

▼ Welcome to ADM004

System Administration -
Background Job Processing Online
Training

▶ Chapter 1: Overview of Job Processing

▶ Chapter 2: JE Processing Model

▶ Chapter 3: JS Processing Model

End of Course

System Administration - Background Job Processing Online Training

Abstract:

This training includes details of two job application - Executor and Scheduler and will show how job processing works, and what kind of business challenges you may need to address using this functionality.

Detailed objectives.

After the training you will know:

- ▶ What are differences between Job Executor and Job Scheduler
- ▶ How they work and how to use all their features

Target audience:

- ▶ People who will need to design, implement, and/or maintain jobs in DELMIA Apriso solutions

Requirements:

- ▶ General understanding of Job Executing, DELMIA Apriso Portal Usage, and Windows Administration.
- ▶ Recommended (but not required) trainings:
 - ADM001 - System Administration Installation
 - ADM002 - System Administration Upgrade
 - ADM003 - System Administration Configuration

Role and level:

- ▶ DELMIA Apriso Administrators



Duration: 60 min

Search...



▼ Welcome to ADM004

System Administration -
Background Job Processing Online
Training

▼ Chapter 1: Overview of Job Processing

Job Executor (JE)

Job Scheduler (JS)

JE and JS Common Features

Job Executor vs Job Scheduler

▶ Chapter 2: JE Processing Model

▶ Chapter 3: JS Processing Model

End of Course

Job Executor (JE)

Job Executor (JE) is the primary DELMIA Apriso component for processing system background tasks. It is designed as a high-performance execution engine for executing queue-like (FIFO - First In, First Out) background tasks. Job Executor is complemented by a second service - Job Scheduler, which is optimized for processing recurring, scheduled jobs.

Job Executor consists of two parts: a background service that actually executes jobs and UI monitoring screens that allow for the monitoring, configuration and troubleshooting of Job Executor.

In JE includes some features that are not supported by JS, mainly:

- ▶ Processing in pools
- ▶ Prioritization of jobs
- ▶ Configurable execution attempts, their number, and sleep time
- ▶ Ability to run multiple instances of the service for maximum performance

ID	Job ID	Name	Description	Created Date	Pool	Synchronization	Action Type	Timeout
14123264	0	FI Invocation fr	10000009	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123263	0	FI Invocation fr	10000017	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123262	0	FI Invocation fr	10000011	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123261	0	FI Invocation fr	10000014	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123260	0	FI Invocation fr	10000013	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123259	0	FI Invocation fr	10000016	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123258	0	FI Invocation fr	10000015	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123257	0	FI Invocation fr	10000018	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123256	0	FI Invocation fr	10000010	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123255	0	FI Invocation fr	10000014	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123254	0	FI Invocation fr	10000016	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123253	0	FI Invocation fr	10000018	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123252	0	FI Invocation fr	10000015	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123251	0	FI Invocation fr	10000013	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123250	0	FI Invocation fr	10000014	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123249	0	FI Invocation fr	10000018	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123248	0	FI Invocation fr	10000008	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123247	0	FI Invocation fr	10000012	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123246	0	FI Invocation fr	10000016	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123245	0	FI Invocation fr	10000015	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123244	0	FI Invocation fr	10000013	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123243	0	FI Invocation fr	10000018	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123242	0	FI Invocation fr	10000014	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00
14123241	0	FI Invocation fr	10000013	06/08/2016 12:	DEFAULT		1 - Invoke Stan	00:30:00

Scroll down to load next page.



▼ Welcome to ADM004

System Administration -
Background Job Processing Online
Training

▼ Chapter 1: Overview of Job Processing

Job Executor (JE)

Job Scheduler (JS)

JE and JS Common Features

Job Executor vs Job Scheduler

▶ Chapter 2: JE Processing Model

▶ Chapter 3: JS Processing Model

End of Course

Job Scheduler (JS)

Job Scheduler (JS) is the secondary DELMIA Apriso component for processing system background tasks. It is designed as a flexible engine for processing complex recurring, scheduled jobs.

Job Scheduler consists of two parts: a background service that actually executes jobs and UI monitoring screens that allow for the monitoring, configuration and troubleshooting of Job Scheduler

With JS, multiple actions per jobs can be processed, as well as multiple schedules per job. Also, there is an alarm service - JS will send alerts in the event of failed execution.

ID	Job Type	Name	Description	Queued On	Status	Synchronization Q	Executor	Disabled On
100367371	1 - User	KPI_APP_Perfor	Equipment Perfomance	05/19/2016 11:40:	0 - Waiting		KPI	
100367384	1 - User	KPI_APP_Quality	Equipment Quality	05/31/2016 08:30:	0 - Waiting		KPI	
100367385	1 - User	KPI_APP_OEE	Overall Equipment Efficiency	05/31/2016 08:30:	0 - Waiting		KPI	
100367387	1 - User	KPI_APP_Availa	Equipment Availability	05/31/2016 08:30:	0 - Waiting		KPI	
100367389	1 - User	ALLWC_Calculate	ALLWIC_Calculate	05/31/2016 08:30:	0 - Waiting		KPI	
100367390	1 - User	KPI_APP_Perfor	Equipment Perfomance	05/31/2016 08:30:	0 - Waiting		KPI	
100367405	1 - User	KPI_APP_Quality	Equipment Quality	05/31/2016 08:40:	0 - Waiting		KPI	
100367406	1 - User	KPI_APP_OEE	Overall Equipment Efficiency	05/31/2016 08:40:	0 - Waiting		KPI	
100367408	1 - User	KPI_APP_Availa	Equipment Availability	05/31/2016 08:40:	0 - Waiting		KPI	
100367410	1 - User	ALLWC_Calculate	ALLWIC_Calculate	05/31/2016 08:40:	0 - Waiting		KPI	
100367411	1 - User	KPI_APP_Perfor	Equipment Perfomance	05/31/2016 08:40:	0 - Waiting		KPI	
100367412	1 - User	APP_CNT_FUTU	Containment Future	05/31/2016 08:40:	0 - Waiting		1	
100367413	1 - User	APP_CNT_FUTU	Containment Future	05/31/2016 08:41:	0 - Waiting		1	
100367414	1 - User	APP_CNT_FUTU	Containment Future	05/31/2016 08:42:	0 - Waiting		1	
100367415	1 - User	APP_CNT_FUTU	Containment Future	05/31/2016 08:43:	0 - Waiting		1	
100367416	1 - User	APP_CNT_FUTU	Containment Future	05/31/2016 08:44:	0 - Waiting		1	
100367417	1 - User	KPI_APP_Electric	Electricity Consumption	05/31/2016 08:45:	0 - Waiting			
100367418	1 - User	KPI_APP_GasCo	Gas Consumption	05/31/2016 08:45:	0 - Waiting			
100367419	1 - User	APP_CNT_FUTU	Containment Future	05/31/2016 08:45:	0 - Waiting		1	
100367420	1 - User	APP_CNT_FUTU	Containment Future	05/31/2016 08:46:	0 - Waiting		1	
100367421	1 - User	APP_CNT_FUTU	Containment Future	06/07/2016 10:19:	0 - Waiting		1	
100367422	1 - User	APP_CNT_FUTU	Containment Future	06/07/2016 10:33:	0 - Waiting		1	
100367423	1 - User	MaintenanceSche		06/07/2016 10:19:	0 - Waiting			
100367424	1 - User	KPI_APP_Electric	Electricity Consumption	06/07/2016 10:19:	0 - Waiting			
100367425	1 - User	KPI_APP_GasCo	Gas Consumption	06/07/2016 10:19:	0 - Waiting			

Scroll down to load next page.

Welcome to ADM004

System Administration -
Background Job Processing Online
Training

Chapter 1: Overview of Job Processing

Job Executor (JE)

Job Scheduler (JS)

JE and JS Common Features

Job Executor vs Job Scheduler

Chapter 2: JE Processing Model

Chapter 3: JS Processing Model

End of Course

JE and JS Common Features

JE and JS share some common features:

- ▶ Monitoring and troubleshooting user interface
- ▶ Synchronization queues
- ▶ Monitoring screens
- ▶ Same model for action type and parameters configuration
- ▶ Automatic retry for jobs failed due to deadlock or concurrency error

The screenshot displays two overlapping software windows. The top window is titled 'Job Executor: Job Pools' and shows a table of 'JE Job Pools' with columns: Name, Maximum Threads, and Priority. It lists five pools: DEFAULT, JE2pool1, JE2pool2, pool1, and pool2. The bottom window is titled 'Job Scheduler: Scheduled Jobs' and shows a table of 'JS Scheduled Jobs' with columns: ID, Name, Description, Job Type, and Enabled. It lists numerous scheduled jobs, such as ALLWC_Calculate, APR_BKG_AutoM, APR_CNT_FUTUR, APR_MNT_RunSc, APR_MNT_RunSc, APR_MPI_Populat, DELETE_FULL_JO, DELETE_KPI_CAL, DELETE_TRANSA, KPI_APB_Availabil, KPI_APB_Electrici, KPI_APB_GasCon, KPI_APB_MTBF, KPI_APB_MTTR, KPI_APB_OEE, KPI_APB_Perform, KPI_APB_Quality, MaintenanceSche, Notifications, OldOpenedLaborE, PlantMPI_DIM_D, ProcessOLAPTes, ProcessPlantMPI, QMSInspectionS, and UPDATE_OPERATI.

