

Connecting External Job Management Systems to the SAP NetWeaver AS ABAP CCMS Job Scheduling System



Interface

SAP

XBP 3.0

External Interface for Background Processing

Version 1.1
23.11.2009

Copyright

© Copyright 2008 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, Informix, i5/OS, POWER, POWER5, OpenPower and PowerPC are trademarks or registered trademarks of IBM Corporation.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries. Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group. Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

MaxDB is a trademark of MySQL AB, Sweden.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

Contents

1	INTRODUCTION	14
2	THIS DOCUMENT - AN OVERVIEW	14
3	THE FUNCTION OF EXTERNAL INTERFACES	15
4	A SHORT INTRODUCTION TO AS ABAP BACKGROUND PROCESSING	15
4.1	Motivation	15
4.2	Concept	16
4.2.1	Creating Jobs	17
4.2.2	Releasing Jobs	18
4.2.3	Starting Jobs (Ready and Active)	19
4.2.4	Ending a Job (Canceled or Finished)	19
4.2.5	Intercepting Jobs	20
4.2.6	Parent/Child Functionality	21
4.2.7	Confirming Jobs	23
4.2.8	Consuming Raised Events	23
4.2.9	Monitoring Performance	24
4.2.10	Obtaining Application Information	24
4.3	Architecture of the AS ABAP Job Scheduling System	25
4.3.1	Job Administration in the Database	25
4.3.2	The Job scheduler	25
4.3.3	The Job Starter and the Job Runtime Environment	26
4.3.4	The Job Log	26
4.3.5	Job Output	26
5	THE EXTERNAL INTERFACE CONCEPT	27
5.1	Range of Interfaces	27
5.2	Naming Conventions	27
5.3	Technical Basics	28
5.3.1	XMI Monitor: External Access	28
5.3.2	RFC Remote Function Call	28
6	XBP - EXTERNAL JOB SCHEDULING INTERFACE (EXTERNAL JOB-API)	31
6.1	What Is Required of the Interface	31
6.2	XBP Interface - Description	32
7	XBP REFERENCE MANUAL	36
7.1	Requirements for Using the XBP Interface	36
7.1.1	Logging on to the AS ABAP System with an External Job Management System	36
7.1.2	External Job Management System - Logging Off	38

7.2	Defining Jobs	<u>39</u>
7.2.1	Opening Jobs	<u>39</u>
7.2.2	Assigning an ABAP Program to a Job Step	<u>40</u>
7.2.3	Assigning an External Program to a Job Step	<u>44</u>
7.2.4	Closing Job Definitions	<u>45</u>
7.2.5	Reading Job Definitions from the AS ABAP System	<u>48</u>
<hr/>		
7.3	Starting a Job	<u>52</u>
7.3.1	Starting Jobs Immediately	<u>52</u>
7.3.2	Starting Jobs as Soon as Possible	<u>53</u>
7.3.3	Triggering an Event from Outside	<u>54</u>
<hr/>		
7.4	Copying Jobs	<u>55</u>
<hr/>		
7.5	Controlling Jobs	<u>57</u>
7.5.1	Modifying Job Global Data	<u>57</u>
7.5.2	Aborting a Job	<u>58</u>
7.5.3	Deleting a Job	<u>60</u>
<hr/>		
7.6	Modifying Steps in a Job	<u>61</u>
7.6.1	Modifying a Job Step Containing an ABAP Program	<u>61</u>
7.6.2	Modifying a Job Step Containing an External Program	<u>65</u>
<hr/>		
7.7	Adding, Changing, and Deleting Job Steps via XMI	<u>67</u>
7.7.1	Adding a Step to a Job via XMI	<u>67</u>
7.7.2	Changing and Deleting a Job Step via XMI	<u>70</u>
<hr/>		
7.8	Intercepting and Confirming Jobs	<u>74</u>
7.8.1	Getting Intercepted Jobs	<u>74</u>
7.8.2	Confirming Jobs	<u>76</u>
7.8.3	Modifying the Criteria Table for Interception	<u>78</u>
<hr/>		
7.9	Finding, Controlling, and Modifying Job Monitor Data	<u>80</u>
7.9.1	Determining the Status of a Job	<u>80</u>
7.9.2	Determining the Status of a Job List	<u>82</u>
7.9.3	Reading Job Logs	<u>83</u>
7.9.4	Reading the Spool List of a Job	<u>86</u>
7.9.5	Getting Information on a Particular Spool List	<u>88</u>
7.9.6	Reading a Particular Spool List	<u>89</u>
7.9.7	Checking the Status of a Job	<u>90</u>
7.9.8	Selecting Jobs	<u>93</u>
7.9.9	Determining the Number of Jobs with Particular Job Names	<u>95</u>
7.9.10	Obtaining Key Job Parameters from Job Headers and Steps	<u>96</u>
7.9.11	Determining Job Children	<u>97</u>
7.9.12	Determining Parent/Child Relation	<u>99</u>
7.9.13	Reading and Changing Intercept Status and Parent/Child Relation	<u>102</u>
7.9.14	Obtaining Application Information	<u>103</u>
7.9.15	Monitoring Performance	<u>105</u>
7.9.16	Consuming Raised Events from Event History	<u>107</u>
7.9.17	Configuring Profiles and Criteria using the Criteria Manager Interface	<u>111</u>
7.9.18	Reading a Particular Spool List as PDF	<u>117</u>
7.9.19	Reading a Particular Binary Spool List	<u>118</u>
<hr/>		
7.10	Searching with Wildcards	<u>119</u>
7.10.1	Searching for ABAP Reports	<u>119</u>
7.10.2	Searching for External Commands	<u>120</u>
7.10.3	Searching for Output Devices	<u>121</u>
7.10.4	Searching for Print Formats	<u>123</u>
7.10.5	Searching for Archive Parameters	<u>124</u>
7.10.6	Searching for Batch Events	<u>125</u>
<hr/>		

7.11	General Help Functions	<u>126</u>
7.11.1	Showing All Defined Variants of an ABAP Program	<u>126</u>
7.11.2	Determining Current Resources for Jobs in the AS ABAP System	<u>128</u>
7.11.3	Checking Available Job Resources at a Particular Time on a Server	<u>129</u>
7.11.4	Checking Available Job Resources at a Particular Time in the Whole SAP System.	<u>131</u>
7.11.5	Reading SAP Factory Calendars	<u>132</u>
7.11.6	Reading SAP Holiday Calendars	<u>133</u>
7.12	Variant Functions	<u>134</u>
7.12.1	Creating a Variant	<u>134</u>
7.12.2	Changing a Variant	<u>135</u>
7.12.3	Copying a Variant	<u>136</u>
7.12.4	Reading Variant Data	<u>138</u>
7.12.5	Deleting a Variant	<u>139</u>
7.12.6	Reading Selection Screen of an ABAP Program	<u>140</u>
7.12.7	Reading Free Selections of an ABAP Program	<u>141</u>
7.13	Synchronizing Jobs	<u>142</u>
7.14	Setting Spool List Recipients.	<u>144</u>
7.14.1	Reading SAP Users	<u>144</u>
7.14.2	Reading SAP Office Distribution Lists	<u>145</u>
8	APPENDIX	<u>146</u>
8.1.1	BAPI Return Structure	<u>146</u>
8.1.2	Message IDs and Their Meaning	<u>147</u>
8.1.3	Document Type Definition for Defining Profiles and Criteria for Event History	<u>150</u>
8.1.4	Language Key Mapping	<u>155</u>
9	INDEX	<u>161</u>

Release Information for XBP Version 3.0 - 2008

XBP 3.0 is an enhancement of XBP 2.0. This means that XBP 2.0 is a subset of XBP 3.0. Therefore, customers who use XBP 2.0 can install XBP 3.0 without any change of functionality in their current applications. XBP 3.0 will be available for SAP Basis releases 7.0 and higher. For the releases that are already on the market, XBP 3.0 will be delivered by Support Packages.

The following tables give you an overview of functions modules that are new in XBP 3.0 or that have been enhanced for XBP 3.0. All these functions are described in detail in the XBP Reference Manual (chapter 7 of this document).

1. Consuming Raised Events (new):

Function module	Feature
BAPI_XBP_BTC_EVTHISTORY_GET	Reading events from the log of raised events (event history).
BAPI_XBP_BTC_EVTHIST_CONFIRM	Changing the status of events from NEW to CONFIRMED.
BAPI_XBP_EVENT_DEFINITIONS_GET	Reading definitions of batch events.

2. Configuring Criteria (new)

Function module	Feature
BAPI_CM_CRITTYPES_GET	Getting a list of available criteria types.
BAPI_CM_PROFILE_ACTIVATE	Activating a criteria profile.
BAPI_CM_PROFILE_CREATE	Creating a criteria profile.
BAPI_CM_PROFILE_DELETE	Deleting an existing criteria profile.
BAPI_CM_PROFILES_GET	Getting a list of profiles.
BAPI_CM_PROFILE_DEACTIVATE	Deactivating an active profile.
BAPI_CM_CRITERIA_GET	Getting criteria in XML format.
BAPI_CM_CRITERIA_SET	Importing criteria from an XML source.

3. Monitoring Performance (new)

Function module	Feature
BAPI_XBP_BTC_STATISTIC_GET	Getting statistic records for a list of jobs.

4. Obtaining Application Information (new)

Function module	Feature
BAPI_XBP_APPL_INFO_GET	Getting the handles of application logs for a particular job or job step.
BAPI_XBP_APPL_LOG_CONTENT_GET	Getting the content of an application log.

5. Getting information about and reading a particular spool list (new)

Function module	Feature
BAPI_XBP_JOB_READ_SINGLE_SPOOL	Reading a particular spool list of a job that has been run.
BAPI_XBP_GET_SPOOL_ATTRIBUTES	Getting information about a particular spool list.

6. Searching for archive parameters (new)

Function module	Feature
BAPI_XBP_GET_ARCHIVE_OBJECTS	Returning SAP Objects and Archive Objects that are defined in a system.

7. Setting a spool list recipient (new)

Function module	Feature
BAPI_XBP_GET_USER_LIST	Reading the SAP users in blocks.
BAPI_XBP_GET_DL_LIST	Getting the list of distribution lists.

8. Selecting all jobs from the SAP system from a certain time period (new)

Function module	Feature
BAPI_XBP_SYNCHRONIZE_JOBS	Reading all SAP jobs, which have been created after a certain point of time. Helps to synchronize the job database of the external job scheduler with the SAP job database.

9. Simplified variant handling (new)

Function module	Feature
BAPI_XBP_VARIANT_CREATE	Creating a variant.
BAPI_XBP_VARIANT_CHANGE	Changing a variant.
BAPI_XBP_VARIANT_COPY	Copying a variant.
BAPI_XBP_VARIANT_DELETE	Deleting a variant.
BAPI_XBP_VARINFO	Reading the data of all variants of an ABAP program.
BAPI_XBP_READ_SELSCREEN	Reading the selection screen of an ABAP program.
BAPI_XBP_GET_FREE_SELECTIONS	Reading the free selections of an ABAP program.

10. Enhanced basic functionality (BAPIs):






Function module	Feature
BAPI_XBP_JOB_CLOSE	This function module for passing a spool list recipient has a new RECIPIENT structure of the RECIPIENT_OBJ input parameter. It allows the external scheduler to pass a spool list recipient and its attributes in plain text.
BAPI_XBP_JOB_DEFINITION_GET BAPI_XBP_JOB_READ	Both function modules have been enhanced with output parameters JOBLG_ATTR and SPOOL_ATTR that return more information about the job log and the spool list(s) created by a job. This allows the external scheduler to know the size of the job log or spool list it has to transfer.
BAPI_XBP_JOB_ADD_ABAP_STEP	This function module can now be supplied with an internal table for specifying free selections and with an internal table for specifying 'normal' selections without referring to an explicit variant.
BAPI_XBP_JOB_SPOOLLIST_READ_20	This function module for reading spool lists has a new table parameter which is narrower than the original one. Also, partial reading of spool lists is now supported
BAPI_XBP_JOB_READ_SINGLE_SPOOL (new)	This function allows to read the contents of a spool list specified by number.
BAPI_XBP_NEW_FUNC_CHECK	Support of new interception mode using Criteria Manger.
BAPI_XBP_OUTPUT_DEVICE_SEARCH	This function module can now search for a certain device type as well as for output devices whose definitions have been changed after a certain date.
BAPI_XBP_JOB_JOBLOG_READ	With the parameter LINES and DIRECTION it is possible to read the first or last n lines of a job log.
BAPI_XBP_GET_SPOOL_AS_PDF (new)	Convert given spool list to PDF and return PDF data.
BAPI_XBP_READ_SPOOL_BIN (new)	Return binary spool request as raw data stream.

What's new in this document version?

Version 1.1:

Description	Chapter	See page
Partial transfer of spool lists	7.9.4 Reading the Spool List of a Job	86
Partial transfer of spool lists	7.9.6 Reading a Particular Spool List	89
Display server groupname of a job	7.9.10 Obtaining Key Job Parameters from Job Headers and Steps	96
Convert spool list to PDF	7.9.18 Reading a Particular Spool List as PDF	117
Get binary spool list	7.9.19 Reading a Particular Binary Spool List	118

Symbols

Symbol	Meaning
	Warning
	Example
	Tip
	Recommendation
	Syntax

1 Introduction

This document deals with the connection of an external job management system, often called an *external scheduler*, to SAP NetWeaver Application Server ABAP (AS ABAP). By external job management, we mean software which allows jobs to be scheduled, run, and monitored from outside SAP NetWeaver AS ABAP.

For this purpose, SAP has defined an open generic interface. This is called XBP, which stands for eXternal interface for job Background Processing. XBP is one of a range of open interfaces which SAP intends to make available in the future for system management tasks. The SAP system's internal CCMS (Computing Center Management System), with this range of interfaces, offers support to software manufacturers by allowing integration with existing system administration tools.

2 This Document - An Overview

We assume that the reader already has a certain degree of knowledge of AS ABAP. Also, he or she should be familiar with terms like application server, dispatcher, scheduler, and work process in relation to SAP AS ABAP systems. Additionally, he or she should be familiar with the Remote Function Call (RFC).

This overview helps you to find the topics relevant for you:

Chapter	Contents
3	General description of the external interface.
4	Short introduction to SAP NetWeaver AS ABAP background processing.
5	Description of the external interface concept.
6	Actual description of the XBP interface; the motivation behind it, its technical background and a functional overview.
7	Technical description of the function modules. This is provided to help with the implementation of external agents.

3 The Function of External Interfaces

The motivation for the development of external interfaces arose from the desire to let SAP installations - especially large ones - be overseen by other software vendors' tools.

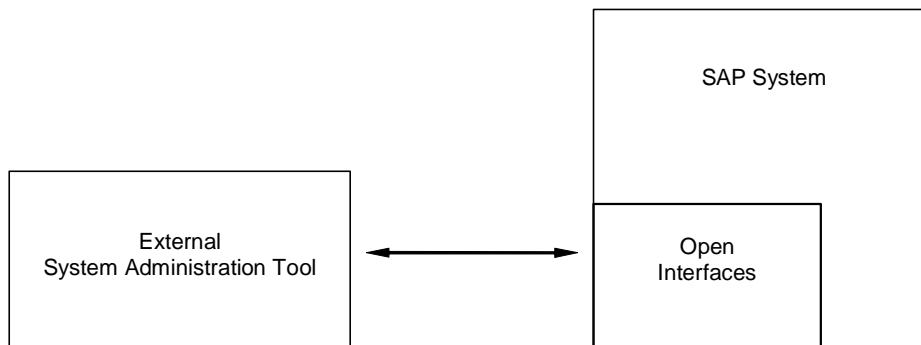


Fig. 3.1: Location of external interfaces

External interfaces allow you to integrate the AS ABAP system simply and seamlessly into both local administrative tools and business-wide system management infrastructures. This integration should not and cannot completely replace the use of CCMS. Complex and security-critical system control tasks will always need to be carried out by CCMS. Furthermore, CCMS basic functionality is expected and required by AS ABAP applications. To explain by example: even if you use an external scheduling tool for background jobs, the internal job system in AS ABAP is still required to carry out the background jobs generated by applications such as AS ABAP archiving or the Workbench.

The aim of the integration is to allow the customer a homogenous information infrastructure. The aim of the interface is to facilitate the flow of information between AS ABAP and external tools.

In summing up, we could say that external tools provide additional flexibility to complement the existing basic functionality of CCMS.

4 A Short Introduction to AS ABAP Background Processing

First, we need to explain why background processing has a place in a dialog-oriented standard application. What is a 'job' within AS ABAP, and how does the system carry it out?

We intend to demonstrate the concept using an example. Afterwards, we will introduce the architecture of the background processing system.

4.1 Motivation

SAP NetWeaver AS ABAP is, above all, an interactive system. In other words, the vast majority of tasks are carried out in dialog with the user. However, there are also good reasons for the inclusion of a background processing system in AS ABAP.

Besides the tasks carried out in dialog, there are numerous tasks processing large amounts of data and requiring lots of performance that do not need user interaction. With the help of the background processing system, such tasks are normally scheduled for times when no users are working in the system (nights, weekends), in order to avoid resource conflicts with the dialog users.

At the scheduled time, these tasks are started by the background processing system and executed without user interaction, even without a connection to any frontend server.

This mechanism is especially useful for tasks that have to be carried out periodically, for example each week or each month. In the background processing system, these tasks – including the period - have to be specified once only. No further action is required from the user with respect to regular execution.

4.2 Concept

A task executed by the background processing system is called a 'background job', 'batch job', or simply 'job'. Technically speaking, a job in the SAP background processing system executes one or more ABAP programs or calls on the OS level, which are referred to as the job steps. The job steps are executed sequentially in the order of their definition.

So, roughly speaking, one can say that defining a job consists of defining the job steps and the job header (start conditions, target server, and other data).

Jobs are identified by their name (Ex.: `PAYROLL_RUN`). However, since these names are not unique if the same application job is repeated, jobs also have job numbers, which ensure that they have a unique identification

In its life cycle, a job always has exactly one of the following statuses:

Status	Description
Scheduled	The job has steps but no start conditions have been defined.
Released	The job definition is complete and the job is waiting for the start conditions to be fulfilled.
Ready	The job is in its start phase, which normally takes only a fraction of a second. This status is of no interest for the end user. However, it is useful for error analysis by SAP support.
Active	The job is being executed.
Finished	The job is finished without errors.
Cancelled	The job execution was terminated due to an error.

Additionally, it is possible to *intercept* jobs. 'Interception' means that the jobs are not started at the moment when their start conditions are fulfilled, but deactivated and restarted later. However, note that *intercepted* is not really a new status in the SAP background processing system (see chapter 4.2.5 'Intercepting jobs').

Figure 4.1 shows the chain of these statuses:

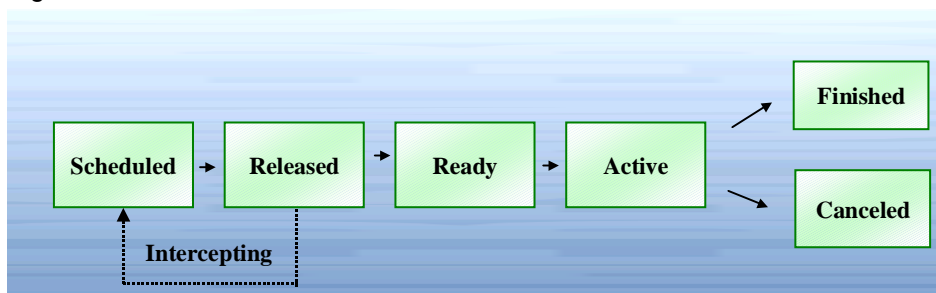


Fig. 4.1: Status chain for jobs.

In the next sections, we follow the life cycle of a job systematically. Figure 4.2 shows an example of the activities which can cause a job to change status. A job 'scheduled' by a program is 'released' by a dialog. It then proceeds through the 'ready' and 'active' statuses with help from AS ABAP system programs. If everything has gone according to plan, the job status moves on to 'finished'. A program error will lead to a final status of 'canceled'. XBP also allows the external scheduler to intercept jobs and to

reschedule and restart them later. This feature allows you to prioritize jobs dynamically as described later in detail.

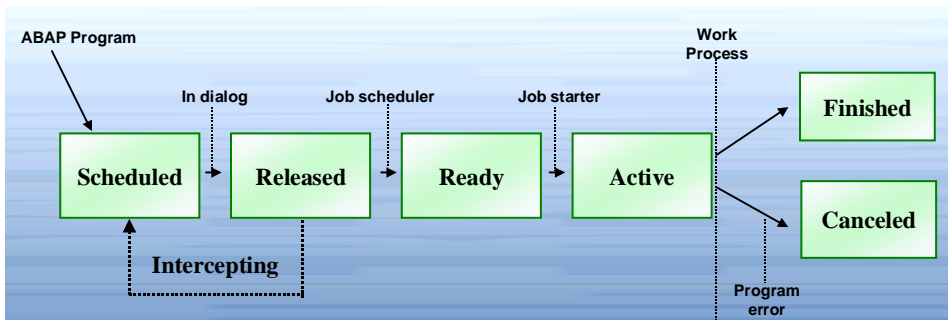


Fig. 4.2: Status chain for jobs with possible causes of status change.

You can see the current status of a particular job in the AS ABAP using the job overview (transaction SM37).

4.2.1 Creating Jobs

A job definition consists of a list of steps and administrative information known as job header. The job header contains the start conditions, job class, and target server.

If a job is created without start conditions, the job receives the status 'scheduled'. If a job is defined completely, the job receives the status 'released'.

Within an SAP system, there are two ways of creating a job:

1. In a dialog (Transaction SM36, or *Tools* → *Administration* → *Jobs* → *Define Job*) you enter the job name, job class and, if necessary, a target machine. Next, you enter a list of steps and a **start time**, if required.
2. New jobs are created from ABAP programs using the JOB_OPEN, JOB_SUBMIT and JOB_CLOSE function modules, which are part of the Batch API. You have the same parameters and degree of freedom here as in the dialog.

In each of these cases, the job number is created by the system itself to ensure that the job has a unique identification. Figure 4.3 shows the specification of a newly created job, with its name, number, status and steps.

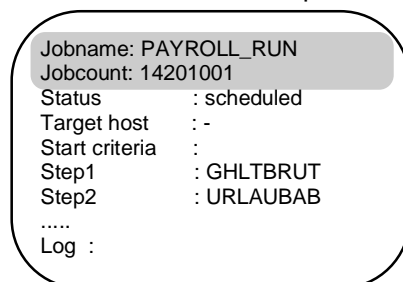


Fig. 4.3: A created job from the system's point of view (greatly simplified)

If **no** target machine is specified for the job, the system decides by itself on which application server the job is to be carried out. This is the normal procedure recommended by SAP, because the system can then carry out its own load balancing. You should only specify the target machine if that server has particular resources necessary for the job processing.

4.2.2 Releasing Jobs

A job is released (set to status 'released') as soon as it has been completely created. This means that the job has to have at least one step and a start condition. The SAP background system offers several types of start conditions, which can be specified by the user.

Note that different start conditions cannot be combined by 'and' or 'or'. Exactly one of the following start conditions has to be specified for each job:

Start condition	Description
Immediate	A job is executed as quickly as possible.
Date/ time	A job is started when its start date and time are reached.
Event	A job is started when a particular event is triggered in the system or by a program at operating system level. For example, a job can be started when data for processing is imported to a server by file transfer.
Preceding job	A job is started once a particular preceding job has run.
Change of operation mode	A job starts once the AS ABAP system has switched into a particular operation mode.
Start on working day	A job is started when a particular working day of the month is reached, for example the last working day of a month.

Once you have released a job by setting a start condition, the **Job scheduler** in the AS ABAP system becomes responsible for the future progress of the job.

For some of the above start criteria, you can also specify that the job should recur. Thus you can ensure that a new job with the same name, but a different job number for each repetition, is released, for example, each time a particular event occurs, or every day from today. A job with such a start condition is called a periodic job.

When selecting start criteria, you can use various calendars, either pre-defined or user-defined. This allows you a still greater degree of freedom, by allowing you to specify deviations in the frequency of periodic jobs (e.g. carry out periodically, but not on Public Holidays).

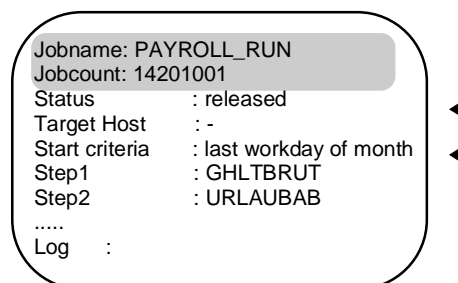


Fig. 4.4: A released job from the point of view of the system (greatly simplified).

Figure 4.4 shows that as well as the status changing, the start criteria have been laid down in the job description.

4.2.3 Starting Jobs (Ready and Active)

Within the system, the **Job scheduler** surveys the start conditions of the released jobs. For each job, the scheduler monitors whether its start condition is fulfilled.



Note that the term 'job scheduler' is somewhat simplified. In fact, there are different kinds of job scheduler (time-based and event-based). And, in a system with several servers the number of job schedulers varies. However, the task is the same for all. They monitor the start conditions of the released jobs and start the released jobs once their start conditions are fulfilled. Therefore, in this documentation 'job scheduler' is used without differentiation.

In the SAP system, there is a special type of work process, the background work process. This work process is reserved for executing jobs. Of course, the job scheduler can only start a job if there is a free background work process.

Technically, a job start functions as follows:

When the start condition of the job is fulfilled, the job scheduler checks if there is a free background work process available. If a free background work process is available, the job scheduler sets the job to the status 'ready' and sends a job start message, which is assigned to a free background work process by the system. The background work process finally sets the job to the status 'active' and starts executing it. The component of the background work process responsible for this is called the 'job starter'.

```
Jobname: PAYROLL_RUN
Jobcount: 14201001
Status      : active
Target host : -
Start criteria : last workday of month
Step1      : GHLTBRUT
Step2      : URLAUBAB
.....
Log       : JOBLGX142010X1234
```

Fig. 4.5: An active job from the point of view of the system (greatly simplified).

If a free background work process is not available at the time when the start condition is fulfilled, the job scheduler monitors the job until a free work process becomes available. Then the job scheduler sends the job start message as described above.

The status 'ready' is a technical status and of no interest for the user. Normally, a job has this status only for a fraction of a second and the user does not even see this status.

As figure 4.5 shows, the job definition has now gained a link to a **job log**.

4.2.4 Ending a Job (Canceled or Finished)

A job logs its steps in a job log, which is visible to the user. If everything runs according to plan and no errors occur, the job's status changes to 'finished'. This can be seen both in the job log and the dialog in transaction SM37.

If an ABAP error or any other interruption occurs at runtime, an error message is recorded in the job log, and the job is terminated.

```
Jobname: PAYROLL_RUN
Jobcount: 14201001
Status      : finished
Target host : -
Start criteria : last workday of month
Step1      : GHLTBRUT
Step2      : URLAUBAB
.....
Log       : JOBLGX142010X1234
```

Fig. 4.6: A finished job from the point of view of the system (greatly simplified).

Figure 4.6 shows the final view of the job. The job status is 'finished'. Further information can be seen in the log.

4.2.5 Intercepting Jobs

Intercepting jobs means that at the moment when the start conditions of the jobs are fulfilled, the jobs are set back to the status 'scheduled' and receive a special attribute. By calling an XBP function, the external scheduler can receive a list of all intercepted jobs. Although 'intercepted' is not really a new status (as described above, an intercepted job has the status 'scheduled' and a specific attribute), it is presented to XBP 2.0 as a new status. If logged on with version 2.0 or higher, the function `BAPI_XBP_STATUS_GET` will return an I for an intercepted job, whereas it will return P, if logged on with version 1.0.

Of course, it is not intended to subject jobs to interception in general. The user can define criteria in the new table `TBCICPT1` (client, job name, job creator including wildcards), and only the jobs that match these intercept criteria are intercepted. For instance, a table entry (100, babu*, *) means that all jobs created in client 100 by users beginning with babu are intercepted. These table entries can be added, changed and deleted with an XBP function.

Note, that the decision as to whether or not a job has to be intercepted, is made by the SAP system at the moment, when the start condition is fulfilled. This means that at that moment the job data are matched against the table contents. A job is intercepted if it matches the intercept criteria at that moment.



The Administrator might want to intercept all jobs of certain users or with certain job names on weekends when long-running and time critical batch jobs are executed. In this case interception provides dynamic job prioritization.

However, the intercept feature may not be used by all customers. Therefore, this functionality can be globally switched off completely. This has the advantage that internal calls of subroutines in the runtime system are avoided.

Functionalities that can be switched on are switched off initially. As of XBP 2.0, you can switch the new functions globally on and off with the ABAP program `INITXBP2` or using `BAPI_XBP_NEW_FUNC_CHECK` function module.



Remember to run `INITXBP2` before using the job interception function.

More information about the technical implementation of the status 'intercepted'

All XBP 2.0 or XBP 3.0 functions that return status information of a job will return an 'I' (= Intercepted) for intercepted jobs. But that does not mean that there will be the value 'I' in the `TBTCO` status column of the `TBTCO` database table. The field `STATUS` in the `TBTCO` table does not contain an 'I', but a 'P'. Only the additional attribute 'intercepted' in the table `TBTCCNTXT` makes it clear that we are dealing with an intercepted job.

Attribute 'intercepted-confirmed'

In order to know if there are any intercepted jobs, the external scheduler calls a function at short intervals (`BAPI_XBP_GET_INTERCEPTED_JOBS`). To prevent this function from returning the same intercepted jobs again and again, the scheduler can confirm a list of intercepted jobs. Confirmation means that the scheduler informs the AS ABAP system that it already knows these intercepted jobs, and that a subsequent call of `BAPI_XBP_GET_INTERCEPTED_JOBS` does not have to return these jobs again.

The confirmation of a list of intercepted jobs is done by calling of `BAPI_XBP_SPECIAL_CONFIRM_JOB`, which sets these jobs to status 'intercepted-confirmed'. However, in some situations (such as after a breakdown) it might be useful to get a list of all intercepted jobs (including the confirmed ones). For this purpose, the function `BAPI_XBP_GET_INTERCEPTED_JOBS` has a special indicator. You can find detailed information in the technical description of these function modules later in this document.

Treatment of periodic intercepted jobs

If a job that matches the interception criteria is periodic, the AS ABAP rescheduling mechanism applies.

This means that if a periodic job is set to status 'intercepted', the AS ABAP system creates the successor immediately after intercepting the predecessor. The successor, of course, has the status 'released'. If the start condition of the successor is fulfilled, it will be set to 'intercepted', and its successor is created, and so on.

Function modules for the status 'intercepted':

BAPI_XBP_MODIFY_CRITERIA_TABLE: A function module for adding and modifying the table with the intercept criteria

BAPI_XBP_GET_INTERCEPTED_JOBS: A function module for returning jobs with status 'intercepted'

BAPI_XBP_SPECIAL_CONFIRM_JOB: A function module for setting special types of confirmation for a list of jobs, such as for intercepted jobs

BAPI_XBP_NEW_FUNC_CHECK: A function module for reading and changing the status of the intercept function and the parent/child functionality

BAPI_XBP_JOB_START_IMMEDIATELY/BAPI_XBP_JOB_START_ASAP: Additional functions for starting intercepted and released jobs

BAPI_XBP_JOBLIST_STATUS_GET: This function module also returns the status "intercepted"

4.2.6 Parent/Child Functionality

In general, a business process that is carried out by a job, or rather, by a collection of jobs, does not only consist of static jobs, which are known in advance and shown right away in SM37. It also comprises jobs that are created at runtime by the static jobs, for example, to dynamically distribute workload. A job that is released by another job is called a child job, and the releasing job is called a parent job.

For a job scheduling system, it is important to know about the existence and the current status of the child jobs of a certain parent job, because in the internal logic of many applications, a parent job is considered as 'finished' only if the parent job itself **and** its child jobs are finished.

By using XBP functions modules, the external scheduler can find out whether or not a job has child jobs. The SAP background processing system stores the parent/child data of jobs automatically and offers functions to access these data. XBP offers functions to access the parent/child data of jobs.

Function modules for parent/child jobs.

- BAPI_XBP_JOB_CHILDREN_GET: A function module returning a list of all child jobs of a certain job.
- BAPI_XBP_NEW_FUNC_CHECK: A function module for reading, setting, and clearing the status of the parent/child functionality.
- BAPI_XBP_JOB_PARENT_CHILD_INFO: A function module returning information as to whether the job is a child or a parent and some other useful information.
- BAPI_XBP_JOB_STATUS_GET: A function that returns also the parent/child information for a single job
- BAPI_XBP_SPECIAL_CONFIRM_JOB: As is the case with intercepted jobs, it is also possible to confirm child jobs. This has the effect that confirmed jobs are not returned anymore by subsequent calls of BAPI_XBP_JOB_CHILDREN_GET.
- BAPI_XBP_JOBLIST_STATUS_GET: A function module receiving a list of jobs, for example the list of all child jobs of a certain job, and completing the list by the addition of status information for each job and a flag that indicates, if a job has child jobs.

However, the parent/child feature may not be used by all customers. If a customer does not use this feature, there is not need to write the parent/child information into the database. So, this feature can be globally switched off with the program **INITXBP2**.



Remember to run INITXBP2 before using the parent/child functionality.

Attribute 'child job-confirmed':

To find out if a job has child jobs, the external scheduler calls a function at short intervals (BAPI_XBP_JOB_CHILDREN_GET). To prevent this function from returning the same child jobs again and again, the scheduler can confirm a list of child jobs. Confirmation means that the scheduler informs the **SAP** system that it already knows these child jobs, and that a subsequent call of BAPI_XBP_JOB_CHILDREN_GET does not have to return these jobs again.

The confirmation of a list of child jobs is done by calling BAPI_XBP_SPECIAL_CONFIRM_JOB, which sets these jobs to status 'child-confirmed' (for more details see the chapter below). However, in some situations (such as after a breakdown) it might be useful to get a list of all child jobs (including the confirmed ones). For this purpose, the function BAPI_XBP_JOB_CHILDREN_GET has a special indicator. You can find detailed information in the technical description of these function modules later in this document.

4.2.7 Confirming Jobs

The concept of confirming jobs is even wider than already described in the context of interception and parent/child jobs. There are three XBP functions for job selection:

BAPI_XBP_JOB_SELECT for general job selection

BAPI_XBP_JOB_CHILDREN_GET for child job selection

BAPI_XBP_GET_INTERCEPTED_JOBS for the selection of intercepted jobs

These functions are normally called at intervals by the external job scheduler and return a list of jobs. If you do not want the system to return the same jobs over and over again, you can confirm them. Confirmation means that the scheduler informs the SAP system that it already knows these jobs, and that a subsequent call of the selection function module does not have to return these jobs again.

There are two types of confirmation:

General: With general confirmation, the job scheduler confirms that it knows a job in general. Jobs are confirmed generally with BAPI_XBP_CONFIRM_JOB. When you use any of the three selection functions, the generally confirmed jobs are not returned if the corresponding indicator (parameter name: SELECTION) is set appropriately.

Special: With special confirmation, the job scheduler confirms that it knows that a job has certain characteristics, for example if a job is an intercepted job or a child job. Child jobs and intercepted jobs are confirmed with BAPI_XBP_SPECIAL_CONFIRM_JOB. When you use BAPI_XBP_JOB_CHILDREN_GET or BAPI_XBP_GET_INTERCEPTED_JOBS the specially confirmed jobs are not returned again, if the corresponding indicator is set appropriately.

However, in some situations (such as after a breakdown) it might be useful to get a list of all intercepted or child jobs (including the confirmed ones). This function has a special indicator for this purpose.

4.2.8 Consuming Raised Events

The external scheduler can use events raised in the AS ABAP system as a start condition for jobs.

All the events that match certain criteria and were received and processed in the AS ABAP system are stored in a log called event history (EH). Event history lists all the received events whether or not they are defined in the system or are processed by the event scheduler.

EH enables the external scheduler to consume, or read raised events. Additionally, the external scheduler can confirm, or mark the events it has read.

- Reading raised events

By calling the BAPI_XBP_BTC_EVTHISTORY_GET function, the external scheduler can obtain a list of the events that were raised in the SAP system.

- Confirming raised events

EH keeps track of the status of events, showing whether the external scheduler has marked the event as read, that is, whether it has confirmed them. By confirming events, the external scheduler can avoid reading the same events in EH more than once.

All events are logged in the event history with status NEW. When the external scheduler has read the events in EH, it can change their status to CONFIRMED by calling the BAPI_XBP_BTC_EVTHIST_CONFIRM function.

Confirming an event is optional. An event in status NEW can remain in this status even after information about it has been polled.

4.2.8.1 Configuring Criteria for Raised Events

By default, the event history logs all events except those with the name SAP_END_OF_JOB. To make the event history log events different from the default setting, or to prevent event history from growing too big, the external scheduler can configure EH to log only events that match custom criteria.

The criteria for the events which should be logged in the event history are controlled by the criteria manager. By using the eXtensible Markup Language (XML), you can define criteria for events as a combination of standard ABAP select options. You can set separate conditions for event names and event arguments. You can combine these conditions and criteria as logical expressions with the logical operators OR and AND.

- Criteria hierarchy

The criteria hierarchy is the set of all the criteria and conditions which event names and/or event argument of the raised events need to fulfill to be logged in the event history. In XML format, you create and combine criteria in a criteria hierarchy. The criteria in the hierarchy may be grouped in nodes that are governed by a logical AND or OR relation:

By using the functions BAPI_CM_CRITERIA_GET and BAPI_CM_CRITERIA_SET the external scheduler can retrieve a criteria hierarchy in XML format from the criteria manager or set a criteria hierarchy which is marked up in XML.

