

Background Job monitoring with SAP Solution Manager

Best Practice for Solution Management

Version Date: November 2005 The newest version of this Best Practice can always be obtained through the SAP Solution Manager

Contents

Applicability, Goals, and Requirements	2
Best Practice Procedure	3
Background Job Monitoring	3
Motivation for Background Job Monitoring	
Real-time Monitoring with SAP Solution Manager	6
Scenario BW process chain for data upload	9
Scenario Demand planning with SAP APO	19
Scenario MRP \rightarrow event \rightarrow BOP	23
Scenario: Mass creation of deliveries	29
Scenario Billing run in five parallel tasks	
Scenario IDoc processing	
Scenario Periodic job chain with SM36	
Further Information	40

Applicability, Goals, and Requirements

To ensure that this Best Practice is the one you need, consider the following goals and requirements.

Goal of Using this Service

This Best Practice will help you to set up background job monitoring properly in the framework of Business Process Monitoring in the SAP Solution Manager. This Best Practice document can be seen as an enhancement to the corresponding chapter in the "Business Process Monitoring – Setup Guide", which can be found on the SAP Marketplace under the Quick Link /BPM \rightarrow Media Library.

It should be clear that the SAP Solution Manager is not purely a background job monitoring tool, although the title of the document may suggest something different. The SAP Solution Manager is mainly intended as a platform to provide the integrated content, tools, and methodologies that are needed to implement, support, operate and monitor your enterprise's solutions from SAP. It can be used to monitor important selected background jobs and assign the possible alerts to the corresponding business process steps within your (core) business processes. This paper will provide you with guidelines how to set up job monitoring on the basis of given examples.

Alternative Practices

You can get SAP experts to deliver this Best Practice on-site if you order a Solution Management Optimization (SMO) service known as the *SAP Business Process Management* service which is a superordinate service comprising the Business Process Monitoring with the SAP Solution Manager.

Staff and Skills Requirements

The background job monitoring as part of the Program Scheduling Management should always be considered within a holistic Business Process Management concept and not isolated as on its own. Hence, to implement a Business Process Management concept and thus this Best Practice, you require the following teams:

Application Management Team

The Business Process Management concept (which the superordinate Best Practice *General Business Process Management* aims to produce) should be created by the Application Management Team. This team combines experts from your company's:

- Business department
- □ Solution support organization (for example, the IT department and the Help Desk)
- Implementation project team

Execution Team

The persons who will be responsible for applying the resulting procedures derived through using this Best Practice include:

- Persons designated to perform business process monitoring and ensure that business processes run smoothly
- □ The business department end users and the IT department
- □ All parties involved in the customer's support and monitoring organization
 - The Business Process Champion or key user for each business process
 - Application Support
 - Development Support
 - Program Scheduling Management
 - Software Monitoring Team
 - System Monitoring Team

More information about roles and responsibilities of these teams can be found in the superordinate Best Practice *General Business Process Management*, which can be obtained through the Solution Manager or through the SAP Service Marketplace via the Quick Link */solutionmanagerbp*. There you will also find a Best Practice *Program Scheduling Management* which should lay the conceptual groundwork for the Best Practice in hand.

System Requirements

None.

Duration and Timing

Duration

Creating a Business Process Management concept depends on the complexity and number of your core business processes, and could take around one week per business process.

Implementing the Business Process Management concept could take around one additional week.

<u>Timing</u>

The best time to apply this Best Practice is during the planning phase or during the implementation phase of your mySAP solution.

Best Practice Procedure

Background Job Monitoring

Motivation for Background Job Monitoring

The monitoring task for background jobs (or programs in general) is only one aspect of a holistic Program Scheduling Management concept (the others are planning and scheduling). Hence we recommend considering your job monitoring not as a separate issue but as part of a holistic Program Scheduling Management concept (as described e.g. in the Best Practice document with the same title) instead. We see this as a necessary prerequisite before starting with your job monitoring. A motivation for Program Scheduling Management in general is given in the corresponding Best Practice document, so that we won't repeat it here. As the monitoring of many hundred or thousand background jobs per day might be very cumbersome when only using transaction SM37 several automatic monitoring tools are provided by SAP. These will be described in the following sections.

The tools for automation also provide the opportunity for faster reaction times in case of problems. Additionally, the root cause analysis should be speeded-up and possible business impacts should be immediately identified.

Background Job Monitoring with CCMS

The CCMS infrastructure provides a set of preconfigured monitors that can be easily used for automated background job monitoring.

General CCMS alerts

The following information is taken from <u>help.sap.com</u>:

"Using this [Background Processing] monitor, you can monitor the status of background processing in your SAP system both for the entire system and itemized by application servers on which background work processes are configured. [...]"



Features

The monitor contains the following monitoring tree elements (MTEs):

MTE	Description
SystemWideQueueLength	Number of jobs that are ready to be executed, have start authorization, and have no target server specified for which there are no free background work processes, averaged over all application servers with background work processes.
SystemWideClassABPWP	System-wide number of background work processes that are reserved for high priority jobs (Class A Jobs)
SystemWideFreeBPWP	Number of free background work processes in the entire system
SystemWideTotalBPWP	Number of background work processes in the entire system
R3Syslog\Background	Messages in the system log for the background processing category; you can set the category in which a message is reported, the message text, and the severity and criticality of the alert using the message ID in transaction SE92
Utilization	Percentage of the background processing capacity currently utilized; the value is averaged over the background work processes and, by default, averaged over the last hour
NumberOfWpBTC	Number of background work processes on an application server
ErrorsInWpBTC	Number of errors in background work processes since the monitoring segment was created (that is, since the application server was started)
ErrorFreqWpBTC	Number of errors in background work processes per minute

EndedWpBTC	Number of background work processes terminated after an error; you can use the process overview (transaction SM50) to determine whether a work process should be restarted after an error
ProgramErrors	Program errors when executing background jobs
ServerSpecificQueueLength	Number of released jobs that are explicitly to be executed on this application server, but for which there are no free background work processes
AbortedJobs	Individual aborted jobs on an application server; a separate red alert is generated for each of these jobs

Specific (Mass) Background Job Monitoring

At this point we would like to emphasize that the SAP Solution Manager is not intended to perform mass background job monitoring. If you are only interested in knowing whether the hundreds or even thousands of background jobs in your SAP system were processed successfully (without cancellation) and within an appropriate runtime without reference to any business process, use the CCMS Infrastructure. This is an easier and more convenient way to realize such monitoring. This functionality described in SAP Note 553953 "RZ20: Monitoring background jobs" offers an overview of the status and runtime of jobs which are regularly running in the system. In addition to the general monitoring of the background runtime environment provided with SM37, alerts can be created for particular job names or name patterns in case of an error. Basically, it is only necessary to enter an SID of the satellite system and a job name or name pattern in table ALBTCMON. For further information refer to the above mentioned SAP Note 553953.

Job monitoring provides the following data:

- The status of the current job in a job chain
- The job log of the current job of a job chain, if it already exists
- The (technical) delay of the current job of a job chain, if the start time has already been exceeded
- The runtime of the current job of a job chain, if the job is already running or is finished
- The total of runtime and delay
- A history of the behavior of the entire job chain

Alerts can also be generated, with the following prerequisites:

- If a job of the job chain terminates
- If a certain message occurs in the job log
- If threshold values are exceeded for a job of the job chain for
 - o Runtime
 - o Delay
 - o The total of runtime and delay

Real-time Monitoring with SAP Solution Manager

The SAP Solution Manager is one possibility for performing automated real-time monitoring. This tool is not intended to work solely as a job monitoring tool but as a platform which provides the integrated content, tools, and methodologies that you need to implement, support, operate and monitor your enterprise's solutions from SAP. This includes providing a tool for monitoring whole business processes. As background jobs mostly play a key role in performing tasks within core business processes the real-time alert monitoring of background jobs is one main focus of the Business Process Monitoring functionality. For further details of how to setup Business Process Monitoring functionality in general, please refer to the SAP Service Marketplace under the Quick Link /BPM. Follow the link "Media Library" and find the latest "Business Process Monitoring – Setup Guide". Background jobs especially, are referred to in chapter 5 "Setup of Business Process Monitoring".

For specific alert monitors there is another document "Application Monitor and User Exit – Setup Guide".

This Best Practice document will explain an appropriate usage of the different alert types and should give a feeling for how to setup background job monitoring for some common examples and business scenarios, respectively. The meaning of the different alert types is not explained explicitly in this document. For such details please refer to the Setup Guide mentioned above.

Simple examples

In the section *Simple examples* a summarization of the most common cases for background jobs is described. For each sort of job, there is a description of what kinds of alerts it is possible to configure, and some background information is provided. In later sections some business scenarios are described which illustrate Best Practices of how to monitor background jobs in specific cases.