ID	Name	Description	Job Type	Enabled
10000001	ALLWC_Calculate	ALLWC_Calculate	1 - User	<input checked="" type="checkbox"/>
10000002	APR_BKG_AutoM	Auto Machine Shift	1 - User	<input checked="" type="checkbox"/>
10000000	APR_CNT_FUTUR	Containment Future	1 - User	<input checked="" type="checkbox"/>
10000000	APR_MNT_RunSc	Schedule for APR	1 - User	<input checked="" type="checkbox"/>
10000000	APR_MNT_RunSc	Schedule for APR	1 - User	<input checked="" type="checkbox"/>
10000000	APR_MNT_RunSc	Call APR_MNT.Run	1 - User	<input checked="" type="checkbox"/>
10000000	APR_MPI_Populat	APR_MPI_Populat	1 - User	<input type="checkbox"/>
10000001	DELETE_FULL_JO	Delete full Job History	1 - User	<input checked="" type="checkbox"/>
10000000	DELETE_KPI_CAL	Call Archiving Procedure	1 - User	<input checked="" type="checkbox"/>
10000001	DELETE_TRANSA	Delete Transactional Data	1 - User	<input checked="" type="checkbox"/>
10000001	KPI_APB_Availabil	Equipment Availability	1 - User	<input checked="" type="checkbox"/>
10000001	KPI_APB_Electrici	Electricity Consumption	1 - User	<input checked="" type="checkbox"/>
10000001	KPI_APB_GasCon	Gas Consumption	1 - User	<input checked="" type="checkbox"/>
10000001	KPI_APB_MTBF	Mean Time Between Failures	1 - User	<input checked="" type="checkbox"/>
10000001	KPI_APB_MTTR	Mean Time To Repair	1 - User	<input checked="" type="checkbox"/>
10000001	KPI_APB_OEE	Overall Equipment Effectiveness	1 - User	<input checked="" type="checkbox"/>
10000002	KPI_APB_Perform	Equipment Performance	1 - User	<input checked="" type="checkbox"/>
10000001	KPI_APB_Quality	Equipment Quality	1 - User	<input checked="" type="checkbox"/>
10000002	MaintenanceSche	Maintenance Schedule	1 - User	<input checked="" type="checkbox"/>
10000000	Notifications	Notifications	1 - User	<input checked="" type="checkbox"/>
10000002	OldOpenedLaborE	Close old open Requests	1 - User	<input checked="" type="checkbox"/>
10000000	PlantMPI_DIM_D	Plant MPI DIMENSION	1 - User	<input checked="" type="checkbox"/>
10000013	ProcessOLAPTes	Process OLAP Test	1 - User	<input checked="" type="checkbox"/>
10000000	ProcessPlantMPI	Process Plant MPI	1 - User	<input checked="" type="checkbox"/>
10000002	QMSInspectionS	QMS Inspection Status	1 - User	<input type="checkbox"/>
10000002	UPDATE_OPERATI	Update Operator Information	1 - User	<input checked="" type="checkbox"/>



▼ Welcome to ADM004

System Administration -
Background Job Processing Online
Training

▼ Chapter 1: Overview of Job Processing

Job Executor (JE)

Job Scheduler (JS)

JE and JS Common Features

Job Executor vs Job Scheduler

► Chapter 2: JE Processing Model

► Chapter 3: JS Processing Model

End of Course

Job Executor vs Job Scheduler

JE and JS also differ in many aspects:

Job Executor (JE)	Job Scheduler (JS)
Jobs for immediate execution.	Jobs scheduled for the future or having recurring schedule.
Large number of single, immediate queue-like (FIFO) jobs.	Small number of recurring jobs scheduled for multiple executions over a period of time (recurring).
Jobs with only one action, only a schedule to be run immediately.	Complex jobs (multiple actions, multiple schedules).
Jobs created by other DELMIA Apriso modules (Business Integrator, Printing, FI).	Jobs usually created manually by Administrators.
Reprocessing performed by creating new jobs (duplicating existing ones).	Reprocessing performed by creating new schedules for the same job.



▼ Welcome to ADM004

System Administration -
Background Job Processing Online
Training

▼ Chapter 1: Overview of Job Processing

Job Executor (JE)
Job Scheduler (JS)
JE and JS Common Features
Job Executor vs Job Scheduler

▼ Chapter 2: JE Processing Model

JE Processing Model, part 1
JE Processing Model, part 2
JE Processing Model - Sequence
JE Processing Model - Double Retry
Mechanism
JE Processing Model - Finishing
JE Self-Monitoring
JE Central Configuration
Job Pool Configuration Screen
Configuration of Job Pools
Sample Configuration of Job Pools -
Correct
Sample Configuration of Job Pools -
Incorrect
Job Pools - Recommendations

► Chapter 3: JS Processing Model

End of Course

Chapter 2: JE Processing Model

In this chapter there will be more details about Job Executor as the the primary DELMIA Apriso component for processing system background tasks.

Here are the chapters to be covered:

1. *Overview of Job Processing*
2. **JE Processing Model**
3. JS Processing Model



▼ Welcome to ADM004:

System Administration -
Background Job Processing Online
Training

▼ Chapter 1: Overview of Job Processing

Job Executor (JE)
Job Scheduler (JS)
JE and JS Common Features
Job Executor vs Job Scheduler

▼ Chapter 2: JE Processing Model

JE Processing Model, part 1
JE Processing Model, part 2
JE Processing Model - Sequence
JE Processing Model - Double Retry Mechanism
JE Processing Model - Finishing
JE Self-Monitoring
JE Central Configuration
Job Pool Configuration Screen
Configuration of Job Pools
Sample Configuration of Job Pools - Correct
Sample Configuration of Job Pools - Incorrect
Job Pools - Recommendations

► Chapter 3: JS Processing Model

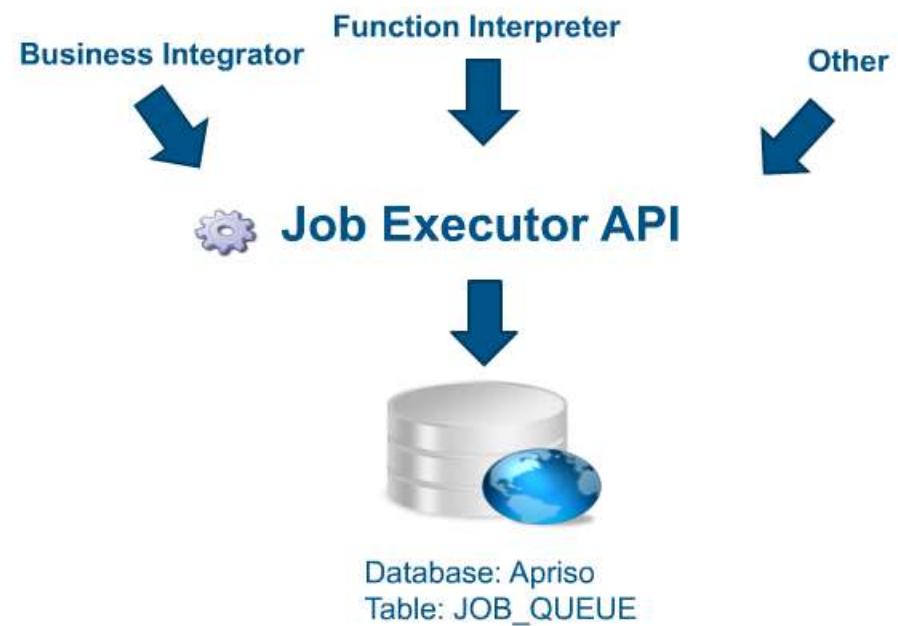
End of Course

JE Processing Model, part 1

Job Executor is a performance-optimized background processing engine. It processes jobs in a queue-like fashion while respecting job pool priorities and thread limitations.

The following section describes details about the Job Executor processing model.

- ▶ JE periodically scans the JOB_QUEUE table and loads new jobs into memory
- ▶ Jobs are loaded in batches
- ▶ Both, the batch size and load frequency are configurable options of JE
- ▶ When new jobs are loaded from the database, they are grouped by a job pool
- ▶ JE keeps a list of pools with assigned jobs and scans the pools in the order of pool priority (there are no priorities on the job level) to find the first job to be executed





▼ Welcome to ADM004:

System Administration -
Background Job Processing Online
Training

▼ Chapter 1: Overview of Job Processing

Job Executor (JE)
Job Scheduler (JS)
JE and JS Common Features
Job Executor vs Job Scheduler

▼ Chapter 2: JE Processing Model

JE Processing Model, part 1
JE Processing Model, part 2
JE Processing Model - Sequence
JE Processing Model - Double Retry Mechanism
JE Processing Model - Finishing
JE Self-Monitoring
JE Central Configuration
Job Pool Configuration Screen
Configuration of Job Pools
Sample Configuration of Job Pools - Correct
Sample Configuration of Job Pools - Incorrect
Job Pools - Recommendations

► Chapter 3: JS Processing Model

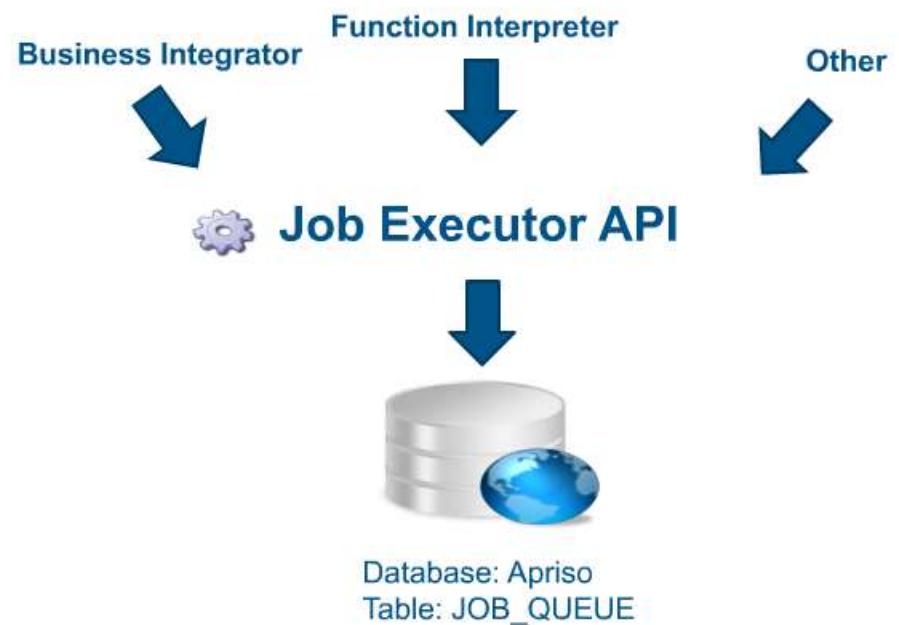
End of Course

JE Processing Model, part 2

Job Executor is a performance-optimized background processing engine. It processes jobs in a queue-like fashion while respecting job pool priorities and thread limitations.