- Criteria profile

A criteria hierarchy is stored in a criteria profile which can be active or inactive. For the criteria in a profile to take effect, the profile has to be active. You can have many profiles, but only one can be active.

By using the functions BAPI_CM_PROFILE_ACTIVATE and BAPI_CM_PROFILE_DEACTIVATE, the external scheduler can set a profile active or inactive, respectively. XBP 3.0 also provides function modules for creating, deleting, and retrieving criteria profiles.

4.2.9 Monitoring Performance

The SAP system stores batch statistic information about the past workload produced by particular jobs or job steps. By using the function BAPI_XBP_BTC_STATISTIC_GET, the external scheduler can obtain statistic workload information about a list of jobs.

This batch statistic information is available in the system and can be retrieved by the external scheduler only within 96 hours after the job has run.

To be able to retrieve batch statistic information, the external scheduler needs the following authorizations:

Authorization object = S_TOOLS_EX

Field	Value
AUTH	S_TOOLS_EX_A

4.2.10 Obtaining Application Information

As a means of passing information, an ABAP application that runs in the background can create an application log and set an application return code in the batch job data.

With the new internal function BP_ADD_APPL_LOG_HANDLE, the application running as a batch job can assign one or more unique application log handles to its batch job data. When the log handle has been assigned, the external scheduler can use the XBP 3.0 function BAPI_XBP_APPL_INFO_GET to retrieve the log handle. Based on the returned log handle, the external scheduler can call BAPI_XBP_APPL_LOG_CONTENT_GET to read the application log and the application return code.

Authorizations

To be able to obtain application information, the external scheduler needs the following authorizations:

Authorization object = S_APPL_LOG

Field	Value
ALG_OBJECT	*
ALG_SUBOBJ	*
ACTVT	03

Chapter 7 contains details of these function modules.

4.3 Architecture of the AS ABAP Job Scheduling System

The essential components of the AS ABAP background processing are the database tables which contain the job administration data and steps, the job scheduler, the job starter, the job log, and the spool subsystem.

Within this document we can only present a rough sketch of how these elements work together. The aim is, however, to provide a simple model of how the system works.

4.3.1 Job Administration in the Database

All essential job administration information and the job steps themselves are stored in the database. This ensures the consistency and security of the relevant information.

The most important of the database tables is the job data table. This contains entries necessary for job administration: job name, job number, target host, desired start time, job log name and much more. The step list for a job is not contained in this table.

The step list table contains a number of ABAP and operating system level programs ('external programs') for each job. Figures 4.3 - 4.6 show data from both tables in one view.

An event table lists all events defined in AS ABAP, along with the jobs that they trigger when the particular event occurs in the system.

4.3.2 The Job scheduler

In actual fact, the job scheduler consists of two schedulers - one for event based and one for time based jobs. According to the start criteria for a particular job, one or the other of these schedulers assumes responsibility for passing it on. For the sake of this overview, we will make no further distinction between the two schedulers.

The job scheduler for time controlled jobs is started regularly on all the application servers in the SAP systems, which carry out background processing and have background work processes for that purpose. You can set the interval at which it is started in the profile parameter `rdisp/btctime` for each application server. The event driven job scheduler is started on the application server on which an event is triggered.

When the scheduler begins its task, it selects jobs from the database (job administration data) which have reached their start date or whose triggering event has taken place. The scheduler also takes into account whether any background work processes are free. If it comes across free processes of the right type, it tries to send as many jobs as possible to these processes, always bearing in mind the priority of the jobs and maintaining a reserve of background work processes for important class A jobs. Jobs which cannot be processed are left untouched. Jobs, on the other hand, which are sent to the background work processes are marked as such in the database ('ready' status) and, after a very short delay, are taken up by the chosen work process and processed completely (without interruption). If the job scheduler starts a periodic job, it immediately reschedules the successor.

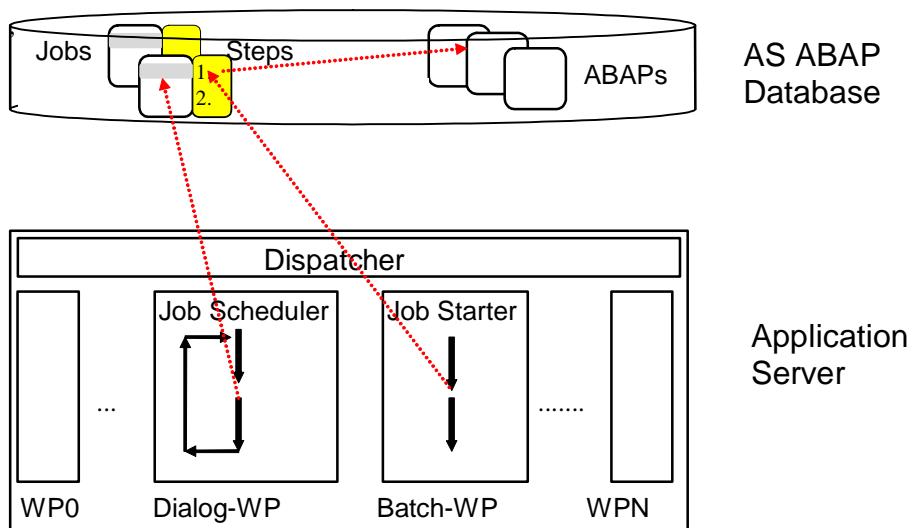


Fig. 4.7: System architecture of AS ABAP background processing (simplified).

4.3.3 The Job Starter and the Job Runtime Environment

The job starter is a compact program which is processed in the background work process and finds a step list in the database. The step list is then processed step by step. If an external program is involved, a UNIX or Windows NT command is sent to the operating system. If the step consists of an ABAP program, this is carried out in the SAP system.

4.3.4 The Job Log

Generally speaking, the job log is not stored in the database. It is a TemSe object, and usually a file at operating system level. The TemSe objects are **temporary** and **sequential** AS ABAP objects, which can be stored system-wide in the database, but are normally kept directly in the file system. In any case, the name of the TemSe object - normally the filename - is stored in the job administration information.

If a job log is requested for a job, the TemSe object is displayed. The user cannot tell from the dialog whether this is a database object or whether it has been saved in a different way.

4.3.5 Job Output

Most ABAP or external programs generate output while they are running. Possible output includes error messages, messages about a program's progress, or lists resulting from a report.

This output and messages are not immediately visible, since the programs are not running in dialog, but in the background. To avoid this output being lost, it is saved so that it can be looked at later. The output lists of a job are looked after by the spool output management. All messages are recorded in the job log. The job log contains information on all job steps.

The step list stores information on the location of the output of each job. The reason for this is that each step can create its own output.

5 The External Interface Concept

The XBP interface described in this document is part of an interface package for external system management tools. On the AS ABAP side, the interfaces are constructed through a pool of function modules. External management systems are able to call these using RFC (Remote Function Call). You can find example calls in the documentation on the interface itself. A short introduction to RFC forms part of the technical basics at the end of this chapter.

In the following section, we are working on the assumption that the external system management tool is represented to the AS ABAP system by an agent. This agent is the communication partner of the function modules.

5.1 Range of Interfaces

The range of interfaces, under the name XM (eXternal System Management), consists at present of the following individual components:

- **XBP** eXternal Interface for Background Processing
- **XBR** eXternal Interface for Backup & Recovery (so far called: BRI or Backint)
- **XMB** eXternal Interface for Monitoring Basics
- **XMI** eXternal Monitor Interface
- **XOM** eXternal Interface for Output Management

For more information about the interfaces, see the SAP Developer Network at sdn.sap.com → **Partners and ISVs** → **Integration and Certification**.

One thing which all of these RFC interfaces have in common is that their function modules call exactly the same function pools as the internal AS ABAP operations. Figure 5.1 shows this using XBP. This does not, of course, apply to the XMI interface, since there is no internal equivalent.

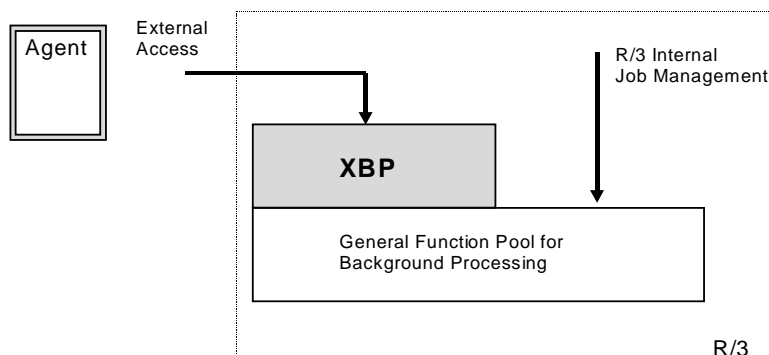


Fig. 5.1: The external interface principle, exemplified by XBP

5.2 Naming Conventions

It is aimed to have a standardized naming convention for function modules, and this is widely kept to. Thus function modules for the XM-interface family have the prefix **BAPI**, an identification for the actual interface and arising from the target object and the intended action.

Syntactic structure for a function module name: `BAPI_<SS>_<Object>_<Action>`

Example of the naming convention: `BAPI_XBP_JOB_OPEN`

In the example, the actual interface is signified by XBP (eXternal interface for Background Processing).

5.3 Technical Basics

The most important technical basics for the external interfaces are XMI and RFC:

XMI is an interface which logs the activities of users and agent programs each time a function module of an external interface is called. In particular, XMI logs the agent's first access to the function module pool. At this point, the name of the external program is recorded and its version number checked.

RFC (Remote Function Call) forms the communications platform for direct calls to the function modules which implement the interface on the AS ABAP side.

5.3.1 XMI Monitor: External Access

Within AS ABAP, all external CCMS interfaces use the same function modules. These function modules can also be collected into an interface themselves. The name XMI (eXternal Monitoring Interface) was established since the interface was intended to log external access.

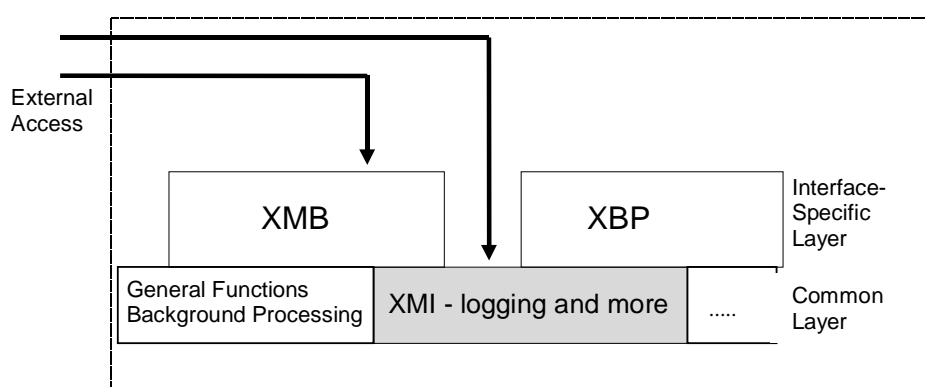


Fig. 5.2: XMI as a common layer for external interfaces.

For developers who want to integrate external tools into CCMS, the XMI interface remains almost invisible, appearing only at the beginning and end of a CCMS session in the form of two functions:

- **BAPI_XMI_LOGON:** Agent logs on to an external interface
- **BAPI_XMI_LOGOFF:** External program logs off from an external interface

You have the option of using further functions from the external tool

- **BAPI_XMI_ENTER_LOGMSG:** Writes a message to the XMI log
- **BAPI_XMI_GET_VERSIONS:** Queries the current version of the interface
- **BAPI_XMI_SELECT_LOG:** Reads the XMI message log

The information which is logged and collected in the access log can be viewed using Transaction RZ15.

The following is displayed: Name of the agent which tried to make contact with the SAP NetWeaver AS ABAP system, and that of its supplier, the user name (if a user logged on) and any changes or attempted changes which were carried out (for example, reset alert).

5.3.2 RFC Remote Function Call

A function within the SAP system can be called from outside via Remote Function Call (RFC), if the function is RFC-enabled. The RFC-enabling is just an indicator in the function attributes.

How it works

RFC is a general concept, allowing communication between SAP systems or between an SAP system and external programs.

In the case of an external system management tool we are dealing with an example of the latter, in which the program adopts the role of a client in relation to the SAP system.

The service offered by the application server involves the delivery of internal system information or carrying out individual system management activities. In order to perform calls via RFC, the client needs the RFC library offered by SAP. This is a collection of necessary functions, which can be called from within C programs.

RFC works in sessions. In other words, the user opens an RFC, carries out RFC tasks and then concludes the session. In opening the session, the AS ABAP logon procedure, with user ID and password, must be carried out. The user must be identifiable to and authorized by the AS ABAP system. In other words, users must exist within the AS ABAP system, which are then used by the external agent. Obviously, you need to set up authorizations for these users which correspond with the activities that they are to carry out.

The Most Important RFC Library Functions

The most important functions of the RFC library at a glance:

Name	Short Description
RfcOpen()	A connection is made with an application server. The connection information is either specified directly or read from a file.
RfcCallReceive()	Synchronous call from an RFC client Activates a function module in an AS ABAP system. After processing, the client calling regains control.
RfcCall()	Calls a function module without waiting for it to end.
RfcLastError()	A function allowing you to analyze RFC errors. Detailed information is given in stdout format.
RfcClose()	RFC connection closed, session finished.
ItCreate() ItDelete()	Storage space needs to be created and released when tables are exchanged between a client program and an AS ABAP system.



The passed parameters should be initialized with 0x20.

Further RFC Documentation

Other RFC functions and call forms are dealt with in the RFC tutorial

AS ABAP Development Workbench, which you can order from SAP.

If you have an AS ABAP installation running, equivalent information is available under 'SAP Library → Basis Components → Basis Services / Communication Interfaces → Remote Communications'.

This documentation answers questions relating to parameter transfer and configuration of an RFC connection (RFC destinations using a file), and also gives the syntax of RFC functions with examples.

Before implementation, you are strongly recommended to consult this documentation.

Framework of a client RFC program in C

In order not to be too theoretical about RFC we have included a practical example. The framework of a program which is linked to the RFC library as an RFC client using the XMB interface would look something like this:

```
#include "saprfc.h"
main()
{
    rfc_handle = RfcOpen(&rfc_opt);
    function = "BAPI_XMI_LOGON";
    .....

    rfc_rc = RfcCallReceive( rfc_handle, function, exporting,
```

```

importing, tables, &exception );

function = "BAPI_XMI_SELECT_LOG";
.....

rfc_rc = RfcCallReceive( rfc_handle, function, exporting,
importing, tables, &exception );

/* a lot more action */

function = "BAPI_XMI_LOGOFF";
.....

rfc_rc = RfcCallReceive( rfc_handle, function, exporting,
importing, tables, &exception );

RfcClose(rfc_handle);

}

```

From this we can see the model for a session between an AS ABAP system and an external management tool. An RFC session contains one or more XMI sessions. Each XMI session contains a series of function calls to the individual interface function module.

Open RFC session	<i>Authorization from AS ABAP</i>
Open XMI session for XBP	<i>Identification of XM</i>
Call XBP functions	<i>Identification of agent user</i>
Open XMI session for XMB	<i>Identification of XM</i>
Call XMB functions	<i>Identification of agent user</i>
Call XBP functions	<i>Identification of agent user</i>
Close XMI session	
Close RFC session	

RFC is being developed further

Another small warning to finish with: the RFC interface is currently being developed further and improved. You should therefore always make sure that you have the current version of the RFC documentation.

6 XBP - External Job Scheduling Interface (external JOB-API)

As explained in the section above, an external interface is a collection of RFC-enabled functions. The XBP interface is an external interface to the SAP background processing system.



The XBP interface must not be confused with the 'normal' batch API, which is a collection of non RFC-enabled functions. The 'normal' batch API is an internal ABAP-API to the SAP background processing system.

Why does SAP offer the XBP interface?

Many customers do not process their data with just one SAP system. They usually have a landscape consisting of one or more SAP systems as well as non-SAP systems. The non-SAP systems usually also have some kind of a background processing system.

There are interdependencies between the systems of such a landscape.



The non-SAP system A creates data using a background job. The SAP system B then processes this data in a job. This means that there is a job Y in SAP system B, which can only start after job X in non-SAP system A has finished.

Such a scenario demonstrates the need for a central job management system. The SAP background processing system, of course, cannot monitor jobs of non-SAP systems. In addition, the interdependencies between jobs even in a single system are sometimes so complex that they cannot be described with the functions of the internal batch API.

A central job management system (often referred to as 'external scheduler') connects to the SAP system via the XBP interface. The functionality of the XBP interface is not more complex than the one offered by the internal batch API, but based on this functionality the external scheduler implements its 'added value', for example graphical editors for job nets and complex start conditions.

In order to manage jobs centrally in a system landscape containing non-SAP systems, the non-SAP systems also have to provide an interface to which the external scheduler can connect.

6.1 What Is Required of the Interface

In order to be able to work in the AS ABAP system, an external job scheduling system must be able to carry out the following activities within the AS ABAP system:

- Create jobs
- Modify jobs
- Delete jobs
- Start jobs (start immediately)
- Terminate active jobs
- Access information about jobs (status, log, and so on)
- Access information about resources in the AS ABAP job scheduling system (number and status of background work processes)

There are XBP functions for carrying out all of these activities.

Any critical changes to AS ABAP jobs (creation, editing, deletion etc) are recorded in the *XMI log* in the AS ABAP system. You can display entries in this log using Transaction RZ15.

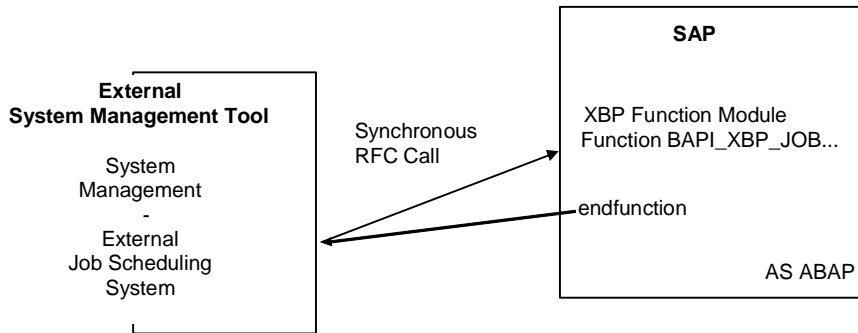


Fig 6.1: How a synchronous RFC function call to XBP works.

For the XBP 3.0 certification, the following functions are required/optional:

1. Configuring criteria for events and interceptions rules (only functions including PROFILE_ are required; the remaining functions are optional)
2. Monitoring performance (optional)
3. Obtaining application information (all functions required)
4. Getting information about and reading a particular spool list (BAPI_XBP_JOB_READ_SINGLE_SPOOL required, BAPI_XBP_GET_SPOOL_ATTRIBUTES optional)
5. Searching for archive parameters (required)
6. Setting a spool list recipient (optional, but it must be possible to set the spool list recipient in BAPI_XBP_JOB_CLOSE by using the new optional parameter RECIPIENT)
7. Selecting all jobs from the SAP system from a certain time period (optional)
8. Simplified variant handling (BAPI_XBP_VARIANT_CREATE, BAPI_XBP_VARIANT_CHANGE, and BAPI_XBP_READ_SELSCREEN required, remaining functions are optional)

6.2 XBP Interface - Description

There follows a short overview of the function modules which make up XBP. Reference information about the modules is given in chapter 7.

The interface can be divided roughly into the following tasks:

Logging on/logging off

Function Modules	Short Description
BAPI_XMI_LOGON	Connect to the external management interface
BAPI_XMI_LOGOFF	End the AS ABAP session of the external job management system

Defining jobs

Function Modules	Short Description
BAPI_XBP_JOB_OPEN	Create a job
BAPI_XBP_JOB_ADD_ABAP_STEP	Add an ABAP step to an existing job
BAPI_XBP_JOB_ADD_EXT_STEP	Add an external program to an existing job
BAPI_XBP_JOB_CLOSE	Finish job creation
BAPI_XBP_JOB_DEFINITION_GET	Read job definitions from the AS ABAP system

Working with jobs

Function Modules	Short Description
BAPI_XBP_JOB_ABORT	Abort job
BAPI_XBP_JOB_DELETE	Delete job
BAPI_XBP_JOB_COPY	Copy a job with all attributes
BAPI_XBP_JOB_HEADER_MODIFY	Modify key job parameters

Modifying jobs

Function Modules	Short Description
BAPI_XBP_JOB_ABAP_STEP_MODIFY	Modify an ABAP step
BAPI_XBP_JOB_EXT_STEP_MODIFY	Modify an external step

Starting jobs and triggering events

Function Modules	Short Description
BAPI_XBP_JOB_START_ASAP	Start as soon as possible
BAPI_XBP_JOB_START_IMMEDIATELY	Start immediately
BAPI_XBP_EVENT_RAISE	Trigger an event from outside

Adding, changing, and deleting job steps via XMI

Function Modules	Short Description
BAPI_XBP_ADD_JOB_STEP	Add and insert a step to a job via XMI
BAPI_XBP_MODIFY_JOB_STEP	Change and delete a step of a job via XMI

Working with raised events

Function Modules	Short Description
BAPI_XBP_BTC_EVTHISTORY_GET	Get the list of raised events
BAPI_XBP_BTC_EVTHIST_CONFIRM	Confirm events
BAPI_XBP_EVENT_DEFINITIONS_GET	Read definitions of batch events
BAPI_CM_CRITYPES_GET	Get a list of available criteria types
BAPI_CM_PROFILE_ACTIVATE	Activate a criteria profile for the criteria for raised events
BAPI_CM_PROFILE_CREATE	Create a criteria profile for the criteria for raised events
BAPI_CM_PROFILE_DELETE	Delete an existing criteria profile for the criteria for raised events
BAPI_CM_PROFILES_GET	Get a list of profiles for the criteria for raised events
BAPI_CM_PROFILE_DEACTIVATE	Deactivate an active profile for the criteria for raised events
BAPI_CM_CRITERIA_GET	Get criteria for raised events in XML format

BAPI_CM_CRITERIA_SET	Import criteria for raised events from XML source
-----------------------------	---

Intercepting and confirming jobs

Function Modules	Short Description
BAPI_XBP_GET_INTERCEPTED_JOBS	Retrieve jobs with the new status 'Intercepted'
BAPI_XBP_CONFIRM_JOB	Confirm jobs in general
BAPI_XBP_SPECIAL_CONFIRM_JOB	Set special confirmation types for jobs (intercept, parent/child)
BAPI_XBP_MODIFY_CRITERIA_TABLE	Modify the criteria table

Monitoring / Controlling the AS ABAP job scheduling system

Function Modules	Short Description
BAPI_XBP_JOB_STATUS_CHECK	Is the job status still correct? Is the WP working?
BAPI_XBP_JOB_COUNT	How many jobs exist with a particular name?
BAPI_XBP_JOB_SELECT	Select sets of jobs by various criteria
BAPI_XBP_JOB_STATUS_GET	Read job status
BAPI_XBP_JOBLIST_STATUS_GET	Determine status of a list of jobs
BAPI_XBP_JOB_JOBLOG_READ	Read job log
BAPI_XBP_JOB_READ	Obtain key job parameters from job header and job steps
BAPI_XBP_JOB_SPOOLLIST_READ_20	Replaces the function modules BAPI_XBP_JOB_SPOOLLIST_READ BAPI_XBP_JOB_SPOOLLIST_READ_RW
BAPI_XBP_JOB_READ_SINGLE_SPOOL	Read a particular spool list of a job that has been run
BAPI_XBP_GET_SPOOL_ATTRIBUTES	Get information about a particular spool list
BAPI_XBP_NEW_FUNC_CHECK	Read and change status of interception and parent/child functionality
BAPI_XBP_JOB_CHILDREN_GET	Get all children of a job
BAPI_XBP_JOB_PARENT_CHILD_INFO	Determine parent/child relation
BAPI_XBP_BTC_STATISTIC_GET	Get statistic records for a list of jobs
BAPI_XBP_APPL_INFO_GET	Get the handles of application logs and application return codes for a particular job
BAPI_XBP_APPL_LOG_CONTENT_GET	Read application log messages

Information about the Background System

Function Modules	Short Description
BAPI_XBP_GET_CURR_BP_RESOURCES	Read current background server and work processes
BAPI_XBP_GET_BP_RESRC_ON_DATE	Background work processes at a particular time
BAPI_XBP_GET_BP_SRVRES_ON_DATE	Is there a background work process on a server at a particular time?
BAPI_XBP_VARIANT_INFO_GET	Does the ABAP program have variants?

Value help functions

Function Modules	Short Description
BAPI_XBP_REPORT_SEARCH	Return a list of ABAP reports available in the current system
BAPI_XBP_EXT_COMM_SEARCH	Return a list of external commands available in the current system
BAPI_XBP_OUTPUT_DEVICE_SEARCH	Return a list of output devices available in the current system
BAPI_XBP_PRINT_FORMAT_SEARCH	Return a list of print formats available for a certain printer
BAPI_XBP_VARIANT_INFO_GET	Get the variants of an ABAP program
BAPI_XBP_EVENT_DEFINITIONS_GET	Get list of batch events
BAPI_XBP_FACT_CALENDERS_GET	Get a list of factory calendars available in the current system
BAPI_XBP_HOL_CALENDERS_GET	Get a list of holiday calendars available in the current system
BAPI_XBP_GET_ARCHIVE_OBJECTS	Return SAP Objects and Archive Objects that are defined in a system

Working with variants

Function Modules	Short Description
BAPI_XBP_VARIANT_CREATE	Create a variant
BAPI_XBP_VARIANT_COPY	Copy a variant
BAPI_XBP_CHANGE	Change a variant
BAPI_XBP_DELETE	Delete a variant
BAPI_XBP_VARINFO	Read information of all variants of an ABAP program
BAPI_XBP_READ_SELSCREEN	Read information about the selection fields of an ABAP program
BAPI_XBP_GET_FREE_SELECTIONS	Read the free selections of an ABAP program

7 XBP Reference Manual

This is the reference manual for the interface function modules. Please, bear in mind that the very latest interface parameter types can only be obtained from your system (Transaction SE37, Function Module - interface).

For using this Reference Manual, note the following points:

- Concerning the following function descriptions the notions IMPORT and EXPORT are described from the view of the respective function module. This means, the module is called with the import parameters and returns export parameters.
- The features that are new or enhanced in XBP 3.0 are marked with 'New in XBP 3.0' or 'Enhanced with XBP 3.0' in the table of the corresponding function description. The features that were introduced in XBP 2.0 are marked with 'Introduced in XBP 2.0'.
- Almost all of the following function descriptions contain the section *Message IDs*. From XBP 3.0 on, this list is not necessarily complete any more, and for some functions no list of message IDs may be given at all.

The reason for this is that in the future the error message, which a function returns in a certain problem case, should be as precise and meaningful as possible for the corresponding case. Therefore, from XBP 3.0 on, SAP reserves the right to return any appropriate error message in an error situation.

What remains unchanged is the format of an error message. It will still be the BAPIRET2 format and an error message will still be of type E. The texts may simply change in the future.

7.1 Requirements for Using the XBP Interface

The requirement for using the XBP interface is an existing XMI session. The technical requirements (RFC, AS ABAP Version) are contained in earlier chapters.

7.1.1 Logging on to the AS ABAP System with an External Job Management System

Before you call a function module in the XBP interface for the first time, it is important that:

- the external job management system logs onto the SAP AS ABAP system first, using an AS ABAP user name and password. (C function RfcOpen).
- the external job management system is authenticated by the CCMS external interface administration using the function module BAPI_XMI_LOGON:

Function name	BAPI_XMI_LOGON
Short description	Connecting to the external Management Interfaces
BAPI object name	SystemMngmtSession
BAPI method name	Logon
RFC interface	<pre> function BAPI_XMI_LOGON importing EXTCOMPANY like BAPIXMLOGR-EXTCOMPANY type RFC_CHAR length 16 EXTPRODUCT like BAPIXMLOGR-EXTPRODUCT type RFC_CHAR length 16 INTERFACE like BAPIXMLOGR-INTERFACE optional type RFC_CHAR length 3 </pre>

	<p>VERSION like BAPIXMLOGR-VERSION optional type RFC_CHAR length 10</p> <p>exporting RETURN structure BAPIRET2 length 548 number of fields 14</p> <p>SESSIONID like BAPIXMLOGR-SESSIONID type RFC_CHAR length 24</p>
Parameter (Input)	<ul style="list-style-type: none"> • EXTCOMPANY is the name of the supplier of the external management tool. • EXTPRODUCT is the product name of external management tool. • INTERFACE (optional) is the interface for logging on. • VERSION (optional) is the version of the interface in AS ABAP required by the external tool.
Parameter (Output)	<ul style="list-style-type: none"> • SESSIONID is the ID of the RFC connection. • BAPIRET2 is the return structure used by BAPIs.
MessageIDs	<ul style="list-style-type: none"> • MSG_ALREADY_LOGGED_ON: This product is already logged onto the interface. • MSG_ALREADY_LOGGED_ON_GEN: The company has already logged on. • MSG_CANT_LOG: Activity was terminated, because the AS ABAP XMI logging mechanism returned an error. • MSG_INVALID_PARAMETERS: EXTCOMPANY and EXTPRODUCT are different within a session. • MSG_LOGON_DENIED: Logon was denied because the AS ABAP user used by the external job management session to log onto the AS ABAP system is not authorized to work with the external job management system. • MSG_LOGON_DENIED_GEN: You have no authorization for general logon. • MSG_PROBLEM_DETECTED: XMI problem which cannot be further specified. • MSG_UNKNOWN_INTERFACE: The interface required by the external tool is not supported by the system. • MSG_UNKNOWN_VERSION: The version required by the external tool is not supported by the system.

The logon will only be successful if you assign the following authorization values to the user for the AS ABAP authorization object **S_XMI_PROD**:

- Name of the company which supplies the external job management system
- Program name of the external job management system
- Name of the interface which the user wants to work with

7.1.2 External Job Management System - Logging Off

Function name	BAPI_XMI_LOGOFF
Short description	If you want to end the external job management system's AS ABAP session, you first need to call the BAPI_XMI_LOGOFF function module.
BAPI object name	SystemMngmtSession
BAPI method name	Logoff
RFC interface	<pre>function BAPI_XMI_LOGOFF importing INTERFACE like BAPIXMLOGR-INTERFACE optional type RFC_CHAR length 3 exporting RETURN like BAPIRET2 structure length 548 number of fields 14</pre>
Parameter (Input)	INTERFACE (optional) from which you should log off.
Parameter (Output)	BAPIRET2 is the return structure used by BAPIs.
MessageIDs	<ul style="list-style-type: none"> • MSG_CANT_LOG: The activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_NOT_LOGGED_ON: There is no connection with the AS ABAP system. • MSG_PROBLEM_DETECTED: XMI problem which cannot be further specified.
Remarks	Afterwards you must close the RFC connection using the C RFC function rfcClose.

7.2 Defining Jobs

You must observe the following procedure in defining AS ABAP jobs:


- Open job (BAPI_XBP_JOB_OPEN)
- Assign (BAPI_XBP_JOB_ADD_ABAP_STEP / BAPI_XBP_JOB_ADD_EXT_STEP) one or more job steps (ABAP Programs or external Programs) to the job.
- Close job and assign start time if required (BAPI_XBP_JOB_CLOSE).

After having defined a job, you can read the definition with the function module BAPI_XBP_JOB_DEFINITION_GET.


7.2.1 Opening Jobs

Function name	BAPI_XBP_JOB_OPEN
Short description	If you want to create a new job, you must first 'open' it. When you open a job, its name is recorded and a job number assigned to it. The job name and number are used as a unique key for all subsequent function calls.
Introduced in XBP 2.0	The optional import parameter JOBCLASS has been added (see description below).
BAPI object name	BackgroundJob
BAPI method name	Open
RFC interface	<pre> function BAPI_XBP_JOB_OPEN importing JOBNAME like BAPIXMJOB-JOBNAME type RFC_CHAR length 32 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 JOBCLASS like BAPIXMJOB-JOBCLASS optional type RFC_CHAR length 1 exporting JOBCOUNT like BAPIXMJOB-JOBCOUNT type RFC_CHAR length 8 RETURN structure BAPIRET2 length 548 number of fields 14 </pre>
Parameter (Input)	<ul style="list-style-type: none"> • JOBCLASS is an optional import parameter. The caller can choose a job class (A, B, or C).
Parameter (Output)	<ul style="list-style-type: none"> • JOBCOUNT is the system-generated job number. • BAPIRET2 is the return structure used by BAPIs.
MessageIDs	<ul style="list-style-type: none"> • MSG_JOBNAME_MISSING : You have not entered a job name. • MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has found an error. • MSG_EXT_USER_MISSING: The name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_CANT_LOG: The activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_NOT_LOGGED_ON: The external management tool has not yet logged on to the CCMS XMI interface. Therefore, the activity cannot be carried out.

7.2.2 Assigning an ABAP Program to a Job Step

Function name	BAPI_XBP_JOB_ADD_ABAP_STEP
Short description	An ABAP program which you intend to run within a job can be assigned to a job step. Enter a valid variant name for the program, if the ABAP program works with variants.
Introduced in XBP 2.0	<p>Two new optional import parameters ALLPRIPAR and ALLARCPAR have been added (see description below).</p>  <p>For corrections to all known issues related with print parameters in XBP, see SAP Note 609462.</p> <p>Note that the reference structures of the structures have been changed. The following applies with SAP Note 609462:</p> <p>ALLPRIPAR LIKE BAPIPRIPAR STRUCTURE BAPIPRIPAR</p> <p>ALLARCPAR LIKE BAPIARCPAR STRUCTURE BAPIARCPAR</p> <p>The SAP System sets default values in the following fields of the print parameter structure if initial values are transferred in the interface:</p> <p>PRIMM = Output immediately (Default = NO) PRREL = Delete after output (Default = NO) PRNEW = New spool request (Default = YES) PRSAP = SAP cover page (Default = Printer settings) PRREC = Recipient (Default = Created by) PRABT = Department (Default = Created by department) PRUNX = Host spooler cover page (Default = Printer settings)</p> <p>If these fields are actually supposed to be transferred empty (for example, if an empty PRIMM overrides the 'Output immediately' setting in the user master), the XBP interface expects the '\$' character to be transferred in this case.</p>
Introduced in XBP 3.0	<p>Two new parameters for enhanced variant handling have been introduced:</p> <ul style="list-style-type: none"> • Import Parameter FREE_SELINFO • Tables Parameter SELINFO
BAPI object name	BackgroundJob
BAPI method name	AddABAPStep
RFC interface	<pre>function BAPI_XBP_JOB_ADD_ABAP_STEP importing JOBNAME like BAPIXMJOB-JOBNAME type RFC_CHAR length 32 JOBCOUNT like BAPIXMJOB-JOBCOUNT type RFC_CHAR length 8 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER</pre>

	<pre> type RFC_CHAR length 16 ABAP_PROGRAM_NAME like BAPIXMREP-REPORTID type RFC_CHAR length 40 ABAP_VARIANT_NAME like BAPIXMREP-VARIANTNAM optional default SPACE type RFC_CHAR length 14 SAP_USER_NAME like BAPIXMSTEP-AUTHCKNAM optional default SY-UNAME type RFC_CHAR length 12 LANGUAGE like BAPIXMSTEP-LANGUAGE optional default SY-LANGU type LANG length 1 PRINT_PARAMETERS structure BAPIXMPRNT 12 optional default SPACE length 48 number of fields 12 ARCHIVE_PARAMETERS structure BAPIXMARCH optional default SPACE length 23 number of fields 3 ALLPRIPAR structure BAPIPRIPAR optional default SPACE length 176 number of fields 22 ALLARCPAR structure BAPIARCPAR optional default SPACE length 328 number of fields 18 FREE_SELINFO type RSDSRANGE_T_SSEL exporting STEP_NUMBER like BAPIXMJOB-STEPSCOUNT type RFC_INT4 length 4 RETURN structure BAPIRET2 length 548 number of fields 14 tables SELINFO like RSPARAMS </pre>
Parameter (Input)	<ul style="list-style-type: none"> • JOBNAME is the name of a background job. • JOBCOUNT is the ID number of a job. Together with the job name, the job number identifies the job uniquely. • EXTERNAL_USER_NAME is the name of the user in the external scheduler who called the function. • ABAP_PROGRAM_NAME is name of the ABAP program that is to be executed in this job step. The program must be type 1 (interactively executable). • ABAP_VARIANT_NAME is an optional parameter to determine variants for the specified ABAP report. • SAP_USER_NAME is an optional parameter to specify an AS ABAP user with whose authorizations the job is processed. • LANGUAGE is an optional parameter to enter the one-digit SAP language key for this job step. For more information see the Language Key Mapping section in the appendix.

