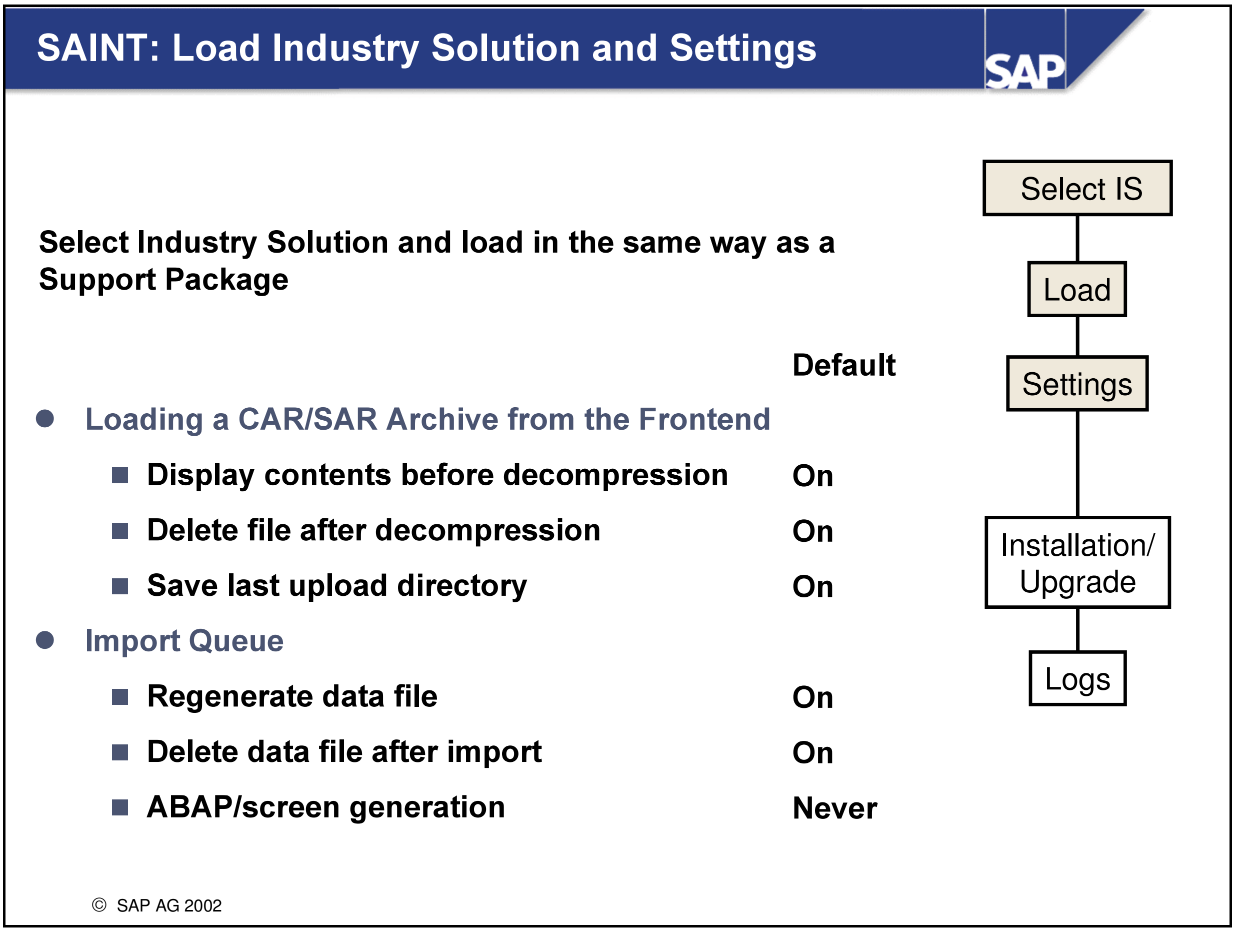
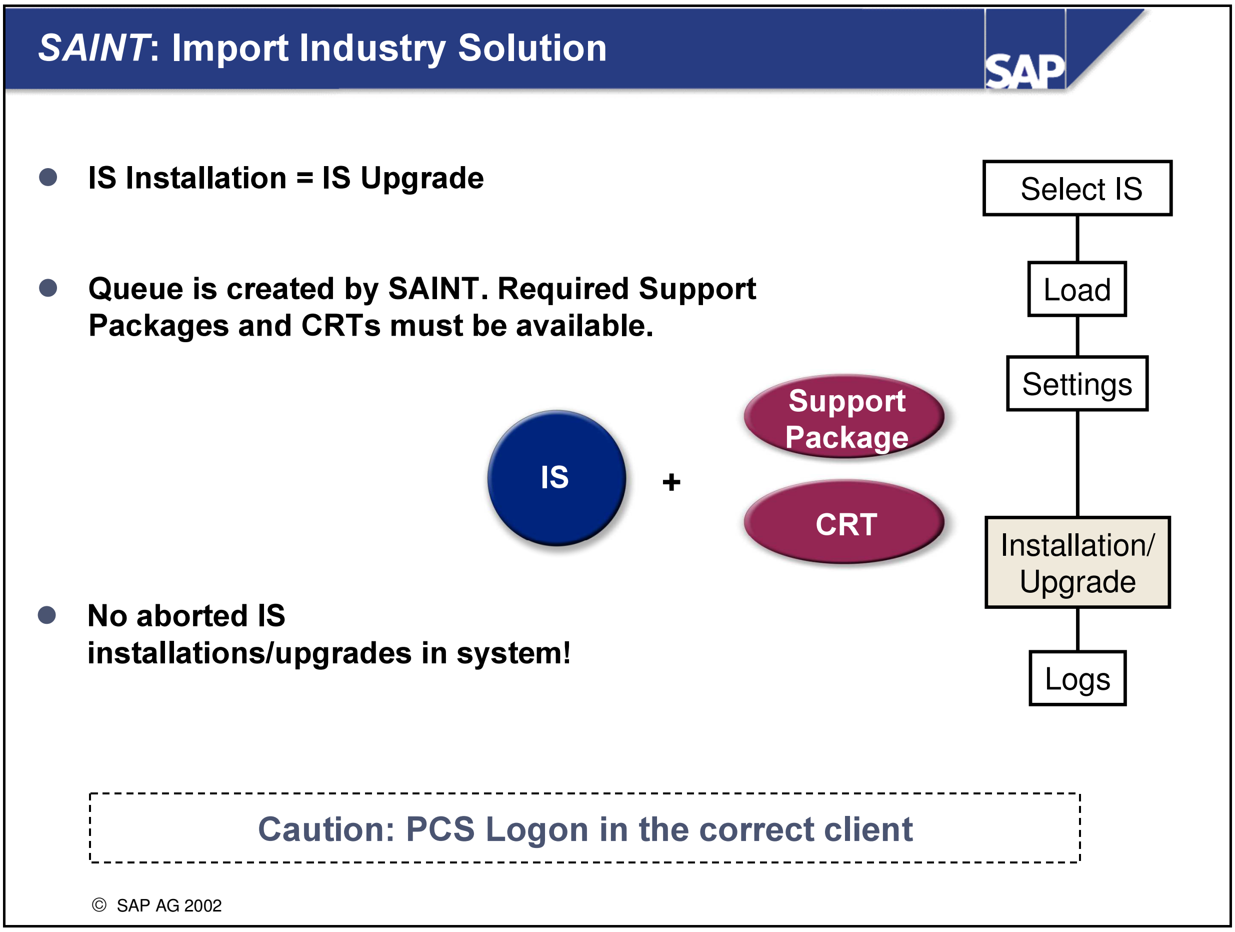


SAINT: Load Industry Solution and Settings

Select Industry Solution and load in the same way as a Support Package

	Default
● Loading a CAR/SAR Archive from the Frontend	
■ Display contents before decompression	On
■ Delete file after decompression	On
■ Save last upload directory	On
● Import Queue	
■ Regenerate data file	On
■ Delete data file after import	On
■ ABAP/screen generation	Never

- Download in the same way as: *SPAM*: Load Support Packages.
- Settings in the same way as: *SPAM*: Settings (as far as they exist for *SAINT*).



SAINT: Import Industry Solution

- **IS Installation = IS Upgrade**
- **Queue is created by SAINT. Required Support Packages and CRTs must be available.**
- **No aborted IS installations/upgrades in system!**

IS + Support Package / CRT

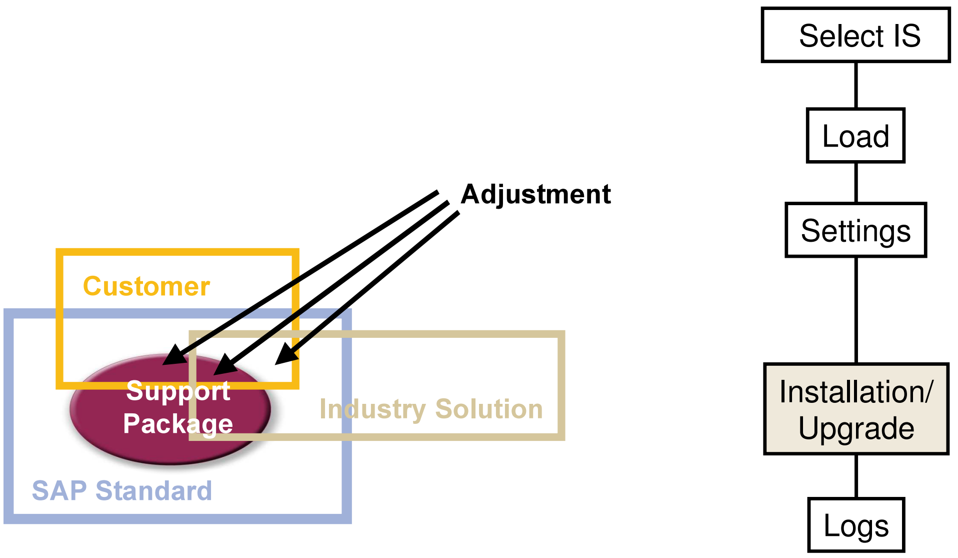
Select IS → Load → Settings → Installation/ Upgrade → Logs

Caution: PCS Logon in the correct client

© SAP AG 2002

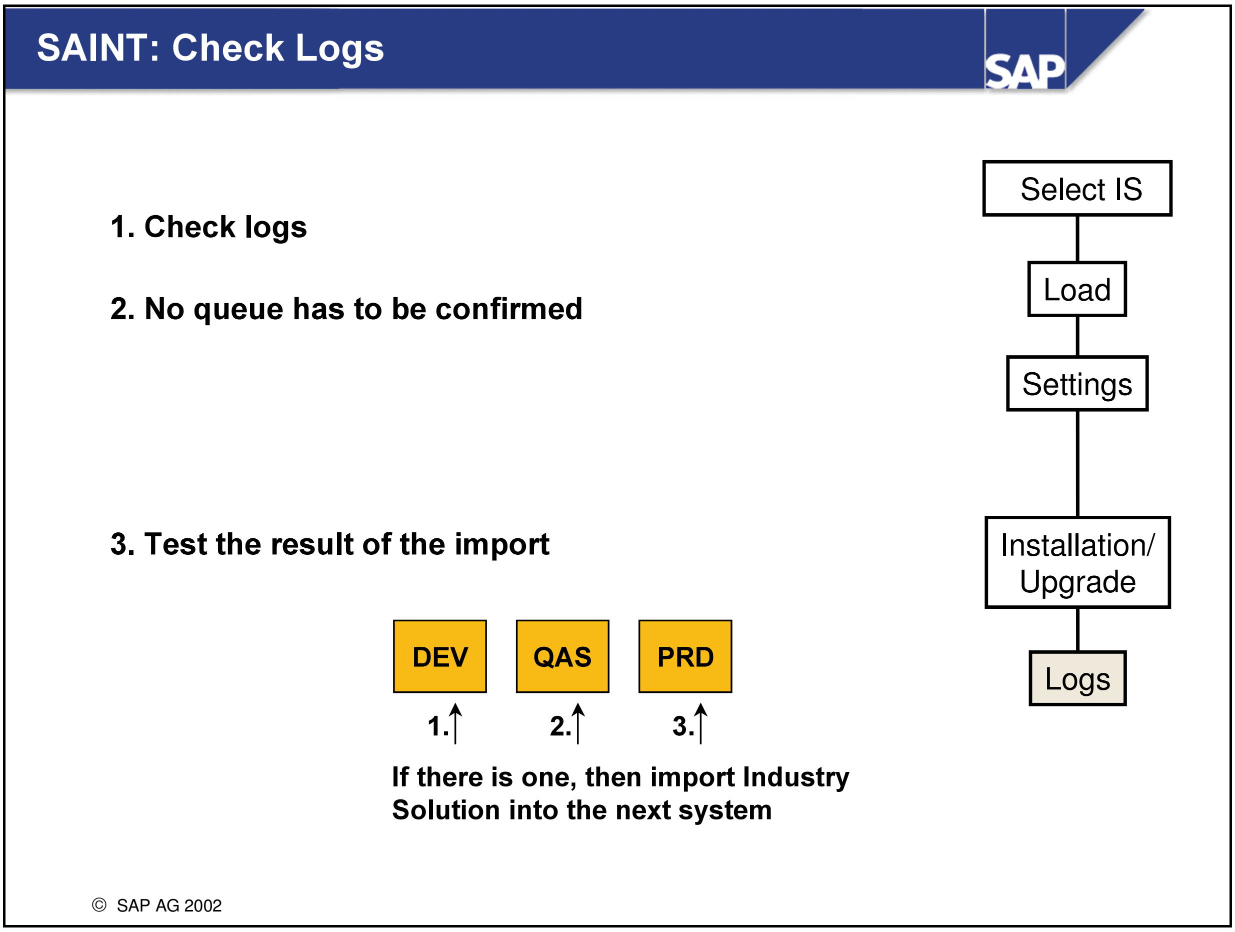
- In principle, transaction *SAINT* can process two different types of delivery packages: Add-On installations and Add-On upgrades. Technically, both of these packages are identical. They differ only in their import prerequisites and in the check for object conflicts with the SAP standard system.
- Prerequisites: You are logged on in client 000 and have loaded the relevant installation/upgrade packages into your system.
- SAP offers Preconfigured Systems (PCS) for certain industries. These systems contain the relevant industry knowledge for typical structures in a particular industry and the corresponding default values for these structures in the SAP system.
- A PCS consists of the following parts: an industry-specific ASAP system, an industry-specific SAP client, and industry-specific Customizing data.
- A PCS must be imported into the intended clients (NOT 000 and 066).
- A PCS also contains test data and industry-specific documentation. You can experiment with examples for various processes and scenarios with this test data. You can find additional information about PCSs in the SAP Service Marketplace, by choosing Quick Link */ASAP*.
- Note: You should only use a PCS as a basis for the configuration of a system installation. First import the PCS only into your test system so that you can check whether the settings meet your requirements.

SAINT: Perform Modification Adjustment



- You must only perform the modification adjustment for objects of the Add-On and its Support Packages. If you have modified SAP objects from the Add-On to be installed, you must adjust these objects during the import (ABAP Dictionary objects are adjusted with transaction *SPDD* and Repository objects with transaction *SPAU*).
- Procedure:

1. The import stops to allow modification adjustment. (*SAINT* will resume processing at step *RUN_SPAU* or *RUN_SPDD*.)
2. So that your developers can perform the modification adjustment, create a request in the Transport Organizer and tasks under this request for the developers.
3. Ask the developers to perform the modification adjustment for these objects.
   As long as you have not yet confirmed the installation, you can perform a context-sensitive switch into *SPDD* and *SPAU* from the initial screen of transaction *SAINT* by choosing *Extras* → *Modif. Adjustment*.
   After the adjustment is complete, the developers must release the tasks and inform you. The adjustment can be performed in any client.
4. Call *SAINT*. The system automatically displays the screen that was displayed before you interrupted the import.
5. Choose *Continue*. The system prompts you again to perform the modification adjustment. As it has already been completed, ignore the message and choose *Continue*. *SAINT* completes the processing and displays the status.



SAINT: Check Logs

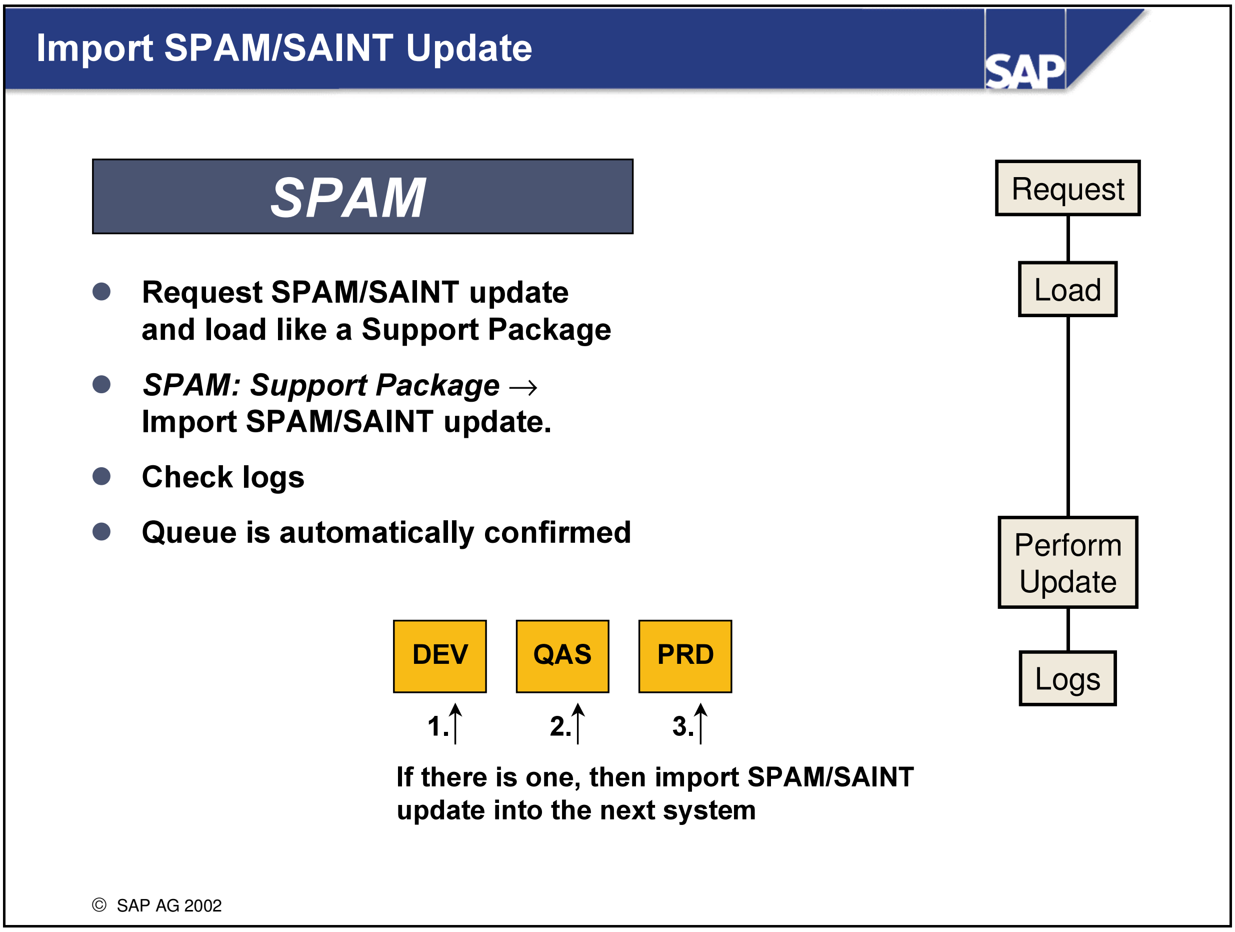
1. Check logs

2. No queue has to be confirmed

3. Test the result of the import

If there is one, then import Industry Solution into the next system

- The *Log* function displays logs for *SAINT* steps that use the transport control program tp.
- After the queue has been successfully imported, you should always check these logs.
- Caution: In the later phases of the import process, the database contents have already been changed. You should then continue the installation from here.
- PCS: You should only use a PCS as a basis for the configuration of a system installation. First import the PCS only into your test system so that you can check whether the settings meet your requirements.