The following section describes details about the Job Executor processing model.

- ▶ When such a job is located, JE checks the number of threads used by other jobs being executed from the same pool, and if the number is not bigger than the maximum number of threads for this pool, it starts the execution of the job
- ▶ If the maximum number of threads is exceeded, JE continues scanning for new jobs belonging to lower priority pools



Search... Q

▼ Welcome to ADM004

System Administration -
Background Job Processing Online
Training

▼ Chapter 1: Overview of Job Processing

Job Executor (JE)
Job Scheduler (JS)
JE and JS Common Features
Job Executor vs Job Scheduler

▼ Chapter 2: JE Processing Model

JE Processing Model, part 1
JE Processing Model, part 2
JE Processing Model - Sequence
JE Processing Model - Double Retry Mechanism
JE Processing Model - Finishing
JE Self-Monitoring
JE Central Configuration
Job Pool Configuration Screen
Configuration of Job Pools
Sample Configuration of Job Pools - Correct
Sample Configuration of Job Pools - Incorrect

Job Pools - Recommendations

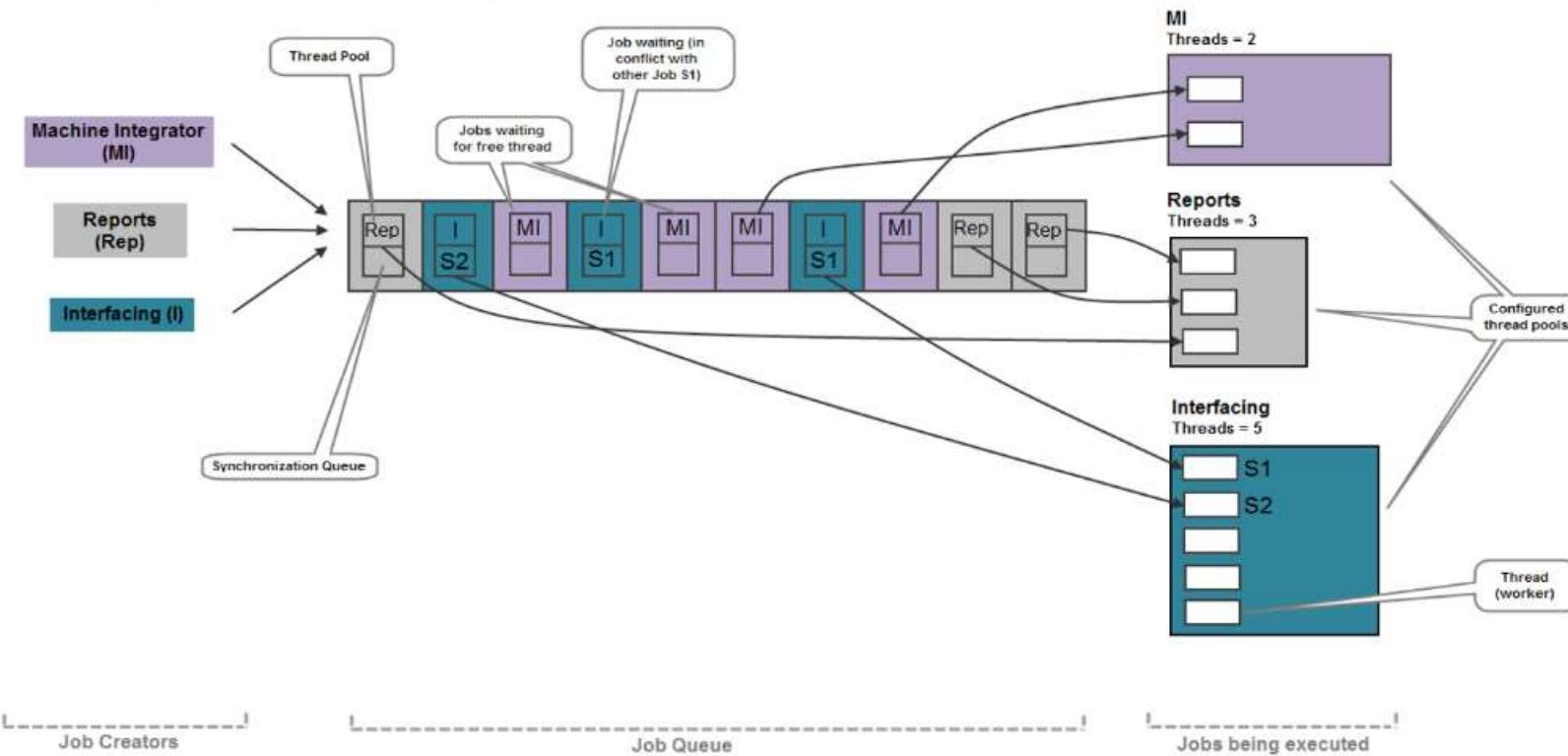
► Chapter 3: JS Processing Model

End of Course

JE Processing Model - Sequence

In cases where jobs must be executed in a particular order, Synchronization Queues should be used.

Each job can have a Synchronization Queue assigned, which is an identifier (string) that groups all jobs from the same pool that should be executed exactly in the same order as they are created.



Welcome to ADM004

System Administration -
Background Job Processing Online
Training

Chapter 1: Overview of Job Processing

Job Executor (JE)

Job Scheduler (JS)

JE and JS Common Features

Job Executor vs Job Scheduler

Chapter 2: JE Processing Model

JE Processing Model, part 1

JE Processing Model, part 2

JE Processing Model - Sequence

JE Processing Model - Double Retry Mechanism

JE Processing Model - Finishing

JE Self-Monitoring

JE Central Configuration

Job Pool Configuration Screen

Configuration of Job Pools

Sample Configuration of Job Pools -
Correct

Sample Configuration of Job Pools -
Incorrect

Job Pools - Recommendations

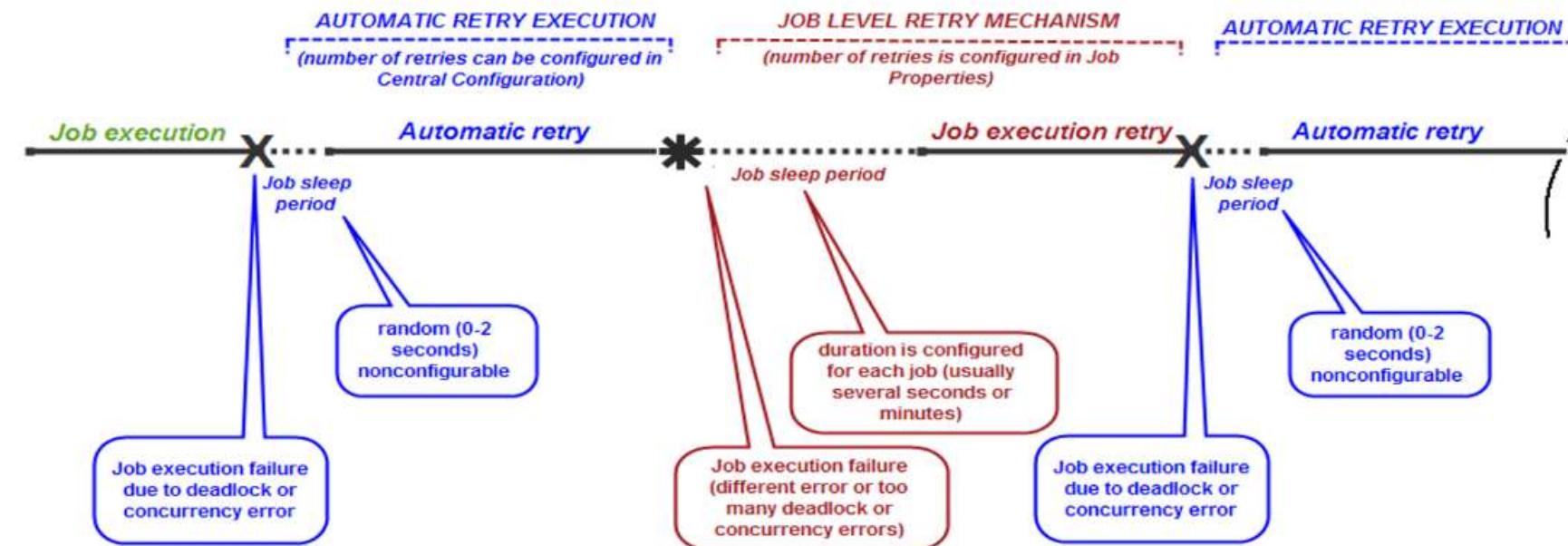
Chapter 3: JS Processing Model

End of Course

JE Processing Model - Double Retry Mechanism

JE also contains a double retry mechanism:

