTLONG, SELINFOSOURCE, SELSOURCESYSTEM, and SELDATASOURCE, are optional.

3.1.2.1 JOB_STATUS

Sm37-Batch-Status.

If this status is filled, only those InfoPackages are selected that have an sm37 status, that is, those that are scheduled or that are currently running in the background ...

Supported status values:

'S' : Loading job is scheduled.

'R' : Loading job is currently running.

'F' : Loading job is complete.

'A' : Loading job terminated.

3.1.2.2 SELTEXTLONG

This table contains possible names of InfoPackages.

Fields:

SIGN:	1-character field: Only 'I' allowed.
OPTION:	2-character field: For single values 'EQ', for an interval, 'BT'.
TEXTLONGLOW:	60-character field: From_value.
TEXTLONGHIGH:	60-character field: To_value.

3.1.2.3 SELINFOSOURCE

This table contains possible InfoSources for which InfoPackages were created.

Fields:

SIGN:	1-character field: Only 'I' allowed.
OPTION:	2-character field: For single values 'EQ', for an interval, 'BT'.
INFOSOURCELOW:	30-character field: From_value.
INFOSOURCEHIGH:	30-character field: To_value.

3.1.2.4 SELSOURCESYSTEM

This table contains possible source systems for which InfoPackages were created.

Fields:

SIGN:	1-character field: Only 'I' allowed.
OPTION:	2-character field: For single values 'EQ', for an interval, 'BT'.
SELSOURCESYSTEMLOW:	10-character field: From_value.
SELSOURCESYSTEMHIGH:	10-character field: To_value.

3.1.2.5 SELDATASOURCE

This table contains possible DataSources for which InfoPackages were created.

Fields:

SIGN:	1-character field: Only 'I' allowed.
OPTION:	2-character field: For single values 'EQ', for an interval, 'BT'.
DATASOURCELOW:	30-character field: From_value.
DATASOURCEHIGH:	30-character field: To_value.

3.1.2.6 INFOPACKAGE_LIST

The table INFOPACKAGE_LIST delivers all InfoPackages that correspond to the selection conditions.

Fields and meanings:

INFOPACKAGE:	30-character field: Technical name of the InfoPackage.
TEXT:	60-character field: Name of the InfoPackage in the logon language.

3.2 BAPI_IPAK_START

3.2.1 Functionality

Schedules an InfoPackage. This method starts an InfoPackage.

3.2.2 Parameters

3.2.2.1 INFOPACKAGE

Technical InfoPackage Name.

3.2.2.2 JOBNAME

Job name the batch job should get at start.

3.2.2.3 REQUESTID

After a successful start, this field contains the request number under which the data is stored.

Key number for a quantity of data and control information, that belongs together and has been requested in BW at the same time. A request consists of data (sometimes split up into several data packets) and additional information about this data.

3.3 BAPI_ISREQUEST_GETSTATUS

3.3.1 Functionality

Determines the Status of a Data Request (Request). Like for RSPC_API_CHAIN_GET_STATUS, the status calculation will examine the request thouroughly and eventually store the status on the SAP database.

Please check CSN-notes, as there have been bugs with that module.

3.3.2 Parameters

3.3.2.1 REQUESTID

Key number for a quantity of data and control information, that belongs together and has been requested in BW at the same time. A request consists of data (sometimes split up into several data packets) and additional information about this data.

3.3.2.2 TECHSTATUS

The parameter TECHSTATUS is the technical status of the request. This always has the following values:

G	Green (Request updated successfully)
Y	Yellow (Request being processed)
R	Red (Request incorrect or terminated)

3.3.2.3 TECHINFO

Descriptive text on request status (technical status)

3.3.2.4 QUALSTATUS

The parameter QUALITYSTATUS is the manually set status for the request. This status may be blank. In SAP BW, QUALITYSTATUS overrides TECHICALSTATUS.

Value range

G	Green (Request updated successfully)
Y	Yellow (Request being processed)
R	Red (Request incorrect or terminated)
<blank>	not set (overall status equal to TECHNICALSTATUS)

3.3.2.5 QUALINFO

Descriptive text on manually set request status.

3.3.2.6 LOCATION

Current Processing Step. The possible values can be found in the domain fixed values of domain RS_ERROR_LOCATION.

3.3.2.7 RETURN

Exeptions during execution of the module. There can be mostly two messages in this structure:

Error RSM 131 (‘Could not generate request &’): The request has not yet started. You should react on this error by repeating the function after a suitable delay.

Error RSM 105 (‘No entries found for request no. &’): BW 2.x: Same as RSM 131. BW 3.x: The request could not been started. This means the creation of the request has finally failed.

3.3.2.8 MESSAGES

Messages in XML Format

Possible tags are:

Special: <msghead ty= id= no= >, <msgbody>

HTML: <h1>, <h2>, <h3>, <p>, <ol>, <ul>, <li>