Singular job

The job (e.g. a Material Requirements Planning-run or Back Order Processing-run) starts at a certain point in time (e.g. 2 am) and runs for an approximately known period (about two hours). The job is **not** periodic in a sense that it is run more than once per day. The start procedure can be any of the four different possibilities *'by time', 'by event, 'by preceding job' or 'use job start condition'*.



Under which circumstances should you use which of the four start conditions?

If you know a planned scheduling time (e.g. 2am) you should use the start condition 'by time' and specify the 02:00:00 as the start time. Please make sure that you do not specify a time that lies ahead of the planned start time. In such a case the corresponding background job would not be found as the data collector doesn't look in the past. With this configuration you are able to configure the start delay alert which is a delay from the planned runtime and not the technical delay that can be seen in transaction SM36.

If the job is **always** started by an event one could use the start condition 'by event' and specify the corresponding event. Additionally one can specify a planned runtime as well so that the start delay alert can be used too as described above. The only thing one has to consider is that if the job is not triggered by an event but is started regularly by time, the job will not be identified by the data collector even when it runs at exactly the same time (e.g. 2am)! If the start condition 'by event' is chosen a job is only found if it is really started by this event no matter if an additional start time is provided or not!

The start condition 'by preceding job' is a special case of the 'by event' case. One can maintain a planned start time as well in order to get a start delay alert but this planned start time has no influence on whether the job is found by the data collector or not. The job will only be identified if it is started by the necessarily specified preceding job.

All the three cases mentioned above have two important things in common. One is that a start delay alert can be configured which is a start delay referring to the planned start time and not the technical delay from SM36 which is normally caused if not enough background work processes or not enough hardware resources are available. The other thing in common is that the data collector is in a way "satisfied" as soon as it found one corresponding job. So another job with the same name (or same ABAP program) that is run just seconds later will not be monitored!

This is different with the fourth start condition *'use job start condition'*. This can especially be used when the job name (or ABAP or external program) is known and it is known that the job should run on a specific day without knowing the exact start time. With this start condition every job that runs that day and fulfills the job (or program) specifications made will be monitored. Hence when using this start condition one should try to be as restrictive as possible when specifying the jobs, e.g. when specifying the job name with a wildcard '*' one would really monitor every single job running in the system on that specific day! Another difference in comparison to the other start conditions is that no matter whether one maintains a start time or not one will only get a start delay alert for the technical delay as can be seen in transaction SM36. This start delay alert has nothing to do with a deviation from the planned start time.

In addition to the start delay alert, all other possible alerts (end delay, maximum duration, job cancellation, parallel processing, out of time window) can be configured no matter what start condition is chosen.

Periodic job

The job (e.g. a job checking if there are any IDocs that should be processed or Post Goods Issue periodically via background job) starts at a certain point in time and runs for a known period. The job is periodic with a periodicity < 24 hours (e.g. every 2 hours). The start procedure can be one of the three different possibilities 'by time', 'by event or 'use job start condition'. It is mandatory to define a start time.



Due to technical reasons it is not possible to define an 'end delay' alert for periodic jobs. Moreover it is important to know, that in the case of a delayed start, the alerts are always assigned to the last customized and not yet satisfied runtime of the job. If you have several customized jobs still waiting for a fitting job to be run and assigned, the most recent of the jobs is taken. See below for examples of this:

Examples:

1. No job started at 2 pm as was planned. When a job fulfilling the job specifications starts at 3 pm then this will be assigned to the 2 pm job and one might get a start delay alert with a delay of one hour (if configured).

- 2. No job started at 2 pm as was planned. When a job fulfilling the job specifications starts at 3.59 pm then this will be assigned to the 2 pm job as well and you might get a start delay alert with a delay of one hour and 59 minutes (if configured). The 4 pm job data collector would wait for another fitting job to run.
- 3. No job started at 2 pm and 4 pm as was planned. When a job fulfilling the job specifications starts at 4.01 pm then this will be assigned to the 4 pm job and you might get a start delay alert with a delay of one minute (if configured). The 2 pm job entry in the SAP Solution Manager would still wait for another fitting job to run.

Another thing one should be aware of is that the background job monitoring in SAP Solution Manager uses basically the same techniques as the CCMS for scheduling jobs in your SAP system. Hence regarding periodic jobs you find the same functional limitations as with transaction SM36. So is it not possible to say that periodic jobs should be monitored only during special time windows e.g. business hours from 8 am to 6 pm. If one configures the monitoring for periodic jobs, these jobs will be monitored 24 hours a day. This may cause unnecessary alerts in the SAP Solution Manager if one uses an external job scheduler which is able to schedule jobs in such a way: schedule job A every 15 minutes but only from 8 am to 6 pm.

For periodic jobs the following alert types can be used: start delay, maximum duration, job cancellation, parallel processing



It is not possible to monitor an End Delay or Out of Time Window alert for periodic jobs.

Start time of the job is not known

In all cases where the start time is not defined (either because of an external job scheduler or the start time is just not known), the start procedure for the background job to be monitored needs to be defined as 'use job start condition'. This could be a singular job or a periodic job. As soon as a job which fulfills the customized settings is scheduled within the SAP system (SM36, SM37) it is considered for monitoring.

Because the actual start time of the job is not known it makes no sense to define a start time (otherwise another start condition could be used). However it is possible to monitor the technical start delay, i.e. the delay time shown in SM37. The technical start delay is the time spent between the release of a background job and the actual start time (e.g. waiting for a free work process). For getting this technical start delay it is necessary to define an (arbitrary) value as a Planned Start time.



With this configuration an additional start delay alert will be raised at 0:01 the next day when the job has not started at all. This is important when the information is needed every day if the job was running or not.



For jobs monitored via "use job start condition" the following alert types can be used: (technical) start delay, end delay, maximum duration, job cancellation, out of time window (if time window defined)

The above mentioned simple examples give a short introduction to what is (theoretically) possible with the job monitoring functionality in the SAP Solution Manager and should provide some background information to the one alert or the other.

There now follow some business scenarios that should illustrate in which cases a special alert is appropriate and how the monitoring should be setup in specific cases. The described scenarios are only loosely connected with each other.

For the following business scenarios we will work with the fictional company called BizMoni.

Scenario BW process chain for data upload

Key words: CCMS monitor, start by event

For uploading data into Info Cubes the company BizMoni defined a Process Chain RSPC_MON_SOL to run from Monday to Friday.

- BI_PROCESS_TRIGGER: The first job in the chain is scheduled daily at 1:30 am.
- BI_PROCESS_ABAP: The next job prepares some data.
- BI_PROCESS_DROPINDEX: This job then deletes the existing index for the involved Info Cube.
- BI_PROCESS_LOADING: This fourth job actually starts the data upload procedure from the corresponding R/3 system by requesting a dynamic request ID REQU_* and should not normally start after 2am. The data upload job BIREQU_* (which is from a SM37 point of view not directly linked to the Process Chain) usually runs no longer than 4.5 hours so that the fifth and last job of the chain BI_PROCESS_INDEX should not start after 6:30am
- BI_PROCESS_INDEX: This job creates a new index for the involved Info Cube. All four
 previous jobs in the Process Chain are started by Events (where the Event ID is always
 RSPROCESS). The complete Process Chain has to be finished uby 7:30am so that the
 corresponding end-users can work with the latest data from the day before.

The following figure shows how the complete chain appears in transaction RSPC.



Suggested Setup:

As a first step one has to decide whether all Process Chain steps should correspond to one business process step respectively or whether the whole Process Chain is subsumed in just one business process step. This decision has to be evaluated in every specific case and further alternatives are thinkable.

In our scenario we define two different business process steps which are both called "Data upload". One step is located on the connected BW system and contains all five BI_PROCESS_* jobs of the Process Chain. The other step is located on the connected R/3 system and contains the BIREQU_* job for the actual data upload.

Starting with the first and most important business process step on the BW system, we list all jobs belonging to the Process Chain.

	Job Identification Add/Remove ABAP Program											
,												
	ID	Monitoring?	Job Step No.							Job Schedul	е	
		•	1	٦						weekly	Ē	
			BI_PROCESS_ABAP	1	Ē						weekly	Ē
		•	BI_PROCESS_DROPINDEX	1	ē						weekly	Ē
		~	1	٦						weekly	Ē	
		~	1	Ð						weekly	Ē	

The job BI_PROCESS_TRIGGER is the only job of the Process Chain which is scheduled by time, namely at 1:30 am.

Job Details	Start Information
Start Procedure	Planned Start [hhmmss]
by time 🛛 🗎	013000

This job is not uniquely identified. In the situation where several Process Chains are started at the same time it is not possible to be certain that the correct triggering job will be monitored as all the triggering jobs have the same generic job name BI_PROCESS_TRIGGER.

All the following jobs of the Process Chain are triggered by the generic event RSPROCESS with a (unique) event parameter, e.g. in our case for BI_PROCESS_ABAP this event parameter is 2FGKWOXTNGIV761WOX3AVX5GE.