- ▶ Automatic retry - in case of deadlock or concurrency error that causes Job failure (it can be configured by JE)
- ▶ Job level retry mechanism - each job can be configured to be reprocessed in case of any error





▼ Welcome to ADM004

System Administration -
Background Job Processing Online
Training

▼ Chapter 1: Overview of Job Processing

Job Executor (JE)
Job Scheduler (JS)
JE and JS Common Features
Job Executor vs Job Scheduler

▼ Chapter 2: JE Processing Model

JE Processing Model, part 1
JE Processing Model, part 2
JE Processing Model - Sequence
JE Processing Model - Double Retry
Mechanism

JE Processing Model - Finishing

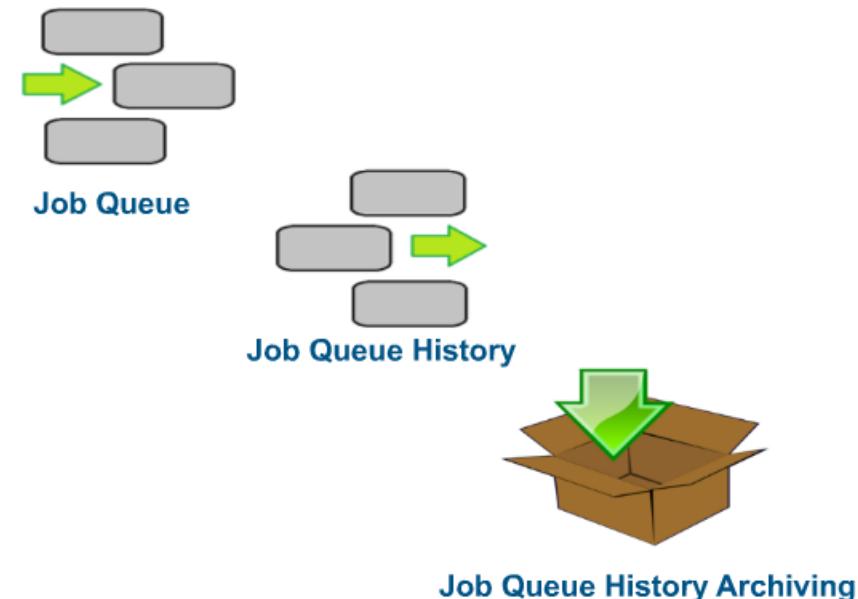
JE Self-Monitoring
JE Central Configuration
Job Pool Configuration Screen
Configuration of Job Pools
Sample Configuration of Job Pools -
Correct
Sample Configuration of Job Pools -
Incorrect
Job Pools - Recommendations

► Chapter 3: JS Processing Model

End of Course

JE Processing Model - Finishing

- ▶ After a job execution is completed, the related row is deleted from the JOB_QUEUE table
- ▶ Then row is copied and inserted into the JOB_QUEUE_HISTORY table - it keeps a history of executions for all jobs together with the original job definition
- ▶ The JOB_QUEUE_HISTORY table should be periodically archived to remove unnecessary old data and avoid database overgrowth





▼ Welcome to ADM004

System Administration -
Background Job Processing Online
Training

▼ Chapter 1: Overview of Job Processing

Job Executor (JE)
Job Scheduler (JS)
JE and JS Common Features
Job Executor vs Job Scheduler

▼ Chapter 2: JE Processing Model

JE Processing Model, part 1
JE Processing Model, part 2
JE Processing Model - Sequence
JE Processing Model - Double Retry Mechanism
JE Processing Model - Finishing

JE Self-Monitoring

JE Central Configuration

Job Pool Configuration Screen

Configuration of Job Pools

Sample Configuration of Job Pools - Correct

Sample Configuration of Job Pools - Incorrect

Job Pools - Recommendations

► Chapter 3: JS Processing Model

End of Course

JE Self-Monitoring

Memory management - this self-check is performed every 5 minutes and is controlled by

- Private Bytes performance counter, where the following memory thresholds apply:
 - For 32-bit environments - 1,5 GB
 - For 64-bit environments - 3 GB
 - I. if the values above are exceeded, service restart is performed
- Database access control - if no database access is detected, the service remains in starting status until the connection is restored; after establishing the connection, the service proceeds to started status
- Job timeout check - the check is performed every 5 minutes; the service is restarted when the following condition is met:
 - Now - Job Timeout > Last Job Timeout + 20%

JE Recycling:

- In the case of database access problems, the service is configured and maintained by infrastructure/operating system and not by Job Executor
- In the case of failure:
 - Non-clustered environment - the behavior is controlled by Windows service properties
 - Clustered environment - several service restarts are performed (depending on cluster configuration); if the problem persists after the specified number of retries, it is switched to another node



▼ Welcome to ADM004:

System Administration -
Background Job Processing Online
Training

▼ Chapter 1: Overview of Job Processing

Job Executor (JE)
Job Scheduler (JS)
JE and JS Common Features
Job Executor vs Job Scheduler

▼ Chapter 2: JE Processing Model

JE Processing Model, part 1
JE Processing Model, part 2
JE Processing Model - Sequence
JE Processing Model - Double Retry Mechanism
JE Processing Model - Finishing
JE Self-Monitoring

JE Central Configuration

Job Pool Configuration Screen
Configuration of Job Pools
Sample Configuration of Job Pools - Correct
Sample Configuration of Job Pools - Incorrect
Job Pools - Recommendations

► Chapter 3: JS Processing Model

End of Course

JE Central Configuration

For proper configuration, it is mandatory to provide the proper data in the Central Configuration file with a path to system services:

```
<add key="SchedulingServices" value="REMOTING:net.tcp://${AppAddress}:32607/JobSchedulerService" />
<add key="JobExecutorServices" value="REMOTING:net.tcp://${AppAddress}:32612/jobexecutorservice" />
<add key="GlobalProcessManagerServices" value="REMOTING:net.tcp://${AppAddress}:32709/gpmService" />
```

```
<FlexNet.JobExecutor>
  <add key="MaxExecutionThreads" value="10" />
  <add key="JobLoadRate" value="1000" />
  <add key="JobExecutorName" value="DefaultJobExecutor" />
  <add key="JobLoadBatchSize" value="50" />
  <add key="NumberOfAutomaticProcessingRetries" value="1" />
</FlexNet.JobExecutor>
```

Search... 

▼ Welcome to ADM004:

System Administration -
Background Job Processing Online
Training

▼ Chapter 1: Overview of Job Processing

Job Executor (JE)
Job Scheduler (JS)
JE and JS Common Features
Job Executor vs Job Scheduler

▼ Chapter 2: JE Processing Model

JE Processing Model, part 1
JE Processing Model, part 2
JE Processing Model - Sequence
JE Processing Model - Double Retry Mechanism
JE Processing Model - Finishing

