Background Processing, Job Scheduling System Test Catalogue for BC-XBP Version 7.2

External Interface for Background Processing

Version 1.0

16.10.2015

Prerequisites regarding the certification of BC-XBP 7.2.

XBP 7.2 is available with support package 16 of SAP NetWeaver release 7.00
All newer Basis Releases: from the beginning

There will be no separate certification for XBP 6.10 any more. Most of the tests, which were formerly contained in the XBP 6.10 test catalogue, are now part of this test catalogue.

FAQ:

1. Question: Why is there no more XBP 6.10 certification ?
Answer:
XBP 7.0 is a superset of XBP 6.10. Therefore, users of XBP 6.10 can install XBP 7.0 without any changes in functionality of their current applications. XBP 7.0 will be available for SAP Basis releases 7.0 (SP14) and higher.

Since today basically every external job scheduler supports XBP 7.0**, the two certifications XBP 6.10 and XBP 7.0 have now been merged to one certification XBP 7.2 (2015).**

The Tests:

1 Basic Functions

1.1 Creating jobs with ABAP steps

Description

Create two jobs XBP_TEST_1 and XBP_TEST_2 with at least two ABAP steps each.
For specifying the ABAP reports as job steps, the scheduler has to offer a value help.

Prerequisites

A connection to an SAP SAP R/3 System is established.
An XBP session has to be established through the XMI interface.

Realization

The jobs should be created using the function modules
BAPI_XBP_JOB_OPEN
BAPI_XBP_JOB_ADD_ABAP_STEP (or BAPI_XBP_ADD_JOB_STEP)
BAPI_XBP_JOB_CLOSE

For the value help for the ABAP reports, the function BAPI_XBP_REPORT_SEARCH has to be used.

For example, use the programs RSPARAM and SHOWCOLO as ABAP steps.
XBP_TEST_1 should have job class A.
XBP_TEST_2 should have job class C.

Test

1. In the SAP job overview (transaction transaction SM37) the two jobs appear. The job class can be verified in the job details.

2. The Job details/Step list shows the correct values

1.2 Starting a job (asap)

Description

Start the job XBP_TEST_1 through BAPI_XBP_JOB_START_ASAP.

Prerequisites

Make sure that Job Interception is switched off.
A connection to an SAP R/3 System is established.
An XBP session has to be established through the XMI interface.
The job XBP_TEST_1 is scheduled and managed within the external job scheduling system.

Realization

Use BAPI_XBP_JOB_START_ASAP.and specify a target server.

Test

1. In the SAP job overview (transaction SM37), the job XBP_TEST_1 appears as *released* (with a delay), *ready*, *running*, or *finished*.

1.3 Starting a job (immediately)

Description

Start the job XBP_TEST_2 using BAPI_XBP_JOB_IMMEDIATELY.

Prerequisites

Make sure that Job Interception is switched off.
A connection to an SAP R/3 System is established.
An XBP session has to be established through the XMI interface.
The job XBP_TEST_2 is scheduled and managed within the external job scheduling system.
There are free batch resources.

Realization

a) Use BAPI_XBP_JOB_START_IMMEDIATELY and specify a target server..

Test

The job XBP_TEST_2 appears in the SAP job overview (transaction SM37) as *released* (with a delay), *ready*, *running*, or *finished*.

1.4 Monitoring a job (reading the status)

Description

The status of a job should be read.

Prerequisites

Make sure that job interception is switched off.
A connection to an SAP R/3 System is established.
An XBP session has to be established through the XMI interface.
Create and start a job XBP_TEST_3 with step BTCLOOP (it is an endless loop) .

Realization

Read the status of the job XBP_TEST_3 with BAPI_XBP_JOB_STATUS_GET.
After performing the next test scenario 1.5. read the job status again.

Test

The returned status should be *running* first case and *cancelled* in the second case.

1.5 Canceling a job

Description

The job XBP_TEST_3 should be cancelled.

Prerequisites

A connection to an SAP R/3 System is established.
An XBP session has to be established through the XMI interface.
The job XBP_TEST_3 is managed within the external job scheduling system.