Job Details Start Information Weekly Schedule							
🔊 i 🛗 i 🗵 i	Ð	🔄 🗎 🌐					
Start Procedure		Event ID	Event Parameter				
by event 🛛 🖺		RSPROCESS	2FGKWOXTNGIV761WOX3AVX5GE				

The (unique) event parameter for all the jobs belonging to one Process Chain can be found in the BW system in table RSPCCHAIN (use transaction SE16).

Enter the name of the Process Chain as CHAIN_ID and 'A' as the OBJVERS. Push F8.

Data Browser: Table RSPCCHAIN: Selection Screen						
🕒 🍪 📑 🚹 Number of Entries						
CHAIN_ID	RSPC_MON_SOL	to	₽			
OBJVERS	A	to	\$			
TYPE		to	⇒			
VARIANTE		to	\$			
LNR		to	⇒			

On the next screen all the corresponding jobs are shown with their event parameter in column EVENTP_START.

RBJVERS	ТҮРЕ	VARIANTE	EVENT_START	EVENTP_START
ну А А А А А	ABAP DROPINDEX INDEX LOADING TRIGGER	ZBW_SOURCE1 ZBW_DROP_INDEX ZBW_CREATE_INDEX ZPAK_ELIQID88IST80GFQZSVM9U1KQ ZBW_TRIGGER	RSPROCESS RSPROCESS RSPROCESS RSPROCESS	2FGKWOXTNGIV761WOX3AVX5GE BEEIHANVHL26YDV142DXK5B90 90LZRCSEIAFWAE11VQVAB9XE0 0WMTTL1DCL2I0FG90Z0CWCT5Q

If you are interested in the information about which was the preceding job that triggered the event you find the information in the column EVENTP_GREEN.

OBJVERS	ТҮРЕ	VARIANTE	EVENT_GREEN	EVENTP_GREEN
A A A	ABAP DROPINDEX INDEX	ZBW_SOURCE1 ZBW_DROP_INDEX ZBW_CREATE_INDEX	RSPROCESS RSPROCESS	BEEIHANVHL26YDV14ZDXK5B90 0WMTTL1DCL2I0F690Z0CWCT5Q
A A	LOADING TRIGGER	ZPAK_ELIQID88IST80GFQZSVM9U1KQ ZBW_TRIGGER	RSPROCESS RSPROCESS	90LZRCSEIAFWAE11VQVAB9XE0 2FGKWOXTNGIV761WOX3AVX5GE

Alternatively you can find out about the different event parameters step by step. Go to transaction RSPC and mark the process chain that should be monitored. Push the button for displaying logs and choose a "Date Selection", e.g. "Today".



Choose one date and time and click on the process chain step you want to configure within the monitoring. On the following pop-up you go to tab strip "Chain". There you find the information regarding on which event parameter the chain step is started and which is the subsequent event parameter.

① 08:55:51	A5R Program ZBW_SOURCE1	
Process Back	CE1 kg Chain	
Start On 11.11.2005 After Event With Parameter	At 08:55:53 Time and 539 MS RSPROCESS 2FGKWOXTNGIV761WOX3AVX5GE	
Generated Instance	/	-
Type Variant Instance	ABAP Program ZBW_SOURCE1 1PQL3VMULAAYE66KP4D79XHFU	
End On 11.11.2005 With Status	At 08:55:58 O'CIk and 39 MS Completed	
Target Event With Parameter	RSPROCESS BEEIHANVHL26YDV14ZDXK5B90	

Continuing with the monitoring setup we mark the days from Monday to Friday in tab strip *Weekly Schedule.*

_	Job Details 🔓 Start Information 🦯 Weekly Schedule								
	Mo Tu We Th Fr Sa Su Period (min								
	•	~	•	~	•				

For the alert setup we configure a *Start Delay* for the triggering job BI_PROCESS_TRIGGER in order to be informed whether the Process Chain started on time or not.

Start Delay End	Delay 🛛 🗍 Out of Time Wi
Start Delay for YELLOW	Start Delay for RED Unit
5	30 min

The triggering job is not uniquely identified as already mentioned above. If several Process Chains are started at the same time it is not certain that the correct triggering job is monitored as all the triggering jobs get the same generic job name BI_PROCESS_TRIGGER. This means that in the end we could monitor the wrong job.

For all jobs we configure the Job Cancellation alert in order to know at which point in the job chain an error occurred, should this be the case.

Start Delay End Delay	Out of Time Window Maximum Duration Job Cancellation
Alert If Job Cancelled	
Red 🛅	

For the two jobs BI_PROCESS_ABAP and BI_PROCESS_DROPINDEX no further alerts are specified.



A Maximum Duration alert is not meaningful for the job BI_PROCESS_LOADING as this job only triggers the upload processing. The runtime of this job correlates in no way with the actual processing time for the data upload.

For the jobs BI_PROCESS_LOADING and BI_PROCESS_INDEX we want to define a *Start Delay* alert respectively. The *Start Delay* alert for BI_PROCESS_INDEX is of particular importance as this will tell us implicitly the runtime of the data upload.

For this purpose we have chosen a special configuration as *Start Information*. We will consider BI_PROCESS_INDEX as an example. The same should be configured for BI_PROCESS_LOADING just with other thresholds.

In order that we have a unique identification for the job we customize the *Start Procedure* as "by event" and enter the corresponding *Event ID* and *Event Parameter* as already described above. Additionally, we enter a *Planned Start* time. We want to customize a planned start time that lies definitely before the actual start time of the job so that we do not have any problems with the monitoring functionality. Hence, we choose the planned start time of the complete Process Chain as reference, i.e. at 1:30 am.

	Job Details	Start Information	Veekly Schedule							
Start Procedure		Planned Start [hhmmss]	Event ID	Event Parameter						
	by event 🛛 🖺	013000	RSPROCESS	90LZRCSEIAFWAE11VQVAB9XE0						

In the *Start Delay* alert configuration we customize the thresholds relative to the overall *Planned Start* time at 1:30 am. After 4 hours (240 minutes) we want to get a Yellow alert at 5:30 am. At 6:30 am, if the job BI_PROCESS_INDEX did not start after 5 hours (300 minutes), a Red alert should be raised.

Start Delay 📔 En	d Delay 🛛 🚪 Out of Time Win					
Start Delay for YELLOW	Start Delay for RED Unit					
240	300 min					

While the configuration of the Start Delay alert for the job BI_PROCESS_INDEX already covers all the important information for our monitoring we set up a second business process step for getting some additional information about the data upload job itself.

Hence, for this second business process step on the R/3 system we configure only the actual data upload job. For this, we specify the job name prefix BIREQU_*.

Job Identification Add/Remove ABAP Program												
Ø			🕒 🗈 🖽									
ID	Monitoring?	Job Name	Job Step No.							Job Sci	nedu	Jle
	>	BIREQU_*	1	٦						weekly		Ē

As the job name is generic and the request number is dynamically generated during the runtime it cannot be specified in advance. This might lead to the situation that a wrong job is monitored.

For the *Job Details* we give a speaking name and specify the user under which the job is running – in our case the generic user BWALEREMOTE. In case that different BIREQU* jobs are running in the system under different users this will make it more likely that the right job is monitored.

Job Details Start Information Weekly Schedule									
Custom Identifier	Critical Job	User	Parallel Jobs?						
Data upload		BWALEREMOTE	No	Ē					

The *Start Information* is now the crucial point for identifying the correct BIREQU* job. This job is scheduled dynamically via some internal BW processing. Hence, the *Start Procedures* "by event" and "by preceding job" are of no use. With the *Start Procedure* "use job start condition" our BIREQU* job of interest would be definitely monitored but every other BIREQU* job in the R/3 system would also be monitored and assigned to the same business process step, which is inappropriate.

So the only appropriate *Start Procedure* is "by time" where only one job per day (if not periodic) is monitored. In order to identify our job of interest the customized start time (which is actually not known and depending on the preceding jobs to be processed beforehand) should be as close as possible to the actual start time. So in a first guess the actual start time lies between 1:30 am (when the triggering job of the Process Chain is started) and 2:00 am due to the scenario description given above.

As a job can be only identified for monitoring if the customized start time lies before the actual start time it is probably inadequate to chose 2:00 am as a start time. Experience showed that the two jobs BI_PROCESS_ABAP and BI_PROCESS_DROPINDEX always had a combined runtime of at least 15 minutes. Hence we chose 1:45 am as a planned start time for BIREQU*.

Job Details / Star	t Information 🛛 🍸 Weekly :
🗠 🛗 🗵 🗈 🛃	
Start Procedure	Planned Start [hhmmss]
by time 👔	014500
by time	
by event	N
by preceding job	42
use job start condition	



Even with this setup we can't be sure that we monitor the correct job. In case that another BIREQU* job under the same user BWALEREMOTE is starting just between 1:45 am and the actual start time of the job we are interested in then the wrong BIREQU* job would be monitored.

On the Weekly Schedule tab strip we mark all days from Monday to Friday.