JE Self-Monitoring
JE Central Configuration

Job Pool Configuration Screen

Configuration of Job Pools

Sample Configuration of Job Pools - Correct

Sample Configuration of Job Pools - Incorrect

Job Pools - Recommendations

► Chapter 3: JS Processing Model

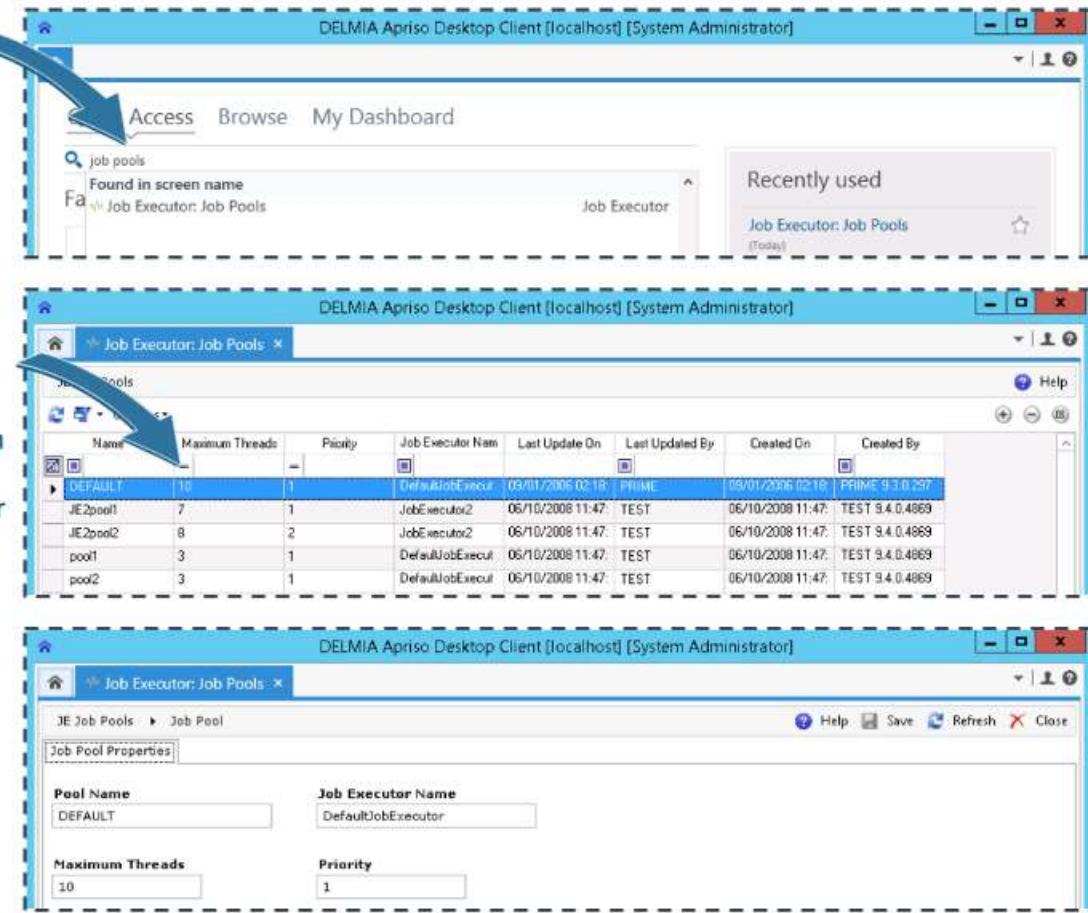
End of Course

Job Pool Configuration Screen

Every job processed by JE must be assigned to one of the configured job pools (job pools allow prioritization of job execution). Each pool contains the following configurable parameters:

- ▶ **Pool name** - User friendly name of the pool.
(This name is used later to select from the list of available pools.)
- ▶ **JE Name** - Name of the JE instance that processes the job assigned to this pool.
(Each JE instance takes its name from its configuration file during start-up.)
- ▶ **Maximum threads** - Number of maximum execution threads used to process jobs assigned to this pool.
(This parameter allows limiting the processing power for the pool to allow other pools to be processed at the same time.)
- ▶ **Priority** - Priority of the pool.
JE always tries to process jobs assigned to higher priority pools first, respecting the maximum number of threads utilized for the pool

All parameters can be configured using the Job Pools configuration screen.



Name	Maximum Threads	Priority	Job Executor Name	Last Update On	Last Updated By	Created On	Created By
DEFAULT	10	1	DefaultJobExecutor	09/01/2006 02:18	PRIME	09/01/2006 02:18	PRIME 9.3.0.297
JE2pool1	7	1	JobExecutor2	06/10/2008 11:47	TEST	06/10/2008 11:47	TEST 9.4.0.4869
JE2pool2	8	2	JobExecutor2	06/10/2008 11:47	TEST	06/10/2008 11:47	TEST 9.4.0.4869
pool1	3	1	DefaultJobExecutor	06/10/2008 11:47	TEST	06/10/2008 11:47	TEST 9.4.0.4869
pool2	3	1	DefaultJobExecutor	06/10/2008 11:47	TEST	06/10/2008 11:47	TEST 9.4.0.4869

▼ Welcome to ADM004:

System Administration -
Background Job Processing Online
Training

▼ Chapter 1: Overview of Job Processing

Job Executor (JE)
Job Scheduler (JS)
JE and JS Common Features
Job Executor vs Job Scheduler

▼ Chapter 2: JE Processing Model

JE Processing Model, part 1
JE Processing Model, part 2
JE Processing Model - Sequence
JE Processing Model - Double Retry Mechanism
JE Processing Model - Finishing

JE Self-Monitoring

JE Central Configuration

Job Pool Configuration Screen

Configuration of Job Pools

Sample Configuration of Job Pools - Correct

Sample Configuration of Job Pools - Incorrect

Job Pools - Recommendations

► Chapter 3: JS Processing Model

End of Course

Configuration of Job Pools

Configuration of job pools has an impact on the way jobs are processed and may influence the performance and responsiveness of JE

- ▶ JE can work with a default configuration that contains only one job pool (all jobs are processed in the same way, have equal priorities, and are executed as they come)
- ▶ It is recommended to define separate job pools for jobs requiring different priority of processing. By limiting the number of available threads it is possible to eliminate the potential issue of "starving" lower priority pools

Name	Maximum Threads	Priority	Job Executor Nam	Last Update On	Last Updated By	Created On	Created By
DEFAULT	10	1	DefaultJobExecut	09/01/2006 05:18:	PRIME	09/01/2006 05:18:	PRIME 9.3.0.297
JE2pool1	7	1	JobExecutor2	06/10/2008 02:47:	TEST	06/10/2008 02:47:	TEST 9.4.0.4869
JE2pool2	8	2	JobExecutor2	06/10/2008 02:47:	TEST	06/10/2008 02:47:	TEST 9.4.0.4869
pool1	3	1	DefaultJobExecut	06/10/2008 02:47:	TEST	06/10/2008 02:47:	TEST 9.4.0.4869
pool2	3	1	DefaultJobExecut	06/10/2008 02:47:	TEST	06/10/2008 02:47:	TEST 9.4.0.4869



▼ Welcome to ADM004

System Administration -
Background Job Processing Online
Training

▼ Chapter 1: Overview of Job Processing

Job Executor (JE)
Job Scheduler (JS)
JE and JS Common Features
Job Executor vs Job Scheduler

▼ Chapter 2: JE Processing Model

JE Processing Model, part 1
JE Processing Model, part 2
JE Processing Model - Sequence
JE Processing Model - Double Retry
Mechanism
JE Processing Model - Finishing

JE Self-Monitoring
JE Central Configuration
Job Pool Configuration Screen
Configuration of Job Pools

Sample Configuration of Job Pools -
Correct

Sample Configuration of Job Pools -
Incorrect

Job Pools - Recommendations

► Chapter 3: JS Processing Model

End of Course

Sample Configuration of Job Pools - Correct

The configuration can be done in few way, but the important parameter to consider is MaximumExecutionThreads.

Let's assume, that MaximumExecutionThreads is set as 10. Now, we can divide the slots in different ways:

Example 1: In this configuration "Printing" and "Default" are executed with the same priority. "Printing" is limited to a maximum of 3 concurrent jobs and "Default" is limited to 5. As a result, all pools can be processed at the same time. The "Interfacing" pool can take all threads only if there are no higher priority jobs. This is one of the correct ways.

Pool name	Priority	Threads
Printing	1	3
Default	1	5
Interfacing	3	10

Example 2: In this configuration, the highest priority belongs to "Machine Processing" jobs. They are executed first using a maximum of 2 threads. Remaining available threads (8) are split between "Print" and "Default" and "Interfacing" jobs. This means that all job pools can be executed at the same time, if required. The "Interfacing" pool can take all threads only if there are no higher priority jobs. This also allocates the slots correctly.

Pool name	Priority	Threads
Machine Processing	1	2
Printing	2	3
Default	3	3
Interfacing	4	10



▼ Welcome to ADM004

System Administration -
Background Job Processing Online
Training

▼ Chapter 1: Overview of Job Processing

Job Executor (JE)
Job Scheduler (JS)
JE and JS Common Features
Job Executor vs Job Scheduler

▼ Chapter 2: JE Processing Model

JE Processing Model, part 1
JE Processing Model, part 2
JE Processing Model - Sequence
JE Processing Model - Double Retry Mechanism
JE Processing Model - Finishing

JE Self-Monitoring
JE Central Configuration
Job Pool Configuration Screen
Configuration of Job Pools

Sample Configuration of Job Pools -
Correct

Sample Configuration of Job Pools -
Incorrect

Job Pools - Recommendations

► Chapter 3: JS Processing Model

End of Course

Sample Configuration of Job Pools - Incorrect

Example 3: The below table shows that all possible threads can be taken by the "Printing" pool. This may result in the "Default" and "Interfacing" pools being "starved" and not processed until all "Printing" jobs are completed. This is not recommended, as it doesn't use the slots efficiently.

Pool name	Priority	Threads
Printing	1	10
Default	2	5
Interfacing	3	10

Example 4: In the table below, all possible (10) threads are not distributed. This means that each job pool is processed in the one-job-at-the-time mode. Increasing the number of threads could improve performance. Again, the available slots are wasted in this example.

Pool name	Priority	Threads
Printing	1	1
MI Events	2	1
Default	3	1
Interfacing	4	1

The configuration of job pools requires knowledge about the type of processing expected in the given deployment of DELMIA Apriso. It may also require some experimenting and performance tuning to obtain the best results.