	<ul style="list-style-type: none"> • PRINT_PARAMETERS (optional): You can use this structure to transfer parameters for the printer products of the job step to the spool system. You can specify on which output device the print requests are to be printed, whether print requests should be retained in the spool system or output immediately and so on. If no structure is transferred, the system uses the print parameters in the default values of the user you used to log onto the AS ABAP system (rfcOpen). • ARCHIVE_PARAMETERS (optional): You can use this structure to transfer parameters for archiving the printer results of the job step to the optical archiving system. You can use this parameter to specify whether or not the spool requests of the job step are to be archived. If no structure is transferred, the system uses the archiving details in the fixed values of the user you used to log onto the AS ABAP system (rfcOpen). • ALLPRIPAR and ALLARCPAR are optional structures for the specification of all print and archive parameters and complement PRINT_PARAMETERS and ARCHIVE_PARAMETERS. If ALLPRIPAR and ALLARCPAR are initial, PRINT_PARAMETERS and ARCHIVE_PARAMETERS are evaluated. If ALLPRIPAR and ALLARCPAR contain a value, this value is used. <p style="text-align: center;"></p> <p>ALLARCPAR is a set of archive parameters used for steps consisting of an ABAP program. The ALLARCPAR parameter contains the whole set of archive parameters that can be used by an ABAP program.</p> <p>ALLPRIPAR is a set of print parameters used for steps consisting of an ABAP program. The ALLPRIPAR parameter contains the whole set of print parameters that can be used by an ABAP program.</p> <ul style="list-style-type: none"> • FREE_SELINFO This parameter actually refers to an internal table. This means that the reference type RSDSRANGE_T_SSEL is a table type in the DDIC. As a matter of fact, it describes a table with a nested table. For technical reasons, a parameter, which refers to such kind of table, cannot be listed in the section of tables parameters of an RFC-enabled function. With the parameter FREE_SELINFO the caller can specify values for the free selections of a report. If this parameter is supplied, a temporary variant is created in the SAP system.
Parameter (Output)	<ul style="list-style-type: none"> • STEP_NUMBER is the number of job steps. • BAPIRET2 is the return structure used by BAPIs.
Parameter (Tables)	<p>SELINFO</p> <p>With this parameter the caller can specify values for the selection fields of the report without referring to an explicit (meaning existing) variant.</p> <p>If this parameter is supplied, a temporary variant is created in the SAP system.</p>
MessageIDs	<ul style="list-style-type: none"> • MSG_JOBID_MISSING: You did not enter a job number. • MSG_JOBNAME_MISSING: You did not enter a job name. • MSG_JOB_DOES_NOT_EXIST: Job does not exist in the AS ABAP system. • MSG_PROGNAME_MISSING: You did not enter an ABAP

	<p>program name.</p> <ul style="list-style-type: none">• MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has found an error.• MSG_EXT_USER_MISSING: The external user name is missing. This is the name of a user in the external job scheduling system.• MSG_INVALID_PRINT_PARAMS: Printer entry invalid.• MSG_INVALID_ARCHIVE_PARAMS: Archiving parameters invalid.• MSG_NO_ARCHIVE_INFO: Archiving information not given.• MSG_CANT_LOG: Activity was terminated because the AS ABAP XMI logging mechanism returned an error.• MSG_NOT_LOGGED_ON: The external management tool is not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.
--	--

7.2.3 Assigning an External Program to a Job Step

Function name	BAPI_XBP_JOB_ADD_EXT_STEP
Short description	You can assign a program to a job step which runs outside the SAP system, for example a C program. Enter the name of the program and that of the host on which the external program is to run
BAPI object name	BackgroundJob
BAPI method name	AddExternalStep
RFC interface	<pre> function BAPI_XBP_JOB_ADD_EXT_STEP importing JOBNAME like BAPIXMJOB-JOBNAME type RFC_CHAR length 32 JOBCOUNT like BAPIXMJOB-JOBCOUNT type RFC_CHAR length 8 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 EXT_PROGRAM_NAME like BAPIXMSTEP-PROGRAM type RFC_CHAR length 128 EXT_PROGRAM_PARAMETERS like BAPIXMSTEP- PARAMETER optional default SPACE type RFC_CHAR length 255 WAIT_FOR_TERMINATION like BAPIXMAUX-CHAR1 optional default 'X' type RFC_CHAR length 1 SAP_USER_NAME like BAPIXMSTEP-AUTHCKNAM optional default SY-UNAME type RFC_CHAR length 12 TARGET_HOST like BAPIXMSTEP-XPGTGTSYS type RFC_CHAR length 32 exporting RETURN structure BAPIRET2 length 548 number of fields 14 STEP_NUMBER like BAPIXMJOB_STEPCOUNT type RFC_INT4 length 4 </pre>
Parameter (Input)	<ul style="list-style-type: none"> • JOBNAME is the name of a background job. • JOBCOUNT is the ID number of a job. Together with the job name, the job number identifies the job uniquely. • EXTERNAL_USER_NAME is the name of the user in the external scheduler who caused the function call. • EXT_PROGRAM_NAME is the name of the program that is to be executed by the background processing system. • EXT_PROGRAM_PARAMETERS is an optional parameter that may be required by the external program at runtime. The parameters are transferred to the external program at the start time as character strings. • WAIT_FOR_TERMINATION is an optional parameter, which has the effect that the background job waits for the external program to finish before processing the next job step (synchronous job step processing). This option is activated by default. • TARGET_HOST is the name of the host computer on which the external program is to be executed. • SAP_USER_NAME is an optional parameter to specify AS ABAP users with whose authorizations the job is processed.

Parameter (Output)	<ul style="list-style-type: none"> STEP_NUMBER is the number of job steps. BAPIRET2 is the return structure used by BAPIs.
MessageIDs	<ul style="list-style-type: none"> MSG_JOBID_MISSING: You did not enter a job number MSG_JOBNAME_MISSING: You did not enter a job name. MSG_JOB_DOES_NOT_EXIST: Job does not exist in AS ABAP system. MSG_PROGNAME_MISSING: You did not specify the name of the external program. MSG_TARGETHOST_MISSING: You did not specify the target host. MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered a problem. MSG_EXT_USER_MISSING: External user name is missing. This is the name of a user in the external job scheduling system. MSG_CANT_LOG: Activity was terminated because the AS ABAP XMI logging mechanism returned an error. MSG_NOT_LOGGED_ON: The external management tool has not logged on to the CCMS XMI interface. Therefore, the activity cannot be carried out.

7.2.4 Closing Job Definitions

Function name	BAPI_XBP_JOB_CLOSE
Short description	You close a job definition using the BAPI_XBP_JOB_CLOSE function module.
Enhanced in XBP 3.0	The optional import parameter RECIPIENT passes a new RECIPIENT structure (see parameter description below).
BAPI object name	BackgroundJob
BAPI method name	Close
RFC interface	<pre> function: BAPI_XBP_JOB_CLOSE importing JOBNAME like BAPIXMJOB_JOBNAME type RFC_CHAR length 32 JOBCOUNT like BAPIXMJOB_JOBCOUNT type RFC_CHAR length 8 EXTERNAL_USER_NAME like BAPIXMLOGR_EXTUSER type RFC_CHAR length 16 TARGET_SERVER like BAPIXMJOB-EXECSERVER optional type RFC_CHAR length 20 RECIPIENT_OBJ structure SWOTOBJID optional length 100 number of fields 4 RECIPIENT structure BAPIXMRECIP optional length 249 number of fields 9 exporting RETURN structure BAPIRET2 length 548 number of fields 14 </pre>
Parameter (Input)	<ul style="list-style-type: none"> JOBNAME is the name of a background job. JOBCOUNT is the ID number of a job. Together with the job name, the job number identifies the job uniquely. EXTERNAL_USER_NAME is the name of the user in the external scheduler who called the function.

	<ul style="list-style-type: none"> • TARGET_SERVER is an optional parameter with which you can have the job executed on the AS ABAP instance that you specify. • RECIPIENT_OBJ passes the SWOTOBJID which is a structure describing the spool list recipient. However, corresponding to the function module job_close, the format is internal and not simply the name of the spool list recipient. SWOTOBJID has the following fields: <ul style="list-style-type: none"> OGSYS char 10 OBJTYPE char 10 OBJKEY char 70 DESCRIBE char 10 • RECIPIENT passes the BAPIXMRECIP structure that allows the external scheduler to pass a spool list recipient (internal SAP Office user, e-mail address, fax number, or distribution list) and send its attributes in plain text. XBP converts the passed values into the internal SAP format.
Parameter (Output)	<ul style="list-style-type: none"> • BAPIRET2 is the return structure used by BAPIs.
MessageIDs	<ul style="list-style-type: none"> • MSG_JOBID_MISSING: You did not enter a job number. • MSG_JOBNAME_MISSING: You did not enter a job name. • MSG_JOB_DOES_NOT_EXIST: Job does not exist in AS ABAP system. • MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered an error. • MSG_EXT_USER_MISSING: Name of external user is missing. This is the name of a user in the external job scheduling system. • MSG_CANT_LOG: The activity was terminated because the AS ABAP XMI logging system returned an error. • MSG_NOT_LOGGED_ON: The external management tool has not logged on to the CCMS XMI interface. Therefore, the activity cannot be carried out. • MSG_CHILD_REGISTER_ERROR: An error occurred during child registration.

7.2.4.1 Fields of the RECIPIENT structure

Name	Data type	Length	Meaning
RECIPIENT	CHAR	241	Recipient in plain text. Fax numbers must be passed in the following format: <ISO country code><area code><number>, for example DE 06227-747474.
REC_TYPE	CHAR	1	Recipient type. The following recipient types are supported: B – SAP Office user name P – personal distribution list C – shared distribution list F – fax number U – internet mail
COPY	CHAR	1	Send copy
BLIND_COPY	CHAR	1	Send blind copy
EXPRESS	CHAR	1	Send express
NO_FORWARDING	CHAR	1	No forwarding is allowed (for internal recipients)

			only)
DELIVER	CHAR	1	Report send status (for external recipients only). The following values are possible: SPACE - use system default A - always report send status E - report send status only in case of an error N - never report send status
NO_PRINT	CHAR	1	Printing not allowed
MAILSTATUS	CHAR	1	Report status by email (for external recipients only). The following values are possible: SPACE - use system default A - always send status e-mail E - send status e-mail only in case of an error N - never send status e-mail

7.2.5 Reading Job Definitions from the AS ABAP System

Function name	BAPI_XBP_JOB_DEFINITION_GET
Short description	You use the BAPI_XBP_JOB_DEFINITION_GET function module to read all the data associated with a job (name, job class, steps, start conditions etc.).
Enhanced in XBP 3.0	<p>This function module is enhanced to return more information about :</p> <ul style="list-style-type: none"> • The job log of a job. Information is returned by the new export parameter JOBLG_ATTR. • The spool list(s) created by a job. Information is returned by the new table parameter SPOOL_ATTR. • Spool list recipient of a job in plain text. <p>The two parameters provide the external scheduler with more information about the size of the job log or spool list. The function module BAPI_XBP_JOB_READ is enhanced with the same two parameters.</p> <p>If BAPI_XBP_JOB_DEFINITION_GET or BAPI_XBP_JOB_READ return high values for the fields JOBLG_ATTR-TMSSIZE or SPOOL_ATTR-TMSSIZE that contain information about the size of the job log or spool list, it is up to the scheduler to decide which function module to call – BAPI_XBP_JOB_JOBLOG_READ or BAPI_XBP_JOB_SPOOLLIST_READ_20. If either of the output parameters JOBLG_ATTR or SPOOL_ATTR return empty, then the external scheduler knows that the desired object does not exist. In addition, if SPOOL_ATTR-DOCTYP is not 'ALI', it is not necessary to call BAPI_XBP_JOB_SPOOLLIST_READ_20, as this function only returns ABAP lists.</p>
BAPI object name	BackgroundJob
BAPI method name	GetDefinition
RFC interface	<pre> function BAPI_XBP_JOB_DEFINITION_GET importing JOBNAME like BAPIXMJOB_JOBNAME type RFC_CHAR length 32 JOBCOUNT like BAPIXMJOB_JOBCOUNT type RFC_CHAR length 8 EXTERNAL_USER_NAME like BAPIXMLOGR_EXTUSER type RFC_CHAR length 16 exporting JOB_HEAD structure BAPIXMJOB length 388 number of fields 35 RETURN structure BAPIRET2 length 548 number of fields 14 JOBLG_ATTR structure BAPIXMJOBLOG length 38 number of fields 5 RECIPIENT structure BAPIXMRECIP length 249 number of fields 9 tables STEP_TBL structure BAPIXMSTEP length 980 number of fields 56 SPOOL_ATTR structure BAPIXMSPOOLID OPTIONAL </pre>

	length 364 number of fields 42
Parameter (Input)	<ul style="list-style-type: none"> • JOBNAME is the name of a background job. • JOBCOUNT is the ID number of a job. Together with the job name, the job number identifies the job uniquely. • EXTERNAL_USER_NAME is the name of the user in the external scheduler who caused the function call.
Parameter (Output)	<ul style="list-style-type: none"> • JOB_HEAD is job header data (name, job class...). • JOBLG_ATTR returns information about the job log of the job. For the fields of the parameter, see the <i>Fields of the JOBLG_ATTR Parameter</i> table below (section 7.2.5.1). • SPOOL_ATTR returns information about the spool list(s) created by the job. For the fields of the parameter, see the <i>Fields of the SPOOL_ATTR Parameter</i> table below (section 7.2.5.2). • BAPIRET2 is the return structure used by BAPIs. • RECIPIENT returns the spool list recipient of a job in plain text.
Tables	STEP_TBL is a table with job step data.
MessageIDs	<ul style="list-style-type: none"> • MSG_JOBID_MISSING: You have not specified the job ID number. • MSG_JOBNAME_MISSING: You did not enter a job name. • MSG_JOB_DOES_NOT_EXIST: Job does not exist in AS ABAP system. • MSG_NO_JOBSTEPS: Job does not yet have any steps. This can occur if a job already exists in the database but no steps have been assigned to it. • MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered an error. • MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_CANT_LOG: The activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.

7.2.5.1 Fields of the JOBLG_ATTR Structure

Name	Data Type	Length	Meaning
TMSNAME	RFC_CHAR	20	TemSe object name of the job log
TMSCLIENT	RFC_CHAR	3	The client, in which the job log was created
CHARCO	RFC_NUM	4	SAP codepage used by the job log
LANGU	RFC_CHAR	1	Language of the job log
TMSSIZE	RFC_INT4	10	Size of the TemSe object

7.2.5.2 Fields of the SPOOL_ATTR Structure

Name	Data Type	Length	Meaning
STEPNO	RFC_INT4	10	Job step ID number
SPOOLID	RFC_INT4	10	Spool request number
CLIENT	RFC_CHAR	3	Client for which the object was generated
NAME	RFC_CHAR	6	Spool request: Name
SUFFIX1	RFC_CHAR	4	Spool request: Suffix 1
SUFFIX2	RFC_CHAR	12	Spool request: Suffix 2
OWNER	RFC_CHAR	12	User name
FINAL	RFC_CHAR	1	Spool request completed
CRTIME	RFC_CHAR	16	The time a spool request was created
DLTIME	RFC_CHAR	16	The expiration date of the spool request
SPOPAGES	RFC_INT2	5	Total number of pages of the spool request
PRINTTIME	RFC_CHAR	1	Spool request to be printed immediately
DELAFTERPRINT	RFC_CHAR	1	Spool request to be deleted automatically after printing
DEVICE	RFC_CHAR	4	Name of output device
COPIES	RFC_NUM	3	Print parameters, number of copies
PRIORITY	RFC_NUM	1	Spool request priority
SPOFORMAT	RFC_CHAR	16	Format type
PJTOTAL	RFC_INT2	5	Number of output requests for a spool request total
PJDONE	RFC_INT2	5	The number of processed output requests
PJPROBLEM	RFC_INT2	5	The number of output requests with problems
PJERROR	RFC_INT2	5	Number of output requests with errors (no printout is produced for these output requests)
WRITER	RFC_INT2	5	Flag, if object is being used (if writer > 0)
SPERROR	RFC_CHAR	1	Error status of spool request
TEMSENAME	RFC_CHAR	20	TemSe object name
TEMSEPART	RFC_INT2	5	TemSe: the number of

			the part of a TemSe object
TEMSECLIENT	RFC_CHAR	3	The client for which the object was generated
TITLE	RFC_CHAR	68	The title of a spool request
SAPCOVER	RFC_CHAR	1	Print SAP cover page
OSCOVER	RFC_CHAR	1	Print operating system cover page
RECEIVER	RFC_CHAR	12	Recipient of spool request
DIVISION	RFC_CHAR	12	Department
AUTHORITY	RFC_CHAR	12	Value for authorization check
MODTIME	RFC_CHAR	16	The time a spool request was last changed
DOCTYP	RFC_CHAR	6	Document type
OSNAME	RFC_CHAR	50	Spool: Long name of printers for host spooler
TMSSIZE	RFC_INT4	10	Size of spool request in bytes
TEMSELOCAT	RFC_CHAR	1	TemSe: Storage type
LINES	RFC_NUM	5	Output lines of a format type
COLUMNS	RFC_NUM	5	Output columns of a format type
LANGU	RFC_CHAR	1	Language key
CODEPAGE	RFC_NUM	4	SAP Character Set Identification
TMSPARTS	RFC_INT4	4	The total number of parts of a TemSe object

7.2.5.3 Field of the Recipient Structure

See section 7.2.4.1.

7.3 Starting a Job

You can start 'scheduled' or 'intercepted' jobs using the XBP interface in the AS ABAP system with the start time types 'start immediately' or 'as soon as possible'. To do this, use the following function modules:

- Start job immediately (BAPI_XBP_JOB_START_IMMEDIATELY)
- Start job as soon as possible (BAPI_XBP_JOB_START_ASAP)

Besides these function modules, you can use the function BAPI_XBP_EVENT_RAISE to trigger a background processing event. All jobs with the status 'released' waiting for this event will then be started by the AS ABAP job scheduler.

Also note the new function BAPI_XBP_JOB_HEADER_MODIFY described in chapter 7.5.1 'Modifying Job Global Data'. With this function a start condition can be assigned to a job with the status 'scheduled'. The AS ABAP job scheduler then takes care of starting the job.

7.3.1 Starting Jobs Immediately

Function name	BAPI_XBP_JOB_START_IMMEDIATELY
Short description	This function attempts to start a job immediately. If the job cannot be started immediately because, for example, all background work processes are busy, the function reports this to its caller. The job is then started after a delay.
Introduced in XBP 2.0	This function module can now also start jobs with status 'intercepted' and 'released'.
BAPI object name	BackgroundJob
BAPI method name	StartImmediately
RFC interface	<pre>function BAPI_XBP_JOB_START_IMMEDIATELY importing JOBNAME like BAPIXMJOB_JOBNAME type RFC_CHAR length 32 JOBCOUNT like BAPIXMJOB_JOBCOUNT type RFC_CHAR length 8 EXTERNAL_USER_NAME like BAPIXMLOGR_EXTUSER type RFC_CHAR length 16 TARGET_SERVER like BAPIXMJOB-EXECSEVER type RFC_CHAR length 20 exporting RETURN structure BAPIRET2 length 548 number of fields 14</pre>
Parameter (Input)	<ul style="list-style-type: none"> • JOBNAME is the name of a background job. • JOBCOUNT is the ID number of a job. Together with the job name, the job number identifies the job uniquely. • EXTERNAL_USER_NAME is the name of the user in the external scheduler who called the function. • TARGET_SERVER is a parameter with which you can have the job executed on the AS ABAP instance that you specify.
Parameter (Output)	BAPIRET2 is the return structure used by BAPIs.
MessageIDs	<ul style="list-style-type: none"> • MSG_JOBID_MISSING: You did not enter a job number. • MSG_JOBNAME_MISSING: You did not enter a job name. • MSG_JOB_DOES_NOT_EXIST: Job does not exist in the AS ABAP system. • MSG_PROBLEM_DETECTED: The AS ABAP job scheduling

	<p>system discovered an error.</p> <ul style="list-style-type: none"> • MSG_NO_IMMEDIATE_START_POSS: Job cannot be started immediately, since no background work processes are free. Note that the Job is not released in this case and is therefore not waiting to run inside the system. • MSG_PRIVILEGE_MISSING: The AS ABAP user with which the external job management system logged on to the AS ABAP system, is not authorized to release the job. • MSG_EXT_USER_MISSING: External user name is missing. This is the name of a user in the external job scheduling system. • MSG_CANT_LOG: Activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_NOT_LOGGED_ON: The external management tool has not logged on to the CCMS XMI interface. Therefore, the activity cannot be carried out.
--	---

7.3.2 Starting Jobs as Soon as Possible

Unlike the `BAPI_XBP_JOB_START_IMMEDIATELY` function, no error is returned if the job cannot be started immediately.

Function name	BAPI_XBP_JOB_START_ASAP
Short description	You use this function to start a job as soon as possible. You can specify the application server this job should be targeted for.
Introduced in XBP 2.0	As of XBP 2.0 this function module can also start jobs with status 'intercepted' and 'released'.
BAPI object name	BackgroundJob
BAPI method name	StartAsSoonAsPossible
RFC interface	<pre>function BAPI_XBP_JOB_START_ASAP importing JOBNAME like BAPIXMJOB_JOBNAME type RFC_CHAR length 32 JOBCOUNT like BAPIXMJOB_JOBCOUNT type RFC_CHAR length 8 EXTERNAL_USER_NAME like BAPIXMLOGR_EXTUSER type RFC_CHAR length 16 TARGET_SERVER like BAPIXMJOB-EXECSERVER type RFC_CHAR length 20 exporting RETURN structure BAPIRET2 length 548 number of fields 14</pre>
Parameter (Input)	<ul style="list-style-type: none"> • JOBNAME is the name of a background job. • JOBCOUNT is the ID number of a job. Together with the job name, the job number identifies the job uniquely. • EXTERNAL_USER_NAME is the name of the user in the external scheduler who called the function. • TARGET_SERVER is a parameter with which you can have the job executed on the AS ABAP instance that you specify.
Parameter (Output)	BAPIRET2 is the return structure used by BAPIs.
MessageIDs	<ul style="list-style-type: none"> • MSG_JOBID_MISSING: You did not enter a job ID number • MSG_JOBNAME_MISSING: You did not enter a job name.

	<ul style="list-style-type: none"> • MSG_JOB_DOES_NOT_EXIST: Job does not exist in the AS ABAP system. • MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system found an error. • MSG_PRIVILEGE_MISSING: The AS ABAP user with which the external job management system logged onto the AS ABAP system is not authorized to release the job. • MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_CANT_LOG: Activity was terminated because the AS ABAP XML logging mechanism returned an error. • MSG_NOT_LOGGED_ON: The external management tool has not logged on to the CCMS XML interface. Therefore, the activity cannot be carried out.
--	--

7.3.3 Triggering an Event from Outside

Function name	BAPI_XBP_EVENT_RAISE
Short description	This function module is the XBP-equivalent of BP_EVENT_RAISE. With this function, you can trigger an event from outside.
Introduced in XBP 2.0	As of XBP 2.0 this function module can also start jobs with status 'intercept' and 'released'.
BAPI object name	BackgroundJob
BAPI method name	EventRaise
RFC interface	<pre>function BAPI_XBP_EVENT_RAISE importing EVENTID like BAPIXMLOGR-EVENTID type RFC_CHAR length 32 EVENTPARM like BAPIXMJOB-EVENTPARM optional type RFC_CHAR length 64 EXTERNAL_USER_NAME like BAPIXMJOB-EXTUSER type RFC_CHAR length 16 exporting RETURN structure BAPIRET2 length 548 number of fields 14</pre>
Parameter (Input)	<ul style="list-style-type: none"> • EVENTID is the name of the event. • EVENTPARM are optional parameters of the event. • EXTERNAL_USER_NAME is the name of the XBP user.
Parameter (Output)	<ul style="list-style-type: none"> • BAPIRET2 is the return structure used by BAPIs.
MessageIDs	<ul style="list-style-type: none"> • MSG_EVENT_DOES_NOT_EXIST: Event does not exist in the AS ABAP system. • MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_CANT_LOG: Activity was terminated because the AS ABAP XML logging mechanism returned an error. • MSG_NOT_LOGGED_ON: The external management tool has not logged on to the CCMS XML interface. The activity cannot therefore be carried out.

	<ul style="list-style-type: none"> • MSG_PARAM_MISSING: EventID is missing. • MSG_EVENT_RAISE_FAILED: Event could not be triggered.
--	---

7.4 Copying Jobs

Function name	BAPI_XBP_JOB_COPY
Short description	<p>With this function module you can copy a job or, to be more precise, a job definition. The job is copied including all definition data, except for the start conditions.</p> <p>The copy has the status 'scheduled'. A name can be specified for the target with the parameter target_jobname. If no name is specified, the target job has the same name as the source job.</p> <p>With the optional STEP_NUMBER parameter you can specify that not all steps of the original job should be copied. In this case this parameter specifies the number of the very first step to start copying from.</p>
Introduced in XBP 2.0	The entire function module was new in XBP 2.0.
BAPI object name	BackgroundJob
BAPI method name	Copy
RFC interface	<pre>function BAPI_XBP_JOB_COPY importing SOURCE_JOBCOUNT like BAPIXMJOB-JOBCOUNT type RFC_CHAR length 8 SOURCE_JOBNAME like BAPIXMJOB-JOBNAME type RFC_CHAR length 32 TARGET_JOBNAME like BAPIXMJOB-JOBNAME optional default SPACE type RFC_CHAR length 32 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 STEP_NUMBER like BAPIXMJOB-STEPCOUNT optional default 0 type RFC_INT4 length 4 exporting TARGET_JOBCOUNT like BAPIXMJOB-JOBCOUNT type RFC_CHAR length 8 RETURN like BAPIRET2 length 548 number of fields 14</pre>
Parameter (Input)	<ul style="list-style-type: none"> • SOURCE_JOBCOUNT is the ID of the job to be copied. • SOURCE_JOBNAME is the name of the job to be copied. • TARGET_JOBNAME (optional) is the name of the target job. If this parameter is not specified, the name of the target job is the name of the source job. • EXTERNAL_USER_NAME is the name of the XBP user. • STEP_NUMBER (optional) specifies the number of the first step to start copying job data from. Valid values of this parameters are 0, 1, ..., <highest step number>. If 0 and 1 all steps are copied.
Parameter (Output)	<ul style="list-style-type: none"> • TARGET_JOBCOUNT: Job ID number of the newly created job. • BAPIRET2: Return structure for function modules used for BAPIs.

<p>MessageIDs</p>	<ul style="list-style-type: none"> • MSG_JOBNAME_MISSING: Name of the job to be copied is missing. • MSG_JOBID_MISSING: Job ID number to be copied is missing. • MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_CANT_LOG: Activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out. • MSG_WRONG_STEP_NUMBER: The value of the parameter STEP_NUMBER is higher than the number of steps of the jobs. • MSG_CANT_SELECT: Specified source job cannot be selected. • MSG_PRIVILEGE_MISSING : Authorization for copying jobs is missing. • MSG_PROBLEM_DETECTED : Problems other than those stated above. • MSG_JOB_DOES_NOT_EXIST: Specified job does not exist.
-------------------	--

7.5 Controlling Jobs

Control functions currently include the modification of job global data, termination of active jobs, and deletion of obsolete – not running - jobs.

7.5.1 Modifying Job Global Data

Global job data can be changed with the following function module. For example, a start condition can be assigned to a job. By defining a start condition, the external scheduler can give control back to the AS ABAP scheduling mechanism. This is useful for downtimes of the external scheduler.

Function name	BAPI_XBP_JOB_HEADER_MODIFY
Short description	This function module is intended for modifying key job parameters, which are stored in the job header. New values can be set using field masks.
Introduced in XBP 2.0	The entire function module was new in XBP 2.0
BAPI object name	BackgroundJob
BAPI method name	
RFC interface	function BAPI_XBP_JOB_HEADER_MODIFY importing JOBNAME like BAPIXMJOB-JOBNAME type RFC_CHAR length 32 JOBCOUNT like BAPIXMJOB-JOBCOUNT type RFC_CHAR length 8 JOB_HEADER structure BP20HEAD length 385 number of fields 29 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 DONT_RELEASE like BAPIXMINFO-DONOTRELE optional type RFC_CHAR length 1 MASK structure BP20HMSK OPTIONAL length 5 number of fields 5 exporting RETURN like BAPIRET2 length 548 number of fields 14
Parameter (Input)	<ul style="list-style-type: none"> • JOBNAME is the name of the job, whose header needs to be modified. • JOBCOUNT is the ID number of the job whose header needs to be modified. • JOB_HEADER is the new job header, some fields of which should replace old ones (see the MASK parameter). • EXTERNAL_USER_NAME: The name of the XBP user. • DONT_RELEASE (optional) specifies whether the job should be released after the header change. • MASK (optional) is a mask with indicators for each field in header. If a field is selected, it is expected that JOB_HEADER has a new value for this field. If a field is left blank, the corresponding field in the header stays intact.
Parameter (Output)	BAPIRET2 is the return structure used by BAPIs.
MessageIDs	<ul style="list-style-type: none"> • MSG_JOBNAME_MISSING : Job name missing.

	<ul style="list-style-type: none"> • MSG_JOBID_MISSING : Job ID number is missing. • MSG_EXT_USER_MISSING : External user name is missing. • MSG_CANT_LOG : Activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_NOT_LOGGED_ON : The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out. • MSG_JOB_DOES_NOT_EXIST : Specified job does not exist. • MSG_CANT_READ_JOBDATA : Unable to read job data. • MSG_INVALID_NEW_JOBDATA : Invalid new job data. • MSG_NO_MODIFY_PRIVILEGE_GIVEN : Current user does not have modify authorization. • MSG_NO_RELEASE_PRIVILEGE_GIVEN : Current user does not have release authorization. • MSG_CANT_ENQ_JOB : An error occurred while locking job in a database table. • MSG_CANT_RELEASE_JOB : Cannot release the job. • MSG_JOB_NOSTEPS : There are no steps in the job. • MSG_JOB_COUNT_MISSING : Job ID number is missing. • MSG_INVALID_TARGET : Invalid target server. • MSG_CANT_START_JOB_IMMEDIATELY : Immediate job start failed. • MSG_INVALID_STARTDATE : Invalid job start date. • MSG_JOB_NOT_MODIFIABLE_ANYMORE : The job is not modifiable anymore.
--	--

7.5.2 Aborting a Job

You can terminate an active job using the function module BAPI_XBP_JOB_ABORT.

Function name	BAPI_XBP_JOB_ABORT
Short description	Abort a running job.
BAPI object name	BackgroundJob
BAPI method name	Abort
RFC interface	<pre>function BAPI_XBP_JOB_ABORT importing JOBNAME like BAPIXMJOB-JOBNAME type RFC_CHAR length 32 JOBCOUNT like BAPIXMJOB-JOBCOUNT type RFC_CHAR length 8 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 exporting RETURN structure BAPIRET2 length 548 number of fields 14</pre>
Parameter (Input)	<ul style="list-style-type: none"> • JOBCOUNT and JOBNAME of the job to be aborted. • EXTERNAL_USER_NAME is the name of the XBP user.
Parameter (Output)	BAPIRET2 is the return structure used by BAPIs.
MessageIDs	<ul style="list-style-type: none"> • MSG_JOBID_MISSING: You did not enter a job ID.

	<ul style="list-style-type: none">• MSG_JOBNAME_MISSING: You did not enter a job name.• MSG_JOB_DOES_NOT_EXIST: Job does not exist in the AS ABAP system.• MSG_EXT_USER_MISSING: The name of the external user is missing. This is the name of a user in the external job scheduling system.• MSG_JOB_NOT_ACTIVE: The job cannot be terminated since it is not active.• MSG_NO_ABORT_PRIVILEGE: The AS ABAP user used by the external job management system to log onto the AS ABAP system is not authorized to terminate the job.• MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered an error.• MSG_CANT_LOG: The activity was terminated because the AS ABAP XMI logging mechanism returned an error.• MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.
--	---

7.5.3 Deleting a Job

Function name	BAPI_XBP_JOB_DELETE
Short description	Delete a - not running - job.
BAPI object name	BackgroundJob
BAPI method name	Delete
RFC interface	<pre>function BAPI_XBP_JOB_DELETE importing JOBNAME like BAPIXMJOB-JOBNAME type RFC_CHAR length 32 JOBCOUNT like BAPIXMJOB-JOBCOUNT type RFC_CHAR length 8 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 exporting RETURN structure BAPIRET2 length 548 number of fields 14</pre>
Parameter (Input)	<ul style="list-style-type: none"> • JOBCOUNT and JOBNAME of the job to be deleted. • EXTERNAL_USER_NAME is the name of the XBP user.
Parameter (Output)	BAPIRET2 is the return structure used by BAPIs.
MessageIDs	<ul style="list-style-type: none"> • MSG_JOBID_MISSING: You did not enter a job number. • MSG_JOBNAME_MISSING: You did not enter a job name. • MSG_JOB_DOES_NOT_EXIST: Job does not exist in the AS ABAP system. • MSG_NO_JOB_FOUND: The job existed when the function module started, but cannot be found anymore for the actual deletion. • MSG_EXT_USER_MISSING: The name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_JOB_RUNNING: The job cannot be terminated since it is currently active. • MSG_NO_DELETE_PRIVILEGE: The AS ABAP user used by the external job management system to log onto the AS ABAP system is not authorized to delete the job. • MSG_CANT_DEL_IN_JOBTABLE: While trying to delete the job one of the tables that contains job data entries can not be deleted. • MSG_CANT_DEL_JOBLOG: Failed to find or delete the job log of the specified job. Deleting the job continues. • MSG_PROBLEM_PRED_SUCC: A problem with the handling of the predecessor or successor of the deleted job occurred. • MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered an error. • MSG_CANT_LOG: The activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out. • MSG_COMMIT_FAILED: Failed to commit changes in database

	tables.
--	---------

7.6 Modifying Steps in a Job


There are two functions which you can use to modify job steps containing ABAP or external programs, namely:

- BAPI_XBP_ABAP_STEP_MODIFY
to modify a job step containing an ABAP program
- and
- BAPI_XBP_JOB_EXT_STEP_MODIFY
to modify a job step containing an external program.