Realization

Use BAPI_XBP_JOB_ABORT.

Test

The job should be shown as cancelled in transaction SM37.

1.6 Deleting a job

Description

The job XBP_TEST_3 should be deleted.

Prerequisites

A connection to an SAP R/3 System is established.
An XBP session has to be established through the XMI interface.
The job XBP_TEST_3 is managed within the external job scheduling system.

Realization

Use BAPI_XBP_JOB_DELETE

Test

The Job XBP_TEST_3 should not appear any more in transaction SM37.

1.7 Raising an event

Description

Raise the event SAP_TEST.

Prerequisites

A connection to an SAP R/3 System is established.
An XBP session has to be established through the XMI interface.
Create within the SAP system a job XBP_TEST_4, which waits for event SAP_TEST.

Realization

Use the function BAPI_XBP_EVENT_RAISE.

Test

The job XBP_TEST_4 should run after raising the event. Check in transaction SM37.

1.8 Reading the job log

Description

Read the job log of the job XBP_TEST_4.

Prerequisites

A connection to an SAP R/3 System is established.
An XBP session has to be established through the XMI interface.
The job XBP_TEST_4 is managed by the external job scheduler.

Realization

Use BAPI_XBP_JOB_JOBLOG_READ.

Test

The job log in the SAP system is identical with the job log shown by the external job scheduler.

1.9 Copying a job

Description

Create a job by copying the job XBP_TEST_4 to XBP_TEST_5.

Prerequisites

A connection to an SAP R/3 System is established.
An XBP session has to be established through the XMI interface.
The job XBP_TEST_5 is managed by the external job scheduler.

Realization

Use BAPI_XBP_JOB_COPY.

Test

Check in transaction SM37, that the jobs XBP_TEST_4 and XBP_TEST_5 have the same job data (except start condition). The new job XBP_TEST_5 should be in status *scheduled* without start condition.

1.10 Selecting jobs

Description

Select all jobs that have been created during this certification procedure by the SAP user, under which the external scheduler logs on to the SAP system.

Prerequisites

A connection to an SAP R/3 System is established.
An XBP session has to be established through the XMI interface.

Realization

Use BAPI_XBP_JOB_SELECT.

Test

Compare with sm37.

1.11 Reading the variants of a report

Description

Read all variants of a given ABAP report in a certain client. Note, that variants (except system variants starting with SAP&) are client-dependent.
The report can be chosen during the test; it should have at least 5 variants.

Prerequisites

A connection to an SAP R/3 System is established.
An XBP session has to be established through the XMI interface.

Realization

Schedule a job XBP_TEST_6 with the above report as a job step. While adding this step to the job, the user of the external scheduler should be able to display a list with all variants of this report. Use BAPI_XBP_VARIANT_INFO_GET.

Test

Display the variants of the report, for example e.g. in transaction se38. They should be the same as those displayed by the external scheduler.

1.12 Setting the Audit Level XMI

Description

Set the audit level for the XMI logging higher/lower. This function is mandatory, because it is the only way to set the audit level within the XMI interface.

Explanation:
The XMI framework, which allows external Management Tools to log on to the SAP system and create XMI-sessions, also offers the possibility of writing a trace of the functions called during an XMI-session. The trace level can be set with the function BAPI_XMI_SET_AUDITLEVEL and can be viewed in the SAP system using transaction rz15.
In the terminology of XMI, the trace level is called audit level.
If the function BAPI_XMI_SET_AUDITLEVEL is not called explicitly in an XMI session, the audit level is 0 by default. Most XBP functions that make changes of some description in the system (create jobs, change jobs etc.), are already traced at audit level 0. Most XBP functions that only read information (such as BAPI_XBP_JOB_STATUS_GET) are only traced with audit level 2 or higher.

Prerequisites

A connection to an SAP R/3 System is established. A (XBP) session has to be established through the XMI interface.

Realization

The audit level of the XMI interface should be raised to 2 using the external job scheduling system. Use BAPI_XMI_SET_AUDITLEVEL . Then a status check of an existing job should be performed by the external scheduler (see 1.8)
If everything worked fine, set the audit level back to the default value 0.