▼ Welcome to ADM004:

System Administration -
Background Job Processing Online
Training

▼ Chapter 1: Overview of Job Processing

Job Executor (JE)
Job Scheduler (JS)
JE and JS Common Features
Job Executor vs Job Scheduler

▼ Chapter 2: JE Processing Model

JE Processing Model, part 1
JE Processing Model, part 2
JE Processing Model - Sequence
JE Processing Model - Double Retry Mechanism
JE Processing Model - Finishing
JE Self-Monitoring

JE Central Configuration

Job Pool Configuration Screen

Configuration of Job Pools

Sample Configuration of Job Pools -
Correct

Sample Configuration of Job Pools -
Incorrect

Job Pools - Recommendations

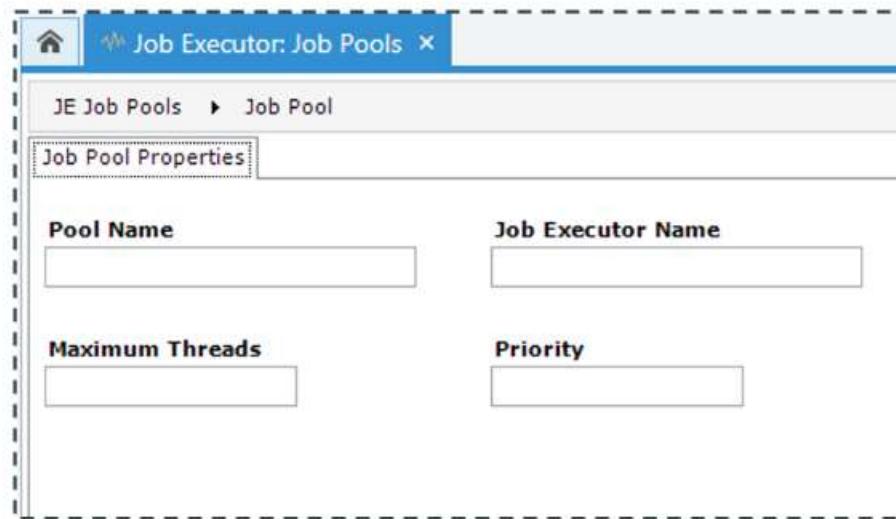
► Chapter 3: JS Processing Model

End of Course

Job Pools - Recommendations

There are some recommendations for Job Pools. Proper configuration of job pools ensures smooth execution of jobs

- ▶ Create several job pools based on different job types
- ▶ Assign priorities based on requirements
- ▶ Do not assign all threads to a single pool
- ▶ Consider the threads as workers:
 - Assign maximum number of workers to a given pool
 - Workers are shared by the pools



Search...	Q
▼ Welcome to ADM004	
System Administration -	
Background Job Processing Online	
Training	
▼ Chapter 1: Overview of Job Processing	
Job Executor (JE)	
Job Scheduler (JS)	
JE and JS Common Features	
Job Executor vs Job Scheduler	
▼ Chapter 2: JE Processing Model	
JE Processing Model, part 1	
JE Processing Model, part 2	
JE Processing Model - Sequence	
JE Processing Model - Double Retry Mechanism	
JE Processing Model - Finishing	
JE Self-Monitoring	
JE Central Configuration	
Job Pool Configuration Screen	
Configuration of Job Pools	
Sample Configuration of Job Pools - Correct	
Sample Configuration of Job Pools - Incorrect	
Job Pools - Recommendations	
▼ Chapter 3: JS Processing Model	
JS Processing Model	

JS Processing Model

Job Scheduler is designed for processing jobs that should be executed at a specific point in time. Each job can be scheduled for a particular moment in time or can have a recurring schedule. It is best used for recurring administration-like tasks.

- ▶ JS periodically scans the JOB table and loads new jobs into memory
- ▶ Load frequency is a configurable option of JS
- ▶ JS can also be notified by a calling component that a new job is created so that JS can load the job without waiting for a full period
- ▶ New jobs are loaded into memory, then JS calculates the nearest execution time for each job based on job schedules
- ▶ When execution time for the job comes, JS publishes the job for execution by inserting it into JOB_EXECUTION_STATUS table
- ▶ The second subsystem of JS loads the action from JOB_EXECUTION_STATUS table and starts the actual execution of the actions using all available threads

Users / Administrators



Job Scheduler
API



DELMIA Apriso components



DELMIA Apriso Tables:

- ▶ JOB
- ▶ JOB_SCHEDULE
- ▶ JOB_ACTION

System Administration -
Background Job Processing Online
Training

▼ Chapter 1: Overview of Job Processing

Job Executor (JE)
Job Scheduler (JS)
JE and JS Common Features
Job Executor vs Job Scheduler

▼ Chapter 2: JE Processing Model

JE Processing Model, part 1
JE Processing Model, part 2
JE Processing Model - Sequence
JE Processing Model - Double Retry
Mechanism
JE Processing Model - Finishing
JE Self-Monitoring
JE Central Configuration
Job Pool Configuration Screen
Configuration of Job Pools
Sample Configuration of Job Pools -
Correct
Sample Configuration of Job Pools -
Incorrect
Job Pools - Recommendations

▼ Chapter 3: JS Processing Model

JS Processing Model
JS Processing Model - Retry

JS Processing Model - Retry

- ▶ JS also supports automatic retry of an action execution when it fails due to database deadlock or concurrency error
- ▶ Each job is executed as a single transaction. If the job execution fails, the whole transaction is rolled back
- ▶ After the job has completed, JS stores history of the execution in tables JOB_HISTORY and JOB_ACTION_HISTORY. Each execution of the job adds new rows into these tables
- ▶ All history rows are related to the single job stored in the JOB table



Background job Processing Online Training

▼ Chapter 1: Overview of Job Processing

- Job Executor (JE)
- Job Scheduler (JS)
- JE and JS Common Features
- Job Executor vs Job Scheduler

▼ Chapter 2: JE Processing Model

- JE Processing Model, part 1
- JE Processing Model, part 2
- JE Processing Model - Sequence
- JE Processing Model - Double Retry Mechanism
- JE Processing Model - Finishing
- JE Self-Monitoring
- JE Central Configuration
- Job Pool Configuration Screen
- Configuration of Job Pools

Sample Configuration of Job Pools - Correct

Sample Configuration of Job Pools - Incorrect

Job Pools - Recommendations

▼ Chapter 3: JS Processing Model

- JS Processing Model
- JS Processing Model - Retry

JS Central Configuration

JS Central Configuration

For proper configuration, it is mandatory to provide the proper data in the Central Configuration file with a path to system services:

```
<add key="SequenceProviderServices" value="REMOTING:net.tcp://${AppAddress}:32601/frameworkservice" />
<add key="SchedulingServices" value="REMOTING:net.tcp://${AppAddress}:32607/JobSchedulerService" />
```

```
<FlexNet.Scheduling>
  <add key="ConfigurationLoaderSleepPeriod" value="300" />
  <add key="NumberOfExecutorThreads" value="5" />
  <add key="MinimalNotificationPeriod" value="10" />
  <add key="NumberOfAutomaticProcessingRetries" value="1" />
  <add key="EnableMaintenanceOldSchedule" value="false" />
</FlexNet.Scheduling>
```



▼ Chapter 1: Overview of Job Processing

- Job Executor (JE)
- Job Scheduler (JS)
- JE and JS Common Features
- Job Executor vs Job Scheduler

▼ Chapter 2: JE Processing Model

- JE Processing Model, part 1
- JE Processing Model, part 2
- JE Processing Model - Sequence
- JE Processing Model - Double Retry Mechanism

JE Processing Model - Finishing

JE Self-Monitoring

JE Central Configuration

Job Pool Configuration Screen

Configuration of Job Pools

Sample Configuration of Job Pools - Correct

Sample Configuration of Job Pools - Incorrect

Job Pools - Recommendations

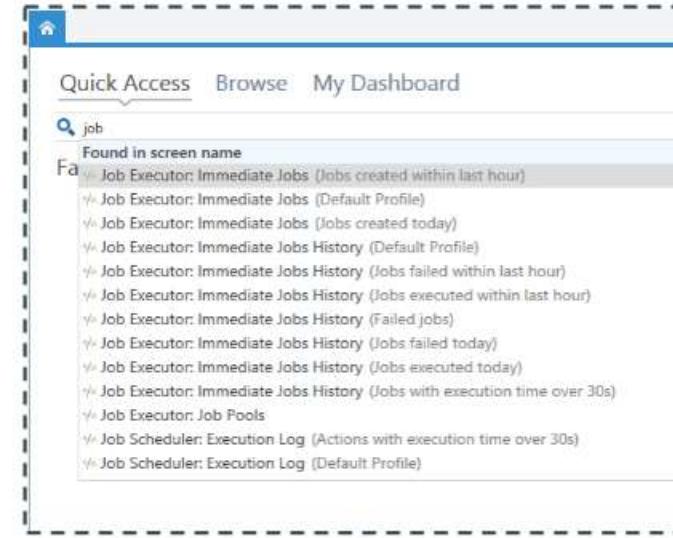
▼ Chapter 3: JS Processing Model

- JS Processing Model
- JS Processing Model - Retry
- JS Central Configuration

Job Executor and Scheduler User Interface

Job Executor and Scheduler User Interface

- ▶ Scheduled Jobs
- ▶ Jobs Currently Executed
- ▶ Synchronization Queues Status
- ▶ XML Messages
- ▶ Execution Log
- ▶ Job Pools
- ▶ Immediate Jobs
- ▶ Immediate Jobs History