7.6.1 Modifying a Job Step Containing an ABAP Program

Function name	BAPI_XBP_JOB_ABAP_STEP_MODIFY
Short description	Modify a job step containing an ABAP program
Introduced in XBP 2.0	New optional structures for the specification of all print and archive parameters ALLPRIPAR and ALLARCPAR. They complement the old parameters PRINT_PARAMETERS and ARCHIVE_PARAMETERS. In addition, two new masks have been introduced: PRINT_MASK and ARCH_MASK (see description below).
BAPI object name	BackgroundJob
BAPI method name	ModifyABAPStep
RFC interface	<pre>function BAPI_XBP_JOB_ABAP_STEP_MODIFY importing JOBNAME like BAPIXMJOB-JOBNAME type RFC_CHAR length 32 JOBCOUNT like BAPIXMJOB-JOBCOUNT type RFC_CHAR length 8 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 ABAP_PROGRAM_NAME like BAPIXMREP-REPORTID type RFC_CHAR length 40 ABAP_VARIANT_NAME like BAPIXMREP-VARIANTNAM optional default SPACE type RFC_CHAR length 14 SAP_USER_NAME like BAPIXMSTEP-AUTHCKNAM optional default SY-UNAME type RFC_CHAR length 12 LANGUAGE like BAPIXMSTEP-LANGUAGE optional default SY-LANGU type LANG length 1 PRINT_PARAMETERS structure BAPIXMPRNT 12 optional length 48 number of fields 12 ARCHIVE_PARAMETERS structure BAPIXMARCH optional default SPACE length 23 number of fields 3 STEP_NUMBER like BAPIXMJOB-STEPCOUNT type RFC_INT4 length 10 ALLPRIPAR structure BAPIPRIPAR optional default SPACE</pre>

	<p>length 176 number of fields 22 ALLARCPAR structure BAPIARCPAR optional default SPACE length 328 number of fields 18 PRINT_MASK structure PRIMASK optional length 22 number of fields 22 ARCH_MASK structure ARCMASK optional length 18 number of fields 18</p> <p>exporting RETURN structure BAPIRET2 length 548 number of fields 14</p>
Parameter (Input)	<ul style="list-style-type: none"> • JOBNAME is the name of a background job. • JOBCOUNT is the ID number of a job. Together with the job name, the job number identifies the job uniquely. • EXTERNAL_USER_NAME is the name of the user in the external scheduler called the function. • ABAP_PROGRAM_NAME is the name of the ABAP program that is to be executed in this job step. The program must be type 1 (interactively executable). • ABAP_VARIANT_NAME is an optional parameter to determine variants for the specified ABAP report. • SAP_USER_NAME is an optional parameter to specify an AS ABAP user with whose authorizations the job is processed • LANGUAGE is an optional parameter to enter the one-digit SAP language key for this job step. For more information see the Language Key Mapping section in the appendix. • PRINT_PARAMETERS (optional): You can use this structure to transfer parameters for the printer products of the job step to the spool system. You can specify on which output device the print requests are to be printed, whether print requests should be retained in the spool system or output immediately and so on. If no structure is transferred, the system uses the print parameters in the default values of the user you used to log onto the AS ABAP system (rfcOpen). • ARCHIVE_PARAMETERS (optional): You use this structure to transfer parameters for archiving the printer results of the job step to the optical archiving system. You can use this parameter to specify whether or not the spool requests of the job step are to be archived. If no structure is transferred, the system uses the archiving details in the fixed values of the user you used to log onto the AS ABAP system (rfcOpen). • STEP_NUMBER specifies the number signifying the position of a job step in the sequence of steps in a background job that you want to modify. The second job step in a background has number 2. • ALLPRIPAR and ALLARCPAR are optional structures for the specification of all print and archive parameters and complement the old parameters PRINT_PARAMETERS and ARCHIVE_PARAMETERS. If ALLPRIPAR and ALLARCPAR are initial, PRINT_PARAMETERS and ARCHIVE_PARAMETERS are evaluated. If ALLPRIPAR and ALLARCPAR contain a value, this value is used. <p>ALLARCPAR is a set of archive parameters used for steps consisting of an ABAP program. The ALLARCPAR parameter contains the whole set of archive parameters that can be used by an ABAP program.</p>

	<p>ALLPRIPAR is a set of print parameters used for steps consisting of an ABAP program. The ALLPRIPAR parameter contains the whole set of print parameters that can be used by an ABAP program.</p> <ul style="list-style-type: none"> PRINT_MASK (optional) refers to ALLPRIPAR and ARCH_MASK (optional) refers to ALLARCPAR. PRINT_MASK and ARCH_MASK were introduced for the following reasons:  <p>If, for example, a field of the parameter ALLPRIPAR is initial, it is impossible to determine, whether the caller wants to overwrite the existing value of the corresponding print parameter with the initial value or whether he wants to leave the existing value untouched. Therefore, the parameter PRINT_MASK was introduced. It contains 22 fields of the same name as the fields of ALLPRIPAR, but all are simply indicator fields of length 1.</p> <p>For example, PRINT_MASK-PRTXT = 'X' means that the existing value of this print parameter should be overwritten with the value in ALLPRIPAR-PRTXT. If an indicator field is blank, the corresponding print parameter remains untouched. If the parameter ALLPRIPAR is initial, the function evaluates the old parameters PRINT_PARAMETERS. If a non-initial parameter ALLPRIPAR is passed by the caller, the parameter PRINT_MASK must not be initial either (see MSG_MASK_ERROR).</p> <p>The above example applies also to the pair ALLARCPAR and ARCH_MASK (optional). If ALLARCPAR is initial, the function evaluates the old parameter ARCHIVE_PARAMETERS.</p>
Parameter (Output)	BAPIRET2 is the return structure used by BAPIs.
MessageIDs	<ul style="list-style-type: none"> MSG_JOBID_MISSING: You did not enter a job number. MSG_JOBNAME_MISSING: You did not enter a job name. MSG_JOB_DOES_NOT_EXIST: Job does not exist in the AS ABAP system. MSG_STEP_COUNT_MISSING: You did not enter the number of the step to be modified. MSG_INVALID_STEP_COUNT: Step number invalid, that is, there is no step with the given number. MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system. MSG_MASK_ERROR: This message refers to ALLPRIPAR and ALLARCPAR: If ALLPRIPAR or ALLARCPAR is not initial, but the corresponding mask is initial, this message is returned. MSG_PROGRAM_MISSING: The ABAP program name is missing. MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered an error. MSG_CANT_LOG: The activity was terminated because the AS ABAP XMI logging mechanism returned an error. MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity

	<p>cannot be carried out.</p> <ul style="list-style-type: none">• MSG_NO_ARCHIVE_INFO: Required fields in archive parameters are missing.• MSG_INVALID_PRINT_PARAMS: One or more print parameters are incorrect.• MSG_INVALID_ARCHIVE_PARAMS: One or more archive parameters are incorrect.
--	--

7.6.2 Modifying a Job Step Containing an External Program

Function name	BAPI_XBP_JOB_EXT_STEP_MODIFY
Short description	To modify a job step containing an external program.
BAPI object name	BackgroundJob
BAPI method name	ModifyExternalStep
RFC interface	<pre> function BAPI_XBP_JOB_EXT_STEP_MODIFY importing JOBNAME like BAPIXMJOB-JOBNAME type RFC_CHAR length 32 JOBCOUNT like BAPIXMJOB-JOBCOUNT type RFC_CHAR length 8 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 EXT_PROGRAM_NAME like BAPIXMSTEP-PROGRAM type RFC_CHAR length 128 EXT_PROGRAM_PARAMETERS like BAPIXMSTEP- PARAMETER optional default SPACE type RFC_CHAR length 255 WAIT_FOR_TERMINATION like BAPIXMAUX-CHAR1 optional default 'X' type RFC_CHAR length 1 TARGET_HOST like BAPIXMSTEP-XPGTGTSYS type RFC_CHAR length 32 SAP_USER_NAME like BAPIXMSTEP-AUTHCKNAM optional default SY-UNAME type RFC_CHAR length 12 STEP_NUMBER like BAPIXMJOB-STEPCOUNT type RFC_INT4 length 4 exporting RETURN structure BAPIRET2 length 548 number of fields 14 </pre>
Parameter (Input)	<ul style="list-style-type: none"> • JOBNAME is the name of a job. • JOBCOUNT is the ID number of a job. Together with the job name, the job number identifies the job uniquely. • EXTERNAL_USER_NAME is the name of the user in the external scheduler who caused the function call. • EXT_PROGRAM_NAME is the name of the program that is to be executed by the background processing system. • EXT_PROGRAM_PARAMETERS is an optional parameter that may be required by the external program at runtime. The parameters are transferred to the external program at the start time as character strings. • WAIT_FOR_TERMINATION is an optional parameter, which has the effect that the background job waits for the external program to finish before processing the next job step (synchronous job step processing). This option is activated by default. • TARGET_HOST is the name of the host computer on which the external program is to be executed. • SAP_USER_NAME is an optional parameter to specify AS ABAP users with whose authorizations the job is processed. • STEP_NUMBER specifies the number that displays the position of a job step in the sequence of steps in a background job that you

	want to modify. The second job step in a background has number 2.
Parameter (Output)	BAPIRET2 is the return structure used by BAPIs.
MessageIDs	<ul style="list-style-type: none"> • MSG_JOBID_MISSING: You did not enter a job number. • MSG_JOBNAME_MISSING: You did not enter a job name. • MSG_JOB_DOES_NOT_EXIST: Job does not exist in the AS ABAP system. • MSG_STEP_COUNT_MISSING: You did not specify the number of the step to be modified • MSG_INVALID_STEP_COUNT: The step number is invalid, that is, there is no step with the given number. • MSG_PROGNAME_MISSING: The name of the external program is missing. • MSG_TARGETHOST_MISSING: The name of the target host is missing. • MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered an error. • MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_CANT_LOG: Activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.

7.7 Adding, Changing, and Deleting Job Steps via XMI

7.7.1 Adding a Step to a Job via XMI


Function name	BAPI_XBP_ADD_JOB_STEP
Short description	Adds and inserts a step to a job via XMI.
Introduced in XBP 2.0	The entire function module was new in XBP 2.0.
BAPI object name	BackgroundJob
BAPI method name	
RFC interface	<pre> function BAPI_XBP_ADD_JOB_STEP importing JOBNAME like BAPIXMJOB-JOBNAME type RFC_CHAR length 32 JOBCOUNT like BAPIXMJOB-JOBCOUNT type RFC_CHAR length 8 STEP structure BPJOBSTEP optional default SPACE length 498 number of fields 17 STEP_NUM like BAPIXMJOB-STEPCOUNT type RFC_INT4 length 4 EXTERNAL_USER_NAME LIKE BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 ALLPRIPAR structure PRI_PARAMS optional default SPACE length 196 number of fields 24 ALLARCPAR structure ARC_PARAMS optional default SPACE length 332 number of fields 19 Exporting STEP_NUMBER like BAPIXMJOB-STEPCOUNT type RFC_INT4 length 4 RETURN structure BAPIRET2 length 548 number of field 14 </pre>
Parameter (Input)	<ul style="list-style-type: none"> • JOBNAME is the name of the job to which a job step is added. • JOBCOUNT is the ID number of the job to which a job step is added. • STEP is a definition of the step to be added or inserted. This parameter has a field TYP for distinguishing between types of steps. Valid values of this field are 'A' for ABAP program, 'C' for external commands, and 'X' for external programs. • STEP_NUM is the position where a new step should be inserted. Use 0 (zero), if you want to add a step at the end of existing steps. • EXTERNAL_USER_NAME is the name of the external user that is used for XMI logging. • ALLPRIPAR (optional) is a set of print parameters, which is only used if the new step consists of an ABAP program. The ALLPRIPAR parameter contains the whole set of print parameters that may be used by an ABAP program. • ALLARCPAR (optional) is a set of archive parameters, which is

	<p>only used if the new step consists of an ABAP program. The ALLARCPAR parameter contains the whole set of archive parameters that may be used by an ABAP program.</p>
Parameter (Output)	<ul style="list-style-type: none"> STEP_NUMBER is the number that displays the position of the inserted or added job step. RETURN is the return code of the function module. If no error occurs, the return code is zero.
Detailed function Description	<p>This function module is for adding and inserting a step to a job. In the case of adding a step, the step will be added at the end of all job steps. In the case of inserting a step, it will be placed at the position STEP_NUM and the rest of steps will be moved down. If the job does not have any steps yet, the new step will be the first step of this job.</p> <p>This function module operates on all types of job steps: ABAP program, external command, and external program. For ABAP programs one can specify print and archive parameters. If these optional parameters are skipped, the function module uses default print and archive parameters, obtained from the function module GET_PRINT_PARAMETERS.</p> <p>Example:</p> <p>Adding or inserting an ABAP program:</p> <pre>step-typ = 'A'. step-program = '<ABAP program name>'. step-parameter = '<Variant name>'. step-language = '<Language>'. step-authcknam = '<User name>'.</pre> <p>Adding or inserting an external command:</p> <pre>step-typ = 'C'. step-program = '<Command name>'. step-opsystem = '<Operating system>'. step-parameter = '<Command parameters>'. step-xpigtgtsys = '<XPG target system>'. step-language = '<Language>'. step-authcknam = '<User name>'. step-termcntl = 'C'.</pre> <p>Adding or inserting an external program:</p> <pre>step-typ = 'X'. step-program = '<External program name>'. step-xpigtgtsys = '<XPG target system>'. step-language = '<language>'. step-authcknam = '<User name>'. step-termcntl = 'C'.</pre>
MessageIDs	<ul style="list-style-type: none"> MSG_JOBID_MISSING: You did not specify a job number. MSG_JOBNAME_MISSING: You did not specify a job name. MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered an error. MSG_CANT_LOG: Activity was terminated because the AS ABAP XML logging mechanism returned an error. MSG_EXT_USER_MISSING: Name of the external user is

	<p>missing. This is the name of a user in the external job scheduling system.</p> <ul style="list-style-type: none"> • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out. • MSG_WRONG_STEP_TYPE: The type of step is wrong. • MSG_NO_ARCHIVE_INFO: Required fields in archive parameters are missing. • MSG_INVALID_PRINT_PARAMS: One or more print parameters are incorrect. • MSG_INVALID_ARCHIVE_PARAMS: One or more archive parameters are incorrect. • MSG_CANNOT_GET_PRIARC_PARAMS: Retrieving new print and archive parameters failed. • MSG_CANNOT_READ_JOB: Reading information about the specified jobs failed. • MSG_CANNOT_MODIFY_JOB: Error while writing new data about the specified job into AS ABAP databases. • MSG_ERROR_MODIFYING_WORKTABLE: Error occurred while modifying worktable. • MSG_WRONG_STEP_NUMBER: There are no steps with the specified step number.
--	--

7.7.2 Changing and Deleting a Job Step via XMI

Function name	BAPI_XBP_MODIFY_JOB_STEP
Short description	Changes and deletes a step of a job via XML.
Introduced in XBP 2.0	The entire function module was new in XBP 2.0.
BAPI object name	BackgroundJob
BAPI method name	
RFC interface	<pre> function BAPI_XBP_MODIFY_JOB_STEP importing JOBNAME like BAPIXMJOB-JOBNAME type RFC_CHAR length 32 JOBCOUNT like BAPIXMJOB-JOBCOUNT type RFC_CHAR length 8 STEP structure BPJOBSTEP optional default SPACE length 498, number of fields 17 STEP_NUM like BAPIXMJOB-STEPCOUNT type RFC_INT4 length 4 DELETE like BAPIXMINFO-DELETESTEP type RFC_CHAR length 1 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 ALLPRIPAR structure PRI_PARAMS optional default SPACE length 196 number of fields 24 ALLARCPAR structure ARC_PARAMS optional default SPACE length 332 number of fields 19 PRINT_MASK structure PPR_MASK optional default SPACE length 22 number of fields 22 ARCH_MASK structure APR_MASK optional default SPACE length 18 number of fields 18 exporting STEP_NUMBER LIKE BAPIXMJOB-STEPCOUNT type RFC_INT4 length 4 RETURN LIKE BAPIRET2 STRUCTURE length 548 number of fields 14 </pre>
Parameter (Input)	<ul style="list-style-type: none"> • JOBNAME is the name of the job whose step is to be changed or deleted. • JOBCOUNT is the ID number of the job whose step is to be changed or deleted. • STEP defines the new step. This parameter has a field TYP for distinguishing between types of steps. Valid values for this field are 'A' for ABAP program, 'C' for external commands, and 'X' for external programs. The STEP parameter is not evaluated, if this function module is used for deleting a step. • STEP_NUM is the number of the step to be changed or deleted. • DELETE is an indicator that specifies if the step should be changed or deleted. If the indicator value is initial, the step is changed.

	<ul style="list-style-type: none"> EXTERNAL_USER_NAME is the name of the external user that is used for XMI logging. ALLPRIPAR (optional) is a set of print parameters used only if the new step consists of an ABAP program. The ALLPRIPAR parameter contains the whole set of print parameters that can be used by an ABAP program. See also PRINT_MASK. ALLARCPAR (optional) is a set of archive parameters used only, if the new step consists of an ABAP program. The ALLARCPAR parameter contains the whole set of archive parameters that can be used by an ABAP program. See also ARCH_MASK. PRINT_MASK (optional) is a mask for print parameters. By setting a value in a mask field, you specify that there is a new field value in the ALLPRIPAR parameter. By leaving a field blank you can specify that the corresponding field should be left intact in the ALLPRIPAR parameter. PRINT_MASK refers to ALLPRIPAR and ARCH_MASK refers to ALLARCPAR. PRINT_MASK and ARCH_MASK were introduced for the following reasons:  <p>If, for example, a field of the parameter ALLPRIPAR is initial, it is impossible to determine whether the caller wants to overwrite the existing value of the corresponding print parameter with the initial value or whether he wants to leave the existing value untouched. Therefore, the parameter PRINT_MASK was introduced. It contains 22 fields of the same name as the fields of ALLPRIPAR, but all are simply indicator fields of length 1.</p> <p>For example, PRINT_MASK-PRTXT = 'X' means that the existing value of this print parameter should be overwritten with the value in ALLPRIPAR-PRTXT. If an indicator field is blank, the corresponding print parameter remains untouched.</p> <p>If a non-initial parameter ALLPRIPAR is passed by the caller, the parameter PRINT_MASK must not be initial either.</p> <p>The same applies to the pair ALLARCPAR and ARCH_MASK. If the parameter ALLPRIPAR is initial, the function evaluates the old parameters PRINT_PARAMETERS.</p> <p>If ALLARCPAR is initial, the function evaluates the old parameters ARCHIVE_PARAMETERS.</p> ARCH_MASK (optional) is a mask for archive parameters. By setting a value in a mask field, you specify that there is a new field value in the ALLARCPAR parameter. By leaving a field blank you can specify that the corresponding field should be left intact in the ALLARCPAR parameter.
Parameter (Output)	<ul style="list-style-type: none"> STEP_NUMBER is the number of the changed or deleted step. RETURN is the return code of the function module. If no error occurs, the return code is zero.
Detailed function description	<p>This function module is for changing and deleting a step of a job. In the case of changing a step, the step does not change its number, but its content is replaced by a new one. This function module operates on all types of job steps: ABAP program, external command, and external</p>

	<p>program. For ABAP programs one can specify print and archive parameters. In the case these optional parameters are skipped, the function module uses default print and archive parameters obtained from the GET_PRINT_PARAMETERS function module.</p> <p>In the case of deleting a step, the step is removed and the rest of the job steps are moved up. If the job has only one step, then deleting fails.</p> <p>Example</p> <p>Changing step N to an ABAP program:</p> <pre> step-typ = 'A'. step-program = '<ABAP program name>'. step-parameter = '<Variant name>'. step-language = '<Language>'. step-authcknam = '<User name>'. step_num = <N>. </pre> <p>Changing step N to an external command:</p> <pre> step-typ = 'C'. step-program = '<Command name>'. step-opssystem = '<Operating system>'. step-parameter = '<Command parameters>'. step-xpigtgtsys = '<XPG target system>'. step-language = '<Language>'. step-authcknam = '<User name>'. step_num = <N>. step-termcntl = 'C'. </pre> <p>Changing step N to an external program:</p> <pre> step-typ = 'X'. step-program = '<External program name>'. step-xpigtgtsys = '<XPG target system>'. step-language = '<language>'. step-authcknam = '<User name>'. step_num = <N>. step-termcntl = 'C'. </pre>
MessageIDs	<ul style="list-style-type: none"> • MSG_JOBID_MISSING: You did not specify a job number. • MSG_JOBNAME_MISSING: You did not specify a job name. • MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered an error. • MSG_CANT_LOG: Activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity

	<p>cannot be carried out.</p> <ul style="list-style-type: none">• MSG_WRONG_STEP_TYPE: The type of step is wrong.• MSG_NO_ARCHIVE_INFO: Required fields in archive parameters are missing.• MSG_INVALID_PRINT_PARAMS: One or more print parameters are incorrect.• MSG_INVALID_ARCHIVE_PARAMS: One or more archive parameters are incorrect.• MSG_CANNOT_GET_PRIARC_PARAMS: Retrieving new print and archive parameters failed.• MSG_CANNOT_READ_JOB: Reading information about the specified jobs failed.• MSG_CANNOT_MODIFY_JOB: Error while writing new data about the specified job into AS ABAP databases.• MSG_ERROR_MODIFYING_WORKTABLE: Error occurred while modifying worktable.• MSG_ERROR_READING_WORKTABLE: Error occurred while reading worktable.• MSG_WRONG_STEP_NUMBER: There are no steps with the specified step number.• MSG_NO_STEP_INFO: Step modification is impossible, because no step information is provided.
--	--

7.8 Intercepting and Confirming Jobs

Interception is a new feature in XBP 2.0. Job interception means that at the moment, when the start condition of the job is fulfilled the job is set back to the status 'scheduled' and receives a special attribute. By calling a new XBP function the external scheduler can receive a list of all intercepted jobs and take control over such jobs.

It is not intended to subject jobs to interception in general. The user can define criteria in the new table TBCICPT1 (client, job name, job-creator including wild cards), and only the jobs that match these intercept criteria, are intercepted. For instance, a table entry (100, babu* , *) means that all jobs created in client 100 by users beginning with babu are intercepted.



Example for the use of interception:

The administrator might want to intercept all jobs of certain users or with certain job names on weekends when long-running and time critical batch jobs are executed. In this case interception provides dynamic job prioritization.

In XBP 3.0 it's also possible to define complex interception criteria using the Criteria Manager Interface.

You can find detailed information on interception in *Intercepting Jobs* on page 16.

7.8.1 Getting Intercepted Jobs

In order to find out if there are intercepted jobs, the external scheduler calls a function at short intervals (BAPI_XBP_GET_INTERCEPTED_JOBS). To prevent this function from returning the same intercepted jobs again and again, the scheduler can confirm a list of intercepted jobs. Confirmation means that the scheduler informs the AS ABAP system that it already knows these intercepted jobs, and that a subsequent call of BAPI_XBP_GET_INTERCEPTED_JOBS does not have to return these jobs again. The confirmation of a list of intercepted jobs is done by calling BAPI_XBP_SPECIAL_CONFIRM_JOB, which is explained in the following section.

Function name	BAPI_XBP_GET_INTERCEPTED_JOBS
Short description	This function module retrieves jobs which have status INTERCEPTED. The list of returned jobs may contain all intercepted jobs, or only those ones, which have not been confirmed with one or more special confirmations.
Introduced in XBP 2.0	The entire function module was new in XBP 2.0.
BAPI object name	BackgroundJob
BAPI method name	
RFC interface	Function BAPI_XBP_GET_INTERCEPTED_JOBS Importing EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 SELECTION type CHAR2 optional DEFAULT 'AL' type RFC_CHAR length 2 CLIENT like TBTCO_AUTHCKMAN optional MORE_INFO like BTCH0000-CHAR1 optional Exporting RETURN structure BAPIRET2 length 548 number of fields 14

	<p>Tables</p> <p>JOBINFO like BPICPINFO length 54 number of fields 4</p> <p>JOBINFO2 like BPICPINF1 length 120 number of fields 10</p>
Detailed function description	<p>This function module retrieves jobs that have the status INTERCEPTED. It is not necessary to retrieve the same jobs again and again. To reduce the list of returned jobs to the jobs that are not known to the caller, confirmation is used.</p> <p>The list of returned jobs may contain all intercepted jobs, or only those jobs that have not been confirmed with one or more confirmations. To specify that confirmation should be taken into account, the optional case-insensitive SELECTION parameter is used.</p>
Parameter (Input)	<ul style="list-style-type: none"> • EXTERNAL_USER_NAME: The name of the XBP user. • SELECTION (optional): This parameter specifies what kind of confirmation should be taken into account when selecting intercepted jobs. <ol style="list-style-type: none"> 1. 'AL' (default) – return all intercepted jobs regardless what confirmation they have. 2. 'NG' – return only those intercepted jobs that do NOT have general confirmation. 3. 'NS' – return only those intercepted jobs that were NOT confirmed as intercepted. 4. 'NC' – return only those intercepted jobs that do NOT have any confirmation. <p>To make a special confirmation for an intercepted job, use the BAPI_XBP_SPECIAL_CONFIRM_JOB function module with CONFIRMATION = 'i'.</p> <p>To make a general confirmation for an intercepted job, use the BAPI_XBP_CONFIRM_JOB function module.</p> • CLIENT (optional): The client, on which jobs should be selected. If no client is specified, jobs from all clients are retrieved. • MORE_INFO (optional): This is an optional flag specifying the function to retrieve more detailed information on the selected jobs. When this flag is left blank, then the JOBINFO table is used; otherwise JOBINFO2
Parameter (Output)	<ul style="list-style-type: none"> • BAPIRET2 is the standard return structure containing return values of the function.
Tables	<ul style="list-style-type: none"> • JOBINFO is the table that contains intercepted jobs. • JOBINFO2 is the table that contains more detailed information on intercepted jobs.
MessageIDs	<ul style="list-style-type: none"> • MSG_CANT_LOG: Activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out. • MSG_WRONG_CLIENT: Wrong client number • MSG_WRONG_SELECTION_PAR: Inconsistent selection parameters.

7.8.2 Confirming Jobs

There are three function modules for job selection:

- BAPI_XBP_JOB_SELECT for general job selection
- BAPI_XBP_JOB_CHILDREN_GET for child job selection
- BAPI_XBP_GET_INTERCEPTED_JOBS for the selection of intercepted jobs

These functions are normally called at intervals by the external job scheduler and return general jobs, child jobs, or intercepted jobs, respectively. If you do not want the system to return the same jobs over and over again, you can confirm them.

Confirmation means that the scheduler informs the AS ABAP system that it already knows these jobs, and that a subsequent call of the selection function module does not have to return these jobs again.

There are two types of confirmation:

- **General:** With the general confirmation the job scheduler confirms, that it knows a job in general. Jobs are generally confirmed with BAPI_XBP_CONFIRM_JOB. When you use BAPI_XBP_JOB_SELECT the generally confirmed jobs are not returned if the corresponding indicator is set.
- **Special:** With the special confirmation the job scheduler confirms, that it knows that a job has certain characteristics. Child jobs and intercepted jobs are confirmed with BAPI_XBP_SPECIAL_CONFIRM_JOB. When you use BAPI_XBP_JOB_CHILDREN_GET or BAPI_XBP_GET_INTERCEPTED_JOBS, the specially confirmed jobs are not returned again if the corresponding indicator is set.

However, in some situations (such as after a breakdown) it might be useful to get a list of all intercepted or child jobs (including the confirmed ones). This function has a special indicator for this purpose.

7.8.2.1 Confirming Jobs Generally

Function name	BAPI_XBP_CONFIRM_JOB
Short description	This function module allows the callers to set general confirmation for a list of jobs.
Introduced in XBP 2.0	The entire function module was new in XBP 2.0.
BAPI object name	BackgroundJob
BAPI method name	
RFC interface	Function BAPI_XBP_CONFIRM_JOB Importing EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 Exporting RETURN structure BAPIRET2 length 548 number of fields 14 Tables JOBS structure BAPIXMJOBS (jobname, jobcount) length 40 number of fields 2
Detailed function description	This function module allows the caller (external scheduler) to confirm jobs that are already known to it. All the jobs that were confirmed this way will not be returned again when a new call to BAPI_XBP_JOB_SELECT is performed.
Parameter (Input)	<ul style="list-style-type: none"> • EXTERNAL_USER_NAME: The name of the XBP user.

Parameter (Output)	<ul style="list-style-type: none"> • BAPIRET2 is the standard return structure containing return values of the function.
Tables	<ul style="list-style-type: none"> • The JOBS table contains the jobs for which general confirmation should be set. After BAPI_XBP_CONFIRM_JOB is called, the JOBS table contains only those jobs, for which this operation failed, perhaps because the job was not intercepted, the job was deleted, or an internal error occurred.
MessageIDs	<ul style="list-style-type: none"> • MSG_EXT_USER_MISSING: External user name is missing. • MSG_CANT_LOG: Activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out. • MSG_JOB_CONFIRMATION_FAILED: Not all jobs were confirmed. The JOBS table contains the jobs that have not been confirmed.

7.8.2.2 Performing a Special Confirmation on a Job

Child jobs and intercepted jobs are confirmed with BAPI_XBP_SPECIAL_CONFIRM_JOB. When you subsequently use BAPI_XBP_JOB_CHILDREN_GET or BAPI_XBP_GET_INTERCEPTED_JOBS, the specially confirmed jobs are not returned again.

However, in some situations (such as after a breakdown) it might be useful to get a list of all intercepted or child jobs (including the confirmed ones). For this purpose, this function has the special indicator CONFIRMATION.

Function name	BAPI_XBP_SPECIAL_CONFIRM_JOB
Short description	This function module allows the caller to set special types of confirmation for a list of jobs.
Introduced in XBP 2.0	The entire function module was new in XBP 2.0.
BAPI object name	BackgroundJob
BAPI method name	
RFC interface	<pre> Function BAPI_XBP_SPECIAL_CONFIRM_JOB Importing EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 CONFIRMATION TYPE CHAR1 Exporting RETURN structure BAPIRET2 length 548 number of fields 14 Tables JOBS structure BAPIXMJOBS (jobname, jobcount) length 40.number of fields 2 </pre>
Detailed function description	With this function module the external scheduler confirms jobs, of which he knows certain special characteristics (parent/child, intercepted). All jobs confirmed this way will not be returned again when jobs with this characteristic are requested. To specify the type of confirmation case-insensitive parameter CONFIRMATION should be used.

Parameter (Input)	<ul style="list-style-type: none"> EXTERNAL_USER_NAME: The name of the XBP user. CONFIRMATION: Currently, the following types of special confirmation are available: <ol style="list-style-type: none"> Confirmation of intercepted jobs: CONFIRMATION = 'i'. Confirmation of child jobs: CONFIRMATION = 'c'. The JOBS table contains the jobs for which confirmation should be set. After BAPI_XBP_SPECIAL_CONFIRM_JOB is called, the JOBS table contains only those jobs, for which this operation failed, perhaps because the job was deleted or an internal error occurred.
Parameter (Output)	<ul style="list-style-type: none"> BAPIRET2 is the standard return structure containing return values of the function.
Tables	<ul style="list-style-type: none"> JOBS is a table with jobs, for which confirmation should be set.
MessageIDs	<ul style="list-style-type: none"> MSG_EXT_USER_MISSING: External user name is missing. MSG_CANT_LOG: Activity was terminated because the AS ABAP XMI logging mechanism returned an error. MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out. MSG_PROBLEM_DETECTED: Impossible to find out whether special functionality is turned on. MSG_INTERCEPTION_INACTIVE: Interception functionality is turned off. MSG_PARENTCHILD_INACTIVE: Parent/child functionality is turned off. MSG_JOB_CONFIRMATION_FAILED: Not all jobs were confirmed. The JOBS table contains the jobs that have not been confirmed. MSG_WRONG_CONFIRMATION_TYPE: The type of confirmation is wrong.

7.8.3 Modifying the Criteria Table for Interception

With the following function module the user can add/modify the table with the intercept criteria. Only jobs matching these criteria will be intercepted.

Function name	BAPI_XBP_MODIFY_CRITERIA_TABLE
Short description	<p>This function module is for modifying the criteria table (TBCICPT1) by replacing or updating its content with the content of the TBCICPT_TABLE table and returning the current content of the criteria table. With the APPEND indicator you can specify whether the content of the criteria table is replaced with the new content or the new information is appended at the end.</p> <p>If the table with new criteria is initial (empty) all the data from the TBCICPT1 table is deleted. When returning the content of the criteria table, the CONTENTS indicator should be set. In this case the function module returns a copy of the criteria table in TBCICPT_TABLE.</p>
Introduced in XBP 2.0	The entire function module was new in XBP 2.0.
BAPI object name	BackgroundJob
BAPI method name	
RFC interface	function BAPI_XBP_MODIFY_CRITERIA_TABLE


	<pre> importing EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 APPEND type CHAR1 DEFAULT 'X' CONTENTS type CHAR1 default SPACE exporting RETURN structure BAPIRET2 length 548 number of fields 14 tables TBCICPT_TABLE structure TBCICPT1 length 47 number of fields 3 </pre>
Parameter (Input)	<ul style="list-style-type: none"> EXTERNAL_USER_NAME is the name of the XBP user. APPEND is an indicator that determines whether the new criteria should be appended ('X') or replace old content (space). CONTENTS is an indicator that specifies that the current call is a request for the contents of the criteria table. In this case, the TBCICPT_TABLE parameter is used for returning a copy of the criteria table.
Parameter (Output)	BAPIRET2 is the standard return structure containing return values of the function.
Table	TBCICPT_TABLE is the table with new criteria information.
MessageIDs	<ul style="list-style-type: none"> MSG_DELETE_LINE_ERROR: Error deleting a line from the TBCICPT1 table. MSG_CANT_LOG: Activity was terminated because the AS ABAP XMI logging mechanism returned an error. MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.

7.9 Finding, Controlling, and Modifying Job Monitor Data

Using an external job management system, you can also monitor AS ABAP jobs, display job logs and spool lists and determine the parent/child relations. This can be done using the function modules below:

7.9.1 Determining the Status of a Job

Function name	BAPI_XBP_JOB_STATUS_GET
Short description	Determines the status of a job by reading AS ABAP information on the job.
Introduced in XBP 2.0	The new parameter HAS_CHILD has been added (see description below). In addition, the function can return now the status 'Intercepted'.
BAPI object name	BackgroundJob
BAPI method name	GetStatus
RFC interface	<pre>function BAPI_XBP_JOB_STATUS_GET importing JOBNAME like BAPIXMJOB-JOBNAME type RFC_CHAR length 32 JOBCOUNT like BAPIXMJOB-JOBCOUNT type RFC_CHAR length 8 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 exporting RETURN structure BAPIRET2 length 548 number of fields 14 STATUS like BAPIXMJOB-STATUS type RFC_CHAR length 1 HAS_CHILD like BAPIXMINFO-HAS_CHILD type RFC_CHAR length 1</pre>
Parameter (Input)	<ul style="list-style-type: none"> • JOBNAME is the name of the job whose status is to be determined. • JOBCOUNT is the ID number of the job whose status is to be determined. • EXTERNAL_USER_NAME is the name of the user in the external scheduler who called the function.
Parameter (Output)	<ul style="list-style-type: none"> • STATUS is the status of a job with the following possible values: <ul style="list-style-type: none"> 'R' - active 'I' - intercepted 'Y' - ready 'P' - scheduled 'S' - released 'A' - cancelled 'F' - finished • BAPIRET2 is the return structure used by BAPIs. • HAS_CHILD returns the information that specifies whether a job is a child, a parent, both, or neither. This is the parent/child information. The following values are possible as parent/child-relation:

	<p>'P' – job is parent/has children 'C' – job is child 'B' – ('both') job is parent and child '' - (blank) job is neither parent nor child</p>  <p>This information might change during the runtime of a job, because children are created at runtime. Before their creation they are not known to the system.</p>
MessageIDs	<ul style="list-style-type: none"> • MSG_JOBID_MISSING: You did not specify a job ID number. • MSG_JOBNAME_MISSING: You did not specify a job name. • MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered an error. • MSG_JOB_DOES_NOT_EXIST: Job does not exist in the AS ABAP system. • MSG_CANT_LOG: Activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out. • MSG_PARENT_CHILD_INCONSISTENCY: Inconsistency in the data found concerning parent/child-relation.

7.9.2 Determining the Status of a Job List

Function name	BAPI_XBP_JOBLIST_STATUS_GET
Short description	Determines the status of a list of jobs by reading AS ABAP information on all jobs.
Introduced in XBP 2.0	The entire function module was new in XBP 2.0.
BAPI object name	BackgroundJob
BAPI method name	GetAllStatus
RFC interface	<pre>function BAPI_XBP_JOBLIST_STATUS_GET importing EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 exporting RETURN structure BAPIRET2 length 548 number of fields 14 tables JOBLIST structure BAPIXMINIFO length 42 number of fields 4 (fields: jobname 32, jobcount 8, status 1, has_child 1)</pre>
Parameter (Input)	<ul style="list-style-type: none"> EXTERNAL_USER_NAME is the name of the user in the external scheduler who caused the function call.
Parameter (Output)	<ul style="list-style-type: none"> BAPIRET2 is the return structure used by BAPIs.
Tables	<ul style="list-style-type: none"> JOBLIST is a table containing the job status consisting of JOBNAME, JOBCOUNT, STATUS and HAS_CHILD. <ul style="list-style-type: none"> STATUS is the status of a job with the following possible values: <ul style="list-style-type: none"> 'R' - active 'I' - intercepted 'Y' - ready 'P' - scheduled 'S' - released 'A' - terminated 'F' - finished 'N' - Job does not exist Possible values for the parent/child-relation HAS_CHILD: <ul style="list-style-type: none"> 'P' – job is parent/has children 'C' – job is child 'B' – ('both') job is parent and child ' ' - (blank) job is neither parent nor child
MessageIDs	<ul style="list-style-type: none"> MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered an error. MSG_CANT_LOG: Activity was terminated because the AS ABAP XML logging mechanism returned an error. MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system. MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XML interface. Therefore, the activity cannot be carried out.

7.9.3 Reading Job Logs

This function has been enhanced after releasing **XBP 2.0**. The enhanced version contains a new optional importing parameter `PROT_NEW` and a new optional table `JOB_PROTOCOL_NEW`. With the enhanced version, it is possible to read the message type of a job log entry. See also note 603919.

The enhanced version has been released with the following support packages:

46B	SAPKB46B52
46C	SAPKB46C44
46D	SAPKB46D33
6.10	SAPKB61032
6.20	SAPKB62021

In XBP 3.0, before reading job logs or spool lists, the external scheduler can obtain additional information about the job log or the spool list size. It can get this information by calling `BAPI_XBP_JOB_DEFINITION_GET` or `BAPI_XBP_JOB_READ` which are enhanced with the additional export parameters `JOBLG_ATTR` and `SPOOL_ATTR` in XBP 3.0. For more information about the enhancements, see the description of the respective function module.

If `BAPI_XBP_JOB_DEFINITION_GET` or `BAPI_XBP_JOB_READ` return high values for the fields `JOBLG_ATTR-TMSSIZE` or `SPOOL_ATTR-TMSSIZE`, it is up to the scheduler to decide which function module to call – `BAPI_XBP_JOB_JOBLOG_READ` or `BAPI_XBP_JOB_SPOOLLIST_READ_20`. If either of the output parameters `JOBLG_ATTR` or `SPOOL_ATTR` return empty, then the external scheduler knows that the desired object does not exist. In addition, if `SPOOL_ATTR-DOCTYP` is not 'ALI', it is not necessary to call `BAPI_XBP_JOB_SPOOLLIST_READ_20`, as this function only returns ABAP lists.

Also new with XBP 3.0: With the parameter `LINES` and `DIRECTION` it is possible to read the first or last n lines of a job log. This functionality is generally available as of SAPKB70018. If you want to use this functionality as of SAPKB70014, you have to apply note 1167524.