As alerts we configure a *Maximum Duration* alert. After a runtime of 3.5 hours (210 minutes) a Yellow alert should be raised and after 4.5 hours (270 minutes) a Red alert should be raised.

_	Start Delay En	d Delay 🛛 👘 Out of	Time	Window	Maximum Duration					
	🔊 i 🛗 i 🗵 🖬 i 🗳	<u>b</u> 🗉 🖽								
	Duration for YELLOW	Duration for RED	Unit							
	210	270	min							

Additionally, a red alert should be raised every time the BIREQU* job cancels.

Additional Setup:

As we described above that there might be the possibility that the wrong BI_PROCESS_TRIGGER job is monitored it is also possible to make use of some standard CCMS functionality to further support your monitoring efforts. The standard BW monitor can be directly found via transaction BWCCMS or under RZ20 \rightarrow SAP BW Monitors \rightarrow BW Monitor $\rightarrow \langle SID \rangle \rightarrow$ Process Chains. This monitor observes every Process Chain in the system automatically and raises an alert if any of the chain steps cancel.

On business process step level chose *Other CCMS Monitor* as an additional *Monitoring Type*.

Monitoring Types	📮 De:
	ē 🌐
Monitoring Type	Rele
Application Monitor	
Background Job	
Dialog Performance	
Update Error	
Due List Log	
Application Log	
Document Volume	
Other CCMS Monitor	v

On the occurring sub-node one can enter an own-defined *Monitoring Name* and the complete *MTE* (*Monitoring Tree Element*) via copy and paste.



The complete name of the MTE can be found in the local RZ20 of the system to be monitored. Mark the name of the MTE with the name of the corresponding Process Chain and then go to *Edit* \rightarrow *Nodes (MTE)* \rightarrow *Display MTE description*.

<u>M</u> onitor <u>I</u>	<u>E</u> dit <u>G</u> oto V <u>i</u> ews Extr <u>a</u> s	System	n <u>H</u> elp				
Ø	Tree	۲	: ۲۵ ایکل 🛍 🗳 ایک	🎦 🗘 🕄 🛛 🗮			
SAP BV	<u>S</u> elections	+	r) - Maintenance	functions			
	<u>N</u> odes (MTE)		Display de <u>t</u> ails	Shift+F6			
🔁 🖬 🛛	Alerts	×	Di <u>s</u> play MTE description	F1			
	Show node (MT <u>E</u>) for aler	t	Log on to R/3 System	Ctrl+Shift+F1			
۸.	<u>F</u> ind	Ctrl+F	Analyze <u>R</u> /3 System	Ctrl+Shift+F2			
	Find ne <u>x</u> t	Ctrl+G					
BW Moni ⁻	C <u>a</u> ncel	F12					
	OMB : System cannot be (85 : System cannot be DLB	accesse accesse	d 141. d 141.				

Copy the MTE name of the occurring pop-up window. For further information on how to use the monitor *Other CCMS Monitor* refer to the document "Business Process Monitoring – Setup Guide" which can be found in the Media Library on the SAP Marketplace under Quick Link /BPM.



As the MTE chosen for monitoring is not a leaf node on lowest level but a branch node in RZ20 it is only possible to get the corresponding alert information in the Business Process Monitoring graphic of the SAP Solution Manager. It is not possible to work with these alerts in the Business Process Monitoring session (e.g. confirm alerts, create Service Desk messages etc.) For that, you would have to insert each of the leaf nodes separately which is not possible as they are created dynamically during the runtime of the respective process chain.

Problem with more complex process chains:

It might happen that the event parameter is not unique as stated above. This happens in the case that one job triggers several jobs at the same time, e.g. the following chain (view from transaction RSPC) where three load jobs are started in parallel.



In table RSPCCHAIN you see that all three jobs are started by the same event parameter 248YORR0B8HXL2TRL6IKQQPQU.

OBJVERS	ТҮРЕ	VARIANTE	EVENT_START	EVENTP_START
A A A A	ABAP AND AND AND AND AND	PM_ROLLBACK EVEN_74FLTKYLIPD15UBMLIWMRBPGE EVEN_74FLTKYLIPD15UBMLIWMRBPGE EVEN_74FLTKYLIPD15UBMLIWMRBPGE EVEN_8BFQWCGZORXFYT7J3FGII3C30	RSPROCESS RSPROCESS RSPROCESS RSPROCESS RSPROCESS	96WB6IN00A1PYI2Y86JHTMUWM 903H0HC5AEOHJ97AA1W68YZMU 40UU06J3D82133QY88E09XBME 84Q7LXT1C9RBEFZ6PEET8BW1H 87NLS6SHLKZ3KTJ7CSYTGFEDP
A A A A	AND AND LOADING LOADING LOADING	EVEN_8BFQWCGZORXFYT7J3FGII3C30 EVEN_8BFQWCGZORXFYT7J3FGII3C30 ZPAK_2HKHG2UQY02H77I75LV0NRUIB ZPAK_30WTZ9Y6CHMAD0DJA06PHP0KI ZPAK_E0XEQT6EXGXC6V0E0N2DLJ786	RSPROCESS RSPROCESS RSPROCESS RSPROCESS RSPROCESS	BR8905V18VVTH5XJ9TB7PU40H EKZ2SA6MSE27468TAVJPKHMEH 248YORR0B8HXL2TRL6IKQQPQU 248YORR0B8HXL2TRL6IKQQPQU 248YORR0B8HXL2TRL6IKQQPQU
A A A	ODSACTIVAT ODSPROCESS TRIGGER	PM_ACTIVATE_ODS PM_UPDATE_TARGETS PM_START	RSPROCESS	78ZETCBQFXBUX7Q646IQ8M30F 3I2QJW2EKH2NL966N3X07LYJK

In such a case you configure only one job for monitoring BI_PROCESS_LOADING and in the "Job Details" you specify in column "Parallel Jobs?" the entry "Yes, number known".

Job Details 🛛 S	tart Informatior	n 🛛 W	eekly Schedule						
Custom Identifier	Critical Job	User	Parallel Jobs?						
bi_process_loading		*	Yes, number known 🛭 🗈						

On alert level the "No. of Para. Jobs" is then set to 3 in the tab strip "Parallel Processing". Please have a look on the Scenario "Billing Run in five parallel tasks" for more information on this alert type.



With this configuration all three jobs are evaluated individually for all alert types which are configured.

Scenario Demand planning with SAP APO

Key words: monthly job, Factory calendar, Application Monitor APO DP log, start after system event, add ABAP program

The company BizMoni creates their monthly demand plan with the help of an SAP APO system on every last Sunday of the corresponding month starting after an event that is triggered after a successful system startup. This is usually at around 3am. The job is expected to run for about 4 hours and should be definitely finished before the users come onto the system at 8am. The underlying ABAP report is the /SAPAPO/TS_BATCH_RUN. Within transaction /SAPAPO/MC8D one defined a job variant with the job number DEM_PLAN. The generation of the corresponding log (could be analyzed with transaction /SAPAPO/MC8K) was flagged during the definition.

Suggested Setup:

As a first step, the *ABAP Program* and its corresponding *Variant* are specified. The *Job Name* has to stay blank as we assume that it is not known. The ABAP Program has to be chosen via the F4 value help which is filled from the entries made in the Business Process Maintenance under Operations Setup \rightarrow Solution Landscape \rightarrow Solution Landscape Maintenance. In case the value help is empty one can add an ABAP Program in the second tab strip *Add/Remove ABAP Program*. After saving this newly added entry the F4 help on the first tab strip *Job identification* is filled. It is important to fill-in the correct *Job Step No.* of the ABAP Program/Variant combination (can be usually identified in the Job Log). If this field is not specified the value "1" is entered per default while saving. A "monthly" *Job Schedule* is chosen.

4	← → I + next open check											
	Job Identification Add/Remove ABAP Program											
	ᡌ			2 I I I I I I I								
	ID	Monitoring?	Job Name	Job Step No.	ABAP Program		Variant			 	Job Schec	lule
		<		1	/SAPAPO/TS_BATCH_RUN	R	DEM_PLAN				monthly	Ē
					/SAPAPO/TS_BATCH_RUN	Ν						Ē

In column *Custom Identifier* it is possible to give a speaking name. Per default, the specified *Job Name* or *ABAP Program* is pre-filled as *Custom identifier*. The job is marked as *Critical Job* in terms that it is wished to get an alert within 5 minutes at the latest. No specific *User* is specified and there is no parallel processing.

CQAutomatic (Done) ✔Do	ne 🛛 🚰 Cheo	sk Job Identificat	ion					
🗢 🔿 📮+ next open check								
Job Details Start In	Job Details Start Information Monthly Schedule							
Custom Identifier	Critical Job	User	Parallel Jobs?					
Monthly demand planning		*	No	Ē				

"by event" is chosen as a *Start Procedure* and the corresponding system event is specified in column *Event_ID*. No specific *Event Parameter* is defined in this case.