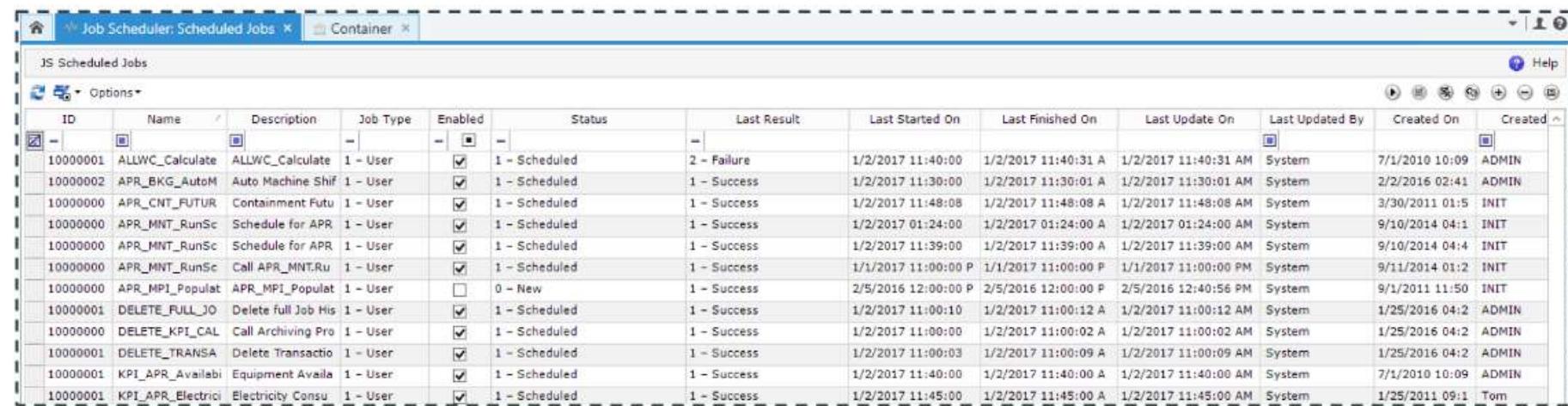
Q

- Job Executor (JE)
- Job Scheduler (JS)
- JE and JS Common Features
- Job Executor vs Job Scheduler
- Chapter 2: JE Processing Model
 - JE Processing Model, part 1
 - JE Processing Model, part 2
 - JE Processing Model - Sequence
 - JE Processing Model - Double Retry Mechanism
 - JE Processing Model - Finishing
 - JE Self-Monitoring
 - JE Central Configuration
 - Job Pool Configuration Screen
 - Configuration of Job Pools
 - Sample Configuration of Job Pools - Correct
 - Sample Configuration of Job Pools - Incorrect
 - Job Pools - Recommendations
- Chapter 3: JS Processing Model
 - JS Processing Model
 - JS Processing Model - Retry
 - JS Central Configuration
 - Job Executor and Scheduler User Interface

Scheduled Jobs

Scheduled Jobs

- Used by JS
- Allows creation of user defined jobs:
- Defines job schedules
- Defines job actions
- Displays all user defined jobs
- Enables and disables jobs
- Executes jobs manually:
- Does not need a schedule to do this
- Does not display jobs related to XML Messages originating from BI



The screenshot shows a software application window titled "Job Scheduler: Scheduled Jobs". The main area displays a table of "JS Scheduled Jobs" with the following columns: ID, Name, Description, Job Type, Enabled, Status, Last Result, Last Started On, Last Finished On, Last Update On, Last Updated By, Created On, and Created. There are 14 rows of data, each representing a scheduled job with its details. The table has a header row and several data rows.

ID	Name	Description	Job Type	Enabled	Status	Last Result	Last Started On	Last Finished On	Last Update On	Last Updated By	Created On	Created
10000001	ALLWC_Calculate	ALLWC_Calculate	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	2 - Failure	1/2/2017 11:40:00	1/2/2017 11:40:31 A	1/2/2017 11:40:31 AM	System	7/1/2010 10:09	ADMIN
10000002	APR_BKG_AutoM	Auto Machine Shif	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:30:00	1/2/2017 11:30:01 A	1/2/2017 11:30:01 AM	System	2/2/2016 02:41	ADMIN
10000000	APR_CNT_FUTUR	Containment Futu	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:48:08	1/2/2017 11:48:08 A	1/2/2017 11:48:08 AM	System	3/30/2011 01:5	INIT
10000000	APR_MNT_RunSc	Schedule for APR	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 01:24:00	1/2/2017 01:24:00 A	1/2/2017 01:24:00 AM	System	9/10/2014 04:1	INIT
10000000	APR_MNT_RunSc	Schedule for APR	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:39:00	1/2/2017 11:39:00 A	1/2/2017 11:39:00 AM	System	9/10/2014 04:4	INIT
10000000	APR_MNT_RunSc	Call APR_MNT.Ru	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/1/2017 11:00:00 P	1/1/2017 11:00:00 P	1/1/2017 11:00:00 PM	System	9/11/2014 01:2	INIT
10000000	APR_MPI_Populat	APR_MPI_Populat	1 - User	<input type="checkbox"/>	0 - New	1 - Success	2/5/2016 12:00:00 P	2/5/2016 12:00:00 P	2/5/2016 12:40:56 PM	System	9/1/2011 11:50	INIT
10000001	DELETE_FULL_JO	Delete full Job His	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:00:10	1/2/2017 11:00:12 A	1/2/2017 11:00:12 AM	System	1/25/2016 04:2	ADMIN
10000000	DELETE_KPI_CAL	Call Archiving Pro	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:00:00	1/2/2017 11:00:02 A	1/2/2017 11:00:02 AM	System	1/25/2016 04:2	ADMIN
10000001	DELETE_TRANSA	Delete Transactio	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:00:03	1/2/2017 11:00:09 A	1/2/2017 11:00:09 AM	System	1/25/2016 04:2	ADMIN
10000001	KPI_APP_Availabi	Equipment Availa	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:40:00	1/2/2017 11:40:00 A	1/2/2017 11:40:00 AM	System	7/1/2010 10:09	ADMIN
10000001	KPI_APP_Electric	Electricity Consu	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:45:00	1/2/2017 11:45:00 A	1/2/2017 11:45:00 AM	System	1/25/2011 09:1	Tom

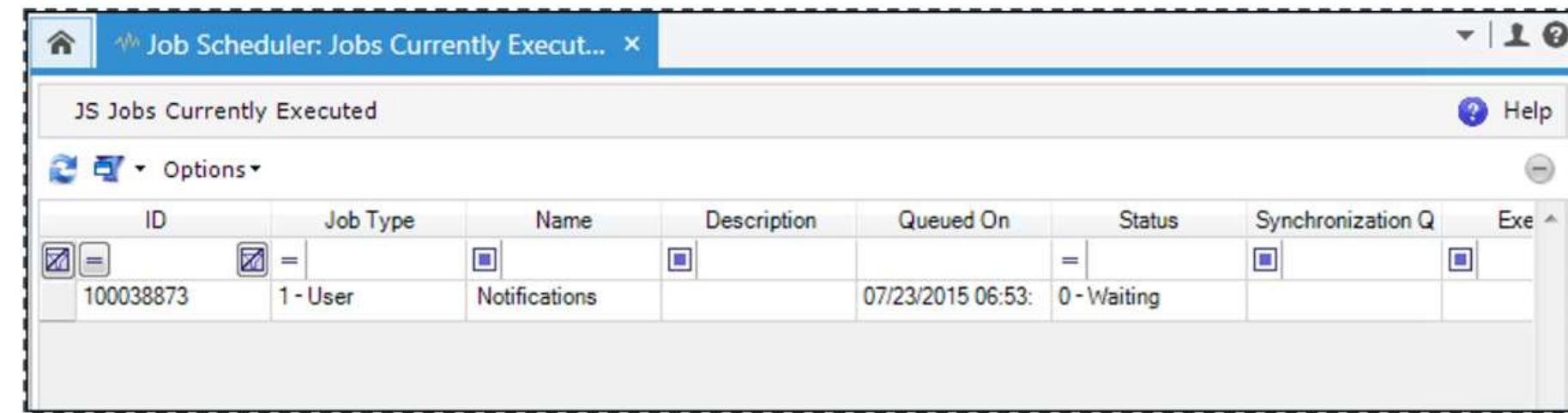
Search... 

- Job Scheduler (JS)
- JE and JS Common Features
- Job Executor vs Job Scheduler
- Chapter 2: JE Processing Model
 - JE Processing Model, part 1
 - JE Processing Model, part 2
 - JE Processing Model - Sequence
 - JE Processing Model - Double Retry Mechanism
 - JE Processing Model - Finishing
 - JE Self-Monitoring
 - JE Central Configuration
 - Job Pool Configuration Screen
 - Configuration of Job Pools
 - Sample Configuration of Job Pools - Correct
 - Sample Configuration of Job Pools - Incorrect
- Job Pools - Recommendations
- Chapter 3: JS Processing Model
 - JS Processing Model
 - JS Processing Model - Retry
 - JS Central Configuration
 - Job Executor and Scheduler User Interface
 - Scheduled Jobs
- Jobs Currently Executed

Jobs Currently Executed

The list of jobs currently executed can be found on the Jobs Currently Executed screen accessible from Job Scheduler main screen:

- ▶ Used by JS
- ▶ Displays jobs that are currently being executed



ID	Job Type	Name	Description	Queued On	Status	Synchronization Q	Exe
100038873	1 - User	Notifications		07/23/2015 06:53:	0 - Waiting		

JE and JS Common Features

Job Executor vs Job Scheduler

Chapter 2: JE Processing Model

JE Processing Model, part 1

JE Processing Model, part 2

JE Processing Model - Sequence

JE Processing Model - Double Retry Mechanism

JE Processing Model - Finishing

JE Self-Monitoring

JE Central Configuration

Job Pool Configuration Screen

Configuration of Job Pools

Sample Configuration of Job Pools - Correct

Sample Configuration of Job Pools - Incorrect

Job Pools - Recommendations

Chapter 3: JS Processing Model

JS Processing Model

JS Processing Model - Retry

JS Central Configuration

Job Executor and Scheduler User Interface

Scheduled Jobs

Jobs Currently Executed

Synchronization Queues Status

Synchronization Queues Status

The Synchronization Queue Status grid displays all the disabled queues (and only the disabled ones) together with the Disabled On and Disabled Until dates.