Function name	BAPI_XBP_JOB_JOBLOG_READ
Short description	Get job log (also called job protocol) for a particular job.
Enhancements	See intro text to this section
BAPI object name	BackgroundJob
BAPI method name	ReadJoblog
RFC interface	<pre> function BAPI_XBP_JOB_JOBLOG_READ importing JOBNAME like BAPIXMJOB-JOBNAME type RFC_CHAR length 32 JOBCOUNT like BAPIXMJOB-JOBCOUNT type RFC_CHAR length 8 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 PROT_NEW like BTCH0000-CHAR1 (optional) type RFC_CHAR length 1 LINES TYPE BTCINT4 (optional) DIRECTION TYPE BTCCHAR1 (optional) exporting RETURN structure BAPIRET2 1 length 552 number of fields 14 </pre>

	<p>tables</p> <p>JOB_PROTOCOL structure BAPIXMPROT length 572 number of fields 14</p> <p>JOB_PROTOCOL_NEW structure TBTC5 (optional) length 573 number of fields 15</p>
Parameter (Input)	<ul style="list-style-type: none"> • JOBNAME is the name of the job whose job log should be retrieved. • JOBCOUNT is the job ID number of the job whose job log should be retrieved. • EXTERNAL_USER_NAME is the name of the user in the external scheduler who caused the function call. • PROT_NEW is an optional flag. If it is not set, the function behaves as before the enhancement. If the flag is set to X, the function returns the table JOB_PROTOCOL_NEW that, in addition to the old table JOB_PROTOCOL, contains also the message type. The table JOB_PROTOCOL is empty in this case. • LINES is the amount of lines and DIRECTION the reading direction (B=from beginning; E= from end). With these parameters it is possible to read the first or last n lines of a job log. This functionality is generally available as of SAPKB70018. If you want to use this functionality as of SAPKB70014, you have to apply note 1167524.
Parameter (Output)	<p>BAPIRET2 is the return structure used by BAPIs.</p>
Tables	<ul style="list-style-type: none"> • JOB_PROTOCOL is the table containing the job log. • JOB_PROTOCOL_NEW is a table filled only, if the flag PROT_NEW is set to X. The table JOB_PROTOCOL_NEW contains, in addition to the old table JOB_PROTOCOL, also the message type.
MessageIDs	<ul style="list-style-type: none"> • MSG_JOBID_MISSING: You have not entered a job ID number. • MSG_JOBNAME_MISSING: You did not specify a job name. • MSG_JOB_DOES_NOT_EXIST: Job does not exist in the AS ABAP system. • MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_NO_PRIVILEGE_GIVEN: The AS ABAP user used by the external management system to log onto the AS ABAP system, is not authorized to read the job log. • MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered an error. • MSG_NO_JOB_PROTOCOL: A log does not yet exist for the specified job. • MSG_JOB_PROTOCOL_IS_EMPTY: Job log is empty. • MSG_CANT_LOG: The activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.
Note	<ul style="list-style-type: none"> ○ The format of the job log (MsgId) has changed in SAP AS ABAP Release 4.0. ○ The term job protocol was common for what is now known as job log. Don't get confused. It is the same thing, but the correct

	tern is JOBLOG .
--	-------------------------

7.9.4 Reading the Spool List of a Job

With XBP 1.0 you had the following possibilities to read spool lists:

- BAPI_XBP_JOB_SPOOLLIST_READ for reading the spool list of a job that has been run.
- BAPI_XBP_SPOOLLIST_READ_RW for reading the spool list of a job in raw format, which is needed by some job scheduling tools to format lists correctly.

These functions modules can still be used as documented in the documentation to XBP 1.0 and XBP 2.0. However, as of XBP 2.0 there is a function available with which you can choose if the list should be read in raw format or not:

- BAPI_XBP_JOB_SPOOLLIST_READ_20 contains a raw format indicator.

Also note that as of XBP 3.0, before reading job logs or spool lists, the external scheduler can obtain additional information about the job log or the spool list size. It can get this information by calling BAPI_XBP_JOB_DEFINITION_GET or BAPI_XBP_JOB_READ which are enhanced with the additional export parameters JOBLG_ATTR and SPOOL_ATTR in XBP 3.0. For more information about the enhancements, see the description of the respective function module.

If BAPI_XBP_JOB_DEFINITION_GET or BAPI_XBP_JOB_READ return high values for the fields JOBLG_ATTR-TMSSIZE or SPOOL_ATTR-TMSSIZE, it is up to the scheduler to decide which function module to call – BAPI_XBP_JOB_JOBLOG_READ or BAPI_XBP_JOB_SPOOLLIST_READ_20. If either of the output parameters JOBLG_ATTR or SPOOL_ATTR returns empty, then the external scheduler knows that the desired object does not exist. In addition, if SPOOL_ATTR-DOCTYP is not 'ALI', it is not necessary to call BAPI_XBP_JOB_SPOOLLIST_READ_20, as this function only returns ABAP lists.



Note that with XBP 3.0, the SAP system stores all the information about the spool lists created by a job. In former versions, only information about the last spool list created by a job step was stored. It is now possible to pass the number of a particular spool list and the system will return information about the spool list (see section 7.9.5) or return its content (see section 7.9.6).

Function name	BAPI_XBP_JOB_SPOOLLIST_READ_20
Short description	Read spool list of a job that has been run. Some job scheduling tools need the raw format to format lists correctly. By setting the corresponding indicator, you can read the spool list also in raw format.
Enhanced in XBP 3.0	In XBP 3.0, the function module provides a new table parameter SPOOL_LIST_PLAIN with a size of 1024 characters. In Release 4.6C, the parameter USE_SPOOL_LIST_PLAIN has to be passed with the call of BAPI_XBP_JOB_SPOOLLIST_READ_20 only if the new narrower structure is filled. With the patches listed in note 1352587, two new importing parameters FIRST_PAGE and LAST_PAGE are introduced. It is now possible to transfer the spool lists in packages. Please note that note 1299738 must be installed in the system. The total page number can be retrieved with BAPI_XBP:JOB_DEFINITION_GET, BAPI_XBP_JOB_READ, or BAPI_XBP_GET_SPOOL_ATTRIBUTES.
BAPI method name	ReadSpoollist
RFC interface	function BAPI_XBP_JOB_SPOOLLIST_READ_20 importing JOBNAME like BAPIXMJOB-JOBNAME

	<pre> type RFC_CHAR length 32 JOB_COUNT like BAPIXMJOB-JOB_COUNT type RFC_CHAR length 8 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 STEP_NUMBER like BAPIXMJOB-STEP_COUNT type RFC_INT4 length 4 RAW like BTCH0000-Char1 (optional) PLAIN LIKE BTCH0000-CHAR1 (optional) FIRST_PAGE TYPE RSPOPAGES (optional) LAST_PAGE TYPE RSPOPAGES (optional) exporting RETURN structure BAPIRET2 length 548 number of fields 14 tables SPOOL_LIST structure BAPIXMSPOW length 4096 SPOOL_LIST_PLAIN structure BAPIXMSPOP Optional length 1024 </pre>
Parameter (Input)	<ul style="list-style-type: none"> • JOBNAME is the name of the job whose spool list should be retrieved. • JOB_COUNT is the job ID number of the job whose spool list should be retrieved. • EXTERNAL_USER_NAME is the name of the user in the external scheduler who caused the function call. • STEP_NUMBER determines from which job step the spool list should be retrieved. • RAW is the raw format indicator. If this indicator is set, you can read the spool list in raw format. • PLAIN indicates that the spool list should be returned in table SPOOL_LIST_PLAIN (recommended) • FIRST_PAGE is the first page of the area to be transferred • LAST_PAGE is the last page of the area to be transferred
Parameter (Output)	<ul style="list-style-type: none"> • BAPIRET2 is the return structure used by BAPIs.
Tables	<ul style="list-style-type: none"> • SPOOL_LIST is the internal table containing the spool list with a length of 4096 characters. • SPOOL_LIST_PLAIN is the internal table with a length of 1024 characters.
MessageIDs	<ul style="list-style-type: none"> • MSG_JOBID_MISSING: You have not entered a job ID number. • MSG_JOBNAME_MISSING: You did not specify a job name. • MSG_JOB_DOES_NOT_EXIST: Job does not exist in the AS ABAP system. • MSG_JOB_DOESNT_HAVE_STEPS: Job has no steps. • MSG_INVALID_STEP_COUNT: The job does not contain a step with the given number. • MSG_STEP_COUNT_MISSING: No step number was specified. • MSG_INVALID_SPOOLID: The spool ID contained in the step is invalid. • MSG_NO_SPOOLLIST: The spool request belonging to the step does not contain an ABAP list. The spool request cannot be

	<p>displayed.</p> <ul style="list-style-type: none"> • MSG_EXT_USER_MISSING: The name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_PRIVILEGE_MISSING: The user used by the external management system to log onto the AS ABAP system is not authorized to read the spool list. • MSG_NO_PRIVILEGE_GIVEN: The user used by the external management system to log onto the AS ABAP system is not authorized to read the spool list. • MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered an error. • MSG_CANT_LOG: The activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.
--	--

7.9.5 Getting Information on a Particular Spool List

With XBP 3.0, the SAP System stores all the information about the spool lists created by a job. In former versions, only information about the last spool list created by a job step was stored. It is now possible to pass the number of a particular spool list and the system will return information about the spool list.

Function name	BAPI_XBP_GET_SPOOL_ATTRIBUTES
Short description	Get information about a particular spool list.
New in XBP 3.0	The entire function module is new in XBP 3.0.
BAPI method name	
RFC interface	<pre> FUNCTION BAPI_XBP_GET_SPOOL_ATTRIBUTES IMPORTING SPOOL_REQUEST TYPE RSPOID type INT4 length 10 EXTERNAL_USER_NAME TYPE BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 EXPORTING RETURN structure BAPIRET2 length 548 number of fields 14 SPOOL_ATTR structure BAPIXMSPOOLID length 364 number of fields 42 </pre>
Parameter (Input)	<ul style="list-style-type: none"> • SPOOL_REQUEST is the number of a spool request in the SAP System. • EXTERNAL_USER_NAME is the name of the user in the external scheduler who caused the function call.
Parameter (Output)	<ul style="list-style-type: none"> • BAPIRET2 is the return structure used by BAPIs. • SPOOL_ATTR returns information about the spool list(s) created by the job. For the fields of the parameter, see the <i>Fields of the SPOOL_ATTR Parameter</i> table (section 7.2.5.2).
MessageIDs	<ul style="list-style-type: none"> • MSG_PARAM_MISSING: No number of a spool request was passed. • MSG_INVALID_SPOOLID: The spool ID contained in the step is invalid. • MSG_NO_SPOOLLIST: The spool request belonging to the step

	<p>does not contain an ABAP list. The spool request cannot be displayed.</p> <ul style="list-style-type: none"> • MSG_EXT_USER_MISSING: The name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_SPOOL_SELECTION_EMPTY: The system could not retrieve the information about the specified request. • MSG_PRIVILEGE_MISSING: The user used by the external management system to log onto the AS ABAP system is not authorized to read the spool list. • MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered an error. • MSG_CANT_LOG: The activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.
--	--

7.9.6 Reading a Particular Spool List

With XBP 3.0, the SAP System stores all the information about the spool lists created by a job. In former versions, only information about the last spool list created by a job step was stored. It is now possible to pass the number of a particular spool list and the system will return its content.

Function name	BAPI_XBP_JOB_READ_SINGLE_SPOOL
Short description	<p>Read a particular spool list of a job that has been run.</p> <p>Some job scheduling tools need the raw format to format lists correctly. By setting the corresponding indicator, you can read the spool list also in raw format.</p>
New in XBP 3.0	<p>The entire function module is new in XBP 3.0.</p> <p>With the patches listed in note 1352587, two new importing parameters FIRST_PAGE and LAST_PAGE are introduced. It is now possible to transfer the spool lists in packages. Please note that note 1299738 must be installed in the system.</p> <p>The total page number can be retrieved with BAPI_XBP:JOB_DEFINITION_GET, BAPI_XBP_JOB_READ, or BAPI_XBP_GET_SPOOL_ATTRIBUTES.</p>
BAPI method name	
RFC interface	<pre> Function BAPI_XBP_JOB_READ_SINGLE_SPOOL importing SPOOL_REQUEST TYPE RSPOID type INT4 length 10 EXTERNAL_USER_NAME TYPE BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 RAW like BTCH0000-Char1 Optional type RFC_CHAR length 1 FIRST_PAGE TYPE RSPOPAGES (optional) LAST_PAGE TYPE RSPOPAGES (optional) exporting RETURN structure BAPIRET2 length 548 number of fields 14 tables </pre>

	<p>SPOOL_LIST structure BAPIXMSPW optional length 4096 SPOOL_LIST_PLAIN structure BAPIXMSPW optional length 1024</p>
Parameter (Input)	<ul style="list-style-type: none"> • SPOOL_REQUEST is the number of a spool request in the SAP system. • EXTERNAL_USER_NAME is the name of the user in the external scheduler who caused the function call. • RAW is the raw format indicator. If this indicator is set, you can read the spool list in raw format. • FIRST_PAGE is the first page of the area to be transferred • LAST_PAGE is the last page of the area to be transferred
Parameter (Output)	<ul style="list-style-type: none"> • BAPIRET2 is the return structure used by BAPIs.
Tables	<ul style="list-style-type: none"> • SPOOL_LIST is the internal table containing the spool list with a length of 4096 characters. • SPOOL_LIST_PLAIN is the internal table with a length of 1024 characters.
MessageIDs	<ul style="list-style-type: none"> • MSG_PARAM_MISSING: No number of a spool request was passed. • MSG_INVALID_SPOOLID: The spool ID contained in the step is invalid. • MSG_NO_SPOOLLIST: The spool request belonging to the step does not contain an ABAP list. The spool request cannot be displayed. • MSG_EXT_USER_MISSING: The name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_PRIVILEGE_MISSING: The user used by the external management system to log on.to the AS ABAP system is not authorized to read the spool list. • MSG_NO_PRIVILEGE_GIVEN: The user used by the external management system to log on.to the AS ABAP system is not authorized to read the spool list. • MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered an error. • MSG_CANT_LOG: The activity was terminated because the AS ABAP XML logging mechanism returned an error. • MSG_NOT_LOGGED_ON: The external management tool has not logged on.to the CCMS XML interface. Therefore, the activity cannot be carried out.

7.9.7 Checking the Status of a Job

Problems with the SAP system (database, network, termination of background work processes) can cause discrepancies between the actual status of a job and its recorded status in the database.



A background work process, with an active job, is terminated manually. The job runtime system cannot set the job status in the database to 'terminated'.
The function module BAPI_XBP_JOB_STATUS_CHECK recognizes these cases and corrects the job status accordingly.

Function name	BAPI_XBP_JOB_STATUS_CHECK
Short description	Check whether the internally displayed status is up to date.
BAPI object name	BackgroundJob
BAPI method name	CheckStatus
RFC interface	<pre>function BAPI_XBP_JOB_STATUS_CHECK importing JOBNAME like BAPIXMJOB-JOBNAME type RFC_CHAR length 32 JOBCOUNT like BAPIXMJOB-JOBCOUNT type RFC_CHAR length 8 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 exporting ACTUAL_STATUS like BAPIXMJOB-STATUS type RFC_CHAR length 1 RETURN structure BAPIRET2 length 548 number of fields 14 STATUS_ACCORDING_TO_DB like BAPIXMJOB-STATUS type RFC_CHAR length 1</pre>
Parameter (Input)	<ul style="list-style-type: none"> • JOBNAME is the name of the job whose status should be checked. • JOBCOUNT is the job ID number of the job whose status should be checked. • EXTERNAL_USER_NAME is the name of the user in the external scheduler who caused the function call.
Parameter (Output)	<ul style="list-style-type: none"> • STATUS_ACCORDING_TO_DB contains the status of a job in the database. • ACTUAL_STATUS is the actual job status with following possible values: 'R' - active 'Y' - ready 'P' - scheduled 'S' - released 'A' - terminated 'F' - finished 'X' - actual status cannot be determined • BAPIRET2 is the return structure used by BAPIs.
MessageIDs	<ul style="list-style-type: none"> • MSG_JOBID_MISSING: You did not specify a job ID number. • MSG_JOBNAME_MISSING: You did not specify a job name. • MSG_JOB_DOES_NOT_EXIST: Job does not exist in the AS ABAP system. • MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered an error. • MSG_PRIVILEGE_MISSING: The SAP user used by the external job management system to log onto the AS ABAP system is not authorized to use this function.

	<ul style="list-style-type: none">• MSG_CANT_LOG: The activity was terminated because the AS ABAP XMI logging mechanism returned an error.• MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.
--	---

7.9.8 Selecting Jobs

Function name	BAPI_XBP_JOB_SELECT
Short description	<p>You can use the function module BAPI_XBP_JOB_SELECT to select a set of jobs in the AS ABAP system that match the selection criteria given. At very least the username and the job name must be partly (using wildcards) specified.</p> <p>From XBP 2.0 on, it is possible to reduce the job list by selecting only non-confirmed jobs (see BAPI_XBP_CONFIRM_JOB and BAPI_XBP_SPECIAL_CONFIRM_JOB).</p>
Introduced in XBP 2.0	The new parameter SELECTION has been added (see description below).
BAPI object name	BackgroundJob
BAPI method name	Select
RFC interface	<pre>function BAPI_XBP_JOB_SELECT importing EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 JOB_SELECT_PARAM structure BAPIXMJSEL length 196 number of fields 18 SYSTEMID like SY-SYSID optional type RFC_CHAR length 8 SELECTION optional type CHAR2 default 'AL' exporting RETURN structure BAPIRET2 length 548 number of fields 14 tables SELECTED_JOBS structure BAPIXMJOBS length 40, number of fields 2 JOB_HEAD STRUCTURE BAPIXMJOB optional length 388 number of fields 35</pre>
Parameter (Input)	<ul style="list-style-type: none"> EXTERNAL_USER_NAME is the name of the XBP user. JOB_SELECT_PARAM are the selection parameters. Use this structure to transfer the job selection criteria to the background processing system. SYSTEMID is the system ID. Parameter SELECTION is optional and case-insensitive, and specifies what kind of confirmation (general or child) should be taken into account. The following values are possible: <ol style="list-style-type: none"> AL (default) returns all child jobs regardless what confirmation they have. NG returns only those child jobs that do NOT have general confirmation. NC returns only those child jobs that do NOT have any confirmation. <p>To make a special confirmation for a child job, use the BAPI_XBP_SPECIAL_CONFIRM_JOB function module with CONFIRMATION='C'.</p> <p>To make a general confirmation for a child job, use the BAPI_XBP_CONFIRM_JOB function module.</p>

Parameter (Output)	<ul style="list-style-type: none"> • BAPIRET2 is the return structure used by BAPIs.
Tables	<ul style="list-style-type: none"> • SELECTED_JOBS is the list of selected jobs. • JOB_HEAD lists job headers of selected jobs.
MessageIDs	<ul style="list-style-type: none"> • MSG_SELECT_PARAM_MISSING: You have not specified select options correctly. • MSG_SELECT_JOBNAME_MISSING: You have not specified a job name. • MSG_SELECT_USERNAME_MISSING: You have not specified a user name. • MSG_NO_JOB_FOUND: No jobs with the given name were found. • MSG_PROBLEM_DETECTED: The AS ABAP job system has found an error. • MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_CANT_LOG: Activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out. • MSG_WRONG_SELECTION_PAR: Inconsistent selection parameters.

7.9.9 Determining the Number of Jobs with Particular Job Names

Function name	BAPI_XBP_JOB_COUNT
Short description	You can use the function module BAPI_XBP_JOB_COUNT to determine the number of jobs which are defined in the AS ABAP system with a particular job name.
BAPI object name	BackgroundJob
BAPI method name	CountByName
RFC interface	<pre>function BAPI_XBP_JOB_COUNT importing JOBNAME like BAPIXMJOB-JOBNAME type RFC_CHAR length 32 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 exporting NUMBER_OF_JOBS like BAPIXMAUX-INT4 type RFC_INT4 length 4 RETURN structure BAPIRET2 length 548 number of fields 14 tables JOB_TABLE structure BAPIXMJOB length 388 number of fields 35</pre>
Parameter (Input)	<ul style="list-style-type: none"> • JOBNAME is the name of a background job. • EXTERNAL_USER_NAME is the name of the user in the external scheduler who caused the function call.
Parameter (Output)	<ul style="list-style-type: none"> • NUMBER_OF_JOBS is the number of jobs found. • JOB_TABLE lists the resources available for background processing for each instance. • BAPIRET2 is the return structure used by BAPIs.
MessageIDs	<ul style="list-style-type: none"> • MSG_JOBNAME_MISSING: You have not specified a job name. • MSG_NO_JOB_FOUND: No jobs with the given name were found. • MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has found an error. • MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_CANT_LOG: Activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.

7.9.10 Obtaining Key Job Parameters from Job Headers and Steps

Function name	BAPI_XBP_JOB_READ
Short description	This function module is intended for obtaining key job parameters from job header and job steps.
Enhanced in XBP 3.0	<p>This function module is enhanced to return more information about :</p> <ul style="list-style-type: none"> • The job log of a job. Information is returned by the new export parameter JOBLG_ATTR. • The spool list(s) created by a job. Information is returned by the new table parameter SPOOL_ATTR. • Spool list recipient of a job in plain text. <p>The two parameters provide the external scheduler with more information about the size of the job log or spool list.</p> <p>The function module BAPI_XBP_JOB_DEFINITION_GET is enhanced with the same two parameters.</p> <p>If BAPI_XBP_JOB_DEFINITION_GET or BAPI_XBP_JOB_READ return high values for the fields JOBLG_ATTR-TMSSIZE or SPOOL_ATTR-TMSSIZE that contain information about the size of the job log or spool list, it is up to the scheduler to decide which function module to call – BAPI_XBP_JOB_JOBLOG_READ or BAPI_XBP_JOB_SPOOLLIST_READ_20. If either of the output parameters JOBLG_ATTR or SPOOL_ATTR return empty, then the external scheduler knows that the desired object does not exist. In addition, if SPOOL_ATTR-DOCTYP is not 'ALI', it is not necessary to call BAPI_XBP_JOB_SPOOLLIST_READ_20, as this function only returns ABAP lists.</p> <p>New with note 1402400: The function can now also return the name of a batch target server group and not only the cryptic GUID.</p>
BAPI object name	BackgroundJob
BAPI method name	ReadJob
RFC interface	<pre>function BAPI_XBP_JOB_READ importing JOBNAME like BAPIXMJOB-JOBNAME type RFC_CHAR length 32 JOBCOUNT like BAPIXMJOB-JOBCOUNT type RFC_CHAR length 8 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 JOB_HEADER_ONLY like BAPIXMINFO-JOBHEADER optional type RFC_CHAR length 1 STEP_NUMBER like BAPIXMJOB-STEPCOUNT optional type RFC_INT4 length 4 SHOW_GROUPNAME type BTCH0000-CHAR1 optional (new with note 1402400) exporting JOBHEAD structure BP20JOB length 668 number of fields 58 RETURN structure BAPIRET2 length 548 number of fields 14 JOBLG_ATTR structure BAPIXMJOBLOG</pre>

	<p>length 38 number of fields 5 RECIPIENT structure BAPIXMRECIP length 249 number of fields 9</p> <p>GROUPNAME type BPSRVGRP (new with note 1402400)</p> <p>tables STEPS structure BP20STEP optional length 1028 number of fields 63 SPOOL_ATTR structure BAPIXMSPPOOLID OPTIONAL length 364 number of fields 42</p>
Parameter (Input)	<ul style="list-style-type: none"> • JOBNAME is the name of the job whose data need to be read. • JOBCOUNT is the ID number of the job whose data need to be read. • EXTERNAL_USER_NAME is the name of the XBP user. • JOB_HEADER_ONLY (optional) is an indicator that allows a choice between two options: retrieve job header only or retrieve job header and step descriptions. • STEP_NUMBER is the number of the step whose information should be retrieved. If STEP_NUMBER is 0 (zero), then information on all steps is retrieved.
Parameter (Output)	<ul style="list-style-type: none"> • BAPIRET2 is the return structure used by BAPIs. • JOBHEAD is the return structure containing the job header. • JOBLOG_ATTR returns information about the job log of the job. For details on the structure fields, see section 7.2.5.1. • SPOOL_ATTR is a table parameter returning information about the spool list(s) created by the job. For details on the structure fields see section 7.2.5.2. • RECIPIENT returns the spool list recipient of a job in plain text.
Tables	<ul style="list-style-type: none"> • STEPS is a table consisting of the requested steps of the job.
MessageIDs	<ul style="list-style-type: none"> • MSG_JOBNAME_MISSING : You have not specified a job name. • MSG_JOBID_MISSING : You have not entered a job ID. • MSG_EXT_USER_MISSING : External user name is missing. • MSG_CANT_LOG : Activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_NOT_LOGGED_ON : The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out. • MSG_JOB_DOES_NOT_EXIST: Specified job does not exist. • MSG_JOB_NOSTEPS: There are no steps in the job. • MSG_JOB_DOESNT_HAVE_THIS_STEP: Job does not have the specified step.


7.9.11 Determining Job Children

In general, a business process that is executed by a job, or rather a collection of jobs, does not only consist of static jobs, which are known in advance, but also of jobs that are created on the fly by the static jobs, such as to dynamically distribute workload. A

job that is released by another job is called a child job and the releasing job is called a parent job.

For a job scheduling system it is important to know about the existence and current status of the child jobs of a certain parent job, because in the internal logic of many applications a parent job is considered as 'finished' only if the parent job itself **and** its child jobs are finished.

Up to now there is no proper way for an external scheduler to find out whether or not a job has child jobs. XBP 2.0 offers functionality to find all children created by a job.


Function name	BAPI_XBP_JOB_CHILDREN_GET
Short description	<p>Get all children created by a job and return them in an internal table. Only the children are returned, not the grandchildren.</p>  <p>Note that this function returns only the children that have already been created at the time of the call. If the function is called while the job is still running, more children might be created after the call.</p> <p>A child job will be returned even if it has been deleted before the call. Therefore, a complete history of the job's children can be retrieved.</p>
Introduced in XBP 2.0	The entire function module was new to XBP 2.0.
BAPI object name	BackgroundJob
BAPI method name	GetChildren
RFC interface	<pre>function BAPI_XBP_JOB_CHILDREN_GET importing JOBCOUNT like BAPIXMJOB-JOBCOUNT type RFC_CHAR length 8 JOBNAME like BAPIXMJOB-JOBNAME type RFC_CHAR length 32 SELECTION type CHAR2 default 'AL' type RFC_CHAR length 2 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 exporting NR_OF_CHILDREN type RFC_INT4 length 4 RETURN structure BAPIRET2 length 548 number of fields 14 tables JOB_CHILDREN structure BAPIXMJOBS length 40, number of fields 2 (jobcount, jobname)</pre>
Detailed description	<p>This function module retrieves jobs that have CHILD status. Since you do not want to retrieve the names of the same jobs over and over again, you can reduce the list of the returned jobs to the ones which are not known to the caller. For this, confirmation is used. The list of returned jobs may contain all child jobs or only those that have not been confirmed with one or more confirmations.</p>
Parameter (Input)	<ul style="list-style-type: none"> • JOBCOUNT and JOBNAME of the job whose children are to be determined.

	<ul style="list-style-type: none"> EXTERNAL_USER_NAME is the name of the XBP user. Parameter SELECTION is optional and incase-sensitive, and specifies what kind of confirmation (general or child) should be taken into account. The following values are possible: <ol style="list-style-type: none"> AL (default) returns all child jobs regardless what confirmation they have. NG returns only those child jobs that do NOT have general confirmation. NS returns only those child jobs that were NOT confirmed as child jobs. NC returns only those child jobs that do NOT have any confirmation. <p>To make a special confirmation for a child job, use the BAPI_XBP_SPECIAL_CONFIRM_JOB function module with CONFIRMATION='C'. To make a general confirmation for a child job, use the BAPI_XBP_CONFIRM_JOB function module.</p>
Parameter (Output)	<ul style="list-style-type: none"> NR_OF_CHILDREN of the given job. BAPIRET2 is the return structure used by BAPIs.
Tables	<ul style="list-style-type: none"> JOB_CHILDREN: Internal table containing the jobs consisting of JOBNAME and JOBCOUNT.
MessageIDs	<ul style="list-style-type: none"> MSG_JOBID_MISSING: You have not entered a job ID number. MSG_JOBNAME_MISSING: You have not entered a job name. MSG_JOB_DOES_NOT_EXIST: Job does not exist in the AS ABAP system. MSG_EXT_USER_MISSING: The name of the external user is missing. This is the name of a user in the external job scheduling system. MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered an error. MSG_WRONG_SELECTION_PAR: Inconsistent selection parameters. MSG_CANT_LOG: The activity was terminated because the AS ABAP XMI logging mechanism returned an error. MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.

7.9.12 Determining Parent/Child Relation

You can use the following function to get information about the parent/child relations of any job.

Function name	BAPI_XBP_JOB_PARENT_CHILD_INFO
Short description	<p>This function returns the child data of a given job (see BAPI_XBP_JOB_STATUS_GET) and the following information :</p> <ul style="list-style-type: none"> If this job is a child, job ID number and job name of the parent job are returned. This is the only function that retrieves the parent of a job. If this job is a parent, the number of children is returned.

	 <p>Note: The child information and the number of children may change during the runtime of a job.</p>
Introduced in XBP 2.0	The entire function module was new to XBP 2.0.
BAPI object name	BackgroundJob
BAPI method name	
RFC interface	<pre>function BAPI_XBP_JOB_PARENT_CHILD_INFO importing JOBNAME like BAPIXMJOB-JOBNAME type RFC_CHAR length 32 JOBCOUNT like BAPIXMJOB-JOBCOUNT type RFC_CHAR length 8 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 exporting HAS_CHILD like BAPIXMINFO-HAS_CHILD type RFC_CHAR length 1 PARENT_NAME like BAPIXMJOB-JOBNAME type RFC_CHAR length 32 PARENT_COUNT like BAPIXMJOB-JOBCOUNT type RFC_CHAR length 8 NR_OF_CHILDREN type INT4 length 4 RETURN structure BAPIRET2 length 548 number of fields 14</pre>
Parameter (Input)	<ul style="list-style-type: none"> • JOBNAME, JOBCOUNT of the job, for which the parent/child information is to be determined. • EXTERNAL_USER_NAME is the name of the XBP user.
Parameter (Output)	<ul style="list-style-type: none"> • Possible values for the parent/child-relation HAS_CHILD: <ul style="list-style-type: none"> 'P' – job is parent/has children 'C' – job is child 'B' – ('both') job is parent and child ' ' - (blank) job is neither parent nor child • PARENT_NAME, PARENT_COUNT of the parent job, if there is one. • NR_OF_CHILDREN of this job at the time of the call. • BAPIRET2 is the return structure used by BAPIs.

MessageIDs	<ul style="list-style-type: none">• MSG_JOBID_MISSING: You did not specify a job ID number.• MSG_JOBNAME_MISSING: You did not specify a job name.• MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered an error.• MSG_CANT_LOG: Activity was terminated, because the AS ABAP XMI logging mechanism returned an error.• MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system.• MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.• MSG_JOB_DOES_NOT_EXIST: The job does not exist in the AS ABAP database.
------------	--

7.9.13 Reading and Changing Intercept Status and Parent/Child Relation

Function name	BAPI_XBP_NEW_FUNC_CHECK
Short description	This function module is intended for reading and changing the status of interception and parent-child functionality.
Introduced in XBP 2.0	The entire function module was new to XBP 2.0.
BAPI object name	BackgroundJob
BAPI method name	
RFC interface	<p>Function BAPI_XBP_NEW_FUNC_CHECK</p> <p>Importing parameters INTERCEPTION_ACTION type CHAR1 OPTIONAL PARENTCHILD_ACTION type CHAR1 OPTIONAL EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16</p> <p>Exporting parameters INTERCEPTION TYPE CHAR1 PARENTCHILD TYPE CHAR1 RETURN structure BAPIRET2 length 548 number of fields 14</p>
Detailed function description	<p>This function module is for reading and changing the status of interception and parent-child functionality. It receives a request to execute an action and returns the statuses as results for this action. A status is a value that is either INITIAL (if functionality is switched off) or 'X'. An action can be one of the following values:</p> <ul style="list-style-type: none"> • 'R' or 'r' or blank for reading the current status • 'S' or 's' for setting the status ON (as with XBP 2.0) • Additionally '3' for INTERCEPTION_ACTION to set the status ON for interception rules set in the Criteria Manager of XBP 3.0. • 'C' or 'c' for removing ON (setting the status OFF) <p>Actions are passed with the optional parameters INTERCEPTION_ACTION and PARENTCHILD_ACTION.</p>
Parameter (Input)	<ul style="list-style-type: none"> • INTERCEPTION_ACTION specifies whether the status of the interception functionality should be read, set, or cleared. See detailed function description above for choosing the correct action. • PARENTCHILD_ACTION denotes whether the status of parent/child functionality should be read, set, or cleared. See detailed function description above for choosing the correct action. • INTERCEPTION_ACTION and PARENTCHILD_ACTION are independent from each other. Statuses are returned via the INTERCEPTION and PARENTCHILD parameters. • EXTERNAL_USER_NAME is the name of the XBP user.
Parameter (Output)	<ul style="list-style-type: none"> • INTERCEPTION is the current status of interception functionality. • PARENTCHILD is the current status of parent-child functionality. • BAPIRET2 is the return structure used by BAPIs.

MessageIDs	<ul style="list-style-type: none"> • MSG_EXT_USER_MISSING : External user name is missing (initial). • MSG_CANT_LOG : Activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_NOT_LOGGED_ON : The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out. • MSG_WRONG_ACTION appears when the name of at least one action is invalid. • MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered an error.
------------	--

7.9.14 Obtaining Application Information

7.9.14.1 Getting the Handle of an Application Log

You can use the following function to get the handle of an application log. By using this handle it can read the application log and the application return code. The application should call in advance the new internal function BP_ADD_APPL_LOG_HANDLE to assign one or more log handles to its batch job data.

Function name	BAPI_XBP_APPL_INFO_GET
Short description	<p>This function module gets handles of the application log and return codes for a specific job. By calling BP_ADD_APPL_LOG_HANDLE, the application should assign a handle to its job data in the runtime.</p> <p>The application log handle returned by BAPI_XBP_APPL_INFO_GET can be used as an input parameter in the function module BAPI_XBP_APPL_CONTENT_GET described in section 7.9.14.2.</p>
New in XBP 3.0	The entire function module is new in XBP 3.0.
BAPI object name	BackgroundJob
BAPI method name	GetApplicationInfo
RFC interface	<pre> FUNCTION: BAPI_XBP_APPL_INFO_GET IMPORTING EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type CHAR length 16 I_JOBNAME like TBTCO-JOBNAME type CHAR length 32 I_JOBCOUNT like TBTCO-JOBCOUNT type CHAR length 8 EXPORTING E_T_LOGHANDLES TYPE BTC_T_LOGHANDLE E_T_RETURN_CODES TYPE BTC_T_APPRC E_JOB_STATUS TYPE BTCSTATUS type CHAR length 1 RETURN structure BAPIRET2 length 548 number of fields 14 </pre>
Parameter (Input)	<ul style="list-style-type: none"> • I_EXTERNAL_USER_NAME is the name of the user in the external scheduler who called the function. • I_JOBNAME is the name of a background job.

	<ul style="list-style-type: none"> • I_JOB_COUNT is the ID number of a job. Together with the job name, the job number identifies the job uniquely.
Parameter (Output)	<ul style="list-style-type: none"> • E_T_LOGHANDLES is a table containing the log handles of the job steps. Each log handle is a GUID that uniquely identifies a log. By using this log handle, the external scheduler can access the log. • E_T_RETURN_CODES is a table containing the return codes of the job steps. • E_JOBSTATUS is the status of the job. • BAPIRET2 is the return structure used by BAPIs.

7.9.14.2 Getting the Content of the Application Log for a Particular Log Handle

Function name	BAPI_XBP_APPL_LOG_CONTENT_GET
Short description	This function module gets the content of the application log for a particular log handle. The application log handle returned by BAPI_XBP_APPL_INFO_GET (described in section 7.9.14.1) can be used as an input parameter.
New in XBP 3.0	The entire function module is new in XBP 3.0.
BAPI object name	BackgroundJob
BAPI method name	GetApplicationLogContent
RFC interface	<pre> FUNCTION BAPI_XBP_APPL_LOG_CONTENT_GET. IMPORTING EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type CHAR length 16 I_LOGHANDLE like BALLOGHNDL type CHAR length 22 I_MESSAGES_FROM type INT4 length 10 default 1 I_MESSAGES_TO type INT4 length 10 default 1000 EXPORTING E_TXT_OBJECT like BAPIBALTEXT type CHAR length 64 E_TXT_SUBOBJECT like BAPIBALTEXT (see above) E_TXT_ALTCODE like BAPIBALTEXT (see above) E_TXT_ALMODE like BAPIBALTEXT (see above) E_TXT_ALSTATE like BAPIBALTEXT (see above) E_TXT_PROBCLASS like BAPIBALTEXT (see above) E_TXT_DEL_BEFORE like BAPIBALTEXT (see above) E_BAL_HEADER structure BAL_S_LOG length 263 number of fields 17 E_BAL_STATISTIC structure BAL_S_SCNT length 101 number of fields 11 E_T_MESSAGES like BAPI_T_APPLOG_MESSAGE (see section 7.9.14.3 below) RETURN structure BAPIRET2 length 548 number of fields 14 </pre>
Parameter	<ul style="list-style-type: none"> • I_EXTERNAL_USER_NAME is the name of the user in the external scheduler

(Input)	<p>who called the function.</p> <ul style="list-style-type: none"> • I_LOGHANDLE reads from the table E_T_LOGHANDLES which is exported by BAPI_XBP_APPL_INFO_GET and contains the log handles of the job steps. • I_MESSAGES_FROM specifies the number of the line in the log message from which the external scheduler should start reading the log. The default value is 1. • I_MESSAGES_TO specifies the number of the line in the log message at which the external scheduler should stop reading the log. The default value is 1000. • The I_MESSAGES_FROM and I_MESSAGES_TO parameters allow the external scheduler to read long application logs in chunks to improve performance.
Parameter (Output)	<ul style="list-style-type: none"> • The E_TXT_* output parameters return additional information about the application log. The data they return is from the log's header. • E_TXT_OBJECT and E_TXT_SUBOBJECT provide a description of the application and/or sub-application that use the application log. • E_TXT_ALTCODE specifies the transaction or program which created the application log. • E_TXT_ALMODE specifies the processing mode in which the log was created (dialog, background, and so on). • E_TXT_ALSTATE specifies the status of the log, that is, whether the log was finished or not. • E_TXT_PROBCLASS provides the level of importance of the log messages. • E_TXT_DEL_BEFORE is a flag which denotes whether the log can be deleted before its expiry date. • E_BAL_HEADER returns the application log header. • E_BAL_STATISTIC returns the amount of log messages grouped by severity. The possible values for the severity are A = "Abort", E = "Error", W = "Warning", I = Information, S= "Status". • E_T_MESSAGES returns a table with log messages. • BAPIRET2 is the return structure used by BAPIs.