Job Details Start Information Monthly Schedule						
🔊 🛗 💈						
Start Proce Event ID Event Parame						
by event 🔳	SAP_SYSTEM_START					

Events can be displayed and maintained via transaction SM62. With transaction SM64 it is possible to trigger events manually.

In order to define that this job only runs every last Sunday of the month a special Factory Calendar has to be defined in the IMG (General Settings \rightarrow Maintain calendar) of the system to be monitored, e.g. one could define a calendar OS (abbreviation for Only Sundays) which contains only Sundays as working days.

Hence, on the tab strip *Monthly Schedule* the *Factory Calendar ID* of the newly defined or already existing calendar has to be maintained. In the defined calendar there will be 4-5 Sundays per month of which only the last one is of interest for our monitoring. Therefore, we chose as *Working Day* "1" (i.e. the first working day of the chosen calendar) and *relative to month's* "end" (i.e. counting the calendar days backwards per month). In column 1st *Month* the month should be entered when the monitoring should start for the first time. The next column *Period [# Months]* specifies the monthly period when the job is executed. As our job is executed on a monthly basis the period is "1".

While theoretically every figure between 1 and 12 could be entered as a period it is usually only meaningful to have a period which is a divider of 12, i.e. 1,2,3,4,6,12.

Clinication Image: Check Job Identification							
← → I 📙+ next open check							
Job Details Start Information Monthly Schedule							
	1						
Factory Calendar ID	Working Day	relative to month's		1 st Month	Period (# Months)		
OS 1 end 🗈 August 🗈 1							
		beginning end					

In case no calendar is maintained in column *Factory Calendar ID* then every single day of the month is considered as a working day.

One business requirement was that the job should be definitely finished at 8am before the end-users log on to the system. Hence, an *End Delay* is configured with a *Planned End* time of 7am. After 30 minutes, i.e. at 7:30 am the background job will be valuated yellow if it has not finished yet. At 7:50am a red alert will be raised. These alerts could be raised with a time delay of up to 5 minutes, i.e. the default collector frequency for critical jobs. If the job was marked as not critical the time delay could be possibly up to 60 minutes per default!

Start Delay End I	Delay Out of Tin	ne Wi	indow 🖌 Maximum Dura			
End Delay for YELLOW	End Delay for RED	Unit	Planned End (hhmmss)			
30	50	min	070000			

Additionally, the *Maximum Duration* alert is configured as it is expected that the normal runtime is about 4 hours. After 4 hours (i.e. 240 minutes) a Yellow and after 5 hours (equal to 300 minutes) a red alert is raised.

Start Delay End	i Delay 🚪 Out of Ti	me Window 🗡	Maximum Duration				
Duration for YELLOW	Duration for RED	Unit					
240	300	min					

Additional setup:

The end-users that have to evaluate the log file with status information of about 20.000 characteristic combinations ask for a possibility of an automatic evaluation. Therefore, one can use the additional Application Monitor "APO DP log". How this monitor is set up is described in the "Application Monitoring and User Exit – Setup Guide" which can be found in the Media Library on the SAP Service Marketplace under Quick Link /BPM. The necessary information one needs is the job number DEM_PLAN and the relevant message type, message ID and message number one wants to evaluate.

_

womitioning Types	📮 Descrij			
Monitoring Type	Relevant?			
Application Monitor	~			
Background Job	~			
		_		
🏽 🕄 Automatic (Done) 🛛 🖌 🖉 Done 🔢	🛐 BLA: Reload Ap	p Mon Objects		
□ 🔿 🛛 🖸 + next open check		Intions 📳		
⇒ →	ED 6 12	Dptions		
Application Monitoring Objects		Options		
Application Monitoring Objects		Dptions		
Application Monitoring Objects		Dptions		
Application Monitoring Objects	Compone	Dptions I	ST-A/	Techn. Nan
Application Monitoring Objects	Compone Compone	Dptions I	ST-A/ 01F	Techn. Nan APAPOLOG

Scenario MRP \rightarrow event \rightarrow BOP

Key words: internal parallelization, Application Monitor MRP, start by preceding job, only Red alerts

The company BizMoni uses an R/3 Enterprise Edition for a daily Material Requirements Planning (MRP) run for plant 0001 starting at 1am. From Monday to Saturday an MRP job with the name PP_MRP_0001_D is scheduled with processing key NETPL. For consistency reasons, on Sunday a job with the name PP_MRP_0001_W is scheduled which uses the processing key NETCH. The MRP runs are internally 4-fold parallelized. The runtime for the daily MRP run should be no longer than 4 hours. The MRP run on Sunday might be longer. As soon as the MRP run finishes an event is triggered so that a subsequent Backorder processing (BOP) job is started which performs a complete rescheduling under the restriction of material availability. The succeeding job is always (Monday to Sunday) called SD_BOP_0001_D and has a normal runtime of up to 2 hours. The Backorder processing should be finished before the users come onto the system, i.e. until 8am at the latest. As there is no online work on Sundays there are no time restrictions on this day and superfluous alerts should be avoided. The only information provided on Sunday should be if the job cancelled or not.

As the given time window for the MRP and BOP processing is at most 7 hours (from 1am to 8am) and it might happen that the total runtime of the job chain might add up to 6 hours (4 hours for the MRP and 2 hours for the BOP) there should be some alarming in case that the MRP didn't start before 2am.

When it comes to scheduling these jobs in SM36, it is not possible to realize this scenario appropriately with the use of the *Start condition* "After job" as this start condition refers only to exactly one preceding job and can't be used with periodically scheduled jobs. Hence the *Start condition* "After job" is of very limited practical use for scheduling and therefore the same applies for the *Start condition* "by preceding job" in the monitoring area. Instead, one can use the workaround where the MRP job consists of two job steps: one is the actual MRP run and the second executes a short ABAP report which raises a specific event that triggers then the BOP run.

Use function module BP_EVENT_RAISE to trigger an event from an ABAP program.

Example

```
* Report processing before triggering event...
*
* Trigger event to start background jobs waiting for the event.
*
DATA: EVENTID LIKE TETCJOB-EVENTID.
DATA: EVENTPARM LIKE TETCJOB-EVENTPARM.
EVENTID = 'SP_TEST_EVENT'. " Event name must be defined
" with transaction SM62.
EVENTPARM = 'EVENT1'. " Optional: a job can be
" scheduled to wait for an
" EVENTID or combination of
" EVENTID and EVENTPARM.
CALL FUNCTION 'BP_EVENT_RAISE' " Event is triggered. Jobs
© 2005 SAP AG
```

```
EXPORTING " waiting for event will be
EVENTID = EVENTID " started.
EVENTPARM = EVENTPARM
TARGET_INSTANCE = ` ` " Instance at which an event
" should be processed. Can
" generally be omitted.
EXCEPTIONS OTHERS = 1. " Exceptions include event not
" defined, no EVENTID
" exported, etc.
```

Suggested Setup:

Within the first business process step "Material Requirement Planning" we would set up both MRP runs – the one from Monday to Saturday and the one only executed on Sundays.

Job Identification Add/Remove ABAP Program								
ᡌ		🛃 🛃 I 🛗 I 🗵	🖻 🛃 🗎 🗄					
ID	Monit	Job Name	Job Step No.					 Job Schedule
1	✓	PP_MRP_0001_D	1	Ð				weekly 🖺
2	 Image: A start of the start of	PP_MRP_0001_W	1	ē				weekly 👔

The first job (Mon-Sat) would be marked as time critical so that the data collector checks for the job every five minutes. Although the MRP run is parallelized we chose "No" in column *Parallel Jobs*. The reason is that the MRP is **parallelized internally**, i.e. the parallel tasks are processed in dialog work processes via RFC calls. The column *Parallel Jobs* refers to **external parallelization**, i.e. all parallel tasks are processed as separate background jobs all running in background work processes.

Job Details	Start Informa	tion 🛛 Weel	dy Schedule	
🔊 🛗 🖾 🗈 🛛	🖪 🛯 🌐			
Custom Identifier	Critical Job	User	Parallel Jobs	?
MRP during week	~	*	No	Ð

The Start Procedure is "by time" as the MRP run is scheduled to start at 1am every day.

_	Job Details	Start Information 🛛 🍸 Weekly Sche
	Start Procedure	Planned Start [hhmmss] 🛛
	by time 🛛 🖹	010000

This first configured MRP run is executed from Monday to Saturday.

ſ	Job Details Start Information Weekly Schedule							
	Мо	Tu	We	Th	Fr	Sa	Su	
		~	~	~	~	•		

We configure some *Start Delay* alert for the PP_MRP_0001_D job so that we get a yellow alert if the MRP did not start before 1:30am and a red alert if the MRP did not even start at 2am.



The *Start Delay* alerts always refer to the *Planned Start* time configured as *Start Information* in the monitoring setup.