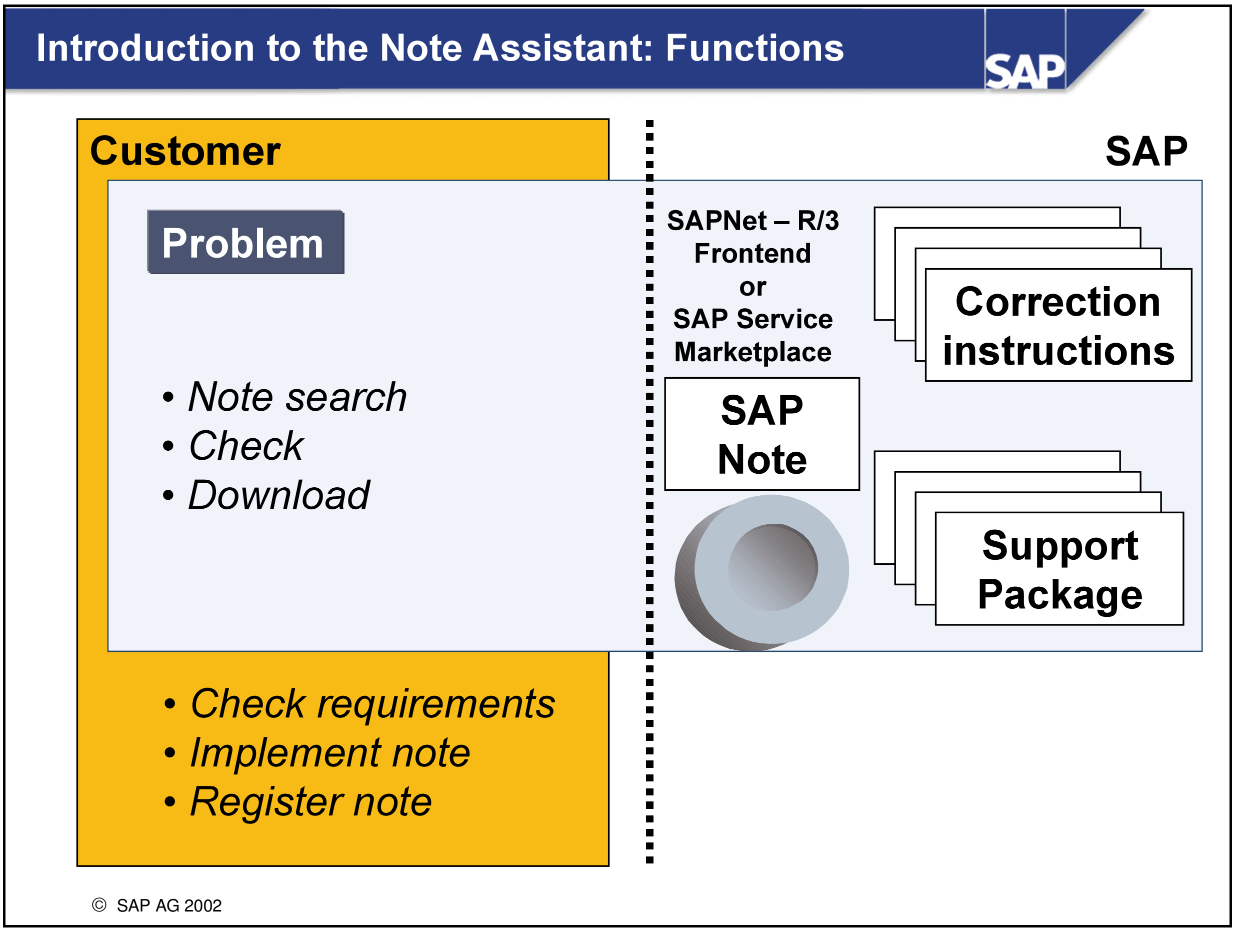
Import SPAM/SAINT Update

SPAM

- Request SPAM/SAINT update and load like a Support Package
- *SPAM: Support Package* → Import SPAM/SAINT update.
- Check logs
- Queue is automatically confirmed

If there is one, then import SPAM/SAINT update into the next system

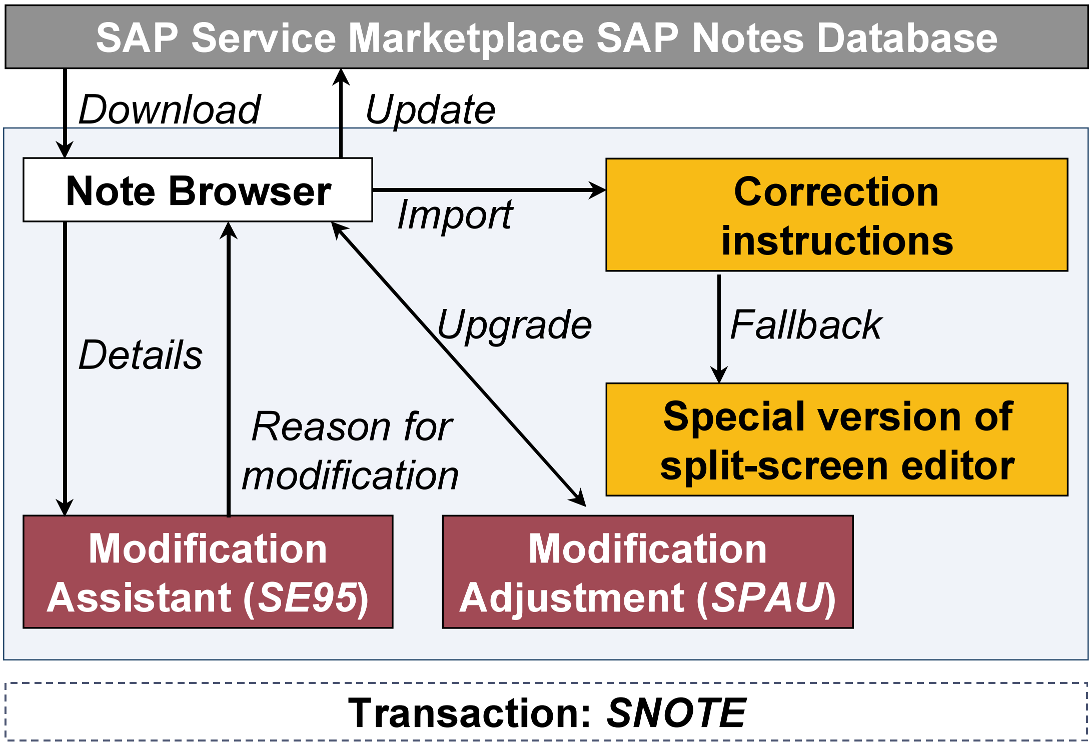
- You receive enhancements and improvements for *SPAM* and *SAINT* with a SPAM/SAINT Update. There is one update for each correction status that is updated in the course of time. You can find out the version from the short description (for example, *SPAM/SAINT update - version <REL>/0001).*
- We recommend that you always import the newest version of a SPAM/SAINT Update before you import other Support Packages.
- You can only import a SPAM/SAINT Update with the transaction *SPAM.*
- You can only successfully import a SPAM/SAINT Update if there are no aborted Support Packages in the system.
  If there are aborted Support Packages, a dialog box informs you of this. You then have two options:
  1. You first import the complete queue and then the SPAM/SAINT update.
  2. Reset the status of the queue, first import the SPAM/SAINT Update and then the queue. You can reset the status of the queue by choosing *Extras* → *Reset Status* → *Queue*.
- To import the newest SPAM/SAINT Update, choose *Support Package* → *Import SPAM/SAINT Update*. SPAM/SAINT updates are automatically confirmed after a successful import.



Introduction to the Note Assistant: Functions

- The Note Assistant allows you to automate, control, and document imports of SAP Notes using an SAP system.
- Not all SAP Notes are suitable for the Note Assistant. Check this for each case, especially for SAP Notes older than April 2001. The SAP Note must contain technical labels so that the Note Assistant can recognize the beginning and the end of the documentation and the ABAP coding in the SAP Note.
- The Note Assistant is also available for older SAP R/3 releases. For more information, see the SAP Service Marketplace, Quick Link */noteassistant.* As of SAP Web AS 6.10, it is contained in the standard SAP system.
- The transaction for the Note Assistant is *SNOTE*.

Introduction to the Note Assistant: Process



SAP Service Marketplace SAP Notes Database

Download

Update

Note Browser

Import

Correction instructions

Details

Upgrade

Fallback

Reason for modification

Special version of split-screen editor

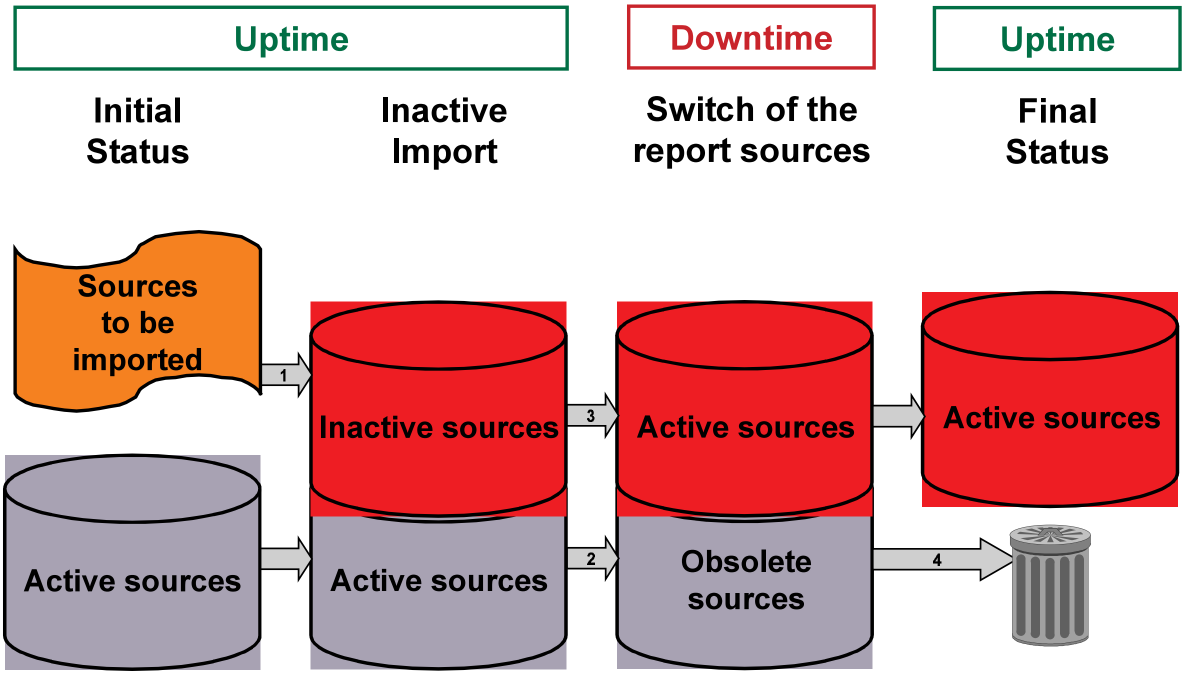
Modification Assistant (*SE95*)

Modification Adjustment (*SPAU*)

Transaction: *SNOTE*

- The following conditions apply in the system for the Note Assistant:
  - The SSCR request does not apply for corrections using the Note Assistant.
  - The corrections are subject to the normal change recording.
  - Changes are noted in the Modification Assistant (only what was changed, not how).
  - There are two new logical transport objects: R3TR NOTE: SAP Note, R3TR CINS: Correction instructions.
  - Changes are fully transportable, for example, to the quality assurance and production systems.
- Note: SAP Notes are not allowed to change DDIC objects. There is therefore no SPDD modification adjustment during an upgrade or when importing Support Packages.

Appendix: New for SAP Web AS 6.10: Inactive Import



- Due to the size and scope of current Support Packages, importing Support Packages requires a long system downtime. The system is not stopped and restarted during the import process, but it should also not be used productively during this time. This restriction is a disadvantage for many production systems.
  To reduce the downtime during the import of Support Packages, an import procedure has been developed that allows you to import a large proportion of the objects to be imported during production operation. These objects are report sources and report texts. The downtime can be significantly reduced if a Support Package contains a high proportion of report sources and report texts. For SAP Basis/SAP Web AS and SAP R/3 Support Packages, the proportion is around 70-80%. With the inactive import, the objects are imported into the database in an inactive state and are therefore invisible to the system. You can therefore continue to use the system productively.
- Procedure:

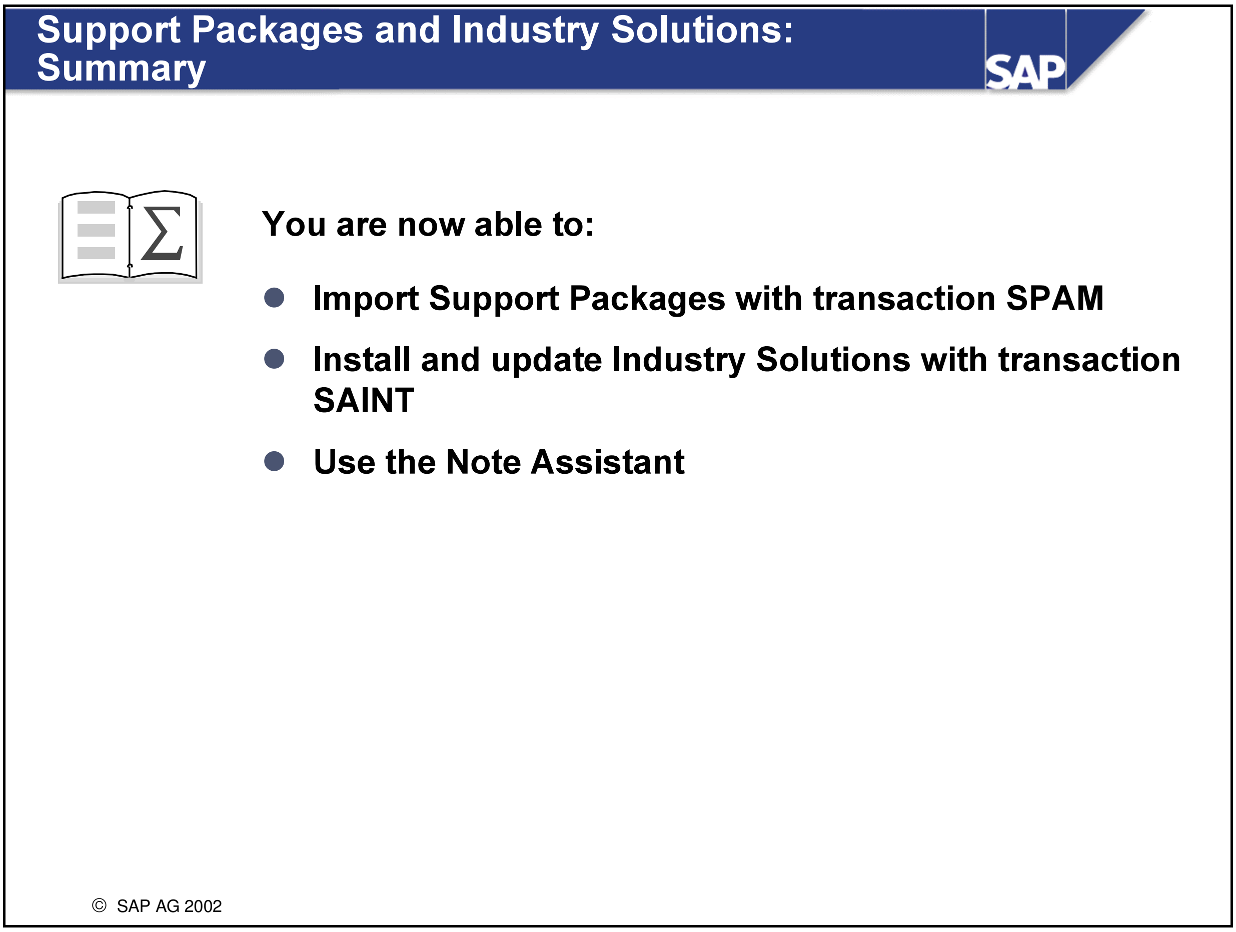
  1. Import the report sources and report texts in an inactive state. Old versions that are still active and new, inactive versions exist in parallel in the database.

  2. Active sources and texts that are to be replaced or deleted are set to obsolete.

  3. Inactive sources and texts are set to active.

  Actions 2 and 3 must be performed during downtime, but usually last less then 10 minutes.

  4. Deletion of sources and texts marked as obsolete from the database.
- Actions one and four, which require large data movements in the database, and are therefore runtime-intensive, can be performed during uptime.



Support Packages and Industry Solutions: Summary

You are now able to:

- Import Support Packages with transaction SPAM
- Install and update Industry Solutions with transaction SAINT
- Use the Note Assistant

Exercises

No.	Exercises
1	**Import a SPAM update** This exercise must be performed first by the *DEV* group and then by the *QAS* group. In the first step, a SPAM update is imported into the SAP system *DEV*. After this import has been successfully completed, the SPAM update is imported into the SAP system *QAS* in a second step. The files required for the SPAM update have already been requested and are on your host in the directory *G:\Additional_Files\ADM100_patches.*
1.1	Check whether a SPAM update has already been imported into your system. If yes: which version?
1.2	Unpack the file that contains the SPAM update at operating system level. Remember that this step must only be performed once for each transport directory.
1.3	Log on to the correct client.
1.4	Load the patch using transaction *SPAM*; that is, inform the SAP system about the file.
1.5	Import the SPAM update into your system. Note any possible errors.
1.6	Check the log for the import process for errors.
2	**Import two ABA Support Packages.** This exercise must be performed first by the *DEV* group and then by the *QAS* group. In the first step, the *ABA Support Packages 77* and *78* are imported in a single queue into the SAP system *DEV*. After this import is successfully completed, the *ABA Support Packages 77* and *78* are imported into the SAP system *QAS*. The files required for the ABA Support Packages have already been requested and are on your host in the directory *G:\Additional_Files\ADM100_patches.*
2.1	Check whether Support Packages have already been imported into your system. Are these confirmed?
2.2	Unpack the files that contain the *ABA Support Packages 77* and *78* at operating system level. Remember that this step must only be performed once for each transport directory. **Caution: On the training systems, these patches are called *SAPKIXT077* and *SAPKIXT078* instead of *SAPKA50A77* and *SAPKA50A78.* This is only the case on these training systems for technical reasons. The normal naming conventions apply in real systems.**
2.3	Log on to the correct client.
2.4	Load the Support Packages, that is, inform the SAP system about the files.

2.5	Check the settings for the import of the Support Packages in transaction *SPAM*.
2.6	Define the queue for importing the *ABA Support Packages 77* and *78* in transaction *SPAM*.
2.7	Import the queue for *ABA Support Packages 77* and *78* into your system. Note any possible errors. If necessary, perform a modification adjustment using transactions *SPDD* and *SPAU*.
2.8	Check the log for the import process for errors.
2.9	Confirm the imported Support Packages after successful import.

Solutions

No.	Solutions
1	**Import a SPAM update** **This exercise must be performed first by the *DEV* group and then by the *QAS* group.** **In the first step, a SPAM update is imported into the SAP system *DEV*. After this import has been successfully completed, the SPAM update is imported into the SAP system *QAS* in a second step.** **The files required for the SPAM update have already been requested and are on your host in the directory *\Additional_Files\ADM100_patches.***
1.1	**Check whether a SPAM update has already been imported into your system. If yes: which version?** Call transaction *SPAM*. The version of SPAM is displayed, such as: *Support Package Manager - Version 6.10/0000* To see whether SPAM updates have already been imported, choose *Imported Support Packages* → *Display*. The system displays an overview of the previously imported SPAM updates.
1.2	**Unpack the file that contains the SPAM update at operating system level.** **Remember that this step must only be performed once for each transport directory.** Go to operating system level. Copy the file *KD610<nn>.CAR* from the directory *G:\Additional_Files\ADM100_patches* to the directory *\usr\sap\trans*. *<nn>* stands for the newest version of the SPAM update. Open a command prompt at operating system level. Switch to the directory *\usr\sap\trans* and unpack the copied file with the command **`sapcar -xvf KD610<nn>.CAR`**. This unpacks the file to the directory *\usr\sap\trans\EPS\in*.
1.3	**Log on to the correct client.** Log on to client *000*. It is only possible to import the SPAM update from client *000*.
1.4	**Load the patch using transaction *SPAM*; that is, inform the SAP system about the file.** Transaction *SPAM: Support Package* → *Load Packages* → *From the application server* Confirm the dialog box to upload.
1.5	**Import the SPAM update into your system. Note any possible errors.** Transaction *SPAM: Support Package* → *Import SPAM Update* Confirm the dialog box to import. Then call transaction *SPAM* again. This takes some time, as the system must regenerate the underlying programs and so on due to the SPAM update.