Test

Check, if there are additional messages in the XMI log for the call of BAPI_XBP_JOB_STATUS_GET.

2 Print Parameters

2.1 Specifying print parameters for a job step

Description

Create a job XBP_PRIPAR that has report RSWATCH0 as its only step.
In the print specifications of the job step specify the following:

Output device = <existing printer>
Title of spool request = XBP 3.0 Certification
Department = SAP Basis

For specifying the output device, a value help has to be offered by the scheduler.

Prerequisites

A connection to an SAP R/3 System is established.
An XBP session has to be established through the XMI interface.

Realization

Create the job as described in 1.1.
For the print parameters, use the appropriate parameter structure of BAPI_XBP_JOB_ADD_ABAP_STEP or BAPI_XBP_ADD_JOB_STEP.
For the value help for the output device, the function BAPI_XBP_OUTPUT_DEVICE_SEARCH has to be used.

Test

After creating the job, check the print parameters using transaction SM37. The print parameters of a job step can be found by displaying the step details and the clicking ‘Print specifications’.

3 Parent/child functionality

For the following tests a report ZMAKECHILD is needed, which creates n jobs CHILD_1, …., CHILD_n.
For testing, n = 4, and one of the four jobs should execute BTCLOOP as a job step.
SAP will provide such a report ZMAKECHILD of this type (see also appendix).
For the following tests, the parent/child function must be switched on, the intercept functionality must be switched off.

3.1 Starting a parent job

Description

Create a job PARENT that has report ZMAKECHILD its only step.
Then start the job PARENT immediately.

Prerequisites

A connection to an SAP R/3 System is established.
An XBP session has to be established through the XMI interface.

Realization

Create and start the job as described in 1.1 and 1.2 / 1.3.

Test

The job PARENT and the four child jobs should be displayed by transaction SM37.

3.2 Retrieving the child jobs of a job

Description

Retrieve the child jobs of the job PARENT (started in 3.1).

Prerequisites

A connection to an SAP R/3 System is established.
An XBP session has to be established through the XMI interface.
The job PARENT is managed within the external job scheduling system.

Realization

Use BAPI_XBP_JOB_CHILDREN_GET.

Test

Compare the returned job names and job counts with those from transaction SM37.

3.3 Reading the status of the child jobs of a job

Description

Read the status of the child jobs of the job PARENT (started in 3.1).

Prerequisites

A connection to an SAP R/3 System is established.
An XBP session has to be established through the XMI interface.
The job PARENT is managed within the external job scheduling system.

Realization

Use BAPI_XBP_JOBLIST_STATUS_GET.

Test

Compare the delivered status data with the data shown in transaction SM37.

4 Variants

In the following scenarios, functions of the function groups SXBP_VAR and SXBP are to be used.

For the scenarios below, use the selection screen and variants of the following program:

```
REPORT  ZVARI_1.

TABLES: tbtco, tbtcp.

PARAMETERS user LIKE tbtco-sdluname default 'scheduler'.

PARAMETERS date LIKE sy-datum default '19720101'.

PARAMETERS chbox AS CHECKBOX.

PARAMETERS radio11 RADIOBUTTON GROUP tst1.
PARAMETERS radio21 RADIOBUTTON GROUP tst1  DEFAULT 'X'.
PARAMETERS radio31 RADIOBUTTON GROUP tst1.

PARAMETERS radio12 RADIOBUTTON GROUP tst2.
PARAMETERS radio22 RADIOBUTTON GROUP tst2  DEFAULT 'X'.
PARAMETERS radio32 RADIOBUTTON GROUP tst2.

SELECT-OPTIONS sel_op1 FOR tbtco-jobname no-extension.
SELECT-OPTIONS sel_op2 FOR tbtco-jobcount no-display.

select-options sel_prog for tbtcp-xpgparams.
parameters par_prog like tbtcp-xpgparams.
```

4.1 Reading the Selection Screen of a Program and Creating a Variant

Description

A variant for the program ZVARI_1 (see above) is to be created. In order to do that, the scheduler needs to read the selection screen of the program first.