7.9.14.3 Table with Application Log Messages (BAPI_T_APPLOG_MESESAGE)

Field name	Type	Description
BAL_MESSAGE	BAL_S_MSG	Application log message in DB format
TXT_MSGTY	BAPIBALTEXT	Message type in text form in logon language
TXT_MSGID	BAPIBALTEXT	Message id in text form in logon language
TXT_DETLEVEL	BAPIBALTEXT	Detail level in text form in logon language
TXT_PROBCLASS	BAPIBALTEXT	Problem class in text form in logon language
TXT_MSG	CHAR255	Message text in logon language

7.9.15 Monitoring Performance

Function name	BAPI_XBP_BTC_STATISTIC_GET
Short description	By calling this function module, the external scheduler can retrieve statistic information about the past workload produced by a job on the system.

New in XBP 3.0	The entire function module is new in XBP 3.0.
BAPI object name	BackgroundJob
BAPI method name	GetStatistics
RFC interface	<pre> FUNCTION BAPI_XBP_BTC_STATISTIC_GET IMPORTING EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type CHAR length 16 I_T_JOBLIST like BTC_T_JOBLIST EXPORTING LOGHANDLE like BALLOGHNDL type CHAR length 22 T_STATDATA like BTC_T_STATDATA (see section 7.9.15.2) RETURN structure BAPIRET2 length 548 number of fields 14 </pre>
Parameter (Input)	<ul style="list-style-type: none"> • I_EXTERNAL_USER_NAME is the name of the user in the external scheduler who called the function. • I_T_JOBLIST is a list of the jobs for which you want to retrieve statistic information. The list should contain the name of the job and the job count of the job, in the following format: Jobname/Jobcount.
Parameter (Output)	<ul style="list-style-type: none"> • LOGHANDLE returns the handle to the error log. • T_STATDATA provides the statistic records for a job list. • BAPIRET2 is the return structure used by BAPIs.
Exceptions	You find a list of exceptions thrown by BAPI_XBP_BTC_STATISTIC_GET in section 7.9.15.1 below.

7.9.15.1 Exceptions Thrown by BAPI_XBP_BTC_STATISTIC_GET

Return Code	Text	Message Class	Message ID
1	No authority	BT	457
2	Job does not exist in status finished or aborted	BT	453
3	Job not found	BT	450
4	Job step not found	BT	098
5	Error reading job log	BT	452
6	RFC communication error	BT	456
7	System error	BT	454
8	Statistic data not found	BT	455

7.9.15.2 Data Elements

The tables below provide detailed information about the input parameter T_STATDATA from type BTC_T_STATDATA of BAPI_XBP_BTC_STATISTIC_GET.

7.9.15.2.1 Structures

BTC_S_STATDATA

Component	Type	Data type	Length	Decimals	Description
INSTANCE	STUNINST	CHAR	39	0	Instance
JOBNAME	BTCJOB	CHAR	32	0	Job name
JOBCOUNT	BTCJOBCNT	CHAR	8	0	Job count
RC	INT1	INTEGER	3		Return code
T_STATISTIC	SWNCGL_T_STATRECS	-	-	-	Statistic records for a particular job in the SWNC Globstat format

7.9.15.2.2 Table Types

Name	Line type	Description
BTC_T_JOBLIST	TBTCK	List of jobs(jobname + jobcount)
BTC_T_STATDATA	BTC_S_STATDATA	Statistic data table

7.9.16 Consuming Raised Events from Event History

By calling the functions below, the external scheduler can read events from the event history and optionally confirm the events that were already read.

7.9.16.1 Reading Events From Event History

Function name	BAPI_XBP_BTC_EVTHISTORY_GET
Short description	This function module retrieves the events that were logged in the event history. It can set the status of read events to CONFIRMED
New in XBP 3.0	The entire function module is new in XBP 3.0.
BAPI object name	BackgroundJob
BAPI method name	GetEventHistory
RFC interface	<pre> FUNCTION bapi_xbp_btc_evthistory_get IMPORTING EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type CHAR length 16 FROM_TIMESTAMP_UTC like BTCTIMESTAMP OPTIONAL type DEC length 15 TO_TIMESTAMP_UTC like BTCTIMESTAMP OPTIONAL type DEC length 15 EVENTIDS like BTCEVENTID OPTIONAL type CHAR length 32 SELECT_STATE like BTCHISTENTRYSTATE OPTIONAL type CHAR length 1 default 'N' ACTION like BTCEVTACTION OPTIONAL type CHAR length 1 default 'C' GUIDS like TT_BTCEVTGUID OPTIONAL PARAMS like BTCEVTPARM OPTIONAL </pre>

	<pre> type CHAR length 64 EXPORTING RETURN structure BAPIRET2 length 548 number of fields 14 TABLES RAISED_EVENTS STRUCTURE BTCEVTHISTORY </pre>
Detailed description	<p>By calling this function, the external scheduler can read events based on</p> <ul style="list-style-type: none"> • The time when the events were logged. You can specify the start and the end of the period within which the events were logged. • Event status. All events are logged with status NEW. To avoid having to read the same portion of events more than once, the scheduler can optionally change the status of NEW events to CONFIRMED. The scheduler can read events that were newly logged (events in status NEW), events that it has already read and confirmed (events in status CONFIRMED), or all events regardless of their status. • Event name (EVENTIDS) and event parameter (PARAMS) • GUIDs (Globally Unique Identifiers)
Parameter (Input)	<ul style="list-style-type: none"> • EXTERNAL_USER_NAME is the name of the user in the external scheduler who called the function. • FROM_TIMESTAMP_UTC is a timestamp (UTC) specifying the beginning of the period within which events were logged. The function returns all events that were logged after this moment. The default value is 01.01.1970 00:00:00. • TO_TIMESTAMP_UTC is a timestamp (UTC) specifying the end of the period within which events were logged. The function returns all events that were logged before this moment. The default value is 31.12.9999. • EVENTIDS specifies the names of the events (EVENTID) which should be read from the event history. The default value is space (empty), which means that all events are read. You can use wildcards. If this parameter is used, the GUIDS parameter will be ignored. • SELECT_STATE specifies the event status of the events which should be read. The following values are possible: <ul style="list-style-type: none"> ○ N means reading all events that are in status NEW. This includes events that were newly logged and have not been read yet, or events that have been read but not marked as confirmed by the external scheduler. ○ C means reading all events which were marked by the external scheduler as CONFIRMED. ○ A means reading all events regardless of their status. • ACTION specifies whether the external scheduler should change the status of read events. The following values are possible: <ul style="list-style-type: none"> ○ C sets the status of successfully read events from NEW to CONFIRMED. ○ N leaves the status of successfully read events as NEW. • GUIDS is an optional list of GUIDs of those events that have already been read before. This parameter can be used to monitor changes of state of raised events. The GUIDS parameter will only be used in the event selection, if the EVENTIDS parameter is empty. • PARAMS is an optional parameter which specifies the event parameters of the events EVENTIDS. This parameter will be used only, if the EVENTIDS parameter is used. You can use wildcards.

Parameter (Output)	<ul style="list-style-type: none"> RAISED_EVENTS is the exported database table containing raised events selected from the event history. BAPIRET2 is the return structure used by BAPIs.
--------------------	---

7.9.16.2 Confirming Events in Event History

Function name	BAPI_XBP_BTC_EVTHIST_CONFIRM
Short description	This function module changes the status of events from NEW to CONFIRMED without reading events.
New in XBP 3.0	The entire function module is new in XBP 3.0.
BAPI object name	BackgroundJob
BAPI method name	ConfirmEvents
RFC interface	<pre> FUNCTION bapi_xbp_btc_evthist_confirm IMPORTING EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type CHAR length 16 EXPORTING RETURN structure BAPIRET2 length 548 number of fields 14 TABLES EVTHIST_GUIDS STRUCTURE BTCEVTGUIDS </pre>
Detailed description	Changing the event status or confirming events enables the external scheduler to avoid reading the same events more than once. Confirming events is optional.
Parameter (Input)	<ul style="list-style-type: none"> EXTERNAL_USER_NAME is the name of the user in the external scheduler who called the function.
Parameter (Output)	<ul style="list-style-type: none"> EVTHIST_GUIDS is the database table containing the exported GUIDs (Globally Unique Identifier) of the event history entries whose status is to be set to CONFIRMED. BAPIRET2 is the return structure used by BAPIs.

7.9.16.3 Reading Event Definitions

Function name	BAPI_XBP_EVENT_DEFINITIONS_GET
Short description	This function reads definitions of batch events. It allows reading a certain number of events beginning from a given event.
New in XBP 3.0	The entire function module is new in XBP 3.0.
BAPI object name	BackgroundJob
BAPI method	GetEvents

name	
RFC interface	<pre> FUNCTION BAPI_XBP_EVENT_DEFINITIONS_GET IMPORTING EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type CHAR length 16 I_EVENTID like BTCEVENTID Optional type CHAR length 32 default % I_FROM type INT4 Optional CHAR length 10 default 0 I_TO type INT4 Optional CHAR length 10 default 0 EXPORTING E_T_EVENTS TYPE BTC_T_EVTINFO E_MORE TYPE BOOLEAN CHAR1 RETURN structure BAPIRET2 length 548 number of fields 14 </pre>
Detailed function description	<p>With this function you can get a full list of batch events available in the current system. The wildcard is specified in the optional and case-insensitive parameter I_EVENTID_MASK. Optional parameters I_FROM and I_TO are used for the reading in packets.</p>
Parameter (Input)	<ul style="list-style-type: none"> • I_EXTERNAL_USER_NAME is the name of the user in the external scheduler who called the function. • I_EVENTID_MASK is a wildcard for the event name. • I_FROM specifies the number of the line in the batch event definitions from which the external scheduler should start reading. The default value is 0: read all events. • I_TO specifies the number of the line in the batch event definitions at which the external scheduler should stop reading. The default value is 0: read all events.
Parameter (Output)	<ul style="list-style-type: none"> • BAPIRET2 is the return structure used by BAPIs. • E_T_EVENTS is a table that contains all batch events that correspond to the given wildcard and are within the package range. • E_MORE has the value 'X' when more batch events than given in I_TO are available in the current system.

7.9.17 Configuring Profiles and Criteria using the Criteria Manager Interface

The function modules described in the following subsections enable the external scheduler to work with the criteria that can control for example which events are to be logged in the event history.

7.9.17.1 Retrieving Information on Available Criteria Types

Function name	BAPI_CM_CRITYPES_GET
Short description	<p>By calling this function module, the external scheduler can retrieve information about the available criteria types. Currently, SAP provides the following criteria types:</p> <ul style="list-style-type: none"> • for event history - criteria specifying which raised events should be logged in the event history. • for reorganization of event history – criteria specifying the reorganization of raised events. • for interception
New in XBP 3.0	The entire function module is new in XBP 3.0.
BAPI object name	BackgroundJob
BAPI method name	GetCriteriaTypes
RFC interface	<pre> FUNCTION BAPI_CM_CRITYPES_GET IMPORTING I_EXTERNAL_USER_NAME like XMILOGEUSR type CHAR length 16 I_CRITTYPE like BTC_CRITTYPE optional type CHAR length 6 DEFAULT % EXPORTING E_T_CRITYTPES BTC_T_CRITTYPE_RAW RETURN structure BAPIRET2 length 548 number of fields 14 </pre>
Parameter (Input)	<ul style="list-style-type: none"> • I_EXTERNAL_USER_NAME is the name of the user in the external scheduler who called the function. • I_CRITTYPE specifies the criteria types that are to be selected. You can use wildcards (*).The default is '%' which means 'select all'. For the default criteria types provided by SAP, the values are: EVTHIS – type ID for event history EVHIRO - type ID for the reorganization of raised events INTERC identifies a criteria type for job interception
Parameter (Output)	<ul style="list-style-type: none"> • E_T_CRITYTPES is the database table containing the returned criteria types and fields (see also section 7.9.17.1.1 below). • BAPIRET2 is the return structure used by BAPIs.

7.9.17.1.1 Line type of E_T_CRITYTPES

Field Name	Field type	Description
TYPE	CHAR 6	Type ID
TYPEDescription	CHAR 40	Type description
CALLBACK_PROG	CHAR 40	Callback program name

CALLBACK_FORM	CHAR 30	Callback subroutine name
FIELD	CHAR 30	Criteria field name
DESCRIPTION	CHAR 40	Criteria field description
SEARCH_HELP	CHAR 30	Criteria field search help

7.9.17.2 Working with Criteria Profiles

Criteria and conditions are stored in profiles which can be active and inactive. Only one profile can be active at a time.

The function modules below enable the external scheduler to create, delete, and activate criteria profiles as well as to retrieve a list of existing criteria profiles.

7.9.17.2.1 Creating a Criteria Profile

Function name	BAPI_CM_PROFILE_CREATE
Short description	This function creates a criteria profile in the SAP system. The created profile is not active. It can be activated by calling the function BAPI_CM_PROFILE_ACTIVATE.
New in XBP 3.0	The entire function module is new in XBP 3.0.
BAPI object name	BackgroundJob
BAPI method name	CreateProfile
RFC interface	<pre> FUNCTION BAPI_CM_PROFILE_CREATE IMPORTING I_EXTERNAL_USER_NAME like XMILOGEUSR type CHAR length 16 I_PROFILE_XML type STRING optional EXPORTING E_PROFILEID like BTCPROFILEID type INT2 length 5 E_PROFILETYPE like BTC_CRITTYPE type CHAR length 6 RETURN structure BAPIRET2 length 548 number of fields 14 </pre>
Parameter (Input)	<ul style="list-style-type: none"> I_EXTERNAL_USER_NAME is the name of the user in the external scheduler who called the function. I_PROFILE_XML contains an XML description of a criteria profile. For more information about the relevant Document type definition, see Chapter 8 <i>Appendix</i>, section <i>Document Type Definition for Defining Profiles and Criteria for Event History</i>.
Parameter (Output)	<ul style="list-style-type: none"> E_PROFILETYPE is the criteria type of the new profile. E_PROFILEID is the ID of the new profile. The type and the ID of the profile identify a profile uniquely. BAPIRET2 is the return structure used by BAPIs.

7.9.17.2.2 Deleting a Criteria Profile

Function name	BAPI_CM_PROFILE_DELETE
Short description	This function module deletes a criteria profile. To be deleted, the criteria profile should not be active. The external scheduler can set an active profile to inactive by calling BAPI_CM_PROFILE_DEACTIVATE.
New in XBP 3.0	The entire function module is new in XBP 3.0.
BAPI object name	BackgroundJob
BAPI method name	DeleteProfile
RFC interface	<pre> FUNCTION BAPI_CM_PROFILE_DELETE IMPORTING I_EXTERNAL_USER_NAME like XMILOGEUSR type CHAR length 16 I_PROFILEID like BTCPROFILEID type INT2 length 5 I_PROFILETYPE like BTC_CRITTYPE type CHAR length 6 EXPORTING RETURN structure BAPIRET2 length 548 number of fields 14 </pre>
Parameter (Input)	<ul style="list-style-type: none"> • I_EXTERNAL_USER_NAME is the name of the user in the external scheduler who called the function. • I_PROFILETYPE is the criteria type of the profile that has to be deleted. For the default criteria types provided by SAP, the values are: EVTHIS identifies a criteria type for event history. EVHIRO identifies a criteria type for the reorganization of raised events. INTERC identifies a criteria type for job interception • I_PROFILEID is the ID of the profile that has to be deleted.
Parameter (Output)	<ul style="list-style-type: none"> • BAPIRET2 is the return structure used by BAPIs.

7.9.17.2.3 Activating a Criteria Profile

Function name	BAPI_CM_PROFILE_ACTIVATE
Short description	This function activates a criteria profile that exists in the SAP system.
New in XBP 3.0	The entire function module is new in XBP 3.0.
BAPI object name	BackgroundJob
BAPI method name	ActivateProfile
RFC interface	<pre> FUNCTION BAPI_CM_PROFILE_ACTIVATE IMPORTING I_EXTERNAL_USER_NAME like XMILOGEUSR type CHAR length 16 I_PROFILEID like BTCPROFILEID type INT2 length 5 I_PROFILETYPE like BTC_CRITTYPE type CHAR length 6 </pre>

	<p>EXPORTING RETURN structure BAPIRET2 length 548 number of fields 14</p>
Parameter (Input)	<ul style="list-style-type: none"> • I_EXTERNAL_USER_NAME is the name of the user in the external scheduler who called the function. • I_PROFILETYPE is the type of the profile that has to be activated. For the default criteria types provided by SAP, the values are: EVTHIS identifies a criteria type for event history. EVHIRO identifies a criteria type for the reorganization of raised events. INTERC identifies a criteria type for job interception • I_PROFILEID is the ID of the profile that has to be activated.
Parameter (Output)	BAPIRET2 is the return structure used by BAPIs.

7.9.17.2.4 Deactivating a Criteria Profile

Function name	BAPI_CM_PROFILE_DEACTIVATE
Short description	This function module sets an active profile to inactive.
New in XBP 3.0	The entire function module is new in XBP 3.0.
BAPI object name	BackgroundJob
BAPI method name	DeactivateProfile
RFC interface	<p>FUNCTION BAPI_CM_PROFILE_DEACTIVATE</p> <p>IMPORTING I_EXTERNAL_USER_NAME like XMILOGEUSR type CHAR length 16 I_PROFILETYPE like BTC_CRITTYPE type CHAR length 6</p> <p>EXPORTING RETURN structure BAPIRET2 length 548 number of fields 14</p>
Parameter (Input)	<ul style="list-style-type: none"> • I_EXTERNAL_USER_NAME is the name of the user in the external scheduler who called the function. • I_PROFILETYPE is the criteria type of the profile that has to be set to inactive. For the default criteria types provided by SAP, the values are: EVTHIS identifies a criteria type for event history. EVHIRO identifies a criteria type for the reorganization of raised events. INTERC identifies a criteria type for job interception
Parameter (Output)	BAPIRET2 is the return structure used by BAPIs.

7.9.17.2.5 Returning a List of Existing Profiles

Function name	BAPI_CM_PROFILES_GET
Short description	This function returns a list of the existing profiles filtered by profile type.
New in XBP 3.0	The entire function module is new in XBP 3.0.
BAPI object name	BackgroundJob
BAPI method name	GetProfiles
RFC interface	<pre> FUNCTION BAPI_CM_PROFILES_GET IMPORTING EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type CHAR length 16 E_PROFILETYPE like BTC_CRITTYPE optional type CHAR length 6 default '%' EXPORTING E_T_PROFILES type BTC_T_PROFILE_RAW RETURN structure BAPIRET2 length 548 number of fields 14 </pre>
Parameter (Input)	<ul style="list-style-type: none"> • I_EXTERNAL_USER_NAME is the name of the user in the external scheduler who called the function. • I_PROFILETYPE is the criteria type of the profile by which the returned criteria profiles are to be filtered. The default is % which means that all profiles are to be selected. You can use wildcards. Only * is supported. <p>For the default criteria types provided by SAP, the values are: EVTHIS identifies a criteria type for event history. EVHIRO identifies a criteria type for the reorganization of raised events. INTERC identifies a criteria type for job interception</p>
Parameter (Output)	<ul style="list-style-type: none"> • E_T_PROFILES is the database table containing the returned criteria profiles with all important parameters. See BTC_T_PROFILE_RAW ABAP dictionary type for details. • BAPIRET2 is the return structure used by BAPIs.

7.9.17.3 Working with Criteria

7.9.17.3.1 Returning the Criteria Hierarchy of a Profile

Function name	BAPI_CM_CRITERIA_GET
Short description	This function module returns the criteria hierarchy of a particular profile in XML format.
New in XBP 3.0	The entire function module is new in XBP 3.0.
BAPI object name	BackgroundJob
BAPI method name	GetCriteria
RFC interface	<pre> FUNCTION BAPI_CM_CRITERIA_GET IMPORTING I_EXTERNAL_USER_NAME like XMILOGEUSR type CHAR length 16 </pre>

	<p>I_PROFILEID like BTCPROFILEID type INT2 length 5</p> <p>I_PROFILETYPE like BTC_CRITTYPE type CHAR length 6</p> <p>EXPORTING</p> <p>E_CRITERIA_XML type STRING</p> <p>RETURN structure BAPIRET2 length 548 number of fields 14</p>
Parameter (Input)	<ul style="list-style-type: none"> • I_EXTERNAL_USER_NAME is the name of the user in the external scheduler who called the function. • I_PROFILETYPE is the criteria type of the profile whose criteria are to be exported in XML format. For the default criteria types provided by SAP, the values are: EVTHIS identifies a criteria type for event history. EVHIRO identifies a criteria type for the reorganization of raised events. INTERC identifies a criteria type for job interception • I_PROFILEID is the ID of the profile whose criteria are to be exported in XML format.
Parameter (Output)	<ul style="list-style-type: none"> • E_CRITERIA_XML returns an XML description of the criteria hierarchy of the profile. • BAPIRET2 is the return structure used by BAPIs.

7.9.17.3.2 Importing Criteria to an Existing Criteria Profile

Function name	BAPI_CM_CRITERIA_SET
Short description	This function module imports criteria to an existing criteria profile. The criteria which you want to import have to be written in XML format. If you import criteria in a profile which already contains certain criteria, the criteria in the profile are overwritten by the imported criteria.
New in XBP 3.0	The entire function module is new in XBP 3.0.
BAPI object name	BackgroundJob
BAPI method name	SetCriteria
RFC interface	<pre>FUNCTION BAPI_CM_CRITERIA_SET IMPORTING I_EXTERNAL_USER_NAME like XMILOGEUSR type CHAR length 16 I_PROFILEID like BTCPROFILEID type INT2 length 5 I_PROFILETYPE like BTC_CRITTYPE type CHAR length 6 I_CRITERIA_XML type STRING EXPORTING RETURN structure BAPIRET2 length 548 number of fields 14</pre>
Parameter (Input)	<ul style="list-style-type: none"> • I_EXTERNAL_USER_NAME is the name of the user in the external scheduler who called the function. • I_PROFILETYPE is the type of the profile in which you want to import criteria.

	<ul style="list-style-type: none"> • I_PROFILEID is the ID of the profile in which you want to import criteria. • I_CRITERIA_XML provides an XML description of the criteria profile and its criteria hierarchy. For more information about the relevant Document Type Definition, see chapter 8. <i>Appendix, section Document Type Definition for Defining Profiles and Criteria for Event History.</i>
Parameter (Output)	BAPIRET2 is the return structure used by BAPIs.

7.9.18 Reading a Particular Spool List as PDF

With this function it's possible to pass the number of a particular spool list and the system will return its content in PDF format.

Function name	BAPI_XBP_GET_SPOOL_AS_PDF
Short description	<p>Convert to PDF format and read a particular ABAP spool list of a job that has been run.</p> <p>If spool list contains more than 100 pages conversion will be performed asynchronous in background.</p> <p>From SAP NW 7.0 PDF-Spool-Lists are compressed using ZIP compress methods.</p>
Introduced in XBP 2.0	Available as XBP 2.0 extension and delivered via correction instruction.
BAPI object name	BackgroundJob
BAPI method name	
RFC interface	<pre> FUNCTION bapi_xbp_get_spool_as_pdf . IMPORTING VALUE(I_SPOOLID) TYPE RSPOID VALUE(I_EXTERNAL_USER_NAME) TYPE XMILOGEUSR EXPORTING VALUE(RETURN) TYPE BAPIRET2 VALUE(E_PDF) TYPE XSTRING VALUE(E_COMPRESSED) TYPE BOOLEAN VALUE(E_JOBNAME) TYPE BTCJOB VALUE(E_JOBCOUNT) TYPE BTCJOBCNT </pre>
Parameter (Input)	<ul style="list-style-type: none"> • I_EXTERNAL_USER_NAME is the name of the XBP user. • I_SPOOLID is the number of a spool request in the SAP system.
Parameter (Output)	<ul style="list-style-type: none"> • E_PDF: Spool List in binary format • E_COMPRESSED: ='X' when spool list is compressed • E_JOBNAME: Jobname of the PDF conversion job • E_JOBCOUNT: Jobcount of the PDF conversion job • RETURN is the standard return structure containing return values of the function.
MessageIDs	<ul style="list-style-type: none"> • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out. • MSG_PRIVILEGE_MISSING: The user used by the external management system to log onto the AS ABAP system is not authorized to read the spool list.

	<ul style="list-style-type: none"> • Errors in text form. Please take a notice of a message text.
--	--

7.9.19 Reading a Particular Binary Spool List

With this function it's possible to read binary spool lists created, for example, via BAPI_XBP_GET_SPOOL_AS_PDF function module.

Function name	BAPI_XBP_READ_SPOOL_BIN
Short description	Read particular binary spool list unconverted in binary format. From SAP NW 7.0 PDF-Spool-Lists are compressed using ZIP compress methods.
Introduced in XBP 2.0	Available as XBP 2.0 extension and delivered via correction instruction.
BAPI object name	BackgroundJob
BAPI method name	
RFC interface	<pre> FUNCTION bapi_xbp_read_spool_bin. <i>IMPORTING</i> VALUE(I_SPOOLID) TYPE RSPOID VALUE(I_EXTERNAL_USER_NAME) TYPE XMILOGEUSR VALUE(I_BYTES_FROM) TYPE INT4 DEFAULT 0 VALUE(I_PORTION) TYPE INT4 DEFAULT 4194304 <i>EXPORTING</i> VALUE(RETURN) TYPE BAPIRET2 VALUE(E_SPOOL) TYPE XSTRING VALUE(E_MIME_TYPE) TYPE SAEMIME VALUE(E_COMPRESSED) TYPE BOOLEAN VALUE(E_MORE) TYPE BOOLEAN </pre>
Parameter (Input)	<ul style="list-style-type: none"> • I_EXTERNAL_USER_NAME is the name of the XBP user. • I_SPOOLID is the number of a spool request in the SAP system. • The I_BYTES_FROM and I_PORTION parameters allow the external scheduler to read long spools in chunks to improve performance.
Parameter (Output)	<ul style="list-style-type: none"> • E_SPOOL: List in binary format • E_COMPRESSED: ='X' when spool list is compressed • E_MIME_TYPE: Mime type (= "PDF" – for PDF spool) • E_MORE has the value 'X' when more bytes than given in I_PORTION are available in the spool list. • RETURN is the standard return structure containing return values of the function.
MessageIDs	<ul style="list-style-type: none"> • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out. • MSG_PRIVILEGE_MISSING: The user used by the external management system to log onto the AS ABAP system is not authorized to read the spool list. • Errors in text form. Please take a notice of a message text.

7.10 Searching with Wildcards

Using the following function modules, you can search with wildcards for ABAP reports, external commands, output devices and print formats available in the current system:

7.10.1 Searching for ABAP Reports

Function name	BAPI_XBP_REPORT_SEARCH
Short description	This function module is a value help function for ABAP reports. It returns a list of ABAP reports available in the current systems, which match a certain wildcard. It is possible to return the list in several parts. These parts can also vary in size from call to call.
Introduced in XBP 2.0	The entire function module was new in XBP 2.0.
BAPI object name	BackgroundJob
BAPI method name	
RFC interface	<pre> Function BAPI_XBP_REPORT_SEARCH importing REPORT like TRDIRT-NAME optional type RFC_CHAR length 40 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 COUNT TYPE I OPTIONAL exporting RETURN structure BAPIRET2 length 548 number of fields 14 Tables REPORTS structure TRDIRT length 111 number of fields 3 </pre>
Detailed function description	<p>With this function you can get a full list of ABAP reports available in the current system, whose names correspond to a certain wildcard and whose descriptions are written in the logon language. The wildcard is given with the optional and case-insensitive parameter REPORT. Valid wildcard special symbols are '*' (asterisk) and '.' (dot). If no wildcard is specified, all reports are returned. Names of reports, languages, and text descriptions are stored in the REPORTS table.</p> <p>If no description for a report is found in the logon language, the entry is returned with any available non-empty description.</p> <p>The optional parameter COUNT is used to limit the size of the output list. This parameter makes it possible to divide the output into parts. The size of each part can vary as well. When the list is divided into parts, each call to the current function module returns the next portion of the specified size unless the output is complete.</p> <p>If a sequence of calls with the same environment, that is, the same REPORT and EXTERNAL_USER_NAME parameters, is interrupted by a call with different parameters, the environment for the first call is lost, and only the interrupting call is able to proceed retrieving the rest of its output.</p>
Parameter (Input)	<ul style="list-style-type: none"> REPORT is a wildcard for names of required ABAP reports.

	<ul style="list-style-type: none"> EXTERNAL_USER_NAME is the name of the XBP user. COUNT is an optional parameter for limiting the size of the output list. This parameter enables you to divide the output into parts.
Parameter (Output)	<ul style="list-style-type: none"> BAPIRET2 is the return structure used by BAPIs.
Tables	<ul style="list-style-type: none"> REPORTS is a table that contains names of all the ABAP reports that will be returned by the function module.
MessageIDs	<ul style="list-style-type: none"> MSG_EXT_USER_MISSING: External user name is missing (initial). MSG_CANT_LOG: Activity was terminated because the AS ABAP XML logging mechanism returned an error. MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XML interface. Therefore, the activity cannot be carried out. MSG_WRONG_COUNTER: Counter has a negative value. MSG_SELECTION_FINISHED: All output has been retrieved.

7.10.2 Searching for External Commands

Function name	BAPI_XBP_EXT_COMM_SEARCH
Short description	This function module is a value help function for external commands. It returns a list of external commands available in the current system, whose names match a certain wildcard.
Introduced in XBP 2.0	The entire function module was new in XBP 2.0.
BAPI object name	BackgroundJob
BAPI method name	
RFC interface	<p>Function BAPI_XBP_EXT_COMM_SEARCH</p> <p>Importing</p> <p>COMMAND LIKE SXPGBCOLIST-NAME optional default ㊿*㊿ type RFC_CHAR length 18</p> <p>OPSYS LIKE SXPGBCOLIST-OPSYSTEM optional type RFC_CHAR length 10</p> <p>EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16</p> <p>Exporting</p> <p>RETURN structure BAPIRET2 length 548 number of fields 14</p> <p>Tables</p> <p>EXT_COMMANDS structure SXPGBCOLIST length 445 number of fields 9</p>
Detailed function description	With this function you can get a full list of external commands available in the current system, whose names and operating systems correspond to certain requirements. The wildcard is specified in the optional and case-insensitive parameter COMMAND. The OPSYS parameter specifies the name of the operating system for which the commands are valid. If no operating system name is specified, all are

	assumed. If no wildcard for command names is specified, then all commands are returned. Names, operating systems, types, underlying commands, etc. of external programs are placed in the EXT_COMMANDS table.
Parameter (Input)	<ul style="list-style-type: none"> COMMAND is a wildcard for required names of external commands. EXTERNAL_USER_NAME is the name of the XBP user. OPSYS is the name of the operating system, for which external commands are searched.
Parameter (Output)	<ul style="list-style-type: none"> BAPIRET2 is the return structure used by BAPIs.
Tables	<ul style="list-style-type: none"> EXT_COMMANDS is a table that contains all external commands that correspond to the given wildcard and the name of the operating system.
MessageIDs	<ul style="list-style-type: none"> MSG_EXT_USER_MISSING: External user name is missing. MSG_CANT_LOG: Activity was terminated because the AS ABAP XMI logging mechanism returned an error. MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out. MSG_PROBLEM_DETECTED: Internal error.

7.10.3 Searching for Output Devices

Function name	BAPI_XBP_OUTPUT_DEVICE_SEARCH
Short description	This function module is a value help function for printer output devices. It returns a list of output devices available in the current system, whose names match a certain wildcard.
Enhanced to XBP 3.0	<p>The function module was new in XBP 2.0 and has been enhanced for XBP 3.0</p> <p>You can now also search for a certain device type as well as for output devices whose definitions have been changed after a certain date.</p>
BAPI object name	BackgroundJob
BAPI method name	
RFC interface	<pre> Function BAPI_XBP_OUTPUT_DEVICE_SEARCH Import parameters OUTPUT_DEVICE_SHORT like TSP03L-PADEST optional type RFC_CHAR length 4 OUTPUT_DEVICE_LONG like TSP03L-LNAME optional type RFC_CHAR length 30 EXTERNAL_USER_NAME LIKE BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 DEVTYPE like RSPOPTYPE optional type RFC_CHAR length 8 CHANGE_DATE type TSPOC-DATUM optional type DATS length 8 CHANGE_TIME type TSPOC-ZEIT optional type TIMS length 6 RETURN_LONG_NAMES type BTCH0000-CHAR1 </pre>

	<p>Optional RFC_CHAR length 1</p> <p>Export parameters RETURN structure BAPIRET2 length 548 number of fields 14</p> <p>Tables OUTPUT_DEVICES structure RSPOLD optional length 92, number of fields 4 OUTPUT_DEVICES_LONG structure RSPODEV Optional length 158, number of fields 7</p>
Detailed function description	<p>With this function you can get a full list of output devices available in the current system, whose short and long names correspond to certain requirements. Long names are non-technical names, which are first converted into short names.</p> <p>The names should be given using the OUTPUT_DEVICE_SHORT and OUTPUT_DEVICE_LONG parameters. Both are optional, case-insensitive parameters, which accept wildcards. If both names are left blank, it is assumed that the long name is '*'. Output devices and some of their properties are stored in the table OUTPUT_DEVICES.</p> <p>As of XBP 3.0, you can also search for a certain device type as well as for output devices whose definitions have been changed after a certain date and time.</p>
Parameter (Input)	<ul style="list-style-type: none"> • OUTPUT_DEVICE_LONG is a wildcard for long names of output devices. These long names will first be converted into short names. • OUTPUT_DEVICE_SHORT is a wildcard for technical names of output devices. • EXTERNAL_USER_NAME is the name of the XBP user. • DEVTYPE is the name of a device type. • CHANGE_DATE is the date when an output device definition has been changed for the last time. • CHANGE_TIME is the time when an output device definition has been changed. • RETURN_LONG_NAMES is a flag that tells the system to return also the long names of output devices in a separate table.
Parameter (Output)	<ul style="list-style-type: none"> • BAPIRET2 is the return structure used by BAPIs.
Tables	<ul style="list-style-type: none"> • OUTPUT_DEVICES is a table that contains all output devices that correspond to the given wildcards for short and long names. • OUTPUT_DEVICES_LONG is a table that contains long and short names of output devices plus some extra information.
MessageIDs	<ul style="list-style-type: none"> • MSG_EXT_USER_MISSING: External user name is missing. • MSG_CANT_LOG: Activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.

7.10.4 Searching for Print Formats

Function name	BAPI_XBP_PRINT_FORMAT_SEARCH
Short description	This function module is a value help function for print formats. It returns a list of print formats available for a certain printer. The names of the print formats match a certain wildcard.
Introduced in XBP 2.0	The entire function module was new in XBP 2.0.
BAPI object name	BackgroundJob
BAPI method name	
RFC interface	<p>Function BAPI_XBP_PRINT_FORMAT_SEARCH</p> <p>Importing</p> <p>PRINTER like TSP03L-PADEST type RFC_CHAR length 4</p> <p>LAYOUT like RSPOLD-LAYOUT optional DEFAULT * type RFC_CHAR length 16</p> <p>EXTERNAL_USER_NAME LIKE BAPIXMLOGR-EXTUSER type RFC_CHAR length 16</p> <p>Exporting</p> <p>RETURN structure BAPIRET2 length 548 number of fields 14</p> <p>Tables</p> <p>LAYOUTS structure RSPOLD length 92, number of fields 4</p>
Parameter (Input)	<ul style="list-style-type: none"> PRINTER specifies a certain printer, for which print formats are retrieved. LAYOUT defines a wildcard for required print formats. If this optional parameter is left blank, '*' is assumed. EXTERNAL_USER_NAME is the name of the XBP user.
Parameter (Output)	<ul style="list-style-type: none"> RETURN is the return structure for function modules used for BAPIs.
Tables	<ul style="list-style-type: none"> LAYOUTS is a table that contains all the print formats that were requested.
MessageIDs	<ul style="list-style-type: none"> MSG_EXT_USER_MISSING: External user name is missing. MSG_CANT_LOG: Activity was terminated because the AS ABAP XMI logging mechanism returned an error. MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out. MSG_WRONG_PRINTER_NAME: There is no printer of the given name in the system.