1.6	**Check the log for the import process for errors.** Transaction *SPAM: Goto → Log → SPAM Update* There should not be any errors displayed.
2	**Import two ABA Support Packages.** **This exercise must be performed first by the *DEV* group and then by the *QAS* group.** **In the first step, the *ABA Support Packages 77* and *78* are imported in a single queue into the SAP system *DEV*. After this import is successfully completed, the *ABA Support Packages 77* and *78* are imported into the SAP system *QAS*.** **The files required for the ABA Support Packages have already been requested and are on your host in the directory *G:\Additional_Files\ADM100_patches.***
2.1	**Check whether Support Packages have already been imported into your system. Are these confirmed?** Transaction *SPAM: Imported Support Packages → Display*. Result: Support Packages have already been imported. They have also already been confirmed. You can tell this from the green traffic light in the bottom right-hand corner of the initial screen of transaction *SPAM*.
2.2	**Unpack the files that contain the *ABA Support Packages 77* and *78* at operating system level.** **Remember that this step must only be performed once for each transport directory.** **Caution: On the training systems, these patches are called *SAPKIXT077* and *SAPKIXT078* instead of *SAPKA50A77* and *SAPKA50A78.* This is only the case on these training systems for technical reasons. The normal naming conventions apply in real systems.** Go to operating system level. Copy the files *KIXT077.CAR* and *KIXT078.CAR* from the directory *G:\Additional_Files\ADM100_patches* to the directory *\usr\sap\trans*. Open a command prompt at operating system level. Switch to the directory *\usr\sap\trans* and unpack the copied file with the commands **`sapcar -xvf KIXT077.CAR`** and **`sapcar -xvf KIXT078.CAR`**. This unpacks the files to the directory *\usr\sap\trans\EPS\in*.
2.3	**Log on to the correct client.** Log on to client *000*. It is only possible to import Support Packages from client *000*.
2.4	**Load the Support Packages; that is, inform the SAP system about the files.** Transaction *SPAM: Support Package → Load Packages → From the application server* Confirm the dialog box to upload.

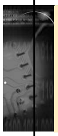
2.5	**Check the settings for the import of the Support Packages in transaction *SPAM*.** Transaction *SPAM: Extras → Settings* Use the default values for the settings.
2.6	**Define the queue for importing the *ABA Support Packages 77* and *78* in transaction *SPAM*.** Transaction *SPAM: Display/Define Queue* Select *SAP_ABA*. Select the queue to *ABA Support Package 78* (is already correctly selected) *Confirm queue* (*Enter*)
2.7	**Import the queue for *ABA Support Packages 77* and *78* into your system. Note any possible errors. If necessary, perform a modification adjustment using transactions *SPDD* and *SPAU*.** Transaction *SPAM: Support Package → Import Queue* Confirm the dialog box to import. Follow the individual steps of the import process, as far as possible, in the status line. If errors occur, check these by opening the error log. Repeat the import of the Support Package. The import process can be repeated. A modification adjustment is not required. Otherwise, the import process would stop and prompt you to perform a modification adjustment with transaction *SPDD* (ABAP Dictionary objects) or *SPAU* (other Repository objects). Confirm the dialog box informing you that the import was successful.
2.8	**Check the log for the import process for errors.** Transaction *SPAM: Goto → Log → Queue* No returncodes greater than eight should be displayed.
2.9	**Confirm the imported Support Packages after successful import.** Transaction *SPAM: Support Package → Confirm*

Scheduling Background Tasks

7. Basics of database administration

8. System landscapes and transport requests

9. Support Packages and Industry Solutions



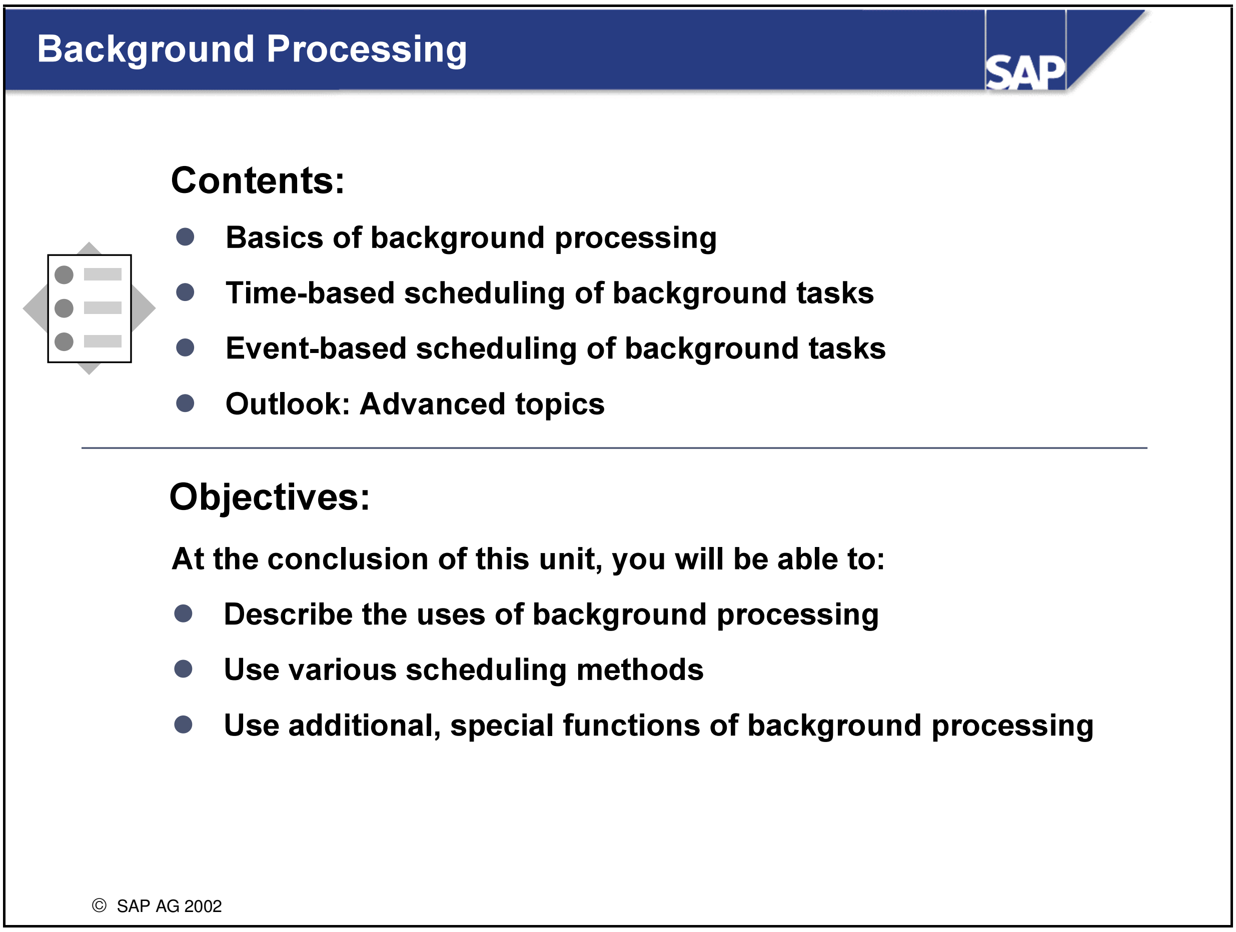
10. Scheduling background tasks

11. Printing from SAP systems

12. Basics of user administration

13. Remote connections

14. SAP Solution Manager



Background Processing

Contents:

- Basics of background processing
- Time-based scheduling of background tasks
- Event-based scheduling of background tasks
- Outlook: Advanced topics

Objectives:

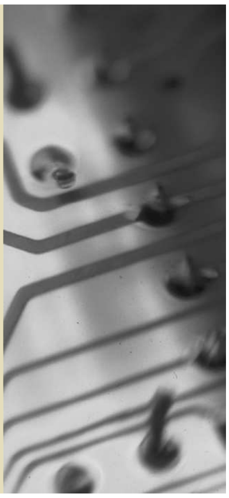
At the conclusion of this unit, you will be able to:

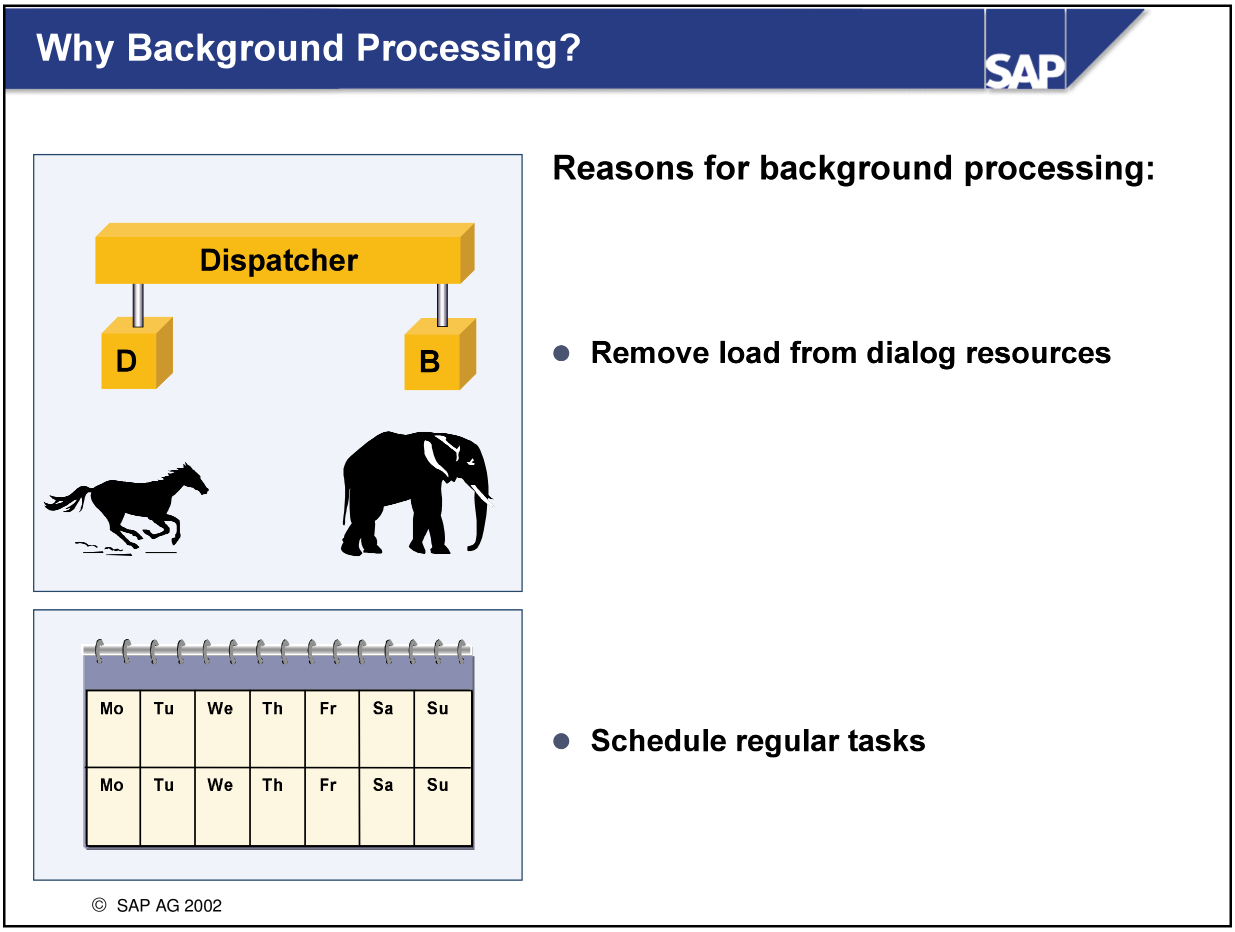
- Describe the uses of background processing
- Use various scheduling methods
- Use additional, special functions of background processing

Basics of Background Processing

Questions:

- Why do we need background processing?
- What is a background job?
- What can be performed in the background?
- What are the start criteria?
- How are jobs scheduled and monitored?
- What status can a job have?





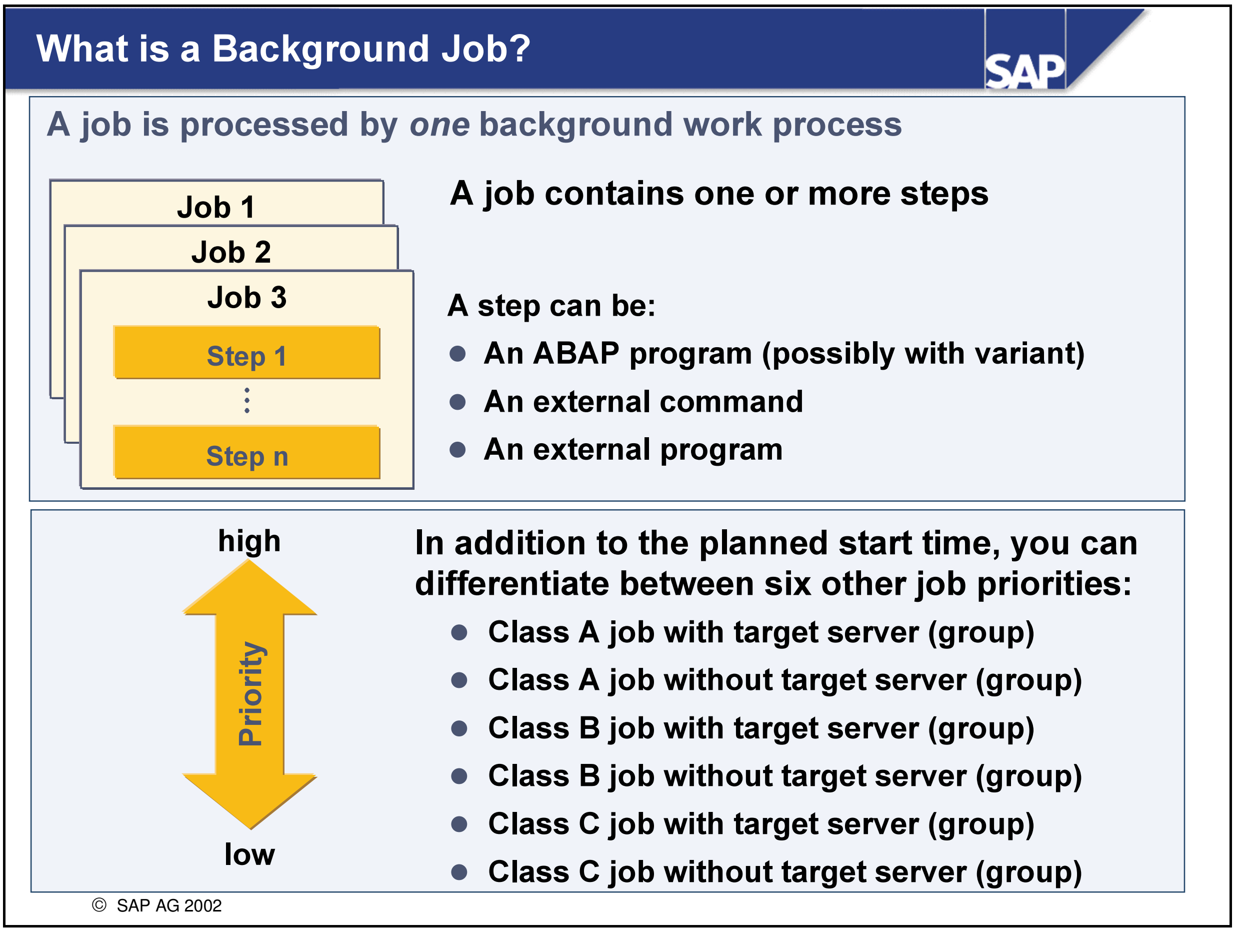
Why Background Processing?

Mo	Tu	We	Th	Fr	Sa	Su
Mo	Tu	We	Th	Fr	Sa	Su

Reasons for background processing:

- **Remove load from dialog resources**
- **Schedule regular tasks**

- Dialog work processes should be able to quickly respond to end users' queries. Dialog resources should therefore not be burdened with long-running programs. This can lead to bottlenecks in the dialog response time. The system profile parameter **rdisp/max_wprun_time** exists for this reason. It limits the maximum runtime of a process step within a dialog process. This should ensure that dialog processes are not blocked by long-running programs, interfering with online operation. After the maximum runtime has elapsed, the program is terminated.
- You can use the background work processes for long-running tasks. These are sometimes called batch work processes.
- Normally, background processing is used not only for long-running tasks, but also for recurrent tasks. Examples of this type of task are the daily database backup or the month end work for financial accounting.
- For more information about parameter **rdisp/max_wprun_time**, see SAP Note 25528.



What is a Background Job?

A job is processed by *one* background work process

Job 1

Job 2

Job 3

Step 1

⋮

Step n

A job contains one or more steps

A step can be:

- An ABAP program (possibly with variant)
- An external command
- An external program

high

Priority

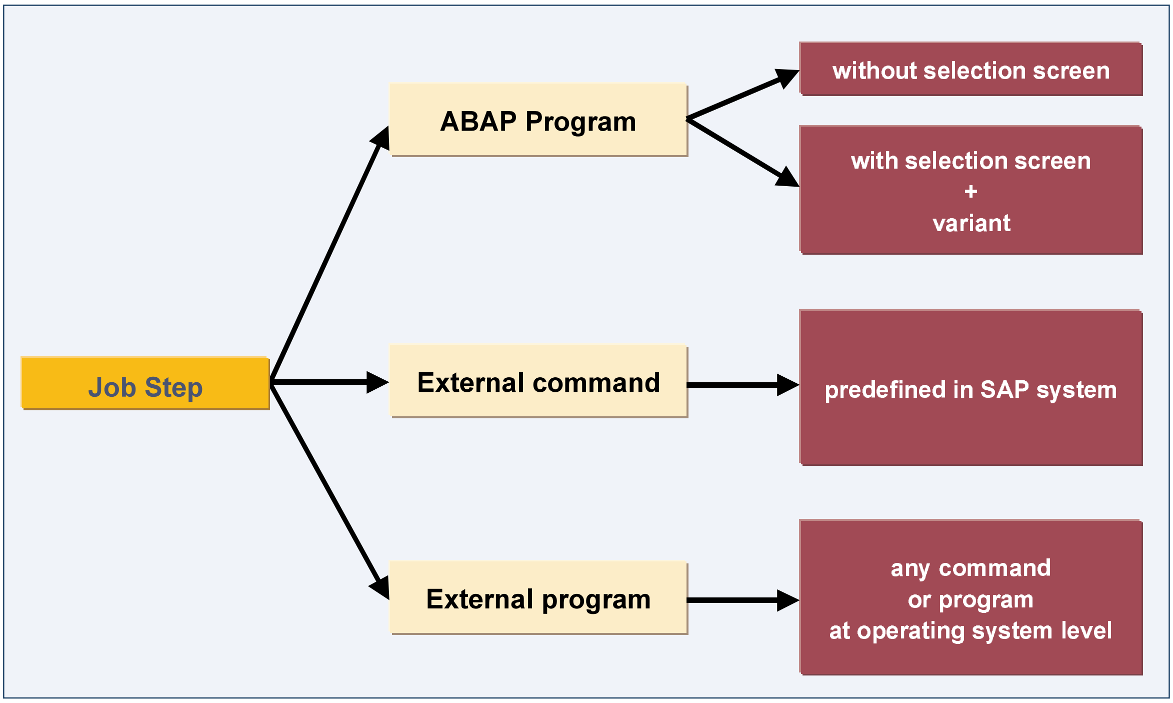
low

In addition to the planned start time, you can differentiate between six other job priorities:

- Class A job with target server (group)
- Class A job without target server (group)
- Class B job with target server (group)
- Class B job without target server (group)
- Class C job with target server (group)
- Class C job without target server (group)

- A background job consists of one or more job steps. A step can be:
  - An ABAP program
  - An external command
  - An external program
- Every job is processed without interruption by one single background work process.
- Background jobs can be scheduled with different priorities:
  - Class A (highest priority)
  - Class B (medium priority)
  - Class C (normal priority)
- If a job should be executed on a particular server or server group, it is handled preferentially compared to other jobs with the same class. This preference only applies if multiple jobs of various priorities request background processing at the same time (for example, because they are scheduled for the same time).
  You should ensure that a large share of all background tasks is normally scheduled as class C **without** target server specification. This share can include 90% (or more) of all tasks. System background tasks are also executed in this way, such as the tasks that you schedule with transaction *DB13*.