Prerequisites

A connection to an SAP system is established.
The corrections of note 1171295 have to be in the test system.
The audit level is set to the value 2, so that the use of the required functions can be verified in the XMI log (RZ15).

Realization

Use BAPI_XBP_READ_SELSCREEN to read the selection screen.
Use BAPI_XBP_VARIANT_CREATE to create the variant.

Verification

Check XMI log via RZ15.
Check in SE38, if a new variant has in fact been created.

4.2 Changing the Variant of a Program

Description

Change the variant created above. Change at least two fields.

Prerequisites

A connection to an SAP system is established.
The corrections of note 1171295 have to be in the test system.

Realization

Use BAPI_XBP_VARIANT_CHANGE to change the variant.

Verification

Check XMI log via RZ15.
Check in SE38, if the values of the variant have been changed.

4.3 Reading the Selection Screen of a Program and Creating a Jobwith a Temporary Variant

Description

A background job with job step ZVARI_1 is to be created. Selection screen data for ZVARI_1has to be specified, so that a temporary variant is created. In order to do that, the schedulerneeds to read the selection screen of the program first.

Prerequisites

A connection to an SAP system is established.
The corrections of note 1171295 have to be in the test system.
The audit level is set to the value 2, so that the use of the required functions can beverified in the XMI log (RZ15).

Realization

Use BAPI_XBP_READ_SELSCREEN to read the selection screen.
Use BAPI_XBP_JOB_ADD_ABAP_STEP to create the job step and the temporary variant.

Verification

Check XMI log via RZ15.
Check in the step data of SM37, if a new temporary variant has in fact been created

5. Criteria Manager Functions

5.1 Defining Criteria for the Event History

Description

Create and activate a criteria profile for an event history criteria type using the following logical condition: Log all SAP events excluding SAP_END_OF_JOB.

Prerequisites

A connection to an SAP system is established.
An XBP session has to be established through the XMI interface.

Realization

The profile should be created using the function modules

BAPI_CM_CRITERIA_SET
BAPI_CM_PROFILES_GET
BAPI_CM_PROFILE_ACTIVATE
BAPI_CM_PROFILE_CREATE
BAPI_CM_PROFILE_DEACTIVATE (optional)
BAPI_CM_PROFILE_DELETE (optional)

Verification

1. In the SAP Criteria Manager (transaction CRIT or program CRITERIA_MANAGER), the Criteria Profile for the Event History appears and is active.

2. The profile details show the correct values.

5.2 Defining Criteria for Job Interception

Description

Create and activate a criteria profile for an interception criteria type using the following logical condition: Intercept all jobs having job class B and a job name starting with TEST excluding TEST_123.

Prerequisites

A connection to an SAP system is established.
An XBP session has to be established through the XMI interface.

Realization

The profile should be created using the function modules

BAPI_CM_CRITERIA_SET
BAPI_CM_PROFILES_GET

BAPI_CM_PROFILE_ACTIVATE
BAPI_CM_PROFILE_CREATE

BAPI_CM_PROFILE_DEACTIVATE (optional)
BAPI_CM_PROFILE_DELETE (optional)

Verification

1. In the SAP Criteria Manager (transaction CRIT or program CRITERIA_MANAGER), the Criteria Profile for the job interception appears and is active.

2. The profile details show the correct values.

6 Event History Functions

6.1 SAP Event-Driven Scheduling with Control in an External Scheduler

Description

In this test scenario it shall be proven, that the external scheduler provides a mechanism to get aware of the fact, that an SAP batch event has been raised internally within the SAP system (e.g. via SM64).

Prerequisites

A connection to an SAP system is established.
An XBP session has to be established through the XMI interface.

Realization

For this test the event SAP_TEST with parameter 123 shall be used.

The external scheduler should show each occurrence of this event with parameter 123 (after it has been raised in the SAP system) in its user interface, or the external scheduler should perform a certain action after each occurrence of the event. This action can be e.g. starting a certain job in the SAP system, and this action would prove, that the external scheduler has become aware of the event.

For the value help of the event names, the function BAPI_XBP_EVENT_DEFINITIONS_GET has to be used.