If the runtime of the MRP exceeds 200 minutes we want to get a first warning. After 240 minutes (4 hours) a red alert should be raised.

Start Delay En	d Delay 📔 Out of	Time	Window	Maximum Duration	
Duration for YELLOW	Duration for RED	Unit			
200	240	min			

Additionally, every job cancellation should result in a red alert.

Start Delay End Delay	🖌 Out of Time Window 📔 Maximum Duration 🦯 Job Cancellation
Alert If Job Cancelled	
Red 🔳	

The MRP run PP_MRP_0001_W on Sunday is far less critical from a business point of view as it won't interfere with any dialog users. Hence, it is sufficient if the data collector runs every 60 minutes (job not marked as critical).

Job Details	Start Inform	ekly Schedule					
Custom Identifier	Critical Job	User	Parallel Jobs?				
MRP on Sunday		*	No 🖹				

The *Start Information* would be just the same as mentioned above for PP_MRP_0001_D. In column *Weekly Schedule* only Sunday is marked.

The only alert which would be configured is the *Job Cancellation* as shown above for the daily job.

Within the second business process step "Backorder Processing" we would customize the Job Name SD_BOP_0001_D twice for monitoring as the business asks for different alert valuation between Monday and Saturday on the one hand and Sunday on the other.

Job Identification Add/Remove ABAP Program											
	ID	Monit	Job Name	Job Step No.					Job Schedule		
	7	~	SD_BOP_0001_D	1	Ē)				weekly 🖺		
		✓	SD_BOP_0001_D	1	Ē)				weekly 🖺		

The first job (Mon-Sat) would be marked as time critical so that the data collector checks for the job every five minutes.

Job Details	Start Informa	ition 👔	Weekly Sched
	🖪 🛯 🌐		
Custom Identifier	Critical Job	User	Parallel Jobs?
BOP during week	 Image: A start of the start of	*	No 🖹

The *Start Procedure* is specified as "by event" where the *Event ID* is Z_START_BOP (to be maintained in SM62) with *Event Parameter* DAILY which is triggered in the second job step of the daily MRP run PP_MRP_0001_D with processing key NETPL.

	Job Details	Weekly Schedule						
🐼 🛗 🗵 🗈 🖽								
	Start Procedure		Event ID	Event Parameter				
	by event 🛛 🖹		Z_START_BOP	DAILY				

This first configured BOP run is executed from Monday to Saturday.

Job Details Start Information Weekly Schedule								
~	<i>N</i>							
∞ 🛱		🖪 i	∎ ∎	1				
Mo Tu	We Th	Fr	Sa	Su				
	••		◄					

As the BOP run should be finished before 8am we configure an *End Delay* alert. The *Planned End* is at 6:30am. Half an hour later at 7am a yellow alert would be raised and again 50 minutes later at 7:50am a red alert would be raised.

Start Delay End [Delay 🛛 Out of Tin	ne Wi	ndow 🍸 Maximum Dura
🗠 i 🛗 i 🔀 🗈 i 🛃			
End Delay for YELLOW	End Delay for RED	Unit	Planned End [hhmmss]
30	80	min	063000

As stated above the normal runtime shouldn't be longer than 2 hours. As we are not interested in a yellow alert in this case we choose the same threshold for a yellow and red alert. Hence, after a runtime of more than 120 minutes a red alert will be raised.

_	Start Delay En	d Delay 🛛 🖌 Out of Ti	ime Window	Maximum Duration						
	Duration for YELLOW	Duration for RED	Unit							
	120	120	min							

Additionally, any *Job Cancellation* will be evaluated red (as already described in other scenarios above).

The second job SD_BOP_0001_D (Sun) would not be marked as time critical as it is sufficient for the data collector to check for the job every 60 minutes.

Job Details Start Information Weekly Schedule							
Custom Identifier	Critical Job	User	Parallel Jobs?				
BOP on Sunday		*	No 🖹				

The *Start Procedure* is also specified as "by event" but differs as the *Event Parameter* is now WEEKLY which is triggered in the second job step of the weekly MRP run PP_MRP_0001_W with processing key NETCH.

Job Details Start Information Weekly Schedu						
	0 B 2 E		la 1 🔠			
	Start Procedure		Event ID	Event Parameter		
	by event 🛛 🖺		Z_START_BOP	WEEKLY		

This job is only executed on Sunday.

_	Job Details Start Information Weekly Schedule								
	ᡌ	間	IΣ			ð 🗄			
	Мо	Tu	We	Th	Fr	Sa	Su		

So from a business point of view it is only interesting whether the job cancelled on Sunday or not.

Start Delay	End Delay	Out of Time Window	K Maximum Duration	Job Cancellation
Alert If Job Cance	lled			
Red	Ē			

Additional setup:

The end-users ask for a possibility of an automatic evaluation of exceptions during the MRP run. Therefore, one can use the additional Application Monitoring "MRP Key Figures Dispo.list". How this monitor is setup is described in the "Application Monitoring and User Exit - Setup Guide" which can be found in the Media Library on the SAP Service Marketplace under Quick Link /BPM.

Monitoring Types	📮 Descri	
	ē 🎛	
Monitoring Type	Relevant?	
Application Monitor		
Background Job	•	

App	olication Monitoring Objects				
Monit	Monitoring Object	Component	Comp. Description	ST-A/PI	Techn. Name
	Dialog Performance MolMonitoring Object	BC	Basis Components	01F	BOPERFMO
	Appl. Monitoring: Due List Log	SD		01F	R3DUELOG
	Appl. Monitoring: MRP Key Figures Dispollist	PP-MRP	Material Requirements Planning	01F	R3PP_MRP

The necessary information needed for the setup is the *Plant*, the *MRP controller(User)* and the *Exception group* to be monitored.

Scenario: Mass creation of deliveries

Key words: unknown number of parallel tasks, external scheduler, use job start condition and Out of time window, due list log of type D, only Yellow alerts

At the end of the working day the company BizMoni starts a mass processing for the creation of deliveries every day from Monday to Friday at around 8pm. The exact starting time of the delivery creation is not known and is depending on the dynamic scheduling of the external Redwood Cronacle scheduler, although it should not start before 7:30pm when the online users stop creating sales orders. Several jobs with the prefix SD_DELCREAT_0001_D* are started in parallel but the exact number of parallel jobs is not known exactly as it varies with the different volumes per day. Usually, the whole process of creating deliveries lasts about 2 hours. It must be finished until 11pm at the latest as this is the time when the mass processing of the billing run starts which could lead to a hardware bottleneck.

Suggested Setup:

The different jobs use the same prefix SD_DELCREAT_0001_D so that we can make use of the wildcard functionality within the *Job Identification* of our business process step "Mass creation of deliveries".

	Job Identifica	tion 🛛 🖌 Add/Remove ABAF	Program			
ᡌ		📕 🛗 🗵 IB 🛃 I				
ID	Monitoring?	Job Name	Job Step No.		 Job Schedu	Jle
	~	SD_DELCREAT_0001_D*	1	ē 🛛	weekly	Ē

In column *Custom Identifier* we specify a speaking name and flag the column *Critical Job*. As we know that in our case every job that is scheduled with the external scheduler runs under the same *User* we specify this User explicitly. As we further know that several jobs should run in parallel for the delivery creation but don't know the exact number due to daily difference in the processed volume we chose "Yes, but number unknown" in column *Parallel Jobs*. This will evaluate if more than one job with the name prefix SD_DELCREAT_0001_D started in parallel within a two minute tolerance time frame.

_	Job Details Start Information Weekly Schedule						
	Custom Identifier	Critical Job	User	Parallel Jobs?			
	Delivery creation	~	CRONACLE	Yes, but number unknown 🗈			
		No					
		Yes, number known					
				Yes, but number unknown			

Since the exact start time is not known, the *Start Procedure* "use job start condition" is chosen so that every job that fulfills the *Job Identification*, *User* and *Weekly Schedule*

customizing will be monitored. Additionally, a *Planned Start* time is configured which might lead to some misconception as in this case the time is chosen to be the same as the really expected start time of the jobs. But every arbitrarily chosen start time could be entered with the same effect.

Job Details Start I	nformation 🛛 🗌 Weekly Sc
Start Procedure	Planned Start [hhmmss]
use job start condition 🔳	200000
by time	
by event	
by preceding job	
use job start condition	

Because the actual start time of the job is not known it makes no sense to define a start time (otherwise another start condition could be used). However, it is possible to monitor the technical start delay, i.e. the delay time shown in SM37. The technical start delay is the time spent between the release of a background job and the actual start time (e.g. time spent waiting for a free work process). For getting this technical delay it is necessary to define an (arbitrary) value for a start delay.

With this configuration an additional *Start Delay* alert will be raised at 0:01 the next day when the job has not started at all. This is important when every day the information if the job was running or not is needed.

On the Weekly Schedule tab strip we would flag all days from Monday to Friday.