What Can Be Performed in the Background?



Job Step

ABAP Program

without selection screen

with selection screen + variant

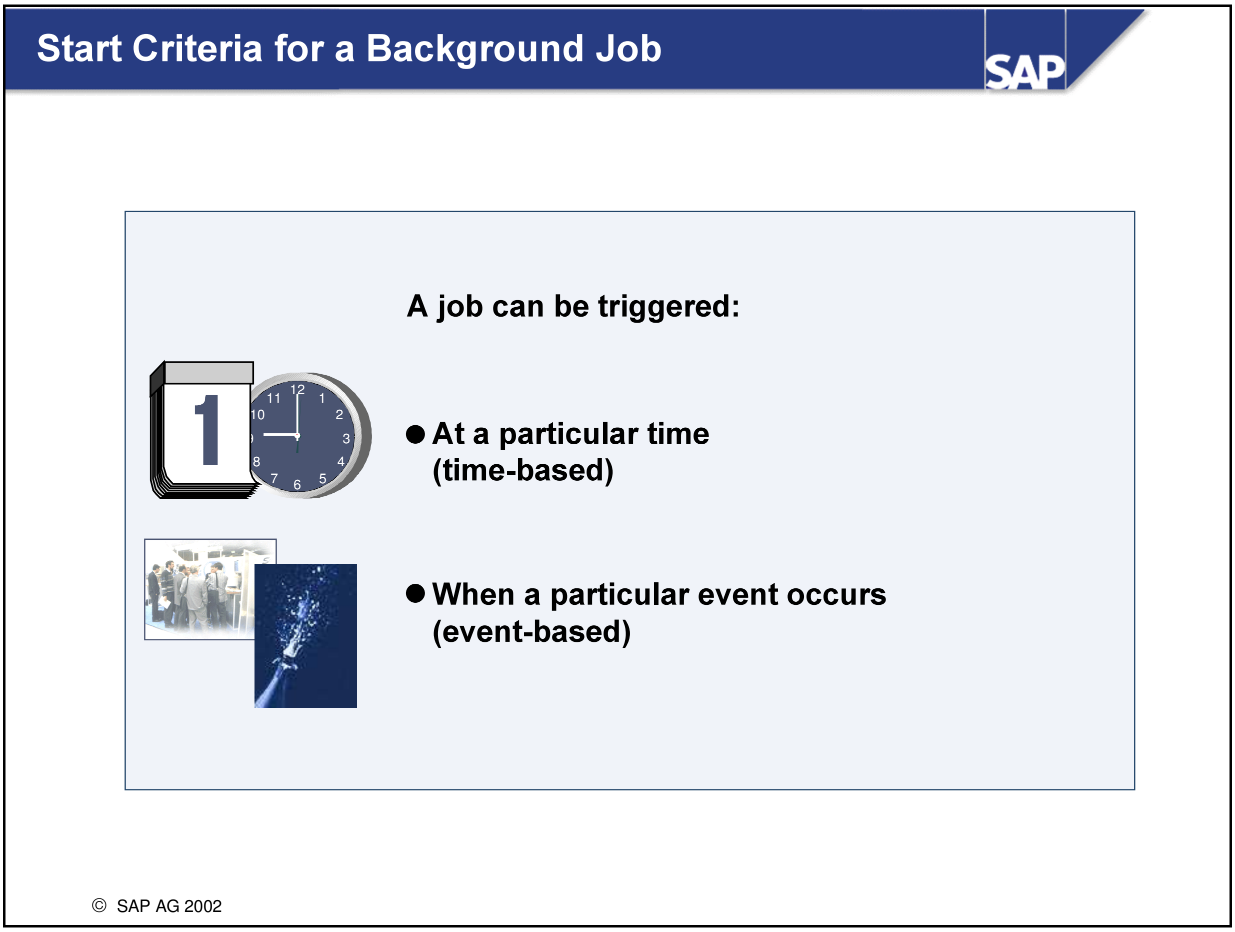
External command

predefined in SAP system

External program

any command or program at operating system level

- A step within a job can call one of the following three actions:
  - Every **ABAP program** can be scheduled as a step of a job. If the ABAP program has one or more selection screens, you must create the input required there in advance in the form of a variant. A variant allows an ABAP program to run in the background, although the program requires input. The values stored in the variant are then used during the execution of the program. If an ABAP program has a screen output as its result, this is directed to a spool list. You can specify a (mail) recipient for this spool list during the definition of the job. This recipient then receives the output of the job after the job has been run. You must also specify a printer for the creation of a spool list, although no direct output to a printer results from background processing. This must be explicitly started later.
  - An **external command** is a call of a predefined script, a command, or a program outside the SAP system. You can mask operating system calls and store them under a simple name with external commands. The execution of an external command can be protected using the SAP authorization concept, so that particular users can only access certain external commands.
  - An **external program** is any operating system command. The SAP authorization concept only specifies whether a user can call external programs or not. Use external commands a more detailed authorization assignment, for example at the level of program names.



Start Criteria for a Background Job

A job can be triggered:

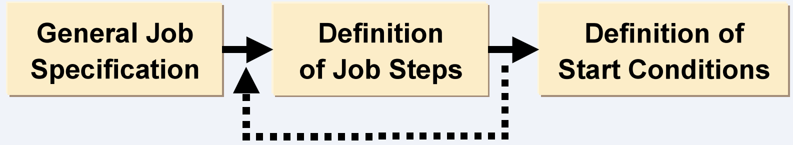
- At a particular time (time-based)
- When a particular event occurs (event-based)

© SAP AG 2002

- A job can be triggered by:
  - Scheduling it on a particular date at a particular time (this includes the start time *immediately*, if there are no free background work processes available when the job is scheduled).
  - The occurrence of a particular event defined in the SAP system (this includes jobs that are to start after other jobs or at operation mode changes, or jobs with *immediate start* if there are free background work processes available when the job is scheduled).

Job Scheduling

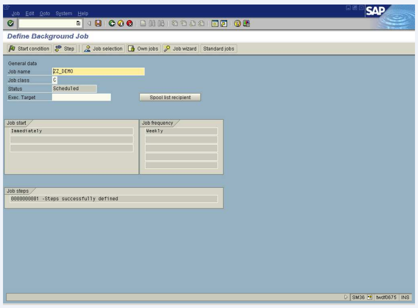
Procedure



General Job Specification → Definition of Job Steps → Definition of Start Conditions

Scheduling Options

Transaction *SM36*



Define Background Job

General data

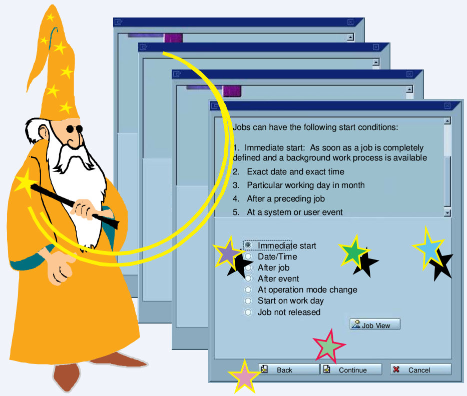
Job name	ZZ_DEMO
Job class	C
Status	Scheduled
Exec. Target	

Job start: Immediately

Job frequency: Weekly

Job steps: 0000000001 -Steps successfully defined

Transaction *SM36WIZ* (Job Wizard)



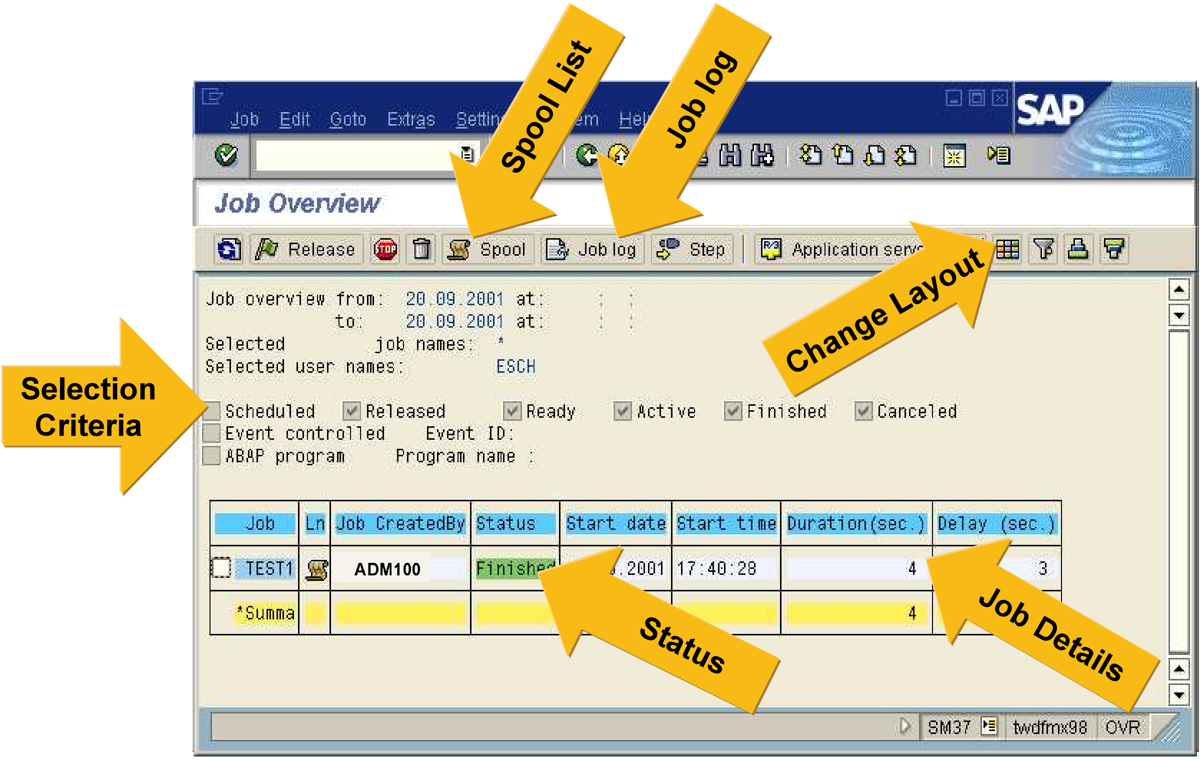
Jobs can have the following start conditions:

1. Immediate start: As soon as a job is completely defined and a background work process is available
2. Exact date and exact time
3. Particular working day in month
4. After a preceding job
5. At a system or user event

● Immediate start
○ Date/Time
○ After job
○ After event
○ At operation mode change
○ Start on work day
○ Job not released

- Use transaction *SM36* to define new jobs. You can also call the Job Wizard (transaction *SM36WIZ*) from here.
- Required specifications when defining a job are:
  - General specifications such as job name, job priority (default *C*) and (optionally) target server (group)
  - Definition of one or more job steps
  - Definition of start conditions (time- or event-based)
- When defining jobs, the Job Wizard supports you by leading you through the individual steps listed above.
- The Job Wizard (unlike classical scheduling) cannot perform individual steps with different users.

Job Monitoring



Job Overview

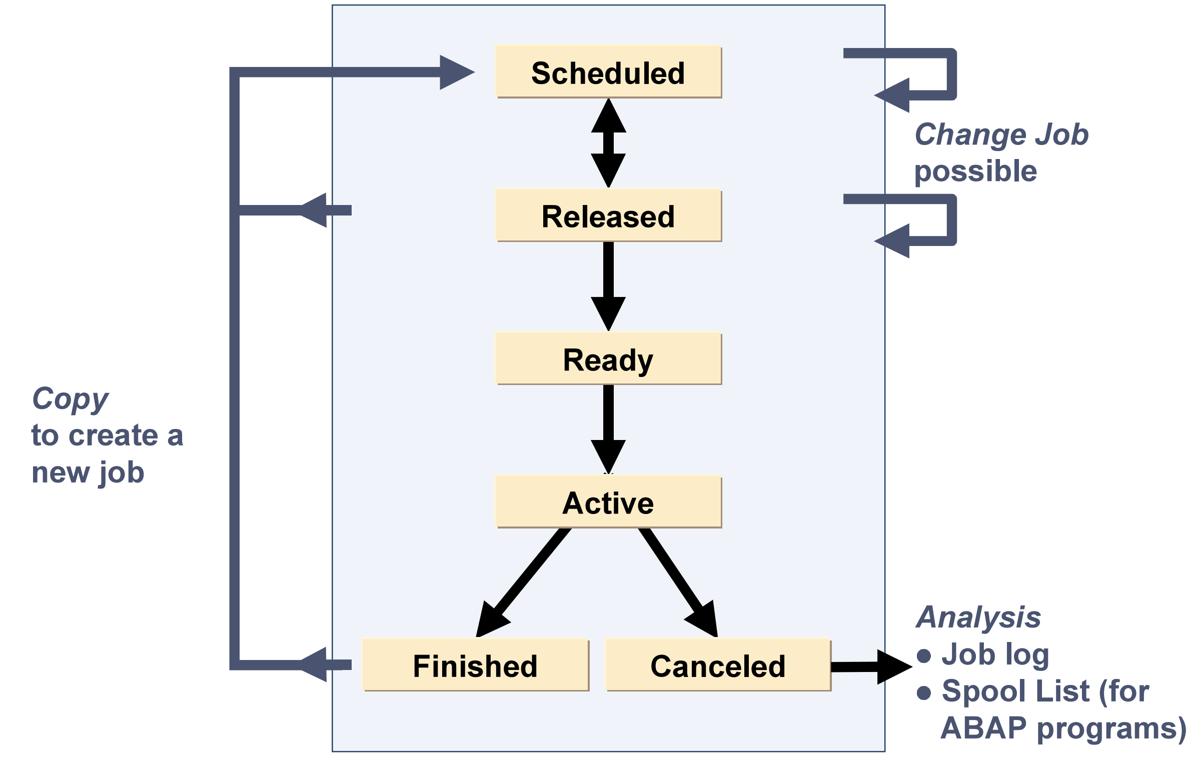
```
Job overview from:  20.09.2001 at:     :  :
             to:    20.09.2001 at:     :  :
Selected         job names:  *
Selected user names:         ESCH

☐ Scheduled   ☑ Released   ☑ Ready   ☑ Active   ☑ Finished   ☑ Canceled
☐ Event controlled    Event ID:
☐ ABAP program     Program name :
```

Job	Ln	Job CreatedBy	Status	Start date	Start time	Duration(sec.)	Delay (sec.)
TEST1		ADM100	Finished	.2001	17:40:28	4	3
*Summa						4	

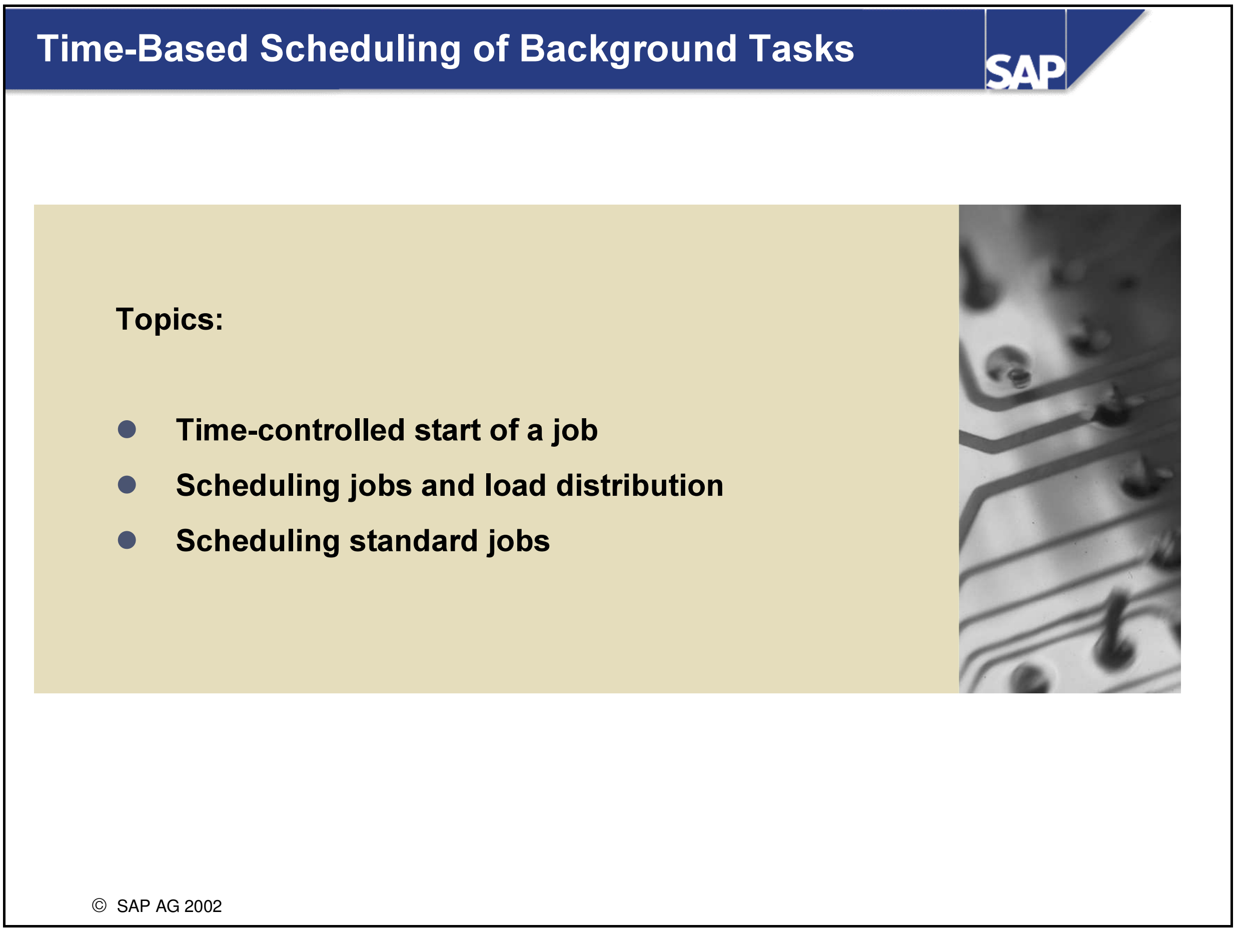
- Use transaction *SM37* to monitor jobs.
- You can select jobs using various criteria on the initial screen of this transaction. In this way, you can display all jobs that contain a particular job step, that have a particular status, or that react to a defined event (for this, you must select a particular event).
- You can navigate to other job-specific views from the job overview shown above:
  - The spool list contains the output lists for ABAP programs (if they exist).
  - The job details contain, among other information, the job definition, duration of the processing of the job, and the start time of the job. You can adjust the layout of the displayed list to your requirements by adding or removing additional fields.
  - All message types output by a background program are stored in the job log. You can display this log to obtain information about a program that terminated early or to perform a detailed investigation about a background processing run.