The event history profile should be created using the function modules:

BAPI_CM_CRITERIA_SET
BAPI_CM_PROFILES_GET
BAPI_CM_PROFILE_ACTIVATE
BAPI_CM_PROFILE_CREATE
BAPI_CM_PROFILE_DEACTIVATE
BAPI_CM_PROFILE_DELETE (optional)

The synchronization should be done using the function modules:

BAPI_XBP_BTC_EVTHISTORY_GET
BAPI_XBP_BTC_EVTHIST_CONFIRM

Verification

1. Raise event SAP_TEST with parameter 123 (using transaction SM64 or program sapevt).
2. Check in the UI of the external scheduler or check, that the specified action (see above) has been performed.
3. Repeat points 1 and 2.

7 Job Interception

For the following tests, job interception XBP 3.0 has to be switched on. This is done by executing the program INITXBP2.
If there are some intercepted jobs in the SAP system from earlier testing, they have to be deleted.

Preparation:

In SM36 define two jobs TEST1 and TEST2 of job class B and choose start condition *immediate start*. According to the interception criteria defined earlier in 5.2 the two jobs must be intercepted.
Check their status in transaction SM37. It should be *intercepted* (in older releases it is displayed as *scheduled*).

7.1 Selecting all intercepted jobs

Description
Show all intercepted jobs.

Prerequisites
A connection to an SAP R/3 System is established.
An XBP session has to be established through the XMI interface.

Realization

Use BAPI_XBP_GET_INTERCEPTED_JOBS.

Test

The two jobs TEST1 and TEST2 should be displayed. Check their status also in transaction SM37.

7.2 Starting an intercepted job asap

Description
Start the job TEST1 as soon as possible.

Prerequisites
A connection to an SAP R/3 System is established.
An XBP session has to be established through the XMI interface.
The job TEST1 is managed within the external job scheduling system.

Realization

Use BAPI_XBP_JOB_START_ASAP.

Test

The job TEST1 should appear in transaction SM37 as released with a delay, ready, running, or finished (or even aborted).

8 Application Information Functions

8.1 Reading Application Log Content and Application Return Code

Description

Schedule a job that executes a program which is creates an application return code and writes entries into the application log. The application return code and the content of the logcan be read. Use BTCTESTN as a test program.

Prerequisites

A connection to an SAP system is established.
An XBP session has to be established through the XMI interface.

Realization

The job should be created using the function modules:
BAPI_XBP_JOB_OPEN
BAPI_XBP_JOB_ADD_ABAP_STEP (or BAPI_XBP_ADD_JOB_STEP);
as ABAP step use program BTCTESTN
BAPI_XBP_JOB_CLOSE

The application information can be read with the function modules:
BAPI_XBP_APPL_INFO_GET
BAPI_XBP_APPL_LOG_CONTENT_GET

Verification

1. Create a job with an ABAP step that creates an application return code and writes entries in the application log, for example with program BTCTESTN.
2. Read the application return information with BAPI_XBP_APPL_INFO_GET. This function module returns the application return code and, if available, a table of handles of the application log.
3. These handles can be used to read the content of the application log with functionmodule BAPI_XBP_APPL_LOG_CONTENT_GET.

9 Spool and Archiving Functions

9.1 Reading the Content of a Spool Request Specified by Number

Description

In this scenario, the content of a spool request that has been specified by its number is to beretrieved via the XBP interface.

Prerequisites

A connection to an SAP system is established.
The corrections of note 1171295 have to be in the test system.
The audit level is set to the value 2, so that the use of the required functions can beverified in the XMI log (RZ15).

Realization

1. Use BAPI_XBP_READ_SELSCREEN to read the parameters of report RSPO0015.
2. Use BAPI_XBP_JOB_ADD_ABAP_STEP to create a job with a temporary variant ofRSPO0015 with RUNS = 3 and PRINTER = LP01.
3. Use BAPI_XBP_JOB_DEFINITION_GET to retrieve the numbers of the spool requests created by the job.
4. Use BAPI_XBP_JOB_READ_SINGLE_SPOOL to read the content of the second spool request.