Hence, after the remark from above we configure a pro forma *Start Delay* with arbitrary values in order to get the information about a technical delay and whether the job started at all on that day.



This information is most applicable if the corresponding job is running just shortly before 12pm. Otherwise the information if the job run at all or not might come far too late just after a few hours.

Start Delay End I	Delay 👖 Out of Time	e Wini		
Start Delay for YELLOW	Start Delay for RED	Unit		
1	2	min		

Additionally we configure an *Out of Time Window* alert. This gives the opportunity to ensure on the one hand that one is informed in case the job doesn't finish in time (which makes then an *End Delay* obsolete) and on the other hand one is informed that the job started too early (before 7:30pm when the online users are still working and perhaps still changing documents).

Start Delay 📔 End Delay	Out of Time Window	K	Maximum Duration 🍸 Pa	rallel Processing 🍸 Job			
Out of Time Win for YELLOW	Out of Time Win for RED	Unit	Time Win. Start [hhmmss]	Time Win. End [hhmmss]			
30	55	min	200000	220000			

With this configuration a yellow alert is raised either if the jobs with prefix SD_DELCREAT_0001_D started before 7:30pm or if they didn't finish after 10:30pm. Accordingly, a red alert is raised if the jobs started before 7:05pm or if they didn't finish after 10:55pm.

Alerts won't be raised for the *Out of Time Window* alert if the corresponding job did not run at all or the job started before a customized start time, i.e. the start time defined as start condition for the job NOT the start time of the time window. This is especially relevant for the start procedure "by time" which we don't use here.

As another alert we configure a *Maximum Duration* alert. This should only provide a yellow but no red alert as we rely more on the *Out of Time Window* alert in this specific scenario. It can be achieved by entering a meaningful threshold for a yellow alert and an arbitrary high threshold for a red alert.

Start Delay En	d Delay 🛛 Out of	Time	e Window	Maximum Duration
Duration for YELLOW	Duration for RED	Unit		
120	999999	min		

In tab strip *Parallel Processing* one decides if a yellow or red alert should be raised in case that just one of the supposedly parallel jobs with prefix SD_DELCREAT_0001_D started.

Start Delay 📔 End Delay 👘	Out of Time Window 🍸 Maximum Duration 🦯 Parallel Processing 👘
Alert for Wrong Parallel Processing	
Yellow 🗈	
Red	
Yellow	

It is not possible to monitor in general if certain (different) background jobs are running at the same time. Jobs are considered to be started in parallel if they start within 2 minutes tolerance.

Additionally we would customize a red alert for every job cancellation as described in the other scenarios.

Additional setup:

The end-users ask for a possibility of an automatic evaluation of exceptions/errors during the delivery creation. Therefore, one can use the *Due List Log* monitoring functionality. How this monitor is setup is described in the "Business Process Monitoring - Setup Guide" which can be found in the Media Library on the SAP Service Marketplace under Quick Link /BPM.

Monitoring Types	🔁 Descrip	1			
	1		Due List I	Log	
Monitoring Type	Relevant?			2000	
Application Monitor			DI Type	User	Aggregation
Background Job	 Image: A set of the set of the		Delivery 🖹	*	Aggregation
Dialog Performance			Denvery		nerlog
Update Error					per day
Due List Log	 Image: A start of the start of				per hour

Scenario Billing run in five parallel tasks

Key words: known number of parallel tasks, due list log of type F, workflow notification

Every day at 11pm five parallel background jobs with prefix SD_BILL_0001_D are scheduled via SM36. The runtime is not seen as so important as it never happened that the whole billing run ended after 5am the very next day which is fine for the company BizMoni from their business point of view. Nevertheless, one would like to check for a *Start Delay*. For an efficient error-handling it is of utmost importance that the person responsible for the job scheduling and monitoring gets an email in case that one of the jobs cancels or less than five jobs are starting to run.

Suggested Setup:

We use the common prefix together with a wildcard '*' to identify the parallel billing jobs.



We enter a speaking name as *Custom Identifier* and do not mark this job as critical so that the data collector will run per default only every 60 minutes instead of every 5 minutes. In column *Parallel Jobs* we chose the entry "Yes, number known".

Job Details Start Information Weekly Schedule						
Custom Identifier	Parallel Jobs?					
Parallelized billing run	Parallelized billing run 🔄 * Yes, number known					
	No					
Yes, number known						
			Yes, but number unknow			

The scenario description said that the job is started every day at 11pm from Monday to Friday.

Job Details Start Information W	Job Details Start Information Weekly Schedule
Start Procedure Planned Start [hhmmss]	Mo Tu We Th Fr Sa Su
by time 🗈 230000	

The interesting alert is the *Parallel Processing* where we enter the exact number of parallel jobs that we expect to run at the same time and the corresponding alert that should be raised in case that more or less than 5 jobs with this prefix SD_BILL_0001_D were started in parallel.

Start Delay End Delay 0	ut of Time Window	K Maximum Duration	Parallel Processing
🔊 🖁 ZD 🛃 🎛			
Alert for Wrong Parallel Processing	No. of Para. Jobs		
Yellow	5		
Red			
Yellow			

It is not possible to monitor in general if certain (different) background jobs are running at the same time. Jobs are considered to be started in parallel if they start within 2 minutes tolerance.

When a *Start Delay* is configured all jobs with prefix SD_BILL_0001_D that have a planned start time of 11pm are considered individually for the *Start Delay* alert. So when we configure a yellow alert to be raised after 60 minutes then all five parallel jobs are evaluated, i.e. if three of the jobs start in time but two have a delay of more than 60 minutes (as all background work processes might be occupied) then three jobs are valuated green and two jobs will raise a yellow alert respectively. No red alert should be raised so we set an arbitrarily high threshold value.

Start Delay End	l Delay	Out of Tir	me W
🔊 🖁 🗵 🗈 🛃			
Start Delay for YELLOW	Start Dela	iy for RED	Unit
60	999999		min

Additionally, we would customize a red alert for every job cancellation as described in the other scenarios.

For setting up some email (or other automatic workflow) notification one has to navigate to node *Notifications* under the corresponding business process step. We would choose *Recipient Type* "U" for regular email addresses.

Workflow Not	ifications Su	pport Notifications]				
🔊 🛗 🔀	a 🛃 a 🖽						
Monitoring Type	Monitoring Objec	t	Sender	Recipient Address	Reci. Type	. etailed	
Background Job	ound Job Bg. Job 12 Parallelized billing run			a.mueller@bizmoni.com			
Update Error Update Error 🕞 Value Help: Reci			ipient Type	9		X	
		Recipient Type	Descrip	otion			
		U	e-mail	address; e.g. "bill.s	smith@our.com	ipany.com"	
		В	SAP use	er name; e.g. "SMITHB'			
R			Remote SAP name; e.g "SMITHB" (specify SID and client, too)				
		C	Shared	distribution list; e	.g. "Business	Process Monitoring"	

Sender	Must be a user within SAP Solution Manager client 000. This user must have a valid e-mail address that is known by the used mail-server
Recipient Address	Depending on the Recipient Type the recipient name is required. This could be any e-mail address, an SAP user name in the same system, a Remote SAP name or a shared distribution list.
Reci. Type	There are currently 4 different Recipient Types as value helps: U (e-mail), B (SAP user name), R (Remote SAP name) and C (shared distribution list). More types can be found under <u>help.sap.com</u>
Detailed	Triggers a long text for e-mails or SAPOffice mails, e.g. name of the solution, name of the business process step,)

Additional setup:

As the creation of invoices is of utmost importance for the core business process of the company BizMoni we also monitor for *Update Errors* that are specifically related to the invoice creation. Depending on how the billing run is executed there are different ABAP reports which could be relevant for monitoring, e.g. SDBILLDL, RV60SBAT or SAPMV60S.

Monitoring Types 🏼 🍟	👎 Descri		
🗠 i 🖁 i 🔀 🗈 i 🛃			
Monitoring Type	Relevant?		
Application Monitor			
Background Job			
Dialog Performance			
Update Error			
Lindstad Errora (Lind	late 2 Errora		
	latezentors		
Object Type Ob	oject Name User	No. of Errs for RED (Upd1)	No. of Errs for RED (Upd2)
ABAP Program 📱 SE	DBILLDL *	1	1
ABAP Program 🛛 🖹 R\	/60SBAT *	1	1
ABAP Program 📱 SA	APMV60S *	1	1

The end-users ask for a possibility of an automatic evaluation of exceptions/errors during the billing run. For this one can use the *Due List Log* monitoring functionality. How this monitor and the *Update Error* monitor are set up in detail is described in the "Business Process Monitoring - Setup Guide" which can be found in the Media Library on the SAP Service Marketplace under Quick Link /BPM.