Status of a Job



© SAP AG 2002

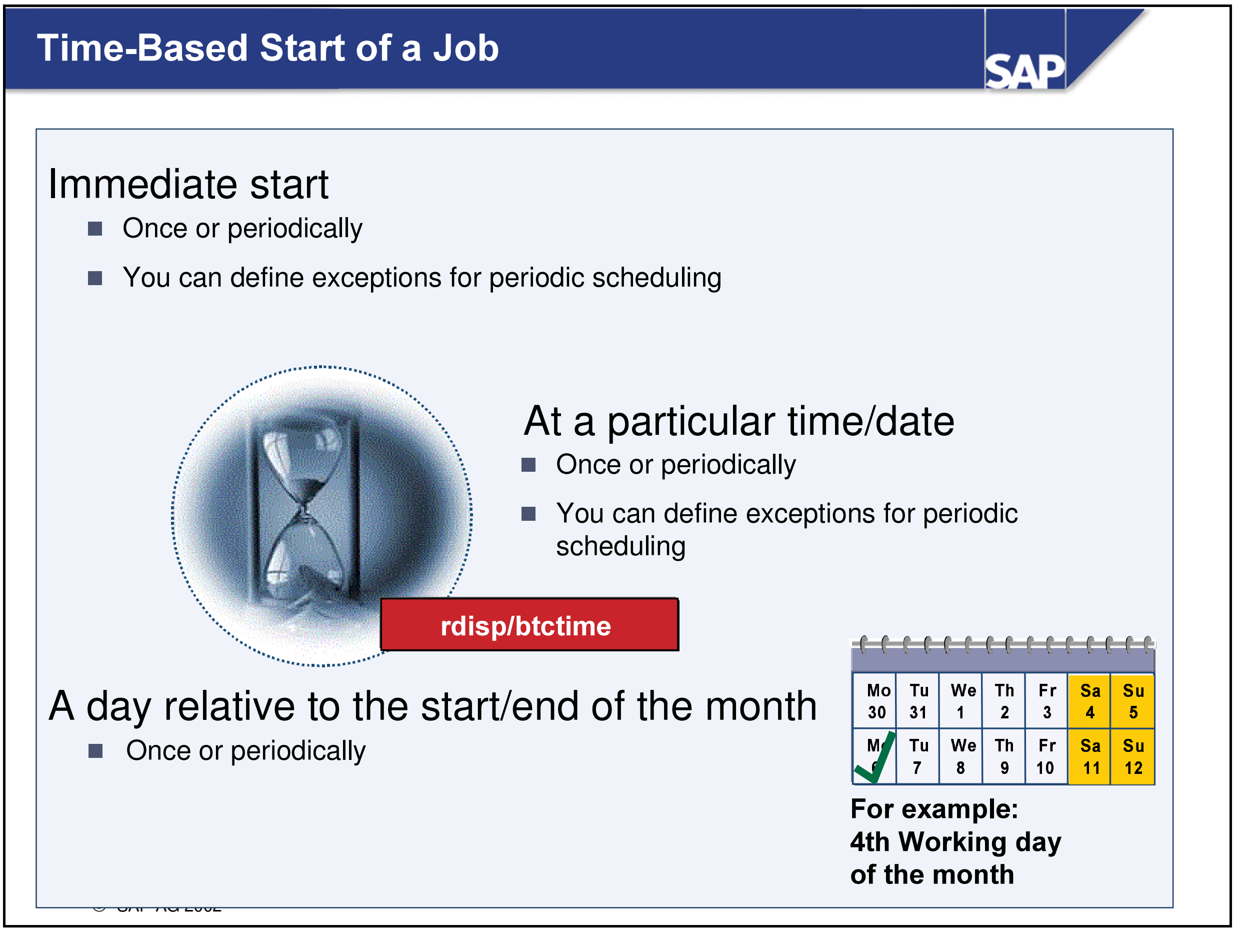
- A job can have the following statuses:
  - *Scheduled*: The steps of which the job consists have already been defined, however the start condition must still be defined.
  - *Released*: The job has been completed defined including its start condition. A job cannot be released without a start condition. Only an administrator or a user with the relevant authorizations for background processing can release a job. This ensures that unauthorized users cannot execute jobs without approval.
  - *Ready*: The start condition of a released job has been fulfilled. A job scheduler has placed the job in the wait queue for a free background work process.
  - *Active*: The job is currently being executed and cannot be deleted or changed.
  - *Finished*: All steps of this job were successfully completed.
  - *Canceled*: The job terminated. This can happen for two reasons:
    - An administrator intentionally terminates the job by choosing *Job* → *Cancel active job*.
    - A job step contains a program that generates an error.
- You can still change a job as long as the job still has the status *Scheduled* or *Released*.
- If the execution of the job has already begun, you can monitor the processing in the job log. If the job contains ABAP programs that create output lists, these are stored in spool lists.
- You can copy a new job from an existing job by choosing *Job* → *Copy*.



Time-Based Scheduling of Background Tasks

Topics:

- Time-controlled start of a job
- Scheduling jobs and load distribution
- Scheduling standard jobs



Time-Based Start of a Job

Immediate start

- Once or periodically
- You can define exceptions for periodic scheduling

At a particular time/date

- Once or periodically
- You can define exceptions for periodic scheduling

rdisp/btctime

A day relative to the start/end of the month

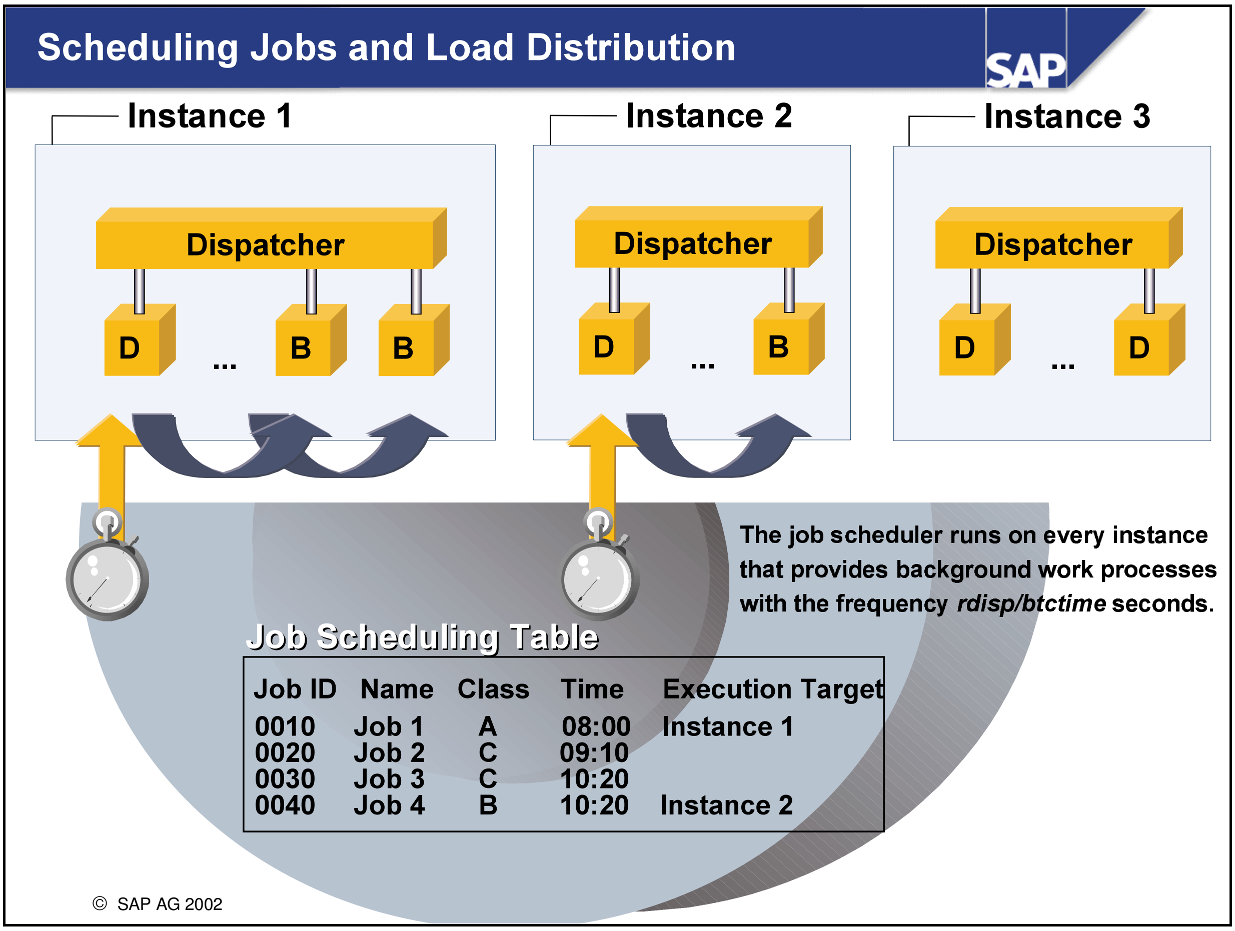
- Once or periodically

Mo 30	Tu 31	We 1	Th 2	Fr 3	Sa 4	Su 5
Mo 6 ✓	Tu 7	We 8	Th 9	Fr 10	Sa 11	Su 12

For example:
4th Working day
of the month

- A job can be started in a time- or event-based manner.
- In the case of a time-based start of a job, you can choose between the following options:
  - The job should be executed immediately.
  - The job should be executed at a particular date/time.
  - The job should be executed on a particular work day.
- You can choose to make all time-based start conditions for jobs recurrent. This means that the job is executed again after a defined period of time has elapsed. You can specify exceptions (such as postponement to the following workday in the case of holidays in the factory calendar).
- The job is started at the specified date and time, in accordance with the job priority and availability of background work processes.
- You can also specify a time period during which the job should be executed. Specify a start time after which the job should no longer be executed. With this function, you prevent periodic jobs being executed at an undesirable time, among other things. For example, a reorganization job that should only be executed at night is delayed. With a start time window, you can avoid this job being executed during the day, when the dialog users are active and there are fewer system resources available.
- The profile parameter *rdisp/btctime* specifies the time period in which the Job Scheduler is active (see next slide). The execution of jobs with the start condition *immediately* avoids the Job Scheduler. In this case, the dialog work process of the scheduling user performs the job scheduling. If no free resources are found, the job concerned is scheduled in a time-based way. The start time then corresponds to the time point at which it should have started.

Scheduling Jobs and Load Distribution



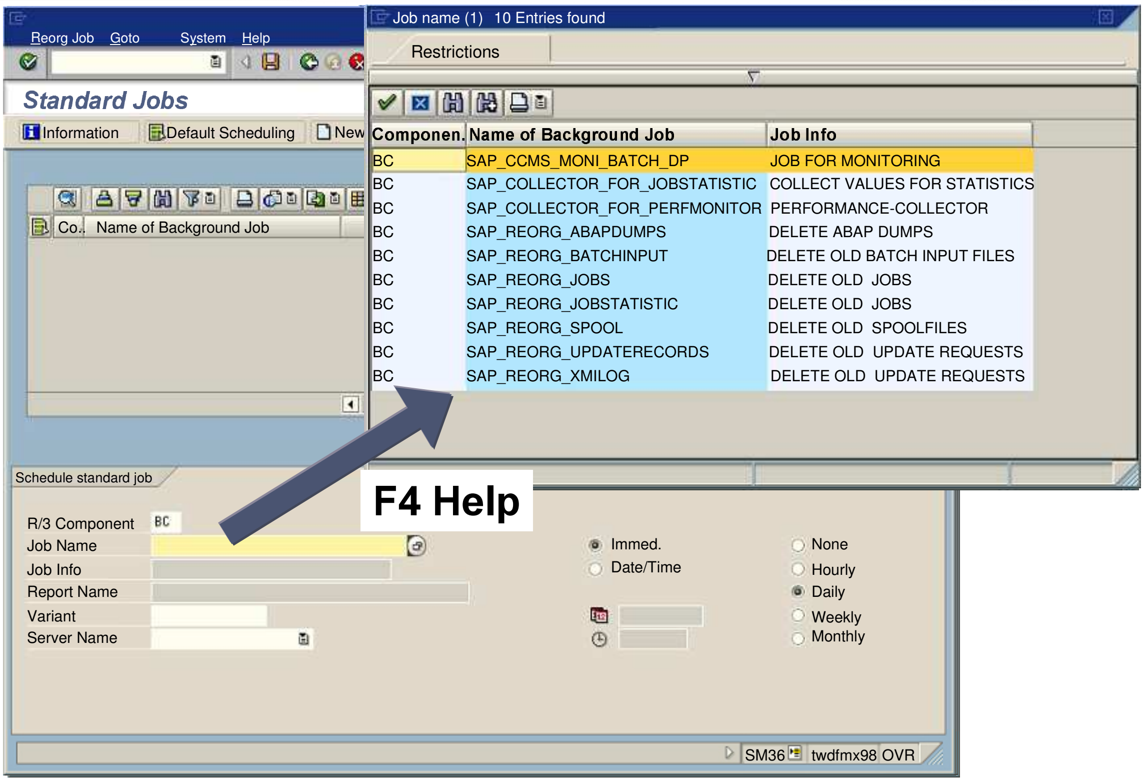
The job scheduler runs on every instance that provides background work processes with the frequency *rdisp/btctime* seconds.

Job Scheduling Table

Job ID	Name	Class	Time	Execution Target
0010	Job 1	A	08:00	Instance 1
0020	Job 2	C	09:10	
0030	Job 3	C	10:20	
0040	Job 4	B	10:20	Instance 2

- Background work processes can be configured on every instance of the SAP system using the profile parameter *rdisp/wp_no_btc*.
- The number of background work processes required in the SAP system depends on the number of tasks to be performed in the background. If the transport system is used, there must be at least two background work processes in the system.
- The combination of job ID and job name uniquely define the job in the system.
- On every SAP instance on which background work processes are defined, the Job Scheduler runs every *rdisp/btctime* seconds (default value 60 seconds). This is an ABAP program (SAPMSSY2 - an Auto ABAP) that automatically runs in a dialog work process.
- The Job Scheduler checks the job scheduling table in the database for jobs that are waiting for processing. These jobs are transferred to free background work processes in the SAP instance, in accordance with their priority and execution target.
  - Jobs that are not assigned any particular execution target can be executed with any free background work process. This means that the workload is automatically distributed between the SAP instances.
  - If a job is explicitly assigned an execution target (such as a selected instance or a group of instances), the special properties of the execution target can be used. For example, you can ensure that a job is performed on a particular operating system, or that the job is executed by a background work process that is running on the same host as the database. You do not have the advantages of automatic load distribution.

Scheduling Standard Jobs



Componen.	Name of Background Job	Job Info
BC	SAP_CCMS_MONI_BATCH_DP	JOB FOR MONITORING
BC	SAP_COLLECTOR_FOR_JOBSTATISTIC	COLLECT VALUES FOR STATISTICS
BC	SAP_COLLECTOR_FOR_PERFMONITOR	PERFORMANCE-COLLECTOR
BC	SAP_REORG_ABAPDUMPS	DELETE ABAP DUMPS
BC	SAP_REORG_BATCHINPUT	DELETE OLD BATCH INPUT FILES
BC	SAP_REORG_JOBS	DELETE OLD JOBS
BC	SAP_REORG_JOBSTATISTIC	DELETE OLD JOBS
BC	SAP_REORG_SPOOL	DELETE OLD SPOOLFILES
BC	SAP_REORG_UPDATERECORDS	DELETE OLD UPDATE REQUESTS
BC	SAP_REORG_XMILOG	DELETE OLD UPDATE REQUESTS

R/3 Component	BC
Job Name	
Job Info	
Report Name	
Variant	
Server Name	

Immed. ○ / Date/Time ○; Period: ● Daily

© SAP AG 2002

- Standard jobs are background jobs that should run regularly in a production SAP system. These jobs mainly perform certain clean up jobs in the system, such as the deletion of obsolete spool requests.

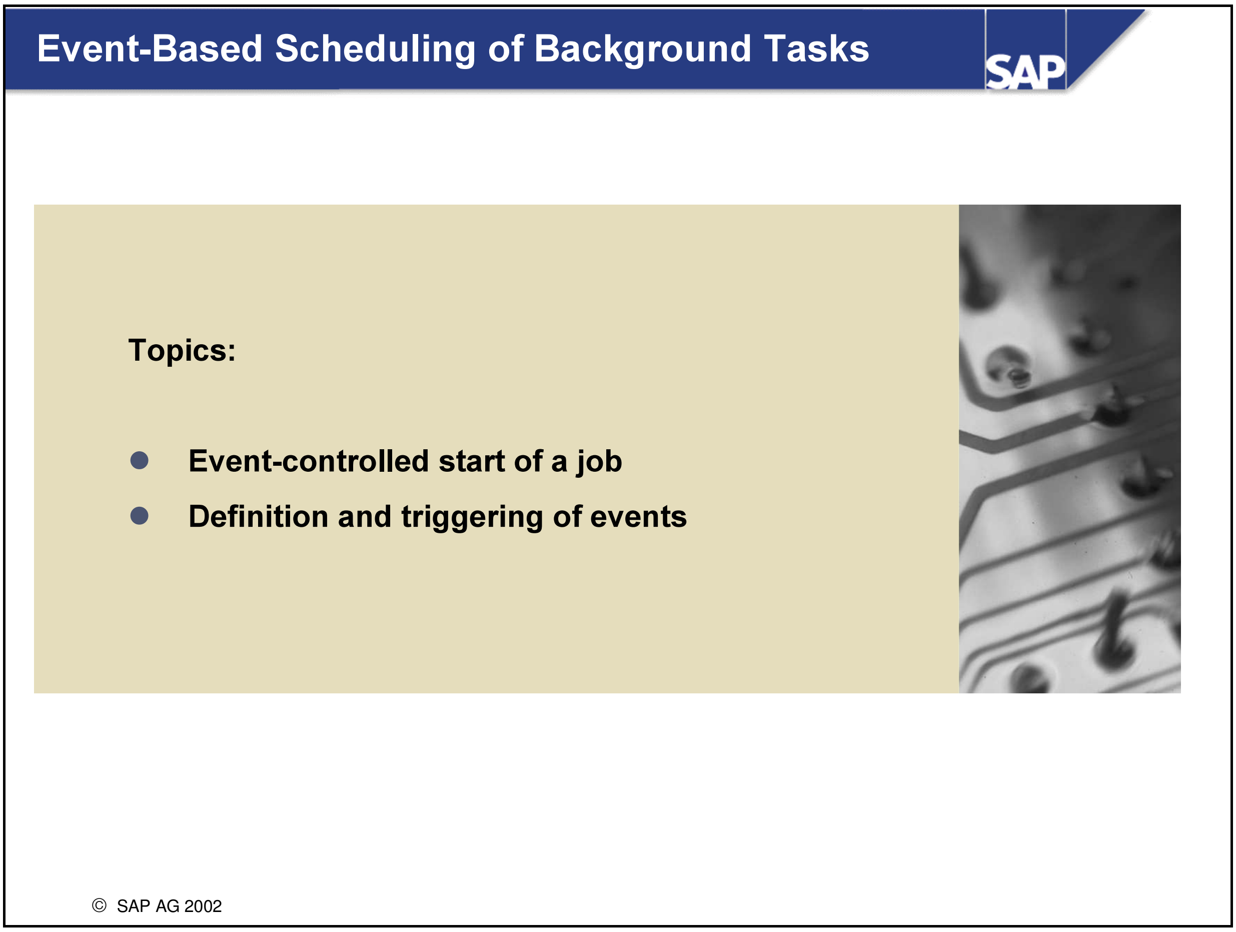
- In the job definition transaction (*SM36*), you can access a selection of important standard jobs that you can schedule, monitor, and edit by choosing *Standard Jobs* (see graphic).

  - If you want to schedule all standard jobs, choose *Default Scheduling*. All standard jobs that are defined in table REORGJOBS are scheduled with the specified variant and period. The system informs you if other jobs exist that execute one of the programs of the standard jobs.

  - To schedule individual jobs, choose the selected job with the input help and specify an execution period.

  - To define an additional standard job that is not yet available in the selection (table REORGJOBS), choose *Predefine new job*.

- For more detailed information about standard jobs, see SAP Note 16083.