7.10.5 Searching for Archive Parameters

Function name	BAPI_XBP_GET_ARCHIVE_OBJECTS
Short description	This function returns SAP Objects and Archive Objects that are defined in a system.
Introduced in XBP 3.0	The entire function module is new in XBP 3.0.
BAPI object name	
BAPI method name	
RFC interface	<pre> FUNCTION BAPI_XBP_GET_ARCHIVE_OBJECTS IMPORTING EXTERNAL_USER_NAME TYPE XMILOGEUSR type CHAR length 16 SAP_OBJECT TYPE SAEANWDID optional default SPACE type CHAR length 10 EXPORTING RETURN structure BAPIRET2 length 548 number of fields 14 TABLES AR_OBJECTS structure BAPIARCOBJECTS length 167 number of fields 5 </pre>
Detailed function description	This function returns a table of SAP and Archive Objects that are defined in a system. The values can be passed as archive parameters to BAPI_XBP_JOB_ADD_ABAP_STEP.
Parameter (Input)	<ul style="list-style-type: none"> EXTERNAL_USER_NAME is the name of the XBP user SAP_OBJECT (optional) is the name of a SAP Object. This parameter does not support wildcards. It is only possible to return Archive Objects for a given single SAP Object.
Parameter (Output)	BAPIRET2 is the return structure used by BAPIs.
Tables	AR_OBJECTS contains SAP Objects and Archive Objects with descriptions. For the description of structure BAPIARCOBJECTS see section 7.10.5.1.

7.10.5.1 Description of structure BAPIARCOBJECTS

Field Name	Field type	Description
SAP_OBJECT	CHAR10	Object type of business object
NTEXT	CHAR80	Description of business object
AR_OBJECT	CHAR10	Document type
OBJECTTEXT	CHAR40	Document type description
RESERVE	CHAR27	Reserved for future usage

7.10.6 Searching for Batch Events

Function name	BAPI_XBP_EVENT_DEFINITIONS_GET
Short description	This function module is a value help function for batch events. It returns a list of batch events that are available in the current system and whose names match a certain wildcard.
Introduced in XBP 3.0	The entire function module is new in XBP 3.0.
BAPI object name	BackgroundJob
BAPI method name	
RFC interface	<pre> Function BAPI_XBP_EVENT_DEFINITIONS_GET IMPORTING EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type CHAR length 16 I_EVENTID_MASK like BTCEVENTID Optional type CHAR length 32 default % I_FROM type INT4 Optional CHAR length 10 default 0 I_TO type INT4 Optional CHAR length 10 default 0 EXPORTING E_T_EVENTS TYPE BTC_T_EVTINFO E_MORE TYPE BOOLEAN CHAR1 RETURN structure BAPIRET2 length 548 number of fields 14 </pre>
Detailed function description	With this function you can get a full list of batch events available in the current system. The wildcard is specified in the optional and case-insensitive parameter I_EVENTID_MASK. Optional parameters I_FROM and I_TO are used for the reading in packets.
Parameter (Input)	<ul style="list-style-type: none"> • I_EXTERNAL_USER_NAME is the name of the user in the external scheduler who called the function. • I_EVENTID_MASK is a wildcard for the event name. • I_FROM specifies the number of the line in the batch event definitions from which the external scheduler should start reading. The default value is 0: read all events. • I_TO specifies the number of the line in the batch event definitions at which the external scheduler should stop reading. The default value is 0: read all events.
Parameter (Output)	<ul style="list-style-type: none"> • BAPIRET2 is the return structure used by BAPIs. • E_T_EVENTS is a table that contains all batch events that correspond to the given wildcard and are within the package range. • E_MORE has the value 'X', when more batch events than stated in I_TO are available in the current system.

7.11 General Help Functions

Below are descriptions of the following functions which allow you to:

- see all variants of a given ABAP program (BAPI_XBP_VARIANT_INFO_GET).
- see the resources currently available for jobs in the AS ABAP system (BAPI_XBP_GET_CURR_BP_RESOURCES).
- determine whether resources are available for a job on a particular server at a particular time (BAPI_XBP_GET_BP_SRVRES_ON_DATE).
- determine whether resources are available for a job on any server in the AS ABAP system at a particular time (BAPI_XBP_GET_BP_RESRC_ON_DATE).
- read syslog entries in the AS ABAP system.
- read SAP Factory calendar (BAPI_XBP_FACT_CALENDERS_GET).
- read SAP Holiday calendar (BAPI_XBP_HOL_CALENDERS_GET).

7.11.1 Showing All Defined Variants of an ABAP Program

Function name	BAPI_XBP_VARIANT_INFO_GET
Short description	For a given ABAP the variants are read.
BAPI object name	BackgroundJob
BAPI method name	GetVariantListForReportname
RFC interface	<pre>function BAPI_XBP_VARIANT_INFO_GET importing ABAP_PROGRAM_NAME like BAPIXMREP-REPORTID type RFC_CHAR length 40 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 VARIANT_SELECT_OPTION like BAPIXMREP- VARSELOPT type RFC_CHAR length 1 exporting RETURN structure BAPIRET2 length 548 number of fields 14 tables ABAP_VARIANT_TABLE structure BAPIXMVAR length 54 number of fields 2</pre>
Parameter (Input)	<ul style="list-style-type: none"> • ABAP_PROGRAM_NAME is name of the ABAP program for which existing variants are to be determined. The program must have type 1 (interactively executable). • EXTERNAL_USER_NAME is the name of the XBP user • VARIANT_SELECT_OPTION defines which type of variables should be chosen. Permissible values are as follows: 'A' - variants that are available for batch as well as for dialog will be selected 'B' - 'batch only' variants will be selected
Parameter (Output)	BAPIRET2 is the return structure used by BAPIs.
Tables	<ul style="list-style-type: none"> • ABAP_VARIANT_TABLE contains all defined variants of the ABAP program.

MessageIDs	<ul style="list-style-type: none"> • MSG_PROGRAM_DOES_NOT_EXIST: The specified ABAP program does not exist in the AS ABAP system. • MSG_PROGNAME_MISSING: You did not enter an ABAP program name. • MSG_INVALID_SELECT_OPTION: Wrong code used to select variants of an ABAP. • MSG_PROGRAM_HAS_NO_VARIANT: The ABAP program has no variants. • MSG_NO_VARIANTS_DEFINED: You have not yet defined any variants. • MSG_PROG_NOT_EXECUTABLE: The program is not executable. • MSG_NO_EXECUTE_PRIVILEGE: The SAP user used by the external job management system to log onto the AS ABAP system is not authorized to execute the ABAP program. • MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered an error. • MSG_CANT_LOG: The activity was terminated because the AS ABAP XMI logging mechanism returned an error. • MSG_EXT_USER_MISSING: The name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.
------------	--

7.11.2 Determining Current Resources for Jobs in the AS ABAP System

Function name	BAPI_XBP_GET_CURR_BP_RESOURCES
Short description	<p>Using this function module you can retrieve the following values from the AS ABAP system:</p> <ul style="list-style-type: none"> Names of the servers which currently have background work processes Number of background work processes on each server and their status (total number, free, working, class A background work processes)
BAPI object name	SystemServiceInfo
BAPI method name	GetCurrentBackgroundResources
RFC interface	<pre>function BAPI_XBP_GET_CURR_BP_RESOURCES importing EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 exporting RETURN structure BAPIRET2 length 552 number of fields 14 tables RESOURCE_INFO structure BAPIXMCRES length 69 number of fields 5</pre>
Parameter (Input)	EXTERNAL_USER_NAME is the name of the XBP user.
Parameter (Output)	BAPIRET2 is the return structure used by BAPIs.
Parameter (Output)	<p>RESOURCE_INFO is a table containing the following information in each line:</p> <ul style="list-style-type: none"> Server name Host name Total number of background work processes on the server Number of free background work processes on the server Number of working background work processes on the server Number of reserved class A background work processes on the server
MessageIDs	<ul style="list-style-type: none"> MSG_PROBLEM_DETECTED: The AS ABAP job scheduling system has discovered an error. MSG_NO_RESOURCES_FOUND: There are no background processing resources in the AS ABAP system. MSG_CANT_LOG: The activity was terminated because the AS ABAP XMI logging mechanism returned an error. MSG_EXT_USER_MISSING: The name of the external user is missing. This is the name of a user in the external job scheduling system. MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.

7.11.3 Checking Available Job Resources at a Particular Time on a Server

In the AS ABAP system, within the framework of switching operation modes on servers, you can assign different work process types at different times. For example, you might only have dialog processes during the day, but at night some of these processes switch to operating as background work processes. You can use the function module `BAPI_XBP_GET_BP_SRVRES_ON_DATE` to determine whether background work processes are available at a particular time **on a particular server**.

Function name	BAPI_XBP_GET_BP_SRVRES_ON_DATE
Short description	Get resource information for a particular server at a certain date and time.
BAPI object name	SystemServiceInfo
BAPI method name	GetBgrdResourcesOnDateOnServer
RFC interface	<pre>function BAPI_XBP_GET_BP_SRVRES_ON_DATE importing EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 SERVER_NAME like BAPIXMJOB-EXECSEVER type RFC_CHAR length 20 DATE like BAPIXMJOB-SDLSTRTDT type RFC_DATE length 8 TIME like BAPIXMJOB-SDLSTRTTM type RFC_TIME length 6 exporting RESOURCE_INFO structure BAPIXMRES length 66 number of fields 4 RETURN structure BAPIRET2 length 548 number of fields 14</pre>
Parameter (Input)	<ul style="list-style-type: none"> EXTERNAL_USER_NAME is the name of the XBP user. SERVER_NAME is the name of the AS ABAP instance on which the availability of resources is to be determined. The name must be specified in the format <host_name> <SAP_system_name>_<SAP_system_number>. The instance name is contained in system profile parameter rdisp/myname. Example: host1234_C11_55 DATE is the date on which the availability of resources is to be determined. You must specify the date in the format YYYYMMDD. Example 2000101. TIME is the time of day at which the availability of resources is to be determined. The time must be specified as HHMMSS. Example: 231255.
Parameter (Output)	<ul style="list-style-type: none"> BAPIRET2 is the return structure used by BAPIs. RESOURCE_INFO is a table that lists the resources available for background processing for each instance: <ul style="list-style-type: none"> Host name Number of background work processes Number of reserved class A background work processes
MessageIDs	<ul style="list-style-type: none"> MSG_NO_RESOURCES_FOUND: There are no background work processes on the server at the given time and date. MSG_PROBLEM_DETECTED: The AS ABAP job scheduling

	<p>system has discovered a problem.</p> <ul style="list-style-type: none">• MSG_INVALID_SERVER_NAME: The specified server name is invalid.• MSG_INVALID_DATE_TIME: The specified date and/or time is invalid.• MSG_CANT_LOG: The activity was terminated because the AS ABAP XMI logging mechanism returned an error.• MSG_EXT_USER_MISSING: The name of the external user is missing. This is the name of a user in an external job scheduling system.• MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.
--	--



The XMI log does not record any calls to this function, since it does not change or output any security-sensitive data.

7.11.4 Checking Available Job Resources at a Particular Time in the Whole SAP System.

Function name	BAPI_XBP_GET_BP_RESRC_ON_DATE
Short description	You can use this function module to determine whether background work processes are available at a particular time on any server in the AS ABAP system.
BAPI object name	SystemServiceInfo
BAPI method name	GetBackgroundResourcesOnDate
RFC interface	<pre> function BAPI_XBP_GET_BP_RESRC_ON_DATE importing EXTERNAL_USER_NAME like BAPIXMLOGR_EXTUSER type RFC_CHAR length 16 DATE like BAPIXMJOB_SDLSTRTDT type RFC_DATE length 8 TIME like BAPIXMJOB_SDLSTRTTM type RFC_TIME length 6 exporting RETURN structure BAPIRET2 length 548 number of fields 14 tables RESOURCE_INFO_TBL structure BAPIXMRES length 66 number of fields 4 </pre>
Parameter (Input)	<ul style="list-style-type: none"> EXTERNAL_USER_NAME is the name of the XBP user. DATE is the date on which the availability of resources is to be determined. You must specify the date in the format YYYYMMDD. Example 20000101. TIME is the time of day at which the availability of resources is to be determined. The time must be specified as HHMMSS. Example: 231255.
Parameter (Output)	<ul style="list-style-type: none"> BAPIRET2 is the return used by BAPIs.
Tables	<ul style="list-style-type: none"> RESOURCE_INFO_TBL is a table that lists the resources available for background processing for each instance. It contains the following values: <ul style="list-style-type: none"> Server name Host name Number of background work processes Number of reserved class A background work processes
MessageIDs	<ul style="list-style-type: none"> MSG_NO_RESOURCES_FOUND: There are no background work processes in the AS ABAP system at the given time and date. MSG_INVALID_DATE_TIME: The specified time and/ or date is invalid. MSG_PROBLEM_DETECTED: The job scheduling system has discovered an error. MSG_CANT_LOG: The activity was terminated because the AS ABAP XMI logging system returned an error. MSG_EXT_USER_MISSING: The name of the external user is missing. This is the name of a user in the external job scheduling system. MSG_NOT_LOGGED_ON: The external management tool has not

	logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.
--	--

7.11.5 Reading SAP Factory Calendars

Function name	BAPI_XBP_FACT_CALENDERS_GET
Short description	You can use this function module to read SAP Factory calendars.
Introduced in XBP 3.0	The entire function module is new in XBP 3.0.
BAPI object name	
BAPI method name	
RFC interface	<pre> Function BAPI_XBP_FACT_CALENDERS_GET Importing I_EXTERNAL_USER_NAME like XMILOGEUSR type CHAR length 16 I_CALENDER_ID like WFCID optional DEFAULT '%' type CHAR length 2 I_GET_DETAILS TYPE BOOLEAN optional DEFAULT ABAP_FALSE EXPORTING T_CAL_DEFINITIONS TYPE BAPI_T_TFACD T_CAL_DETAILS TYPE BAPI_T_TFACS RETURN structure BAPIRET2 length 548 number of fields 14 </pre>
Parameter (Input)	<ul style="list-style-type: none"> • I_EXTERNAL_USER_NAME: Name of the XBP user. • I_CALENDER_ID: Wildcard for the calendar ID. The default is '%' which means 'read all'. • I_GET_DETAILS: When flag: ='X', the detail info is transferred to the external scheduler (Table <i>T_CAL_DETAILS</i>).
Parameter (Output)	<ul style="list-style-type: none"> • BAPIRET2 is the return structure used by BAPIs. • T_CAL_DEFINITIONS: Table with SAP factory calendar definition(s). • T_CAL_DETAILS: Table with details of SAP factory calendar(s).
MessageIDs	<ul style="list-style-type: none"> • MSG_CALENDER_DOES_NOT_EXISTS: There were no calendars found with the specified mask • MSG_EXT_USER_MISSING: The name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.


7.11.6 Reading SAP Holiday Calendars

Function name	BAPI_XBP_HOL_CALENDERS_GET
Short description	You can use this function module to read SAP Holiday calendars.
Introduced in XBP 3.0	The entire function module is new in XBP 3.0.
BAPI object name	
BAPI method name	
RFC interface	<p>Function BAPI_XBP_HOL_CALENDERS_GET</p> <p>Importing</p> <p>I_EXTERNAL_USER_NAME like XMILOGEUSR type CHAR length 16</p> <p>I_CALENDER_ID like WFCID optional DEFAULT '%' type CHAR length 2</p> <p>I_GET_DETAILS TYPE BOOLEAN optional DEFAULT ABAP_FALSE</p> <p>EXPORTING</p> <p>T_CAL_DEFINITIONS TYPE BAPI_T_TFACD</p> <p>T_CAL_DETAILS TYPE BAPI_T_TFACS</p> <p>RETURN structure BAPIRET2 length 548 number of fields 14</p>
Parameter (Input)	<ul style="list-style-type: none"> I_EXTERNAL_USER_NAME is the name of the XBP user. I_CALENDER_ID: Wildcard for the calendar ID. The default is '%' which means 'read all'. I_GET_DETAILS: When flag: ='X', the detail info is transferred to the external scheduler (Table <i>T_CAL_DETAILS</i>).
Parameter (Output)	<ul style="list-style-type: none"> BAPIRET2 is the return used by BAPIs. T_CAL_DEFINITIONS: Table with SAP holiday calendar definition(s). T_CAL_DETAILS: Table with details of SAP holiday calendar(s).
MessageIDs	<ul style="list-style-type: none"> MSG_CALENDER_DOES_NOT_EXISTS: There were no calendars found with the specified mask. MSG_EXT_USER_MISSING: The name of the external user is missing. This is the name of a user in the external job scheduling system. MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.

7.12 Variant Functions


With XBP 3.0 some functions for simplified variant handling are introduced.

7.12.1 Creating a Variant

Function name	BAPI_XBP_VARIANT_CREATE
Short description	<p>Create a variant of an ABAP program</p>  <p>Regarding support of full length 132 (instead of 45) for simple selection fields, see note 1144882.</p>
New in XBP 3.0	This function is new in XBP 3.0
BAPI object name	
BAPI method name	
RFC interface	<pre>Function BAPI_XBP_VARIANT_CREATE IMPORTING ABAP_PROGRAM_NAME type BAPIXMREP-REPORTID type CHAR length 40 ABAP_VARIANT_NAME type BAPIXMREP-VARIANTNAM type CHAR length 14 ABAP_VARIANT_TEXT type RVART_VTXT Type CHAR length 30 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 EXPORTING RETURN structure BAPIRET2 length 548 number of fields 14 TABLES VARIANT_INFO like BAPIVARINFO length 196 number of fields 10</pre>
Parameter (Input)	<ul style="list-style-type: none"> • ABAP_PROGRAM_NAME: Name of the ABAP program for which a variant is to be created. • ABAP_VARIANT_NAME: Is the name of the variant to be created. • ABAP_VARIANT_TEXT: Short text of the variant. • EXTERNAL_USER_NAME: Name of the XBP user (user of the external management system).
Parameter (Output)	BAPIRET2 is the return structure used by BAPIs.
Tables	VARIANT_INFO: Contains the information of a report variant.
MessageIDs	<ul style="list-style-type: none"> • MSG_ERROR_IN_FUNCTION: A function called within this function returned an error. This error message contains the name and return code of the called function as variable parts. • MSG_CANT_LOG: Activity was terminated because the SAP XML logging mechanism returned an error. • MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling

	<p>system.</p> <ul style="list-style-type: none"> • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out. • MSG_PROGNAME_MISSING • MSG_NO_VARIANT_SPECIFIED: No variant name has been specified • MSG_PROG_DOES_NOT_EXIST: • MSG_NO_EXECUTE_PRIVILEGE: The user, who wants to create a new variant for a report, does not have the authorization to execute this report • MSG_PROG_NOT_EXECUTABLE: The specified program is not an executable program • MSG_VARIANT_ALREADY_EXISTS: The specified variant already exists • MSG_WRONG_CLIENT: the caller tries to create a system variant, but the logon client is other than 000
--	---

7.12.2 Changing a Variant

Function name	BAPI_XBP_VARIANT_CHANGE
Short description	<p>Change a variant of an ABAP program.</p>  <p>Regarding support of full length 132 (instead of 45) for simple selection fields, see note 1144882.</p>
New in XBP 3.0	This function is new in XBP 3.0
BAPI object name	
BAPI method name	
RFC interface	<pre> Function BAPI_XBP_VARIANT_CHANGE IMPORTING ABAP_PROGRAM_NAME type BAPIXMREP-REPORTID type CHAR length 40 ABAP_VARIANT_NAME type BAPIXMREP-VARIANTNAM type CHAR length 14 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 EXPORTING RETURN structure BAPIRET2 length 548 number of fields 14 TABLES VARIANT_INFO like BAPIVARINFO length 196 number of fields 10 </pre>
Parameter (Input)	<ul style="list-style-type: none"> • ABAP_PROGRAM_NAME: Name of the ABAP program for which a variant is to be changed. • ABAP_VARIANT_NAME: Is the name of the variant to be changed.


	<ul style="list-style-type: none"> EXTERNAL_USER_NAME: Name of the XBP user (user of the external management system).
Parameter (Output)	BAPIRET2 is the return structure used by BAPIs.
Tables	VARIANT_INFO: Contains the information about a report variant.
MessageIDs	<ul style="list-style-type: none"> MSG_ERROR_IN_FUNCTION: A function called within this function returned an error. This error message contains the name and return code of the called function as variable parts. MSG_CANT_LOG: Activity was terminated because the SAP XMI logging mechanism returned an error. MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system. MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out. MSG_PROGNAME_MISSING MSG_NO_VARIANT_SPECIFIED: No variant name has been specified MSG_PROG_DOES_NOT_EXIST MSG_VARIANT_DOES_NOT_EXIST MSG_NO_EXECUTE_PRIVILEGE: The user, who wants to create a new variant for a report, does not have the authorization to execute this report MSG_PROG_NOT_EXECUTABLE: The specified program is not an executable program MSG_WRONG_CLIENT: the caller tries to create a system variant, but the logon client is other than 000

7.12.3 Copying a Variant

Function name	BAPI_XBP_VARIANT_COPY
Short description	Copy a variant of an ABAP program
New in XBP 3.0	This function is new in XBP 3.0
BAPI object name	
BAPI method name	
RFC interface	<pre> Function BAPI_XBP_VARIANT_COPY IMPORTING ABAP_PROGRAM_NAME type BAPIXMREP-REPORTID type CHAR length 40 SOURCE_VARIANT type BAPIXMREP-VARIANTNAM type CHAR length 14 TARGET_VARIANT type BAPIXMREP-VARIANTNAM type CHAR length 140 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 EXPORTING RETURN structure BAPIRET2 length 548 number of fields 14 </pre>

Parameter (Input)	<ul style="list-style-type: none"> • ABAP_PROGRAM_NAME: Name of the ABAP program from which a variant is to be copied. • SOURCE_VARIANT: Name of the variant to copy. • TARGET_VARIANT: Name of the copied variant. • EXTERNAL_USER_NAME: Name of the XBP user (user of the external management system).
Parameter (Output)	BAPIRET2 is the return structure used by BAPIs.
MessageIDs	<ul style="list-style-type: none"> • MSG_CANT_LOG: Activity was terminated because the SAP XMI logging mechanism returned an error. • MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out. • MSG_PROGNAME_MISSING: No program name has been specified • MSG_NO_VARIANT_SPECIFIED: No variant name has been specified • MSG_PROG_DOES_NOT_EXIST • MSG_VARIANT_DOES_NOT_EXIST: The source variant does not exist • MSG_VARIANT_ALREADY_EXISTS: The target variant already exists • MSG_PROG_NOT_EXECUTABLE: The specified program is not an executable program • MSG_WRONG_CLIENT: the caller tries to create a system variant, but the logon client is other than 000. • MSG_ERROR_IN_FUNCTION: A function called within this function returned an error. This error message contains the name and return code of the called function as variable parts.

7.12.4 Reading Variant Data

Function name	BAPI_XBP_VARINFO
Short description	<p>Read the data of all variants of an ABAP program</p>  <p>Regarding support of full length 132 (instead of 45) for simple selection fields, see note 1144882.</p>
New in XBP 3.0	This function is new in XBP 3.0
BAPI object name	
BAPI method name	
RFC interface	<pre> Function BAPI_XBP_VARINFO IMPORTING ABAP_PROGRAM_NAME type BAPIXMREP-REPORTID type CHAR length 40 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 VARIANT_SELECT_OPTION type BAPIXMREP- VARSELOPT optional Default 'A' Type CHAR length 1 EXPORTING RETURN structure BAPIRET2 length 548 number of fields 14 TABLES VARIANT_INFO like BAPIVARINFO length 196 number of fields 10 </pre>
Parameter (Input)	<ul style="list-style-type: none"> • ABAP_PROGRAM_NAME: Name of the ABAP program of which you want to read all variants. • EXTERNAL_USER_NAME: Name of the XBP user (user of the external management system). • VARIANT_SELECT_OPTION: The default value is 'A' which means that all variants are read. 'B' means that only variants for background processing are selected.
Parameter (Output)	BAPIRET2 is the return structure used by BAPIs
Table	VARIANT_INFO: Contains the information about a report variant.
MessageIDs	<ul style="list-style-type: none"> • MSG_ERROR_IN_FUNCTION: A function called within this function returned an error. This error message contains the name and return code of the called function as variable parts. • MSG_CANT_LOG: Activity was terminated because the SAP XMI logging mechanism returned an error. • MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out. • MSG_PROGNAME_MISSING: No program name has been specified

	<ul style="list-style-type: none"> • MSG_PROG_DOES_NOT_EXIST • MSG_NO_EXECUTE_PRIVILEGE: The user, who wants to create a new variant for a report, does not have the authorization to execute this report • MSG_PROG_NOT_EXECUTABLE: The specified program is not an executable program
--	--

7.12.5 Deleting a Variant

Function name	BAPI_XBP_VARIANT_DELETE
Short description	Delete a variant of an ABAP program
New in XBP 3.0	This function is new in XBP 3.0
BAPI object name	
BAPI method name	
RFC interface	<p>Function BAPI_XBP_VARIANT_DELETE</p> <pre> IMPORTING ABAP_PROGRAM_NAME type BAPIXMREP-REPORTID type CHAR length 40 ABAP_VARIANT_NAME type BAPIXMREP-VARIANTNAM type CHAR length 14 EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type CHAR length 16 CLIENT_ONLY type CHAR1 optional EXPORTING RETURN structure BAPIRET2 length 548 number of fields 14 </pre>
Parameter (Input)	<ul style="list-style-type: none"> • ABAP_PROGRAM_NAME: Name of the ABAP program of which a variant is to be deleted. • ABAP_VARIANT_NAME: Name of the variant to be deleted. • EXTERNAL_USER_NAME: Name of the XBP user (user of the external management system). • CLIENT_ONLY: The flag X means that the variant is to be deleted only in current client.
Parameter (Output)	BAPIRET2 is the return structure used by BAPIs.
MessageIDs	<ul style="list-style-type: none"> • MSG_CANT_LOG: Activity was terminated because the SAP XMI logging mechanism returned an error. • MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out. • MSG_PROGNAME_MISSING: No program name has been specified • MSG_NO_VARIANT_SPECIFIED: No variant name has been specified • MSG_PROG_DOES_NOT_EXIST

	<ul style="list-style-type: none"> • I_EXTERNAL_USER_NAME is the name of the XBP user. • I_CALENDER_ID is a wildcard for the calendar ID, default '%': read all. • MSG_VARIANT_DOES_NOT_EXIST: The source variant does not exist • MSG_NO_EXECUTE_PRIVILEGE: The user, who wants to create a new variant for a report, does not have the authorization to execute this report • MSG_PROG_NOT_EXECUTABLE: The specified program is not an executable program • MSG_WRONG_CLIENT: The caller tries to create a system variant, but the logon client is other than 000. • MSG_ERROR_IN_FUNCTION: A function called within this function returned an error. This error message contains the name and return code of the called function as variable parts.
--	---

7.12.6 Reading Selection Screen of an ABAP Program

Function name	BAPI_XBP_READ_SELSCREEN
Short description	Read information about the selection fields of an ABAP program.
New in XBP 3.0	This function is new in XBP 3.0
BAPI object name	
BAPI method name	
RFC interface	<pre> Function BAPI_XBP_READ_SELSCREEN IMPORTING PROGRAM like BAPIXMREP-REPORTID type CHAR length 40 DEFAULT_VALUES like BTCH0000-CHAR1 optional Default 'X' EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type CHAR length 16 EXPORTING RETURN structure BAPIRET2 length 548 number of fields 14 TABLES SELSCREEN_INFO like BAPI_SELSCREEN_INFO (The last 4 fields contain the default values, if there are any, represented in the well known format) </pre>
Parameter (Input)	<ul style="list-style-type: none"> • PROGRAM: Name of the ABAP program of which the selection fields are to be read. • DEFAULT_VALUES: If marked, the default values are also returned. • EXTERNAL_USER_NAME: Name of the XBP user (user of the external management system).

Parameter (Output)	BAPIRET2 is the return structure used by BAPIs
Tables	SELSCREEN_INFO: Information on the selection fields of an ABAP program.

7.12.7 Reading Free Selections of an ABAP Program

Function name	BAPI_XBP_GET_FREE_SELECTIONS
Short description	Read the free selections of an ABAP program.
New in XBP 3.0	This function is new in XBP 3.0
BAPI object name	
BAPI method name	
RFC interface	<pre> Function BAPI_XBP_GET_FREE_SELECTIONS IMPORTING EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type CHAR length 16 ABAP_PROGRAM_NAME type BAPIXMREP-REPORTID type CHAR length 40 EXPORTING FREE_SELINFO type RSDSRANGE_T_SSEL (this is a table with a nested table) FREE_SEL_EXISTS type SYCHAR01 (char 1) RETURN structure BAPIRET2 length 548 number of fields 14 </pre>
Parameter (Input)	<ul style="list-style-type: none"> EXTERNAL_USER_NAME: Name of the XBP user (user of the external management system). ABAP_PROGRAM_NAME: Name of the ABAP program of which free selections are to be read.
Parameter (Output)	<ul style="list-style-type: none"> FREE_SELINFO: Contains information on the free selections. FREE_SEL_EXISTS: The parameter has the value Y, if there are free selections. Otherwise, the value is N. BAPIRET2 is the return structure used by BAPIs.

7.13 Synchronizing Jobs

Synchronizing jobs means, that the external scheduler reads all jobs from the SAP system, which have been created from a certain point of time on. Thus the external scheduler can synchronize its jobs with the job database of an SAP system.

Function name	BAPI_XBP_SYNCHRONIZE_JOBS
Short description	Read the SAP jobs (in order to synchronize the database of an external job scheduler with the SAP job database).
New in XBP 3.0	This function is new in XBP 3.0
BAPI object name	
BAPI method name	
RFC interface	<p>Function BAPI_XBP_SYNCHRONIZE_JOBS</p> <p>IMPORTING</p> <p>JOBNAME type BAPIXMJOB-JOBNAME optional type CHAR length 32</p> <p>USERNAME type BAPIXMJOB-SDLUNAME optional type CHAR length 12</p> <p>NOT_THIS_USER flag optional type CHAR length 1</p> <p>FROM_CREATEDATE type BTCSDLDATE optional type DATS length 8</p> <p>FROM_CREATETIME type BTCSDLTIME optional type TIMS length 6</p> <p>MAX type I default 5000 optional</p> <p>ONLY_THIS_CLIENT optional type CHAR1 default 'N'</p> <p>EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type CHAR length 16</p> <p>EXPORTING</p> <p>RETURN structure BAPIRET2 length 548 number of fields 14</p> <p>NR_OF_ENTRIES type I</p> <p>TO_CREATEDATE type BTCSDLDATE type DATS length 8</p> <p>TO_CREATETIME type BTCSDLTIME type TIMS length 6</p> <p>MORE type BTCH0000-CHAR1</p> <p>TABLES</p> <p>JOBS like BP30JOB</p>
Parameter (Input)	<ul style="list-style-type: none"> • JOBNAME: Wildcard * is possible. If no jobname is specified, * is assumed. • USERNAME: Creator of the jobs to be selected. Wildcard * is possible. • NOT_THIS_USER: If set to X, jobs of all other users than the user specified in USERNAME are selected. • FROM_CREATEDATE, FROM_CREATETIME: Only jobs,

	<p>which were created after the point of time specified by these two parameters, are selected. The upper limit of the time interval is the system time.</p> <ul style="list-style-type: none"> • ONLY_THIS_CLIENT: If set to X, only jobs of the current client are selected. If empty, jobs of all clients are selected. • MAX: In order to cause not too much traffic, when this function is called via RFC, not more than MAX jobs are selected. If MAX is empty or less than 5000, it is internally set to 5000. If there are more than MAX jobs, which fit the selection criteria, it is not intended to return exactly MAX jobs. <p>Example: Assume that system time is 2006/4/5, 16:00:00 and the function is called with FROM_CREATEDATE, FROM_CREATETIME = 2006/04/01 , 00:00:00. Assume further that 12000 jobs have been created since that point of time, and assume JOBNAME = * and USERNAME = * MAX = 5000 ONLY_THIS_CLIENT is empty</p> <p>Since more than 5000 jobs have been created in the time interval under consideration, the function divides the time interval into two halves, the first half being 2006/04/01 , 00:00:00 - 2006/04/03 , 08:00:00 Then the function checks, how many jobs have been created in the first interval. If more than 5000, the process is repeated. If less, these jobs are returned. It can even happen, that 0 jobs are returned. Moreover the output-parameter MORE is set to X to indicate that there are more jobs in the original time interval and that the function has to be called again. The lower limit of the time interval to be considered in the next call is returned in the output parameters TO_CREATEDATE and TO_CREATETIME.</p>
Parameter (Output)	<ul style="list-style-type: none"> • NR_OF_ENTRIES: Number of entries returned • TO_CREATEDATE; TO_CREATETIME : See above • MORE: See above • JOBS: Internal table containing the jobs selected. Note that the status of a job is returned as 'I', if it has been intercepted.
MessageIDs	<ul style="list-style-type: none"> • MSG_INVALID_PARAMETERS: There is something wrong with the specified date and / or time, which is the lower limit of the time interval to be considered. Probably, the date is in the future. • MSG_CANT_LOG: Activity was terminated because the SAP XMI logging mechanism returned an error. • MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.

7.14 Setting Spool List Recipients.

The following functions help to read values (SAP users or distribution lists), which can be used as spool list recipients for jobs. The values returned by the functions contained in this section can be passed to parameter RECIPIENT of BAPI_XBP_JOB_CLOSE.

7.14.1 Reading SAP Users

With this function the SAP users can be read in blocks.

Function name	BAPI_XBP_GET_USER_LIST
Short description	Read the SAP users.
New in XBP 3.0	This function is new in XBP 3.0
BAPI object name	
BAPI method name	
RFC interface	<pre> Function BAPI_XBP_GET_USER_LIST IMPORTING START_ROW type RFC_INT4 optional NR_OF_ROWS type RFC_INT4 optional EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 EXPORTING ROWS type BAPIUSMISC-BAPIROW RETURN structure BAPIRET2 length 548 number of fields 14 TABLES USERLIST like BAPIUSNAME (length 172, number of fields 4) optional </pre>
Parameter (Input)	<p>With this function the SAP users can be read in blocks. With the parameters START_ROW and NR_OF_ROWS (number of rows) the size of the block can be determined. In a subsequent call of this function, the parameter START_ROW should be (START_ROW + NR_OF_ROWS) of the previous call.</p>
Parameter (Output)	<p>The parameter ROWS contains the number of users returned in the table USERLIST. If ROWS < NR_OF_ROWS, no additional block needs to be read.</p>
MessageIDs	<ul style="list-style-type: none"> • MSG_PROBLEM_DETECTED: The SAP job scheduling system has discovered an error • MSG_CANT_LOG: Activity was terminated because the SAP XMI logging mechanism returned an error. • MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.

7.14.2 Reading SAP Office Distribution Lists

Function name	BAPI_XBP_GET_DL_LIST
Short description	Read the SAP Office distribution llists
New in XBP 3.0	This function is new in XBP 3.0
BAPI object name	
BAPI method name	
RFC interface	<pre> Function BAPI_XBP_GET_DL_LIST IMPORTING START_ROW type RFC_INT4 optional NR_OF_ROWS type RFC_INT4 optional EXTERNAL_USER_NAME like BAPIXMLOGR-EXTUSER type RFC_CHAR length 16 EXPORTING ROWS type BAPIUSMISC-BAPIROW RETURN structure BAPIRET2 length 548 number of fields 14 TABLES DLISTS like BAPIDLI (length 75, number of fields 4) optional </pre>
Parameter (Input)	<p>With this function the SAP Office distribution lists can be read in blocks.</p> <p>With the parameters START_ROW and NR_OF_ROWS (number of rows) the size of the block can be determined.</p> <p>In a subsequent call of this function, the parameter START_ROW should be START_ROW + NR_OF_ROWS of the previous call.</p>
Parameter (Output)	<p>The parameter ROWS contains the number of distribution lists returned in the table DLIST. If ROWS < NR_OF_ROWS, no additional block needs to be read.</p> <p>BAPIRET2 is the return structure used by BAPIs.</p>
MessageIDs	<ul style="list-style-type: none"> • MSG_PROBLEM_DETECTED: The SAP job scheduling system has discovered an error • MSG_CANT_LOG: Activity was terminated because the SAP XMI logging mechanism returned an error. • MSG_EXT_USER_MISSING: Name of the external user is missing. This is the name of a user in the external job scheduling system. • MSG_NOT_LOGGED_ON: The external management tool has not logged onto the CCMS XMI interface. Therefore, the activity cannot be carried out.

8 Appendix

8.1.1 BAPI Return Structure

Each XBP function module has a RET structure of type BAPIRET2 as export parameter. With the help of this structure messages are reported from the SAP system to the caller.

STRUCTURE	BAPIRET2	
Field name	Type	Short description
TYPE	CHAR 1	Message typ: S Success, E Error ...
ID	CHAR 20	Message-ID
NUMBER	NUMC 3	Message-Number
MESSAGE	CHAR 220	Message Text
LOG_NO	CHAR 20	Appication-Log: Protocol Number
LOG_MSG_NO	NUMC 6	Application-Log: actual Number
MESSAGE_V1	CHAR 50	Message-Variable
MESSAGE_V2	CHAR 50	Message-Variable
MESSAGE_V3	CHAR 50	Message-Variable
MESSAGE_V4	CHAR 50	Message-Variable
PARAMETER	CHAR 32	Parameter Name
ROW	INT4 10	Line in Parameter
FIELD	CHAR 30	Field in Parameter
SYSTEM	CHAR 10	Logical System; origin of message
Notes		This structure is used for all BAPIs and is not related to XBP directly
Related to		There have been older structures with different names such as: BAPIRETURN1

8.1.2 Message IDs and Their Meaning

In the table below you find a list of the message numbers as they are in the transaction SE91 and the corresponding aliases that are used in the XBP functions described in chapter 7. Note that only messages from the message class XM are used.