Monitoring Types 🏼 🍟	🟳 Descrip	1			
	ũ 🌐		Due List	Log	
Monitoring Type	Relevant?		∞間	201 🖪 🗈	
Application Monitor			DI Type	Licor	Angregation
Background Job	~		Dilling D	*	Aggregation
Dialog Performance			Binnig E		
Update Error					per log
Due List Log	~				per day

Scenario IDoc processing

Key words: periodic job, Interface monitoring, Service Desk notification

During the day it happens all day long that stock transfer orders are created in dialog which should be processed as soon as possible in the warehouse WHS. The company BizMoni uses a third party legacy system to manage their warehouse WHS. In order to guarantee a quick processing a periodic job LE_TRANS_IDOC_0001 is scheduled every 15 minutes from Monday to Friday which collects all newly created transfer orders and sends them as one IDoc (of type WMTORD) to the connected legacy system. The first start of the job happens five minutes past the full clock-hour. For the job scheduling team it is important to know if every job started at all and in time as well as if each job finished successfully

Suggested Setup:

For the Job Identification we choose the easiest way and enter the correct Job Name.

	Job Identifica	ation 🛛 🗍 Add/Remove AB/	AP Program					
) 🗋 🛃	3 (H) ZID (B) D						
ID	Monitoring?	Job Name	Job Step No.		 	Job S	chedule	9
	~	LE_TRANS_IDOC_0001	1	Ē)		week	ly	Ē

As the job runs every 15 minutes and is important it would be senseless if one marked the job as not critical so that the data collector only checks every 60 minutes for new jobs. Within this period there will be normally four job runs that are then evaluated with a big delay. So the job is marked as time critical. If one knows the exact user under which the job is scheduled it is reasonable to specify it.

Job Details Start Information Weekly Schedule												
🔊 🛗 🗵 🗈 🛃 🗉												
Custom Identifier	Critical Job	User	Parallel Jobs?									
IDoc to WHS	v	BTCIDOC	No 🗈									

The job is started "by time". As a start time one can choose an arbitrary time at which one of the many periodic jobs should normally start. In our case a *Planned Start* at 1:20am, 1:35am, 1:50am or 2:05am would work just as good.

Job Details	Start Information 🛛 🛛 🛛 🗸
Start Procedure	Planned Start [hhmmss]
by time 🔳	010500

We mark the working days from Monday to Friday in the *Weekly Schedule* and enter the *Period* in minutes.

Job Details Start Information Weekly Schedule									
∞	1 (11)	2							
Mo Tu We Th Fr Sa Su Period [min] (just for periodic jobs)									
•	•	•	•	•			15		



The maximum value for a period would be 1439 minutes which equals 23 hours and 59 minutes. But this is surely no meaningful period for a job to run. Meaningful periods are usually divider of 60 or multiples of 15.

The *Start Delay* is configured in a way that after one minute a yellow alert is raised indicating that one job has not started in time, and after 14 minutes a red alert is raised indicating that one job didn't run at all before the next job is kicked-off.

Start Delay 🛛 Max	imum Duration 🏼 🖌 Job C
Start Delay for YELLOW	Start Delay for RED Unit
1	14 min

It is not possible to monitor an *End Delay* or *Out of Time Window* alert for periodic jobs.

Additionally we would customize a red alert for every job cancellation as described in the other scenarios.

Additional setup:

In order to be aware about the IDoc sending and its status it is possible to use the Interface Monitoring functionality within the Business Process Monitoring that would then check only for outbound IDocs of type WMTORD. Alerts could be raised for status IDoc generated, IDoc ready for dispatch, IDoc in external system, IDoc dispatched, Error in IDoc interface, Error in external system, IDoc with delete flag, IDoc processing in target system.

Similar alerts could also be raised for other IDoc types and for inbound direction of course.

Additionally, one could configure the automatic Service Desk or Support Desk notification in case that e.g. more than 4 yellow or more than 2 red alerts have been raised. This message would be put on the *Queue* BC-MID-ALE for interface problems. If one customized all required information then the column *Automatic* should be ticked after saving.

	Workflow Notifications Support Notifications											
Monitoring Type Monitoring Object Queue Processor Automatic Priority Reporter Num of YELLOW Num							Num of RED					
	Background Job	Bg. Job 6 IDoc to	BC-MID-ALE	MUELLER		×	High 🖺	SCHMITT	5	3		

For further information regarding interface monitoring and Support Notifications please refer to the "Business Process Monitoring - Setup Guide" on the SAP Service Marketplace under the Quick Link /BPM \rightarrow Media Library.

Scenario Periodic job chain with SM36

Key words: job with several steps, periodic job chain with SM36

For some specific vendors the company BizMoni wants to create deliveries every four hours (starting at 6 am) in background, post the goods issue afterwards and then immediately create the corresponding invoices. Due to functional limitations within standard transaction SM36 it is not possible to directly define such a periodic job chain (see also scenario MRP \rightarrow event \rightarrow BOP). Instead one defines a periodic job with three different job steps. As a first step ABAP program RVV50R10C is started, as a second step ABAP program WS_MONITOR_OUTB_DEL_GDSI is triggered and then as a third step ABAP program SDBILLDL is scheduled.

It is important to get the information if the job cancels and if the runtime of the respective job is shorter than four hours so that there is no overlap between two succeeding jobs.

Suggested Setup:

For the identification of the corresponding job it is possible to use different configurations. Each of the three configurations below is just as good as the others and would identify the same job. Just out of convenience we stick with the first one.

	Job Identifica	ation 🛛 Ad	d/Remove AB/	AP Program							
ᡌ			2010								
ID	Monitoring?	Job Name	Job Step No.	ABAP Program	Variant				Job Sched	ule	
7	✓	*	1	RVV50R10C	SPECVEND				weekly	Ē	
8		*	2	WS_MONITOR_OUTB_DEL_GDSI 🖺	SPECVEND				weekly	Ē	
9		*	3	SDBILLDL 🗎	SPECVEND				weekly	Ē	

It is not possible to monitor just a specific job step of a background job. Always the whole job is monitored and valuated. That's why the three above mentioned configurations are all equivalent.

We give the job a speaking name, enter the *Start Information* and customize the job period of 4 hours, i.e. 240 minutes.

Job Details Start Information W	eekly Schedule	
Custom Identifier Critical Job Use	er Parallel Jobs?	
Job chain for spec. vendors 📃 *	No	1
Job Details Start Information w Job Details Start Information Weekly Schedule		
Start Procedure Planned Start [hhmmss]	Mo Tu We Th Fr Sa	Su Period [min] (just for periodic jobs)
by time 📓 060000		240

We configure the *Start Delay* in a way that only a red is raised in case that an overlap between two succeeding jobs occurs, i.e. after 4 hours.

Start Delay Ma	ximum Duration 🛛 🍸 Job Can	
Start Delay for YELLOV	Start Delay for RED Unit	
240	240 min	

Additionally we would customize a red alert for every job cancellation which can be compared with other scenarios.

Further Information

Troubleshooting

If executing this Best Practice did not produce the desired results, see the following SAP Notes

Note Number	Short Description
16083	Standard jobs, reorganization jobs
24092	Distribution of background jobs on application server
31503	FAQ: Background jobs
36280	Background Work Processes Reserved for Job Class A
39412	How many work processes to configure
195157	Application log: Deletion of log
519059	FAQ: Background processing system
541538	FAQ: Reorganizations
553953	RZ20: Monitoring background jobs
564391	Other method during the job immediate- and event start
588668	FAQ: Database statistics
599835	Problems because of too many short jobs
706478	Measures against intensively increasing Basis tables

Feedback and Questions

Send any feedback by formulating an SAP customer message to component SV-GST-SMC. You can do this at http://service.sap.com/message.

© Copyright 2005 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[®], NT[®], EXCEL[®], Word[®], PowerPoint[®] and SQL Server[®] are registered trademarks of Microsoft Corporation.

IBM[®], DB2[®], OS/2[®], DB2/6000[®], Parallel Sysplex[®], MVS/ESA[®], RS/6000[®], AIX[®], S/390[®], AS/400[®], OS/390[®], and OS/400[®] are registered trademarks of IBM Corporation.

ORACLE[®] is a registered trademark of ORACLE Corporation. INFORMIX[®]-OnLine for SAP and Informix[®] Dynamic ServerTM are registered trademarks of Informix Software Incorporated.

UNIX[®], X/Open[®], OSF/1[®], and Motif[®] are registered trademarks of the Open Group. HTML, DHTML, XML, XHTML 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.

SAP, SAP Logo, R/2, RIVA, R/3, ABAP, SAP ArchiveLink, SAP Business Workflow, WebFlow, SAP EarlyWatch, BAPI, SAPPHIRE, Management Cockpit, mySAP.com Logo and mySAP.com are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other products mentioned are trademarks or registered trademarks of their respective companies.

Disclaimer: SAP AG assumes no responsibility for errors or omissions in these materials. These materials are provided "as is" without a warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.

SAP shall not be liable for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials. SAP does not warrant the accuracy or completeness of the information, text, graphics, links or other items contained within these materials. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third party web pages nor provide any warranty whatsoever relating to third party web pages.