Event-Based Scheduling of Background Tasks

Topics:

- Event-controlled start of a job
- Definition and triggering of events

© SAP AG 2002

Event-Based Start of a Job

- After an event
  - Once or periodically
  - With or without parameters

- After a (predecessor) job
  - Optional: Status-dependent start

- At operation mode change

Events are received here: **rdisp/btcname**

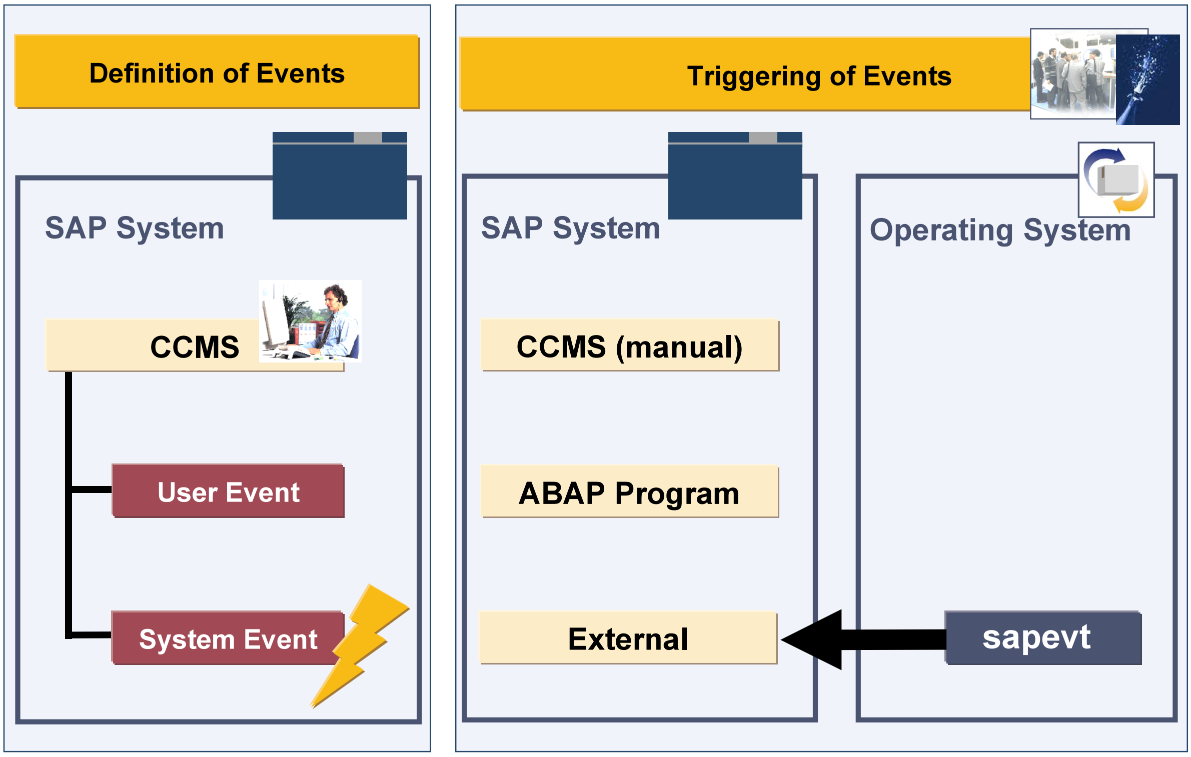
- An event is a signal to the background processing system that a particular status has been achieved in the system. The background processing system receives events and then starts all jobs that are linked to this event.

- An application server is specified for the processing of events that are triggered within the SAP system in profile parameter **rdisp/btcname**. An event-dependent Job Scheduler is started on this server (unlike the time-controlled Job Scheduler that runs periodically on all background servers). This scheduler checks whether a job is waiting for the event that has been received. It is therefore important that the parameter **rdisp/btcname** contains the name of an active background server.

- Event-based jobs can be scheduled with one of the following three start conditions:
  - *After Event*: The job starts after a defined event is received by the SAP system.
  - *At Operation Mode*: With this option, you can link a job to the activation of an operation mode, when scheduling the job.
  - *After Job*: You can create simple job chains in which the execution of the successor job can be made dependent on the status of the predecessor job.

- To be able to react to external events, the SAP system must be active. Otherwise, an event triggered by an external program is lost.

Definition and Triggering of Events



Definition of Events

SAP System

CCMS

User Event

System Event

Triggering of Events

SAP System

CCMS (manual)

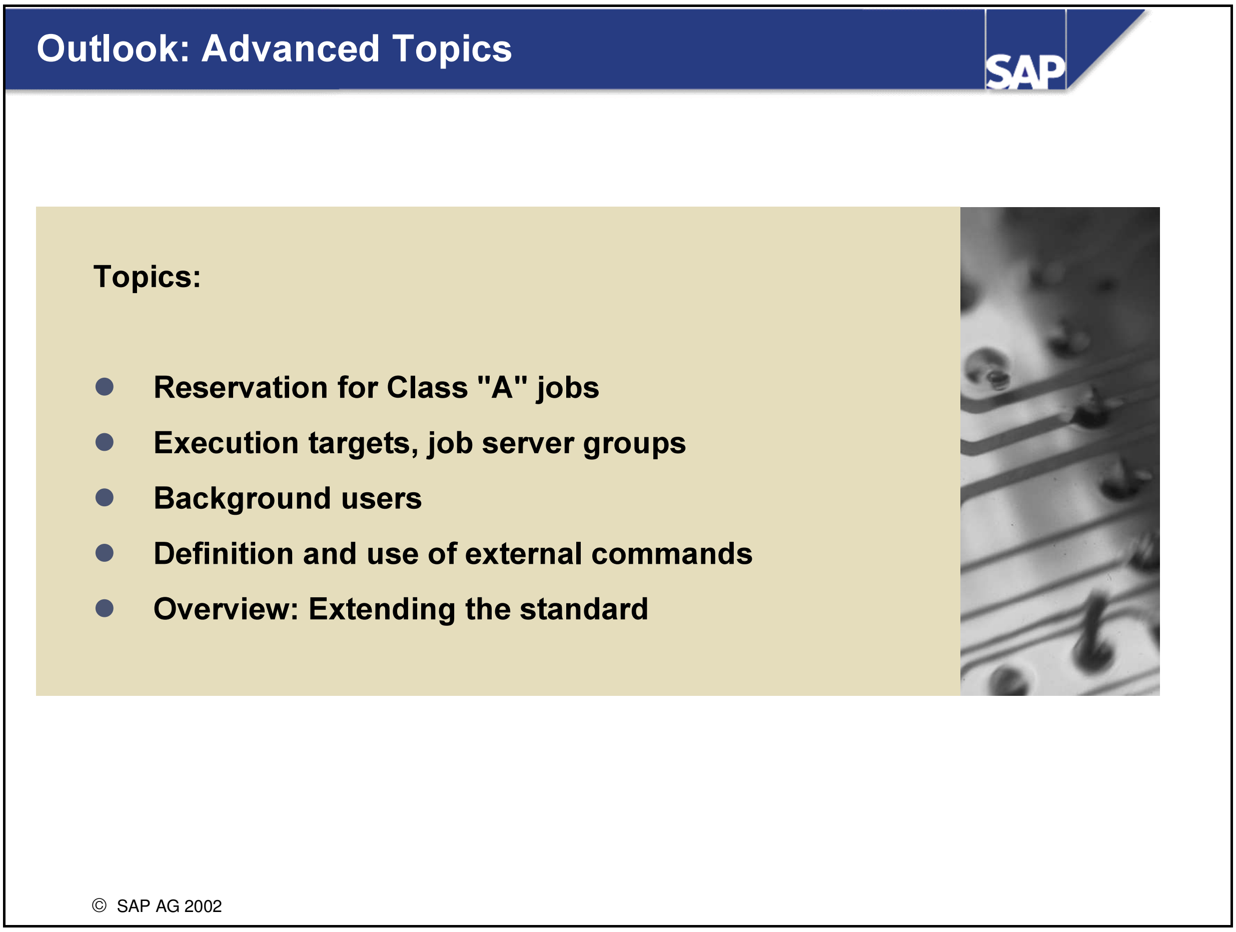
ABAP Program

External

Operating System

sapevt

- New events are defined by the system administrator in CCMS (transaction *SM62*). When doing so, the administrator differentiates between system events and user events. System events are events predefined by SAP that you can neither modify nor trigger.
- Events can be triggered in various ways:
  - Manually in CCMS for test purposes (transaction *SM64*)
  - Within an ABAP program (through use of the function module BP_EVENT_RAISE)
  - Outside the SAP system at operating system level through the program *sapevt*
- A parameter can also be transferred when an event is triggered. In this way, you can define jobs that wait for the occurrence of the event together with the specified parameter.
- The syntax for the program *sapevt* is:
  sapevt <event> [*-p* <parameter>] [*-t*] <[*pf* = <profile>]> | <[*name* = <SID>] [*nr* = <instance>]>
  - <event> : Name of the event, as defined in CCMS (required)
  - <parameter> : specified parameter (optional)
  - *-t* : Creates a trace file called *dev_evt*. If *sapevt* discovers problems, you can find out about these in this trace file (optional)
  - <profile> : Path name of the profile (optional)
  - <SID> : SAP system name (optional)
  - <instance> : Instance number of the SAP system (optional)
  - Example: *sapevt zmyevent name=DEV nr*=00.



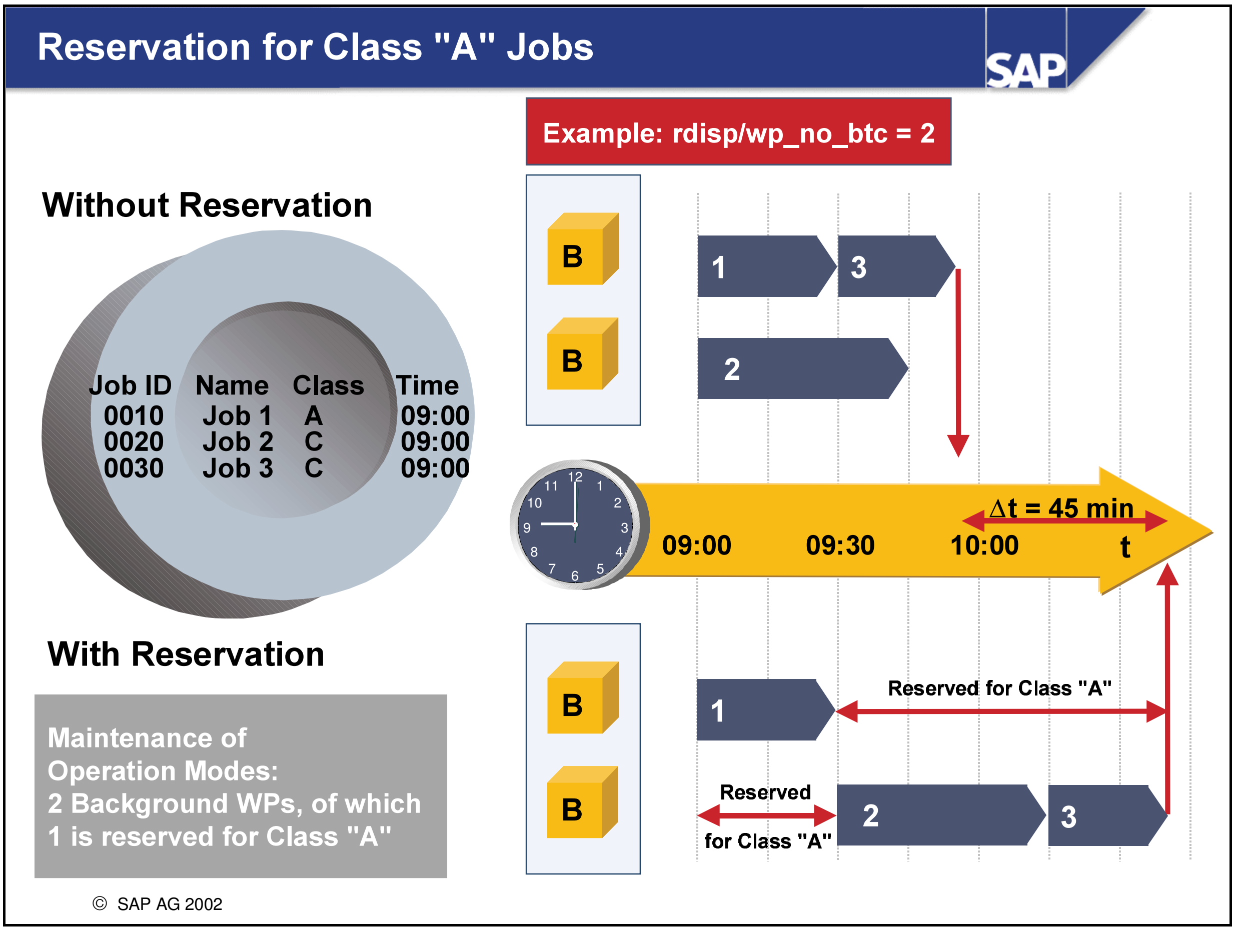
Outlook: Advanced Topics

Topics:

- Reservation for Class "A" jobs
- Execution targets, job server groups
- Background users
- Definition and use of external commands
- Overview: Extending the standard

© SAP AG 2002

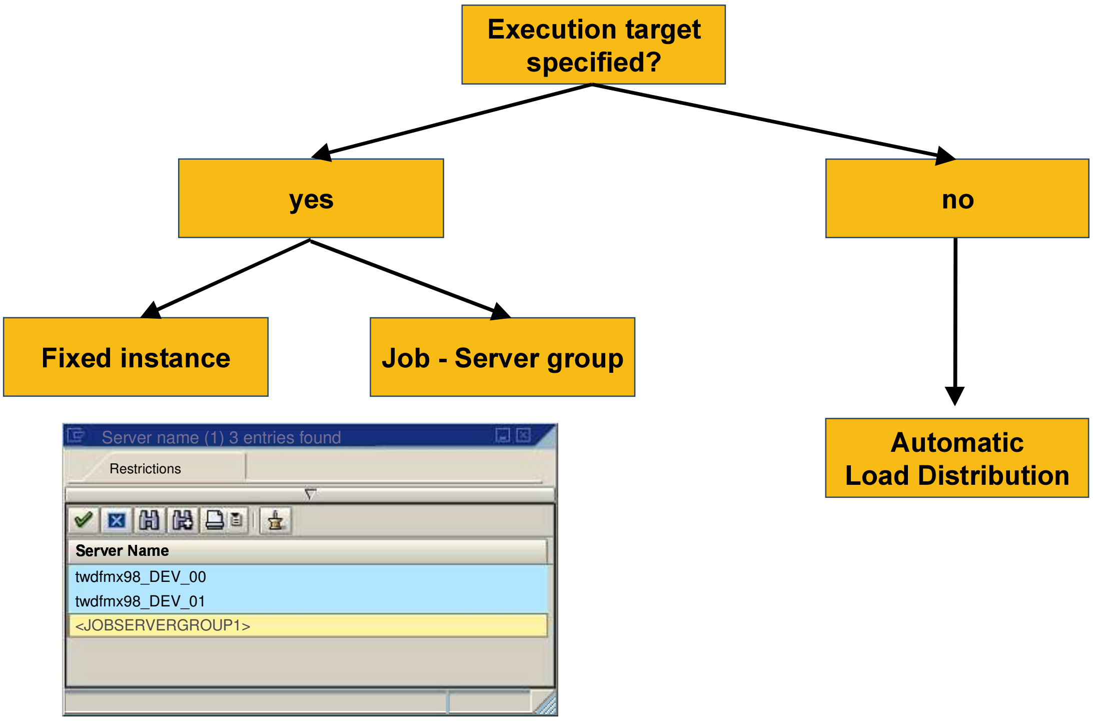
Reservation for Class "A" Jobs



Maintenance of Operation Modes:
2 Background WPs, of which
1 is reserved for Class "A"

- Every work process processes jobs of every priority in normal operation.
- You can reserve as many of the configured background work processes as desired for high priority jobs (class A jobs). The reservation of work processes for class A jobs **does not reserve any particular work processes** in a server. It ensures that **a particular number** of work processes is always kept free. Jobs of job classes "B" and "C" can only be started if the defined number of work processes for possible class "A" jobs remains free.
- To set the number of reserved background work processes for class "A", you can define an operation mode in transaction *RZ04* (maintenance of operation modes and instances) and maintain the work process allocation for this operation mode. When doing so, you have the option of reserving work processes.
- If the class "A" workload is small, or bottlenecks rarely occur in background processing (at least one background work process is usually free) the reservation of a work process for class A jobs probably provides no advantages. In this case, reservation will simply mean that a work process is seldom used.
- You should not reserve more than one background work process for processing class "A" jobs. It is usually sufficient to reserve one background work process for processing class "A" jobs.

Execution Targets and Job Server Groups



Execution target specified?

yes

no

Fixed instance

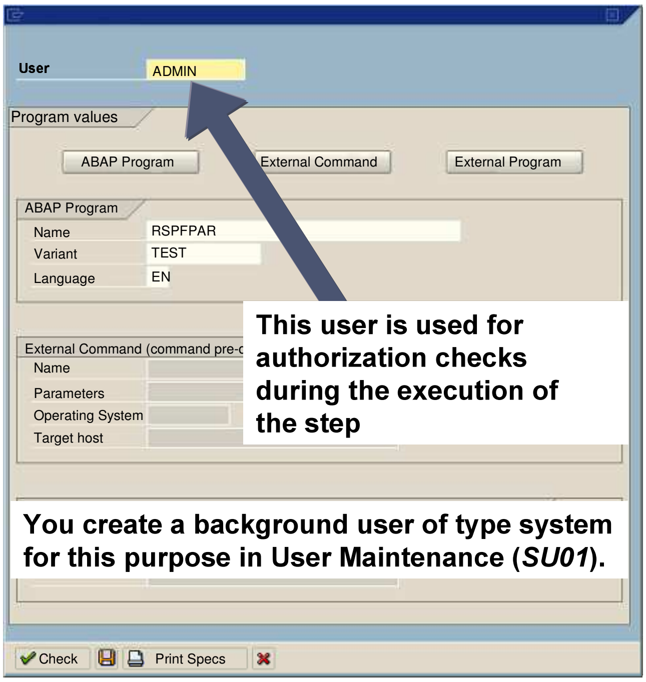
Job - Server group

Automatic Load Distribution

Server Name
twdfmx98_DEV_00
twdfmx98_DEV_01
<JOBSERVERGROUP1>

- Only instances with background work processes or a job server group can be used for the targeted scheduling of a background task.
- A job server group contains one or more instances with available background work processes. Groups of this type can be used in the same way as logon groups for dialog users. It is therefore possible to process background tasks from different applications on selected instances.
- You can set up a job server group in transaction *SM61* (Path: *Tools* → CCMS → *Jobs* → *Background Objects*). To do this, call the transaction and then choose *Job Server Groups*. You can then define your job server group and assign your instances.

Background Users



User ADMIN

Program values

ABAP Program

Name RSPFPAR

Variant TEST

Language EN

External Command (command pre-

Name

Parameters

Operating System

Target host

This user is used for authorization checks during the execution of the step

You create a background user of type system for this purpose in User Maintenance (*SU01*).

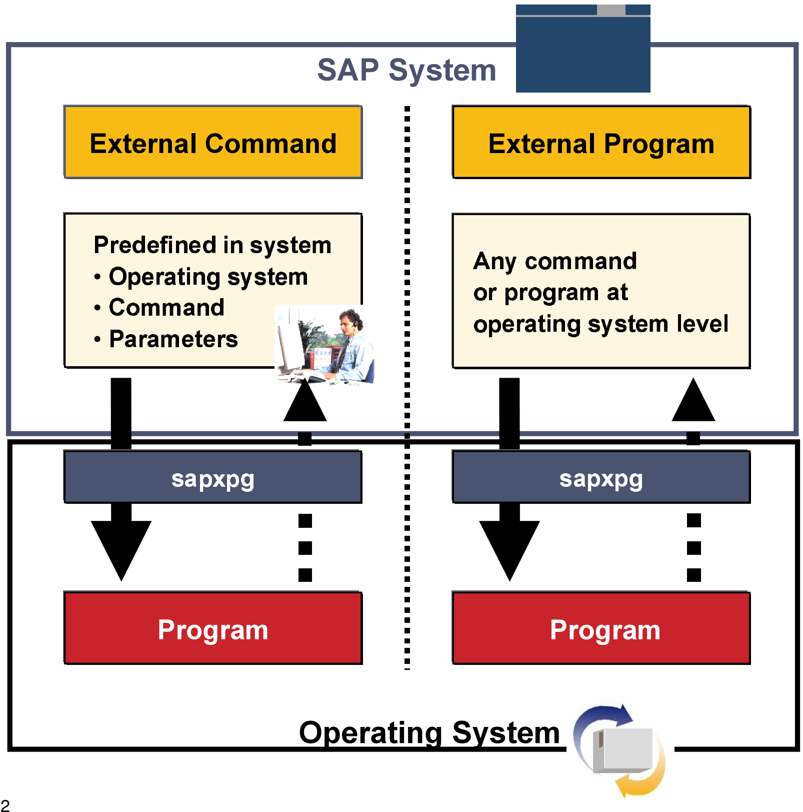
- During the definition of jobs using transaction *SM36*, you can assign each step of the job to a user (see graphic above). The specified user is used for the authorization checks during the execution of the step. By default, your user name appears in this field, and your job is performed using your authorizations. Enter a different user name, if your job should not be performed using your own authorizations. You must have the appropriate authorization (authorization S_BTCH_NAM) to enter names other than your own in the *User* field.