You will be able to see the actual text of the message once you analyze the BAPIRET2 values. For your convenience we include here the constants which are referenced in the function module descriptions.

xmi_messages	like sy-msgid	value 'XM',
msg_logon_gen	like sy-msgno	value '9',
msg_logon	like sy-msgno	value '010'
msg_logoff_gen	like sy-msgno	value '011'
msg_logoff	like sy-msgno	value '012'
msg_auditlevel_set	like sy-msgno	value '013'
msg_versions_get_gen	like sy-msgno	value '014'
msg_versions_get	like sy-msgno	value '015'
msg_version_check	like sy-msgno	value '016'
msg_interface_describe	like sy-msgno	value '017'
msg_logmsg_enter	like sy-msgno	value '018'
msg_log_select	like sy-msgno	value '019'
msg_message_formats_upload	like sy-msgno	value '020'
msg_already_logged_on_gen	like sy-msgno	value '021',
msg_already_logged_on	like sy-msgno	value '022',
msg_unknown_interface	like sy-msgno	value '023',
msg_unknown_version	like sy-msgno	value '024',
msg_logon_denied_gen	like sy-msgno	value '025',
msg_logon_denied	like sy-msgno	value '026',
msg_not_logged_on_gen	like sy-msgno	value '027',
msg_not_logged_on	like sy-msgno	value '028',
msg_invalid_range	like sy-msgno	value '029',
msg_cant_select	like sy-msgno	value '030',
msg_cant_log	like sy-msgno	value '031',
msg_cant_upload	like sy-msgno	value '032',
msg_invalid_parameters	like sy-msgno	value '033',
msg_problem_detected	like sy-msgno	value '034',
msg_reorg	like sy-msgno	value '035',
msg_reorg_gen	like sy-msgno	value '037',
msg_jobname_missing	like sy-msgno	value '046',
msg_jobid_missing	like sy-msgno	value '047',
msg_ext_user_missing	like sy-msgno	value '048',
msg_job_does_not_exist	like sy-msgno	value '049',

msg_progname_missing	like sy-msgno	value '050',
msg_no_archive_info	like sy-msgno	value '051'
msg_invalid_print_params	like sy-msgno	value '052',
msg_invalid_archive_params	like sy-msgno	value '053',
msg_no_release_privilege	like sy-msgno	value '054',
msg_job_not_active	like sy-msgno	value '055',
msg_no_abort_privilege	like sy-msgno	value '056',
msg_no_job_found	like sy-msgno	value '057',
msg_targethost_missing	like sy-msgno	value '058'
msg_no_jobsteps	like sy-msgno	value '059',
msg_no_job_protocol	like sy-msgno	value '060',
msg_empty_job_protocol	like sy-msgno	value '061',
msg_step_count_missing	like sy-msgno	value '062',
msg_no_spoolist	like sy-msgno	value '063',
msg_privilege_missing	like sy-msgno	value '064',
msg_invalid_spoolid	like sy-msgno	value '065',
msg_no_immediate_start_poss	like sy-msgno	value '066',
msg_no_resources_found	like sy-msgno	value '067',
msg_invalid_date_time	like sy-msgno	value '068',
msg_invalid_server_name	like sy-msgno	value '069',
msg_prog_has_no_variant	like sy-msgno	value '070',
msg_prog_does_not_exist	like sy-msgno	value '071',
msg_no_execute_privilege	like sy-msgno	value '072',
msg_prog_not_executable	like sy-msgno	value '073',
msg_no_variants_defined	like sy-msgno	value '074'
msg_invalid_select_option	like sy-msgno	value '075'
msg_select_param_missing	like sy-msgno	value '076'
msg_trace_before_call	like sy-msgno	value '077'
msg_select_jobname_missing	like sy-msgno	value '078'
msg_select_username_missing	like sy-msgno	value '079'
msg_cant_del_in_jobtable	like sy-msgno	value '080'
msg_cant_del_joblog	like sy-msgno	value '081'
msg_problem_pred_succ	like sy-msgno	value '082'
msg_commit_failed	like sy-msgno	value '083'
msg_no_delete_privilege	like sy-msgno	value '084'
msg_job_running	like sy-msgno	value '085'
msg_interface_reorg	like sy-msgno	value '086'
msg_interface_reorg_gen	like sy-msgno	value '087'
msg_parent_child_inconsistency	like sy-msgno	value '088'

msg_child_register_error	like sy-msgno	value '089'
msg_mask_error	like sy-msgno	value '090'
msg_param_missing	like sy-msgno	value '091'
msg_event_does_not_exist	like sy-msgno	value '092'
msg_event_raise_failed	like sy-msgno	value '093'
msg_job_confirmation_failed	like sy-msgno	value '094'
msg_wrong_confirmation_type	like sy-msgno	value '095'
msg_wrong_selection_par	like sy-msgno	value '096'
msg_parentchild_inactive	like sy-msgno	value '097'
msg_interception_inactive	like sy-msgno	value '098'
msg_wrong_counter	like sy-msgno	value '099'
msg_wrong_printer_name	like sy-msgno	value '100'
msg_selection_finished	like sy-msgno	value '101'

msg_cant_enq_job	like sy-msgno	value '194'
msg_cant_read_jobdata	like sy-msgno	value '195'
msg_cant_release_job	like sy-msgno	value '196'
msg_cant_set_jobstatus_in_db	like sy-msgno	value '197'
msg_cant_start_job_immediately	like sy-msgno	value '198'
msg_cant_update_jobdata	like sy-msgno	value '199'
msg_eventcnt_generation_error	like sy-msgno	value '200'
msg_invalid_dialog_type	like sy-msgno	value '201'
msg_invalid_new_jobdata	like sy-msgno	value '202'
msg_invalid_new_jobstatus	like sy-msgno	value '203'
msg_invalid_startdate	like sy-msgno	value '204'
msg_job_edit_failed	like sy-msgno	value '205'
msg_job_modify_canceled	like sy-msgno	value '206'
msg_job_not_modifiable_anymore	like sy-msgno	value '207'
msg_nothing_to_do	like sy-msgno	value '208'
msg_no_batch_on_target_host	like sy-msgno	value '209'
msg_no_batch_server_found	like sy-msgno	value '210'
msg_no_batch_wp_for_jobclass	like sy-msgno	value '211'
msg_no_modify_privilege_given	like sy-msgno	value '212'
msg_no_release_privilege_given	like sy-msgno	value '213'
msg_no_startdate_no_release	like sy-msgno	value '214'
msg_invalid_targetgroup	like sy-msgno	value '216'
msg_conflicting_targets	like sy-msgno	value '217'
msg_job_doesnt_have_steps	like sy-msgno	value '218'
msg_wrong_step_type	like sy-msgno	value '219'

msg_job_doesnt_have_this_step	like sy-msgno	value '220'
msg_cannot_get_priarc_params	like sy-msgno	value '221'
msg_cannot_read_job	like sy-msgno	value '222'
msg_cannot_modify_job	like sy-msgno	value '223'
msg_wrong_step_number	like sy-msgno	value '224'
msg_error_modifying_worktable	like sy-msgno	value '225'
msg_job_nosteps	like sy-msgno	value '227'
msg_jobcount_missing	like sy-msgno	value '228'
msg_invalid_target	like sy-msgno	value '229'
msg_error_reading_worktable	like sy-msgno	value '230'
msg_delete_line_error	like sy-msgno	value '231'
msg_no_step_info	like sy-msgno	value '232'
msg_wrong_action	like sy-msgno	value '233'
msg_no_change_authority	like sy-msgno	value '234'
msg_invalid_jobclass	like sy-msgno	value '235'
msg_wrong_client	like sy-msgno	value '236'
msg_calender_does_not_exist	like sy-msgno	value '267'

8.1.3 Document Type Definition for Defining Profiles and Criteria for Event History

8.1.3.1 Overview

By calling the function modules BAPI_CM_PROFILE_CREATE and BAPI_CM_CRITERIA_SET, the external scheduler can create a criteria profile in event history and import a criteria hierarchy in a criteria profile from an XML source file. The XML source file provides a description of that profile and criteria hierarchy respectively. This section provides and explains the Document Type Definition (DTD) to be used when preparing an XML description of a criteria profile or of a criteria hierarchy.

- Criteria hierarchy

A criteria hierarchy is the set of all the **criteria and conditions** which event names and/or event arguments of the raised events need to fulfill to be logged in the event history.

When you prepare an XML description of a criteria hierarchy for event history, you specify conditions for the **fields** EVENTID (event name) and EVENTPARM (event argument). Conditions are grouped into criteria. A criterion may contain separate conditions for EVENTID and EVENTPARM, or only for one of the fields EVENTID and EVENTPARM.

Criteria are grouped into **nodes**. Nodes impose the logical relationship that governs the conditions and/or criteria in it. A node can be of one of two types: AND or OR, which impose an AND or OR relationship respectively. A node can contain one or more criteria.

All criteria and nodes in a hierarchy are grouped into the **root** of the hierarchy. The relationship between the criteria and nodes in the root is always an AND relationship.

- Criteria profile

A criteria hierarchy is stored in a criteria profile which can be in one of two states: active and inactive.

8.1.3.2 Setting a Criteria Hierarchy

The excerpt below provides the DTD for a criteria profile and its criteria hierarchy for event history. The DTD elements and their attributes are explained in the text the accompanying texts.



```

<!ELEMENT criterion EMPTY>
<!ATTLIST criterion
  sign CDATA #REQUIRED
  opt CDATA #REQUIRED
  low CDATA #REQUIRED
  high CDATA #IMPLIED>

<!ELEMENT field (#PCDATA|criterion)*>

<!ELEMENT item (field)+>
<!ATTLIST item
  description CDATA #REQUIRED>

<!ELEMENT node ((item)*,(node)*,(item)*,(node)*,(item)*,(node)*)>
<!ATTLIST node
  type CDATA #REQUIRED>

<!ELEMENT root ((item)*,(node)*,(item)*,(node)*,(item)*,(node)*)>

<!ELEMENT profile (root)>
<!ATTLIST profile
  type CDATA #REQUIRED
  id CDATA #IMPLIED
  description CDATA #REQUIRED
  state CDATA #IMPLIED
  lastchuser CDATA #IMPLIED
  lastchtmstamp CDATA #IMPLIED
  createuser CDATA #IMPLIED>

```

<!ELEMENT criterion EMPTY>

<!-- The element criterion specifies the conditions for the values of the EVENTID and/or EVENTPARM fields. A criterion may contain separate conditions for both fields or for just one of the them. -->

<!ATTLIST criterion

sign CDATA #REQUIRED

<!-- The attribute sign of a criterion specifies the general operation between the field (EVENTID or EVENTPARM) and the field values. The possible values for sign are:

I = include.

E = exclude. -->

opt CDATA #REQUIRED

<!--The criterion attribute opt specifies the operation between the field and the field values. The possible values for opt are:

BT – between. You can specify a range of values for a field.

NB – exclude (not between). You can exclude a range of values for a field.

EQ – equal. You can provide an exact value.

GE – greater or equal to and exact value.

GT – greater than and exact value.

LE – less or equal to an exact value.

LT – less than an exact value.

NE – not equal to. You can exclude exact values.

CP – contains pattern. You can specify a range of values by using a pattern. The following wildcard patterns are supported: *, and ?.

NP – no pattern. You can exclude values by using a pattern. The following wildcard patterns are supported: *, and ?-->

low CDATA #REQUIRED

<!--Specifies an exact value, for example, when opt = EQ, or the lowest exact value of a range, for example, when opt = GT, or GE -->

high CDATA #IMPLIED>

<!-- Specifies the highest value of a range, for example, when opt = LE or LT -->

<!ELEMENT field (#PCDATA|criterion)*>

<!--The element field specifies the field – EVENTID or EVENTPARG, for which the conditions in the criterion apply and is a placeholder for the criterion. -->

<!ELEMENT item (field)+>

<!-- The element item is a placeholder for the criteria applicable to the fields EVENTID and EVENTPARG. -->

<!ATTLIST item

description CDATA #REQUIRED>

<!--The attribute description provides a description of a criterion. -->

<!ELEMENT node

((item)*,(node)*,(item)*,(node)*,(item)*,(node)*)>

<!-- By grouping item elements, the element node groups criteria in the hierarchy and imposes a logical relationship between them. Depending on its type, the node imposes a logical AND or OR relationship between items. -->

<!ATTLIST node

type CDATA #REQUIRED>

<!-- The attribute type of a node specifies whether the node is an AND or OR node and determines the logical relationship which the node imposes on the criteria it contains The possible values for type are:

A = AND. The node of type AND imposes an AND relationship between the criteria it contains.

O = OR. The node of type OR imposes and OR relationship between the criteria it contains.

-->

<!ELEMENT root

((item)*,(node)*,(item)*,(node)*,(item)*,(node)*)>

<!-- The element root groups all the nodes in the criteria hierarchy. The root always imposes an AND relationship between the nodes it contains. Criteria do not need to be grouped in nodes: you can insert criteria directly in the root element. In this case, criteria are related by the AND relationship imposed by the root.-->

<!ELEMENT profile (root)>

<!-- The element profile is a placeholder for the whole criteria hierarchy. -->

<!ATTLIST profile

type CDATA #REQUIRED

<!-- The attribute type specifies the type of the criteria type. The value for a profile for event history is EVTHIS. -->


```

id CDATA #IMPLIED
description CDATA #REQUIRED
  <!-- The attribute description specifies a free-text description of the profile.--
  >
state CDATA #IMPLIED
lastchuser CDATA #IMPLIED
lastchtmstp CDATA #IMPLIED
createuser CDATA #IMPLIED>

```



The external scheduler has to create a profile named *My_Profile* with criteria specifying which events are to be logged in the event history. Event history has to log all events with event name MY_EVENT_1 and event argument greater than 123, as well as all events with an event name starting with ABC and an event argument equal to 123.

The excerpt below provides an XML description of this criteria profile and the corresponding criteria hierarchy:



```

<?xml version="1.0"?> <!DOCTYPE profile SYSTEM
"criteria_profile.dtd">
<profile type="EVTHIS" id="15" description="My_Profile"
<root>
  <node type="0">
    <item description="Criterion 1">
      <field>
        EVENTID
        <criteria sign="I" opt="EQ" low="MY_EVENT_1" high=""/>
      </field>
      <field>
        EVENTPARG
        <criteria sign="I" opt="GT" low="123" high=""/>
      </field>
    </item>
    <item description="Criterion 2">
      <field>
        EVENTID
        <criteria sign="I" opt="EQ" low="ABC*" high=""/>
      </field>
      <field>
        EVENTPARG
        <criteria sign="I" opt="EQ" low="123" high=""/>
      </field>
    </item>
  </node>
</root>
</profile>

```

8.1.3.3 Creating a Blank Criteria Profile

If you want to create a blank criteria profile (BAPI_CM_PROFILE_CREATE) without setting any criteria hierarchy, you use the same DTD as for setting a criteria hierarchy. However, the XML source has to provide a description only for the profile attributes. The excerpt below shows an XML source for a criteria profile for event history with name *My_Profile*.

```
<?xml version="1.0"?> <!DOCTYPE profile SYSTEM  
"criteria_profile.dtd">  
<profile type="EVTHIS" id="<ID>" description="My_Profile">  
<root/>  
</profile>
```

In the XML source, you can specify an integer value for the ID attribute. If there is no profile with this ID, the profile is created with the ID you specified. If the ID is already occupied by another profile, an ID is generated for the profile you create from the XML source.

8.1.4 Language Key Mapping

The following table shows an overview of the mapping of one-digit SAP language keys to two-digit ISO language codes.

SAP language key	UC representation of SAP language key	ISO language code	Name of Language
1	U+0031	ZH	Chinese
2	U+0032	TH	Thai
3	U+0033	KO	Korean
4	U+0034	RO	Romanian
5	U+0035	SL	Slovene
6	U+0036	HR	Croatian
7	U+0037	MS	Malaysian
8	U+0038	UK	Ukrainian
A	U+0041	AR	Arabic
B	U+0042	HE	Hebrew
C	U+0043	CS	Czech
D	U+0044	DE	German
E	U+0045	EN	English
F	U+0046	FR	French
G	U+0047	EL	Greek
H	U+0048	HU	Hungarian
I	U+0049	IT	Italian
J	U+004A	JA	Japanese
K	U+004B	DA	Danish
L	U+004C	PL	Polish
M	U+004D	ZF	Chinese trad.
N	U+004E	NL	Dutch
O	U+004F	NO	Norwegian
P	U+0050	PT	Portuguese
Q	U+0051	SK	Slovakian
R	U+0052	RU	Russian
S	U+0053	ES	Spanish
T	U+0054	TR	Turkish
U	U+0055	FI	Finnish
V	U+0056	SV	Swedish
W	U+0057	BG	Bulgarian
Z	U+005A	Z1	Customer reserve

ﻻ	U+B282	1B	Arabic_QA
ﻻ	U+B283	1C	Chinese_HK
ﻻ	U+B2E2	2B	Arabic_SY
ﻻ	U+B342	3B	Arabic_TN
ﻻ	U+B466	6F	French_HT
ﻻ	U+B46E	6N	English_GB
ﻻ	U+B4C6	7F	French_LU
ﻻ	U+B4CE	7N	English_ZW
ﻻ	U+B526	8F	French_ML
ﻻ	U+B585	9E	English_JM
ﻻ	U+B586	9F	French_MC
ﻻ	U+B881	AA	Afar
ﻻ	U+B882	AB	Abkhazian
ﻻ	U+B883	AC	Achinese
ﻻ	U+B885	AE	Avestan
ﻻ	U+B88B	AK	Akan
ﻻ	U+B88D	AM	Amharic
ﻻ	U+B88E	AN	Aragonese
ﻻ	U+B893	AS	Assamese
ﻻ	U+B896	AV	Avaric
ﻻ	U+B899	AY	Aymara
ﻻ	U+B89A	AZ	Azerbaijani
ﻻ	U+B8E1	BA	Bashkir
ﻻ	U+B8E5	BE	Belarusian
ﻻ	U+B8E8	BH	Bihari
ﻻ	U+B8E9	BI	Bislama
ﻻ	U+B8ED	BM	Bambara
ﻻ	U+B8EE	BN	Bengali
ﻻ	U+B8EF	BO	Tibetan
ﻻ	U+B8F2	BR	Breton
ﻻ	U+B8F3	BS	Bosnian
ﻻ	U+B938	C8	CreolesPidgins P
ﻻ	U+B945	CE	Chechen
ﻻ	U+B948	CH	Chamorro
ﻻ	U+B94F	CO	Corsican

뤿	U+B952	CR	Cree
뤼	U+B955	CU	Church Slavic
뤽	U+B956	CV	Chuvash
뤼	U+B959	CY	Welsh
뎡	U+B9B6	DV	Divehi
뎡	U+B9BA	DZ	Dzongkha
뎡	U+BA0F	EO	Esperanto
뎡	U+BA15	EU	Basque
뎡	U+BA61	FA	Persian
뎡	U+BA66	FF	Fulah
뎡	U+BA6A	FJ	Fijian
뎡	U+BA6F	FO	Faroese
뎡	U+BA79	FY	Frisian
뎡	U+BAC1	GA	Irish
뎡	U+BAC4	GD	Gaelic
뎡	U+BACC	GL	Gallegan
뎡	U+BACE	GN	Guarani
뎡	U+BAD5	GU	Gujarati
뎡	U+BAD6	GV	Manx
뎡	U+BB21	HA	Hausa
뎡	U+BB29	HI	Hindi
뎡	U+BB2F	HO	Hiri Motu
뎡	U+BB34	HT	Haitian
뎡	U+BB39	HY	Armenian
뎡	U+BB3A	HZ	Herero
뎡	U+BB81	IA	Interlingua
뎡	U+BB85	IE	Interlingue
뎡	U+BB87	IG	Igbo
뎡	U+BB89	II	Sichuan Yi
뎡	U+BB8B	IK	Inupiaq
뎡	U+BB8F	IO	Ido
뎡	U+BB95	IU	Inuktitut
뎡	U+BBF6	JV	Javanese
뎡	U+BC41	KA	Georgian
뎡	U+BC49	KI	Kikuyu
뎡	U+BC4A	KJ	Kuanyama

벵	U+BC4B	KK	Kazakh
바	U+BC4C	KL	Greenlandic
박	U+BC4D	KM	Khmer
밖	U+BC4E	KN	Kannada
밭	U+BC52	KR	Kanuri
밭	U+BC53	KS	Kashmiri
밖	U+BC55	KU	Kurdish
밖	U+BC56	KV	Komi
밖	U+BC57	KW	Cornish
밖	U+BC59	KY	Kirghiz
백	U+BCA1	LA	Latin
백	U+BCA2	LB	Luxembourgish
벨	U+BCA7	LG	Ganda
백	U+BCA9	LI	Limburgish
벨	U+BCAE	LN	Lingala
벵	U+BCAF	LO	Lao
벵	U+BCB5	LU	Luba-Katanga
붓	U+BD07	MG	Malagasy
붓	U+BD08	MH	Marshall
붕	U+BD09	MI	Maori
붓	U+BD0B	MK	Macedonian
북	U+BD0C	ML	Malayalam
분	U+BD0E	MN	Mongolian
붕	U+BD0F	MO	Moldavian
밖	U+BD12	MR	Marathi
판	U+BD14	MT	Maltese
밖	U+BD19	MY	Burmese
릴	U+BD61	NA	Nauru
뵤	U+BD62	NB	Norwegian Bokmal
뵤	U+BD64	ND	Ndebele, North
북	U+BD65	NE	Nepali
붓	U+BD67	NG	Ndonga
붓	U+BD69	NI	Niger- Kordofani
뵤	U+BD6E	NN	Norweg. Nynorsk

불	U+BD72	NR	Ndebele, South
북	U+BD76	NV	Navajo
붕	U+BD79	NY	Nyanja
뵈	U+BDC3	OC	Occitan
뵊	U+BDCA	OJ	Ojibwa
뵋	U+BDCD	OM	Oromo
뵌	U+BDD2	OR	Oriya
뵍	U+BDD3	OS	Ossetian
부	U+BE21	PA	Punjabi
픽	U+BE29	PI	Pali
푼	U+BE33	PS	Pushto
뵠	U+BE95	QU	Quechua
뵡	U+BEED	RM	Rhaeto- Romance
뵢	U+BEEE	RN	Rundi
뵣	U+BEF7	RW	Kinyarwanda
뵤	U+BF41	SA	Sanskrit
뵦	U+BF43	SC	Sardinian
뵧	U+BF44	SD	Sindhi
뵨	U+BF45	SE	Northern Sami
뵩	U+BF47	SG	Sango
뵪	U+BF49	SI	Sinhalese
뵫	U+BF4D	SM	Samoan
뵬	U+BF4E	SN	Shona
뵭	U+BF4F	SO	Somali
뵮	U+BF51	SQ	Albanian
뵯	U+BF53	SS	Swati
뵰	U+BF54	ST	Sotho, Southern
뵱	U+BF55	SU	Sundanese
뵲	U+BF57	SW	Swahili
뵳	U+BFA1	TA	Tamil
뵴	U+BFA5	TE	Telugu
뵵	U+BFA7	TG	Tajik
뵶	U+BFA9	TI	Tigrinya
뵷	U+BFAB	TK	Turkmen

뵓	U+BFAC	TL	Tagalog
뵔	U+BFAE	TN	Tswana
뵕	U+BFAF	TO	Tonga
뵖	U+BFB3	TS	Tsonga
뵗	U+BFB4	TT	Tatar
뵘	U+BFB7	TW	Twi
뵙	U+BFB9	TY	Tahitian
뵚	U+C007	UG	Uighur
뵛	U+C012	UR	Urdu
뵜	U+C01A	UZ	Uzbek
뵝	U+C065	VE	Venda
뵞	U+C069	VI	Vietnamese
뵟	U+C06F	VO	Volapuk
샹	U+C0C1	WA	Walloon
샹	U+C0CF	WO	Wolof
셧	U+C128	XH	Xhosa
셧	U+C189	YI	Yiddish
슌	U+C18F	YO	Yoruba

9 INDEX

A

ABAP reports
 search with wildcards · 117

Aborting jobs
 (BAPI_XBP_JOB_ABORT) · 60

Activating Criteria Profiles · 113

Adding ABAP steps
 (BAPI_XBP_JOB_ADD_ABAP_STEP) · 42

Adding job step via XMI
 (BAPI_XBP_ADD_JOB_STEP) · 69

Appendix · 146

Application Information
 obtaining · 103

Archive parameter search · 122

Assigning ext. program to steps
 (BAPI_XBP_JOB_ADD_EXT_STEP) · 46

B

Background processing
 introduction · 14

BAPI return structure · 146

BAPI_CM_CRITERIA_GET · 115

BAPI_CM_CRITERIA_SET · 116

BAPI_CM_CRITTYPES_GET · 111

BAPI_CM_PROFILE_ACTIVATE · 113

BAPI_CM_PROFILE_CREATE · 112

BAPI_CM_PROFILE_DEACTIVATE · 114

BAPI_CM_PROFILE_DELETE · 113

BAPI_CM_PROFILE_GET · 115

BAPI_XBP_ADD_JOB_STEP · 69

BAPI_XBP_APPL_CONTENT_GET · 104

BAPI_XBP_APPL_INFO_GET · 103

BAPI_XBP_BTC_EVTHIST_CONFIRM · 109

BAPI_XBP_BTC_EVTHISTORY_GET · 107

BAPI_XBP_CONFIRM_JOB · 78

BAPI_XBP_EVENT_DEFINITIONS_GET · 110,
 123

BAPI_XBP_EVENT_RAISE · 56

BAPI_XBP_EXT_COMM_SEARCH · 118

BAPI_XBP_FACT_CALENDERS_GET · 131

BAPI_XBP_GET_ARCHIVE_OBJECTS · 122

BAPI_XBP_GET_BP_RESRC_ON_DATE · 130

BAPI_XBP_GET_BP_SRVRES_ON_DATE ·
 128

BAPI_XBP_GET_CURR_BP_RESOURCES ·
 127

BAPI_XBP_GET_DL_LIST · 144

BAPI_XBP_GET_FREE_SELECTIONS · 140

BAPI_XBP_GET_INTERCEPTED_JOBS · 76

BAPI_XBP_GET_USER_LIST · 143

BAPI_XBP_HOL_CALENDERS_GET · 132

BAPI_XBP_JOB_COPY · 57

BAPI_XBP_JOB_READ_SINGLE_SPOOL · 91

BAPI_XBP_JOB_ABAP_STEP_MODIFY · 63

BAPI_XBP_JOB_ABORT · 60

BAPI_XBP_JOB_ADD_ABAP_STEP · 42

BAPI_XBP_JOB_ADD_EXT_STEP · 46

BAPI_XBP_JOB_CHILDREN_GET · 98

BAPI_XBP_JOB_CLOSE · 47

BAPI_XBP_JOB_COUNT · 96

BAPI_XBP_JOB_DEFINITION_GET · 50

BAPI_XBP_JOB_EXT_STEP_MODIFY · 67

BAPI_XBP_JOB_GET_SPOOL_ATTRIBUTES ·
 90

BAPI_XBP_JOB_HEADER_MODIFY · 59, 62

BAPI_XBP_JOB_JOBLOG_READ · 85

BAPI_XBP_JOB_OPEN · 41

BAPI_XBP_JOB_PARENT_CHILD_INFO · 100

BAPI_XBP_JOB_READ · 97

BAPI_XBP_JOB_SELECT · 94

BAPI_XBP_JOB_SPOOLLIST_READ_20 · 88

BAPI_XBP_JOB_START_ASAP · 55

BAPI_XBP_JOB_START_IMMEDIATELY · 54

BAPI_XBP_JOB_STATUS_CHECK · 92

BAPI_XBP_JOB_STATUS_GET · 82

BAPI_XBP_JOBLIST_STATUS_GET · 84

BAPI_XBP_MODIFY_CRITERIA_TABLE · 80

BAPI_XBP_MODIFY_JOB_STEP · 72

BAPI_XBP_NEW_FUNC_CHECK · 102

BAPI_XBP_OUTPUT_DEVICE_SEARCH · 119

BAPI_XBP_PRINT_FORMAT_SEARCH · 121

BAPI_XBP_READ_SELSCREEN · 139

BAPI_XBP_REPORT_SEARCH · 117

BAPI_XBP_SPECIAL_CONFIRM_JOB · 79

BAPI_XBP_SYNCHRONIZE_JOBS · 141

BAPI_XBP_VARIANT_CHANGE · 134

BAPI_XBP_VARIANT_COPY · 135

BAPI_XBP_VARIANT_CREATE · 133

BAPI_XBP_VARIANT_DELETE · 138

BAPI_XBP_VARIANT_INFO_GET · 125

BAPI_XBP_VARINFO · 137

BAPI_XMI_LOGOFF · 40

BAPI_XMI_LOGON · 38

Batch events
 search with wildcards · 123

C

Changing a Variant
 (BAPI_XBP_VARIANT_CHANGE) · 134

Changing job steps via XMI
 (BAPI_XBP_MODIFY_JOB_STEP) · 72

Checking job resources on any server
 (BAPI_XBP_GET_BP_RESRC_ON_DATE) ·
 130

Checking job status
 (BAPI_XBP_JOB_STATUS_CHECK) · 92

Closing job definitions
 (BAPI_XBP_JOB_CLOSE) · 47

Configuring Profiles and Criteria · 111

Confirming Events in Event History
 (BAPI_XBP_BTC_EVTHISTORY_CONFIRM
) · 109

Confirming jobs · 22, 76, 78

Confirming jobs (special confirm)
 (BAPI_XBP_SPECIAL_CONFIRM_JOB) · 79

Confirming jobs generally
 (BAPI_XBP_CONFIRM_JOB) · 78

Consuming Raised Events · 107

Copying a Variant
 (BAPI_XBP_VARIANT_COPY) · 135

Copying jobs
 (BAPI_XBP_JOB_COPY) · 57

Create jobs · 16

Creating a Blank Criteria Profile · 154

Creating a Variant
 (BAPI_XBP_VARIANT_CREATE) · 133

Creating Criteria Profiles · 112

Criteria

- import in profile · 116
- Criteria Hierarchy
 - get · 115
- Criteria Manager
 - configuring profiles and criteria · 111
- Criteria Profiles
 - activate · 113
 - create · 112
 - deactivate · 114
 - delete · 113
 - get · 115
 - working with · 112
- Criteria table · 19
 - modify · 80
- Criteria Types
 - finding information · 111

D

- Database · 24
- Deactivating Criteria Profiles · 114
- Define jobs · 41
- Deleting a Variant
 - (BAPI_XBP_VARIANT_DELETE) · 138
- Deleting Criteria Profiles · 113
- Deleting jobs
 - (BAPI_XBP_JOB_DELETE) · 62
- Determining current job resources
 - (BAPI_XBP_GET_CURR_BP_RESOURCES) · 127
- Determining job children
 - (BAPI_XBP_JOB_CHILDREN_GET) · 98
- Determining job list status
 - (BAPI_XBP_JOBLIST_STATUS_GET) · 84
- Determining job status
 - (BAPI_XBP_JOB_STATUS_GET) · 82
- Determining jobs with particular name
 - (BAPI_XBP_JOB_COUNT) · 96
- Determining parent/child relation
 - (BAPI_XBP_JOB_PARENT_CHILD_INFO) · 100
- Document Type Definition · 150
- DTD for Defining Profiles and Criteria for Event History · 150
- Dynamic job prioritization · 19

E

- End jobs · 19
- Event
 - trigger from outside · 56
- Events
 - confirming · 109, 110
 - consuming · 107
 - reading from history · 107
- External commands
 - search with wildcards · 118
- External interface · 27
 - function description · 14
 - types · 27

F

- Factory Calendars

- Reading · 131
- Free Selections
 - Reading · 140

G

- Getting Criteria Hierarchy · 115
- Getting Criteria Profiles · 115
- Getting information on a particular spool list · 90
- Getting intercepted jobs
 - (BAPI_XBP_GET_INTERCEPTED_JOBS) · 76

H

- Help functions (general) · 125
- Holiday Calendars
 - Reading · 132

I

- Importing Criteria in a Profile · 116
- Intercept jobs · 19, 76
- Intercept status
 - read and change · 102
- Interface description · 32

J

- Job
 - abort · 60
 - confirm · 76, 78
 - confirmation · 22
 - control · 59
 - copy · 57
 - define · 41
 - delete · 62
 - intercept · 76
 - log · 25
 - open · 41
 - output · 25
 - prioritization · 19
 - select · 94
 - start · 54
 - start asap · 55
 - start immediately · 54
- Job children
 - determine · 98
- job header
 - modify · 59
- Job list status
 - determine · 84
- Job log
 - read · 85
- Job monitor data
 - find, control, and modify · 82
- Job scheduler · 24
- Job Scheduling Architecture · 24
- Job Starter · 25
- Job status
 - check · 92
 - determine · 82
- Job step

add via XMI · 69
change via XMI · 69
delete via XMI · 69
modify · 63
Jobs
Synchronize · 141

L

Language Key Mapping · 155
Logging off (BAPI_XMI_LOGOFF) · 40
Logging on
(BAPI_XMI_LOGON) · 38

M

Message IDs and their meanings · 147
Modify job steps (ext. program)
(BAPI_XBP_JOB_EXT_STEP_MODIFY) · 67
Modifying criteria table
(BAPI_XBP_MODIFY_CRITERIA_TABLE) · 80
Modifying global data
(BAPI_XBP_JOB_HEADER_MODIFY) · 59
Modifying job step containing a report
(BAPI_XBP_JOB_ABAP_STEP_MODIFY) · 63
Monitoring Performance
(BAPI_XBP_BTC_SATISTIC_GET) · 106

N

Naming conventions · 27

O

Obtaining Application Information · 103
Obtaining key job parameter from headers and steps
(BAPI_XBP_JOB_READ) · 97
Opening jobs
(BAPI_XBP_JOB_OPEN) · 41
Output device
search with wildcards · 119

P

Parent/Child functionality · 21
Parent/child relation
determine · 100
read and change · 102
Periodic intercepted jobs · 20
Print formats
search with wildcards · 121

R

Reading and changing (Status Intercept - Parent/Child)
(BAPI_XBP_NEW_FUNC_CHECK) · 102

Reading Event Definitions in Event History
(BAPI_XBP_EVENT_DEFINITIONS_GET) · 110
Reading Events From Event History
(BAPI_XBP_BTC_EVTHISTORY_GET) · 107
Reading Factory Calendars
(BAPI_XBP_FACT_CALENDERS_GET) · 131
Reading Free Selections
(BAPI_XBP_GET_FREE_SELECTIONS) · 140
Reading Holiday Calendars
(BAPI_XBP_HOL_CALENDERS_GET) · 132
Reading job definitions
(BAPI_XBP_JOB_DEFINITION_GET) · 50
Reading job logs
(BAPI_XBP_JOB_JOBLOG_READ) · 85
Reading job spool list · 88
Reading SAP Office Distribution Lists
(BAPI_XBP_GET_DL_LIST) · 144
Reading SAP Users
BAPI_XBP_GET_USER_LIST · 143
Reading Selection Screen
(BAPI_XBP_READ_SELSCREEN) · 139
Reading Variant Data
(BAPI_XBP_VARINFO) · 137
Reference manual · 38
Release information · 9
Release jobs · 17
Remote function call · 29
Retrieving Information on Criteria Types
(BAPI_CM_CRITTTYPES_GET) · 111

S

SAP Office distribution lists
reading · 144
SAP users
reading · 143
Searching for ABAP reports with wildcards
(BAPI_XBP_REPORT_SEARCH) · 117
Searching for archive parameters · 122
Searching for batch events with wildcards
(BAPI_XBP_EVENT_DEFINITIONS_GET) · 123
Searching for external commands with wildcards
(BAPI_XBP_EXT_COMM_SEARCH) · 118
Searching for output devices with wildcards
(BAPI_XBP_OUTPUT_DEVICE_SEARCH) · 119
Searching for print formats with wildcards
(BAPI_XBP_PRINT_FORMAT_SEARCH) · 121
Searching with wildcards · 117
Selecting jobs
(BAPI_XBP_JOB_SELECT) · 94
Selection Screen
Reading · 139
Server resource information (date and time)
(BAPI_XBP_GET_BP_SRVRES_ON_DATE) · 128
Setting a Criteria Hierarchy · 151
Setting Spool List Recipients · 143, 162
Showing all defined variants for an ABAP program.
(BAPI_XBP_VARIANT_INFO_GET) · 125
Spool list

getting information on a particular one · 90
Spool list read
 job spool list · 88
Spool list recipients
 setting · 143
Start jobs · 18
Starting jobs asap
 (BAPI_XBP_JOB_START_ASAP) · 55
Starting jobs immediately
 (BAPI_XBP_JOB_START_IMMEDIATELY)
 · 54
Statistic Information
 obtaining · 106
Status
 intercept · 20
 intercept-confirmed · 20
Symbols · 12
Synchronize Jobs · 141

T

TBCICPT1 · 19
TBCICPT1 (criteria table)
 modify · 80
Triggering event from outside
 (BAPI_XBP_EVENT_RAISE) · 56

V

Variant
 getting variant info for ABAP progr. · 125
Variant Functions · 133
Variants
 Changing · 134
 Copying · 135
 Creating · 133
 deleting · 138
 Reading Variant Data · 137

W

Wildcard search · 117
Working with Criteria Profiles) · 112

X

XMI Monitor · 28
XML Description of a Criteria Profile or of a
 Criteria Hierarchy · 150