Verification

Check XMI log via RZ15.
Use UI of scheduling tool to display the contents of the spool list.

9.2 Passing an E-Mail Address as Recipient for the Spool List of aJob

Description

In this scenario, an e-mail address is to be passed as the recipient for the spool list createdby a background job.

Prerequisites

A connection to an SAP system is established.
The corrections of note 1171295 have to be in the test system.
The audit level is set to the value 2, so that the use of the required functions can beverified in the XMI log (RZ15).

Realization

Create a job and add an e-mail address as recipient to the job via function BAPI_XBP_JOB_CLOSE.

Verification

Check XMI log via RZ15.
Check with transaction SOST that a mail request has been created for the specifiedmail address.

10. Appendix

```
*&---------------------------------------------------------------------*
*& Report  ZMAKECHILD                                                  *
*&                                                                     *
*&---------------------------------------------------------------------*
*&                                                                     *
*&                                                                     *
*&---------------------------------------------------------------------*

REPORT  ZMAKECHILD  .

data: jobname  like tbtcjob-jobname.
data: jobcount like tbtcjob-jobcount.

data: report   like sy-repid.

data: start_day  like sy-datum.
data: start_time like sy-uzeit.

data: rel like BTCH0000-CHAR1.
data: cnt(2) type n.


start_day = sy-datum.
start_time = sy-uzeit + 60.

DO 4 times.

write sy-index to cnt.
concatenate 'CHILD_' cnt into jobname.

* job_open creates a job-header and returns the
* jobcount, which is needed in the following calls
* of job_submit and job_close.

CALL FUNCTION 'JOB_OPEN'
  EXPORTING
    JOBNAME               = jobname
    jobclass              = 'C'
 IMPORTING
   JOBCOUNT              =  jobcount
 EXCEPTIONS
   CANT_CREATE_JOB       = 1
   INVALID_JOB_DATA      = 2
   JOBNAME_MISSING       = 3
   OTHERS                = 4
        .
IF SY-SUBRC <> 0.
   write: / 'Error in job_open, sy-subrc =', sy-subrc.
   exit.
ENDIF.


* job_submit must be called for each job step.
if sy-index = 1.
   report = 'BTCLOOP'.
```

```
else.
   report = 'RSWATCH0'.
endif.

CALL FUNCTION 'JOB_SUBMIT'
  EXPORTING
    AUTHCKNAM                   = sy-uname
    JOBCOUNT                    = jobcount
    JOBNAME                     = jobname
    REPORT                      = report
 EXCEPTIONS
   BAD_PRIPARAMS                = 1
   BAD_XPGFLAGS                 = 2
   INVALID_JOBDATA              = 3
   JOBNAME_MISSING              = 4
   JOB_NOTEX                    = 5
   JOB_SUBMIT_FAILED            = 6
   LOCK_FAILED                  = 7
   PROGRAM_MISSING              = 8
   PROG_ABAP_AND_EXTPG_SET      = 9
   OTHERS                       = 10
          .

IF SY-SUBRC <> 0.
  write: / 'Error in Job_Submit, sy-subrc =', sy-subrc.
  exit.
ENDIF.


* job_close specifies the global job data, e.g. the start condition

CALL FUNCTION 'JOB_CLOSE'
  EXPORTING
    JOBCOUNT                 = jobcount
    JOBNAME                  = jobname
    SDLSTRTDT                = start_day
    SDLSTRTTM                = start_time
  IMPORTING
    JOB_WAS_RELEASED                 = rel
  EXCEPTIONS
    CANT_START_IMMEDIATE             = 1
    INVALID_STARTDATE                = 2
    JOBNAME_MISSING                  = 3
    JOB_CLOSE_FAILED                 = 4
    JOB_NOSTEPS                      = 5
    JOB_NOTEX                        = 6
    LOCK_FAILED                      = 7
    OTHERS                           = 8
          .
IF SY-SUBRC <> 0.
   write: / 'Error in Job_Close, sy-subrc =', sy-subrc.
   exit.
ENDIF.

ENDDO.
```