- It is useful to set up background users for various work areas that have the necessary authorizations for the required activities, and that can be used by users with the same authorizations, to schedule background tasks in this work area (such as for system administration). Background users have user master records that are specially intended for background processing authorizations.

- Use the *System* user type when creating background users. A dialog logon with a user of this type is not possible. In the same way, users of this type are excluded from the usual settings for the validity period of a password. The password can only be changed by user adminstrators using transaction *SU01* (*User* → *Change Password*).

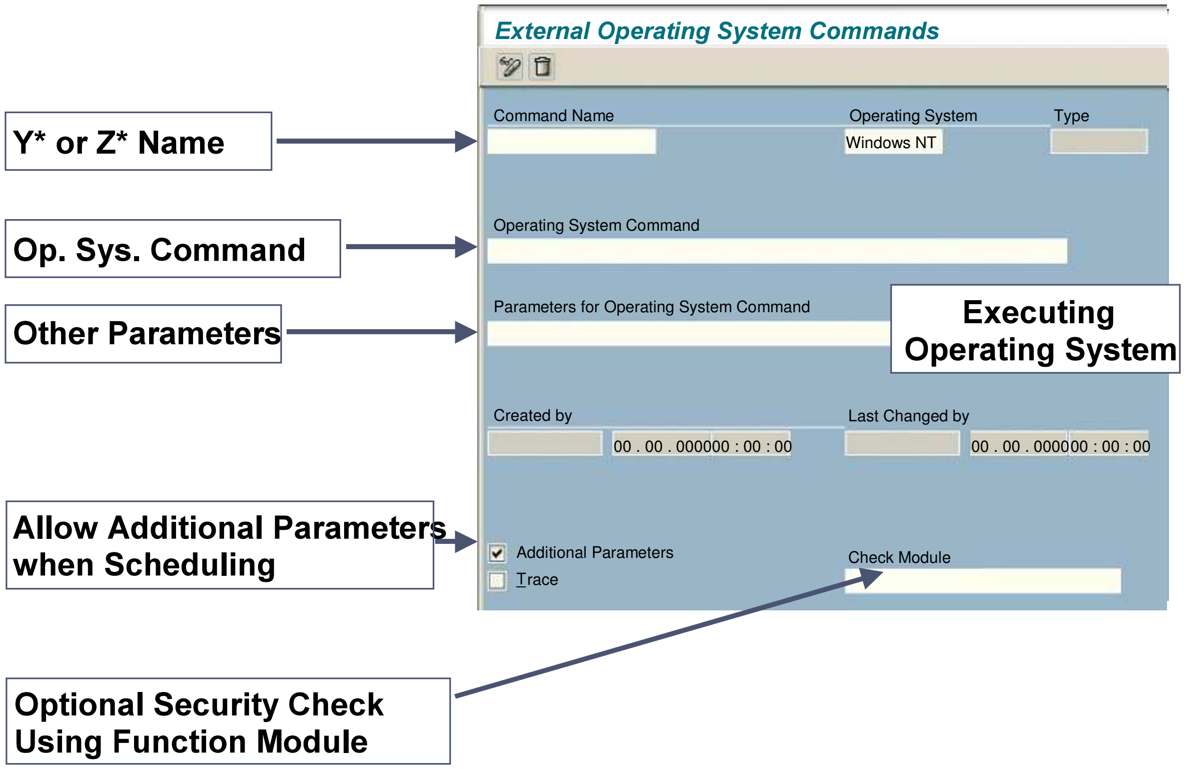
- Your own user is always used for the authorization checks when a job is defined using the Job Wizard.

Use of External Programs



- The background processing system differentiates between *external commands* for normal users and *external programs* for system administrators. The purpose of this differentiation is to give system administrators the ability to run any required *external program*, while normal users are restricted to *external commands* for which there are authorization checks. In both cases, the program sapxpg is called at operating system level. sapxpg then starts the relevant program at operating system level.
- External commands are host system commands or programs for end users predefined in the SAP system by an administrator. They are protected by authorizations so that normal end users can only schedule the commands for which the system administrator has assigned them the authorization. You can therefore provide functions outside the SAP system, at operating system level, for users of the SAP system.
- External programs are unrestricted commands that are neither predefined nor restricted by authorizations. A user with administration authorization can enter any external program in a job step. No SAP authorization check is performed before the execution of the command. External programs provide an administrator (user with administration authorization for external commands (Authorization object S_RZL_ADM: *Background Processing: Background Administrator*)) the flexibility to perform any required host system command in the SAP system without administrative preparation.

Definition and Use of External Commands



External Operating System Commands

Command Name

Operating System: Windows NT

Type

Operating System Command

Parameters for Operating System Command

Created by 00 . 00 . 000000 : 00 : 00

Last Changed by 00 . 00 . 000000 : 00 : 00

☑ Additional Parameters

□ Trace

Check Module

Y* or Z* Name

Op. Sys. Command

Other Parameters

Executing Operating System

Allow Additional Parameters when Scheduling

Optional Security Check Using Function Module

- The creation of external command requires the following steps:

1. Call transaction *SM69* (*CCMS* → *Configuration* → *External Commands*).

2. Choose *Display/Change* and then *Create*.

3. Enter a name for your command. External commands are uniquely identified with a name (beginning with Y or Z) and an operating system type. The field *Type* is filled automatically.

4. Specify an executable operating system command (if required with the complete path) and specify any additional required or optional parameters.

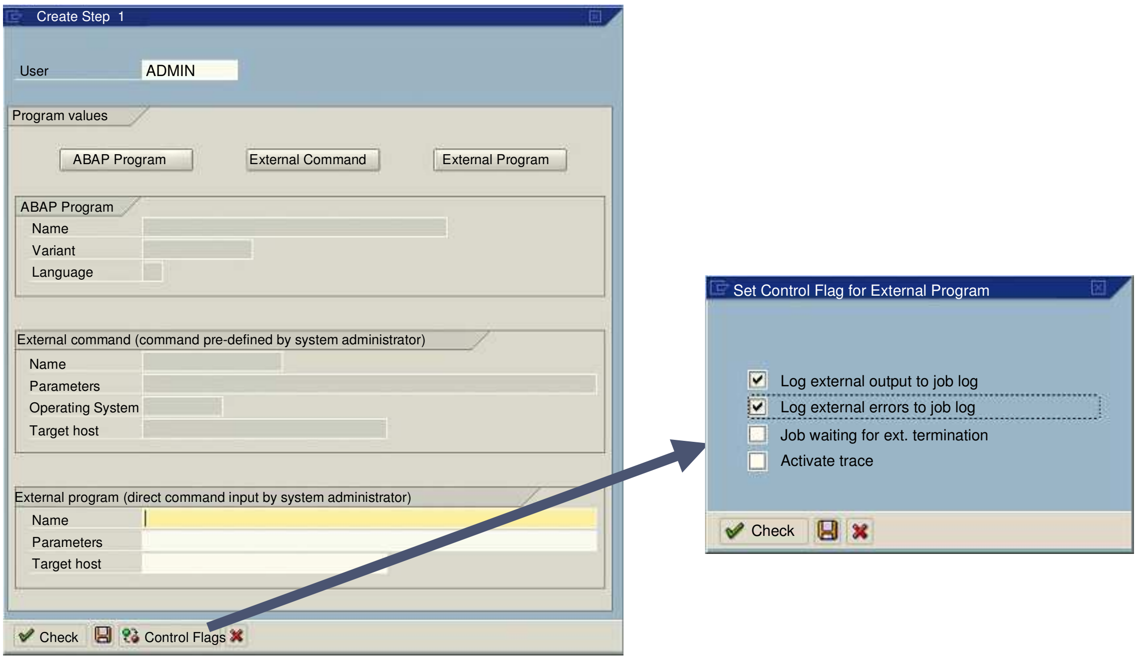
5. Select the *Additional parameters allowed* field if users want to specify additional parameters when they schedule the command in a background job. The additional parameters are added to the parameter strings under *Parameters for Operating System Command.*

6. The *Trace* field should usually be left blank. To follow the execution of an external command, use the trace parameter for the function module SXPG_COMMAND_EXECUTE.

7. If you have defined an additional authorization check, enter the name of the function module that performs the check under *Check Module*, usually a customer copy of the function module SXPG_DUMMY_COMMAND_CHECK. The system automatically calls the function module if a user tries to schedule the external command in a background job.

8. Save the external command both in the screen shown above and in the initial screen of transaction *SM69.*

Control Flags for External Programs/Commands



Create Step 1

User ADMIN

Program values

ABAP Program
- Name
- Variant
- Language

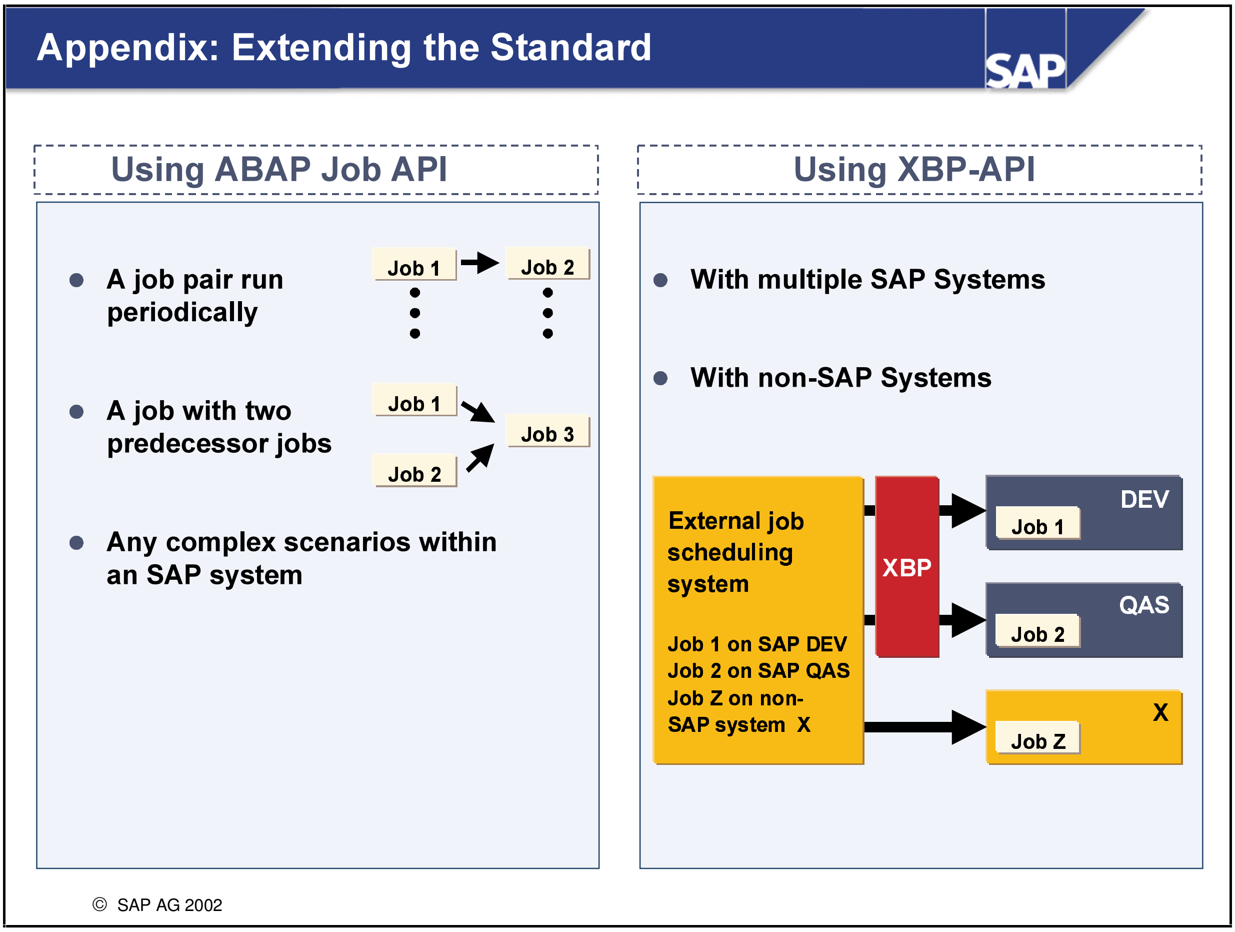
External command (command pre-defined by system administrator)
- Name
- Parameters
- Operating System
- Target host

External program (direct command input by system administrator)
- Name
- Parameters
- Target host

Set Control Flag for External Program

☑ Log external output to job log
☑ Log external errors to job log
□ Job waiting for ext. termination
□ Activate trace

- Make specifications about the task and other runtime options using *control flags*. You do not normally need to change the default values here.
- For example, you can specify:
  - Whether the process should be logged. The output data is written to the log in the way that it was output by the external program. The language can differ here from your logon language. You can also call additional information about the data of an external program in the job log.
  - How the SAP control program should end and whether the external program triggers an event. After you have started service programs with the background processing system (such as daemons in UNIX systems), they remain active after the start. They are not ended and do not return to the SAP background processing system, as is the case with other programs. If you start a service program, you should not use the control flag setting *Job waiting for ext. termination* when scheduling the job.
- You can display additional information about the meaning of the control flags using the F1 field help on the *Set Control Flag for External Programs* screen.



Appendix: Extending the Standard

Using ABAP Job API

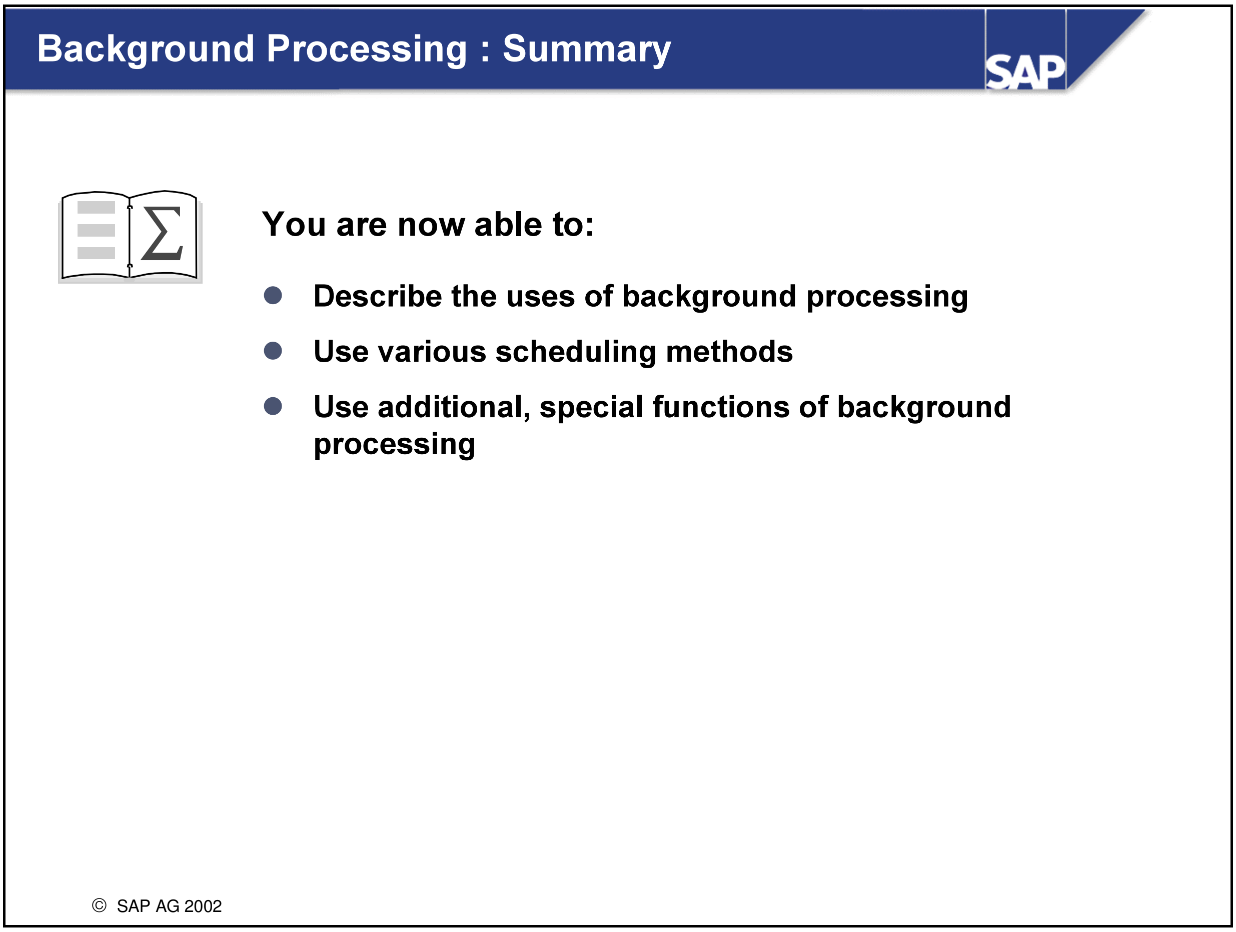
- **A job pair run periodically**
- **A job with two predecessor jobs**
- **Any complex scenarios within an SAP system**

Using XBP-API

- **With multiple SAP Systems**
- **With non-SAP Systems**

© SAP AG 2002

- The job scheduling options described in this unit do not cover all possible requirements. You can realize more complex scenarios in the following ways:
  - The SAP system has various internal function modules that support you when defining your own job processes. These function modules are in the function groups *BTCH* and *BTC2*. You can define any complex scenarios with the help of these function modules.
  - SAP provides a set of interfaces that allow the connection of other system management environments to the SAP Computing Center Management System (CCMS):
    - The eXternal Monitoring Interface API (XMI-API) interface logs the activities of external users and programs.
    - The eXternal interface for Background Processing (XBP-API) interface allows the use of external job scheduling programs.
  - With these tools, you can schedule background jobs beyond the boundaries of the SAP system and include non-SAP systems.
- You can find a list of certified solutions on the SAP Service Marketplace under the Quick Link *softwarepartner* (Software Partner Program), by branching to the Software Partner Directory and searching in the software category *Job Scheduling*.



Background Processing : Summary

You are now able to:

- Describe the uses of background processing
- Use various scheduling methods
- Use additional, special functions of background processing

- For more information about the distribution of background jobs to application servers, see SAP Note 24092.

Exercises

No.	Exercises
1	(Optional) **Check the configuration of your SAP system with regard to background processing** The following steps familiarize you with the settings of the SAP system that are relevant for background processing
1.1	How many background processes are configured in your SAP system?
1.2	What determines the number of available background work processes?
1.3	How many background work processes can work at the same time?
1.4	Check the frequency with which the background job scheduler runs.
1.5	Find out if background work processes are reserved for jobs of job class A in your system.
1.6	Assume that all background work processes in your system are currently processing and there are therefore no free resources for other jobs. Can you set up additional background work processes without restarting the SAP system?
2	**Creating, Scheduling and Monitoring a Job** Schedule jobs and monitor their execution. Check the job events.
2.1	Run report *RSUSR000* in dialog (transaction *SA38*).
2.2	Run report *RSUSR000* in the background using transaction *SA38* (start time *Execute immediately*).
2.3	Define the job **`SIMPLE_##`**, where **`##`** is your group number. The job should have the following properties: • Job class C • Consists of the ABAP program *RSUSR000* as the only step • Immediate start
2.4	Check whether your job *SIMPLE_##* has run successfully. Find data that gives details about the start time, steps, and job details (such as client and user name). Is the spool list automatically printed?
2.5	(Optional) Determine the execution time of your job *SIMPLE_##* and find out whether it was executed with a delay.
3	**Schedule an ABAP program with a variant and an external program** Schedule an ABAP program with a variant and an external program.
3.1	Define the job **`STEPS_##`**, where **`##`** is your group number. The job should have the following properties: • Job class C • Step 1 Execution of the report *RSPFPAR* with variant *DISPATCHER* Step 2: Execution of the external program `whoami` (check the control flags for this) • Start in 2 minutes (referring to system time)

	Check whether the job was successfully processed.
3.2	(Optional) Set up your own variant for report *RSPFPAR* in transaction *SA38*, and schedule a job that uses this variant.
4	**External commands and event-controlled jobs** Define a user-defined event and then schedule a job that starts after this event occurs. Trigger the event in the Computing Center Management System (CCMS) and from operating system level.
4.1	Display a list of external commands. What does the command *ZKERNEL* do?
4.2	Define the user-defined event **`MYEVENT_##`** in the CCMS.
4.3	Define a job **`EVENT_##`**. The job should have the following properties: • Job class C • Step 1 Execution of the external command *`ZKERNEL`* • Starts after the event *`MYEVENT_##`* (event-periodic) Display your job in the job overview.
4.4	Trigger the event *`MYEVENT_##`* `in the` CCMS in another session and refresh the job overview.
4.5	Trigger the event at operating system level and refresh the job overview.
5	(Optional) **Definition of job chains** Define a job that depends on the job in exercise 4.
5.1	Define the job **`SUCCESSOR_##`** (where **`##`** is your group number), with the following properties: • Job class C • Step 1 Execution of ABAP program *`RSUSR000`* • Start after the job from exercise 4 has been successfully completed Check the job status in the job overview.
5.2	What must you do so that the job *`SUCCESSOR_##`* is executed?

Solutions

No.	Solutions
1	(Optional) **Check the configuration of your SAP system with regard to background processing** The following steps familiarize you with the settings of the SAP system that are relevant for background processing
1.1	**How many background processes are configured in your SAP system?** Using transaction *SM66*, check the number of work processes of type *BTC* in your system (To do this, choose *Select process* and set the indicator *Status → wait*).
1.2	**What determines the number of available background work processes?** The parameter *rdisp/wp_no_btc* initially specifies the number of available background work processes on the current instance. You can check the value of the parameter with either transaction *RZ11* or the report *RSPFPAR*. If you have defined operation modes (transaction *RZ04*), this value can be overwritten. In this case, use the CCMS Control Panel (transaction *RZ03*) to specify the current operation mode.
1.3	**How many background work processes can work at the same time?** You can process as many jobs concurrently as there are background work processes available.
1.4	**Check the frequency with which the background job scheduler runs.** Check the parameter *rdisp/btctime*. This parameter specifies the frequency with which the background job scheduler runs on your background server. You can check the value of the parameter with either transaction *RZ11* or the report *RSPFPAR*.
1.5	**Find out if background work processes are reserved for jobs of job class A in your system.** You can only check whether you have defined background work processes for jobs of job class A using transaction *RZ04* (Maintain Operation Modes and Instances). Choose *Instances/Operation Modes* there and check column *BPA*.
1.6	**Assume that all background work processes in your system are currently processing and there are therefore no free resources for other jobs. Can you set up additional background work processes without restarting the SAP system?** Using the concept of operation modes, you can define an operation mode with additional background work processes in transaction *RZ04*. You can then perform an operation mode switch using transaction *RZ03*. You do not need to restart your system when doing this.
2	**Creating, Scheduling and Monitoring a Job** Schedule jobs and monitor their execution. Check the job events.
2.1	**Run report *RSUSR000* in dialog (transaction *SA38*).** Select the report *RSUSR000* in the initial screen of transaction *SA38* and

	choose *Execute*.
2.2	**Run report *RSUSR000* in the background using transaction *SA38* (start time *Execute immediately*).** Select the report *RSUSR000* in the initial screen of transaction *SA38* and choose *Background*. On the next screen, choose *Execute Immed.* To check the job status and the spool list for this job, choose *Job Overview*. The system displays the initial screen of transaction *SM37*, on which you can display the results of your job (*Execute*).
2.3	**Define the job SIMPLE_##, where ## is your group number. The job should have the following properties:** • **Job class C** • **Consists of the ABAP program *RSUSR000* as the only step** • **Immediate start** In transaction *SM36*, enter the job name **SIMPLE_##**, select job class *C* and leave the execution target empty. Choose *Step*. On the following screen, choose *ABAP Program* and enter the ABAP program name **RSUSR000**. Save your step. Then choose *Start Condition*. Choose *Immediately* and save the start time values. Then save your job. The system displays a message in the status line that your job was saved with the status *released*.
2.4	**Check whether your job *SIMPLE_##* has run successfully.** **Find data that gives details about the start time, steps, and job details (such as client and user name).** **Is the spool list automatically printed?** Use transaction *SM37* to check the status of your job, and choose *Execute* on the initial screen. To analyze job details, double click the job and choose *Job Details*. The details include, for example, the client, the job ID number, the user that scheduled, changed, and released the job, and the work process that executed the job. To see the individual steps of the job, choose *Step*. Spool lists are only created for job steps that contain ABAP programs. They are only printed immediately if you have made the corresponding setting in the printer settings during the definition of the relevant job step or in your user defaults.
2.5	**(Optional) Determine the execution time of your job *SIMPLE_##* and find out whether it was executed with a delay.** You can display the duration of your job in transaction *SM37* (see solution to exercise 2.4). You can also see here whether your job was started with a delay (caused, for example, by a high load on your system or due to the fact that the background job scheduler runs periodically).
3	**Schedule an ABAP program with a variant and an external program** Schedule an ABAP program with a variant and an external program.
3.1	**Define the job STEPS_##, where ## is your group number. The job should have the following properties:** • **Job class C** • **Step 1 Execution of the report *RSPFPAR* with variant *DISPATCHER*** **Step 2: Execution of the external program *whoami* (check the control**

	flags for this) • **Start in 2 minutes (referring to system time)** **Check whether the job was successfully processed.** In transaction *SM36*, enter the job name **`STEPS_##`**, select job class *C* and leave the execution target empty. Choose *Step*. On the following screen, choose *ABAP Program* and enter the ABAP program name **`RSPFPAR`** and the variant **`DISPATCHER`**. Save your step. Then choose *Step* again and choose *Create*. As a next step, choose *External Program* and enter the program name **`whoami`** with no parameters. Enter the name of your own host as the target host. Check the *Control flags* and save the default values. Save your step. Return to the initial screen of the transaction and choose *Start Condition*. Choose *Date/Time* and enter a time that is approximately two minutes in the future and save the start time values. Then save your job. The system displays a message in the status line that your job was saved with the status *released*. Check whether your job was successfully processed as described for exercise 2.4.
3.2	**(Optional) Set up your own variant for report *RSPFPAR* in transaction *SA38*, and schedule a job that uses this variant.** Enter the report name **`RSPFPAR`** in the initial screen of transaction *SA38* and choose *Goto → Variants*. Enter a name for your variant and choose *Create*. Fill out the selection screen as you wish and choose *Attributes*. Enter a meaningful description for your variant here and save the variant. You can now schedule a job in transaction *SM36* (in the same way as in exercise 3.1) that executes the report *RSPFPAR* with your own variant.
4	**External commands and event-controlled jobs** Define a user-defined event and then schedule a job that starts after this event occurs. Trigger the event in the Computing Center Management System (CCMS) and from operating system level.
4.1	**Display a list of external commands. What does the command *ZKERNEL* do?** To display a list of external commands, choose *Tools → CCMS → Jobs → External Commands* (transaction *SM49*) from the SAP Easy Access menu. To execute a command from this list, select the command, and choose *Execute*. On the following screen, you can specify a target host. Choose *Execute* again. The external command *ZKERNEL* displays information about the kernel (*disp+work*).
4.2	**Define the <u>user</u>-defined event** `MYEVENT_##` **in the CCMS.** In the SAP Easy Access Menu, choose *Tools → CCMS → Jobs → Maintain Event* (transaction SM62). Do not change any *System event names*. Choose *Maintain* under the point *User event names*, and choose *Enter*. Choose *Display*. Enter the event description **`MYEVENT_##`** (where `##` is your group number) and a description of the event, and then choose *Save*.

4.3 **Define a job EVENT_##. The job should have the following properties:**

- **Job class C**
- **Step 1 Execution of the external command *ZKERNEL***
- **Starts after the event *MYEVENT_##* (event-periodic)**

Display your job in the job overview.

In transaction *SM36*, enter the job name **EVENT_##** (where **##** is your group number), select job class *C* and leave the execution target empty.

Choose *Step*. On the following screen, choose *External Command* and enter the name **ZKERNEL** with no other parameters. Select an operating system and specify your own host as the target host. Save your step.

Choose *Start Condition*. Choose *After Event*, enter the event **MYEVENT_##** without other parameters, set the flag *Periodic job*, and save the start time values. Then save your job. The system displays a message in the status line that your job was saved with the status: released

As in exercise 2.4, use transaction *SM37* to check whether your job was successfully scheduled. To do this, you must make an appropriate selection on the initial screen of *SM37* under the job start condition in the field *or after event* (such as *).

4.4 **Trigger the event *MYEVENT_##* in the CCMS in another session and refresh the job overview.**

In the SAP Easy Access Menu, choose *Tools* → *CCMS* → *Jobs* → *Trigger Event* (transaction *SM64*). Select your event and trigger it without additional parameters.

If you refresh the job overview in transaction *SM37*, you can see that your job has been processed (status *complete*) and has scheduled itself again (status *released*).

4.5 **Trigger the event at operating system level and refresh the job overview.**

Run the command **sapevt MYEVENT_## name=<SID> nr=<nr>** at operating system level of your application server. If you refresh the job overview in transaction *SM37*, you can see that, as in exercise 4.4, your job has been processed (status *complete*) and has scheduled itself again (status *released*).

5 (Optional) **Definition of job chains**

Define a job that depends on the job in exercise 4.

5.1 **Define the job SUCCESSOR_## (where ## is your group number), with the following properties:**

- **Job class C**
- **Step 1 Execution of ABAP program *RSUSR000***
- **Start after the job from exercise 4 has been successfully completed**

Check the job status in the job overview.

In transaction *SM36*, enter the job name **SUCCESSOR_##**, select job class *C*, and leave the execution target empty.

Choose *Step*. On the following screen, choose *ABAP Program* and enter the

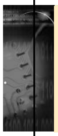
	ABAP program name **RSUSR000** without a variant. Save your step. Choose *Start Condition*. Choose *After job*, enter the predecessor job **EVENT_##**, and choose *Start status-dependent*. Save your start time values. Then save your job. The system displays a message in the status line that your job was saved with the status *released*. As in exercise 2.4, use transaction *SM37* to check whether your job was successfully scheduled. To ensure that your job is also displayed in the job overview, choose *Extended Job Selection* on the initial screen of transaction *SM37*. On the following screen, select your job (or *****) in the field *OR Start after job* on the *Start condition* tab page and choose *Execute*.
5.2	**What must you do so that the job *SUCCESSOR_##* is executed?** Trigger the event *MYEVENT_##* in another session, in the same way as with exercise 4.4 or 4.5. Check that your job chain was successfully executed by choosing *Refresh* in the job overview.

Printing from SAP Systems

8. System landscapes and transport requests

9. Support Packages and Industry Solutions

10. Scheduling background tasks



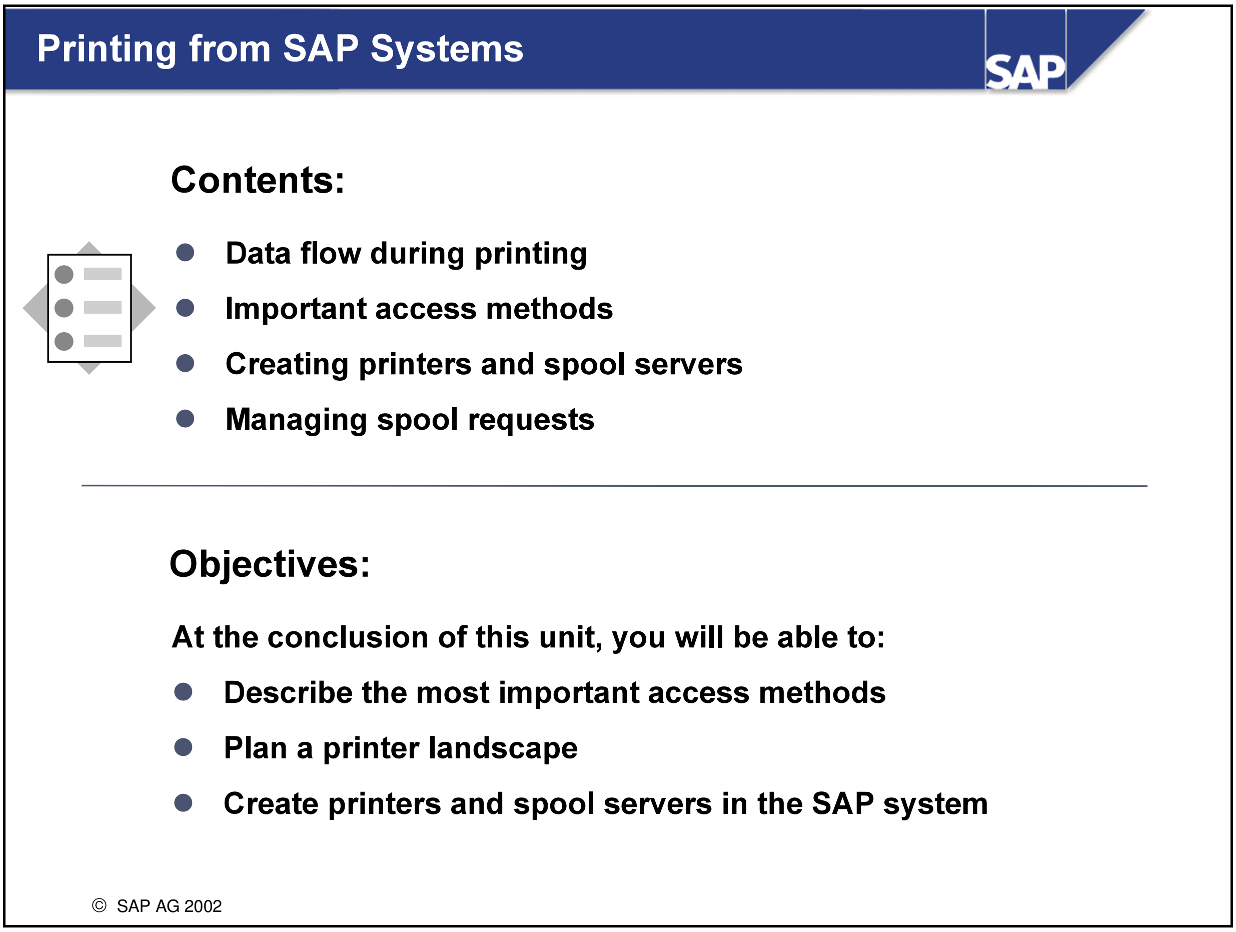
11. Printing from SAP systems

12. Basics of user administration

13. Remote connections

14. SAP Solution Manager

15. System monitoring



Printing from SAP Systems

Contents:

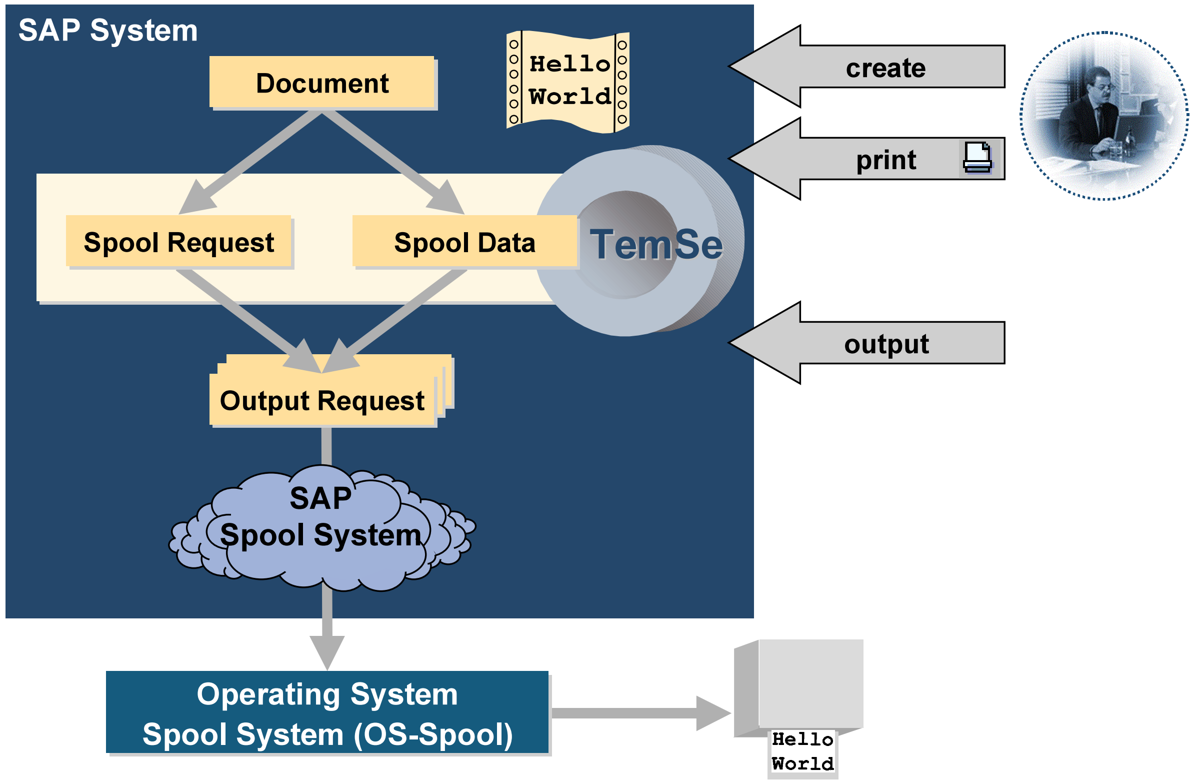
- Data flow during printing
- Important access methods
- Creating printers and spool servers
- Managing spool requests

Objectives:

At the conclusion of this unit, you will be able to:

- Describe the most important access methods
- Plan a printer landscape
- Create printers and spool servers in the SAP system

Data Flow During Printing



SAP System

Document

Hello World

create

print

Spool Request

Spool Data

TemSe

output

Output Request

SAP Spool System

Operating System Spool System (OS-Spool)

Hello World

- There are various document classes in the SAP system (such as report lists, SAPscript or Smart Forms documents). Although the way in which documents are created may be completely different, the output on paper is always performed using the same mechanism in two steps: First a **spool request** is created. The spool request contains device-independent print data and includes administrative information (such as author, date, and number of copies) and the actual print data. Only when the spool request is to be output on a particular device is an **output request** created. The device-independent print data from the spool request is converted to the printer language that the selected output device understands. This procedure allows the user to display a spool request before output. There can also be several output requests for one spool request. This can avoid recreating (possibly with a great deal of effort) a spool request, if, for example, the toner in a printer is exhausted, or the wrong paper was in the tray. The user can of course create a spool request and an output request at the same time (immediate print) by choosing the *Print immediately* option.
- The actual print data of a spool request is stored in **TemSe** (for temporary sequential objects), for which you define the storage location with the profile parameter *rspo/store_location*. Value *db* (the default value): Spool requests are stored in database table TST03 (advantage: backup as part of the database). Value *G*: Stored at operating system level in global directory (advantage: performance). SAP Note 20176 contains additional possible values for *rspo/store_location*.
- The creation of an output request prompts the SAP spool system to send a (usually) printer-dependent, completely formatted data stream to a printer using an operating system spooler (OS spool). This means that the addressed printer model must be known to the SAP system. Definitions of this type are described as **device types**.
- If a printer cannot be controlled at operating system level, it cannot be used from the SAP system.