- ▶ Used by JS
- ▶ Allows user to manually disable a synchronization queue
- ▶ Can specify start and stop times for disabling the queue
- ▶ When disabled, pauses the processing of all messages assigned to the queue

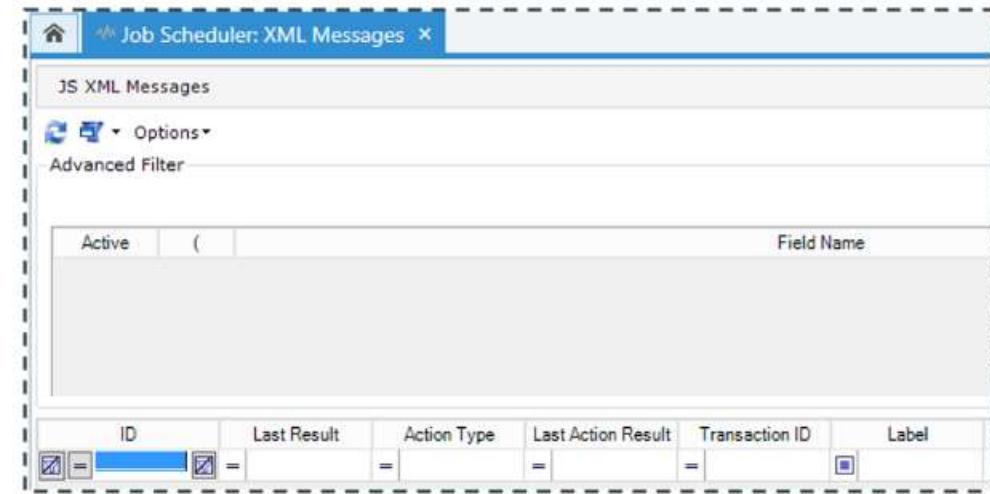
The screenshot shows a software interface titled "JS Synchronization Queues Status". At the top right are buttons for Help, zoom, and window control. Below the title is a toolbar with icons for refresh, search, and options, which is currently set to "Options". The main area is a grid table with three columns: "Synchronization Queue", "Disabled On", and "Disabled Until". There are two rows in the grid. The first row is highlighted with a blue selection bar and contains the text "SAPBCAdapter" in the first column, and "08/28/2015 01:10" in both the second and third columns. The second row contains the text "SAPBCAdapter_Production" in the first column, and "08/28/2015 01:10" in both the second and third columns. The entire grid has a dashed border.

Synchronization Queue	Disabled On	Disabled Until
SAPBCAdapter	08/28/2015 01:10	08/28/2015 01:10
SAPBCAdapter_Production	08/28/2015 01:10	08/28/2015 01:10

[Job Executor vs Job Scheduler](#)**Chapter 2: JE Processing Model**[JE Processing Model, part 1](#)[JE Processing Model, part 2](#)[JE Processing Model - Sequence](#)[JE Processing Model - Double Retry Mechanism](#)[JE Processing Model - Finishing](#)[JE Self-Monitoring](#)[JE Central Configuration](#)[Job Pool Configuration Screen](#)[Configuration of Job Pools](#)[Sample Configuration of Job Pools - Correct](#)[Sample Configuration of Job Pools - Incorrect](#)[Job Pools - Recommendations](#)**Chapter 3: JS Processing Model**[JS Processing Model](#)[JS Processing Model - Retry](#)[JS Central Configuration](#)[Job Executor and Scheduler User Interface](#)[Scheduled Jobs](#)[Jobs Currently Executed](#)[Synchronization Queues Status](#)[XML Messages Screen](#)

XML Messages Screen

- ▶ Displays all jobs related to XML messages originating from BI
- ▶ Mainly used for backward compatibility
- ▶ Displays the history of executed jobs
- ▶ Allows re-executing of jobs
- ▶ Allows viewing of job details



<input type="text" value="Search..."/>	
Chapter 2: JE Processing Model	
JE Processing Model, part 1	
JE Processing Model, part 2	
JE Processing Model - Sequence	
JE Processing Model - Double Retry Mechanism	
JE Processing Model - Finishing	
JE Self-Monitoring	
JE Central Configuration	
Job Pool Configuration Screen	
Configuration of Job Pools	
Sample Configuration of Job Pools - Correct	
Sample Configuration of Job Pools - Incorrect	
Job Pools - Recommendations	
Chapter 3: JS Processing Model	
JS Processing Model	
JS Processing Model - Retry	
JS Central Configuration	
Job Executor and Scheduler User Interface	
Scheduled Jobs	
Jobs Currently Executed	
Synchronization Queues Status	
XML Messages Screen	
Execution Log	

Execution Log

- ▶ Used by JS
- ▶ Displays history of executed job actions
- ▶ Allows re-executing job actions
- ▶ Allows viewing job action details

The screenshot shows a software window titled "Job Scheduler: Execution Log". The window has a toolbar with icons for home, search, and help. Below the toolbar is a menu bar with "Options" and "Grid profile: Default Profile". The main area is a table titled "Execution Log" with the following columns: Job Name, Job Description, Job Type, Action Name, Action Description, Action Type, and Result. There are five rows of data:

Job Name	Job Description	Job Type	Action Name	Action Description	Action Type	Result
APR_MPI_Popula	APR_MPI_Popula	1 - User	InvokeAPR_MPI_	APR_MPI_Popula	1 - Invoke Standar	2 - Failure
APR_CNT_FUTU	Containment FUTU	1 - User	Enqueue	Enqueue	1 - Invoke Standar	1 - Success
APR_CNT_FUTU	Containment FUTU	1 - User	Process	Process	1 - Invoke Standar	1 - Success
APR_CNT_FUTU	Containment FUTU	1 - User	Enqueue	Enqueue	1 - Invoke Standar	1 - Success
APR_CNT_FUTU	Containment FUTU	1 - User	Process	Process	1 - Invoke Standar	1 - Success

At the bottom of the table, there is a message: "Scroll down to load next page."

Search...	Q
JE Processing Model, part 1	
JE Processing Model, part 2	
JE Processing Model - Sequence	
JE Processing Model - Double Retry Mechanism	
JE Processing Model - Finishing	
JE Self-Monitoring	
JE Central Configuration	
Job Pool Configuration Screen	
Configuration of Job Pools	
Sample Configuration of Job Pools - Correct	
Sample Configuration of Job Pools - Incorrect	
Job Pools - Recommendations	
Chapter 3: JS Processing Model	
JS Processing Model	
JS Processing Model - Retry	
JS Central Configuration	
Job Executor and Scheduler User Interface	
Scheduled Jobs	
Jobs Currently Executed	
Synchronization Queues Status	
XML Messages Screen	
Execution Log	
Immediate Jobs History	

Immediate Jobs History

- ▶ Used by JE
- ▶ Displays JE History
- ▶ Allows viewing of failed and processed jobs
- ▶ Commonly used to view failed integration messages
- ▶ Allows viewing and editing of detailed job properties
- ▶ Allows two ways to reprocess jobs:
 - Reprocess
 - Edit and reprocess



JE Immediate Jobs History										Help	
		Options		Grid profile: Default Profile							
ID	Job ID	Name	Description	Created Date	Pool	Synchronization	Action Type	Timeout			
25363758	0	FI Invocation fr	100000031	03/18/2016 09:	DEFAULT				1 - Invoke Stan	00:30:00	
25363755	0	AttoERP	ATORDER	03/02/2016 10:	DEFAULT				8 - XML - Proce	00:05:00	
25363754	0	AttoERP	ATORDER	03/02/2016 09:	DEFAULT				8 - XML - Proce	00:05:00	

JE Processing Model, part 2
JE Processing Model - Sequence
JE Processing Model - Double Retry Mechanism
JE Processing Model - Finishing
JE Self-Monitoring
JE Central Configuration
Job Pool Configuration Screen
Configuration of Job Pools
Sample Configuration of Job Pools - Correct
Sample Configuration of Job Pools - Incorrect

Job Pools - Recommendations

▼ Chapter 3: JS Processing Model

JS Processing Model
JS Processing Model - Retry
JS Central Configuration
Job Executor and Scheduler User Interface
Scheduled Jobs
Jobs Currently Executed
Synchronization Queues Status
XML Messages Screen
Execution Log
Immediate Jobs History

Logging

Logging

As with every other application within DELMIA Apriso, the Job Executor and Job Scheduler keep track of all activities in execution logs. By default these logs can be found in the C:\Temp\AprisoLogs\directory.

- ▶ JE and JS log all activity using the DELMIA Apriso logging framework
- ▶ The C:\Temp\AprisoLogs\ directory contains the following log files:
 - AprisoJobExecutorService_*.log for JE
 - AprisoSchedulingService_*.log for JS
- ▶ Commonly used to find more detailed error messages for failed jobs
- ▶ Performance counters can be used to observe the health of the system

Name	Date modified	Type	Size
AprisoArchiving_ErrorRolling	6/9/2016 2:00 AM	Text Document	3 KB
AprisoArchiving_InfoRolling	6/9/2016 2:00 AM	Text Document	6 KB
AprisoArchiving_WarningRolling	6/9/2016 2:00 AM	Text Document	3 KB
AprisoJobSchedulerService_InfoRolling	6/5/2016 5:00 AM	Text Document	0 KB
AprisoJobSchedulerService_InfoRolling.lo...	6/5/2016 5:00 AM	1 File	10,241 KB

Search... 

JE Processing Model - Sequence

JE Processing Model - Double Retry Mechanism

JE Processing Model - Finishing

JE Self-Monitoring

JE Central Configuration

Job Pool Configuration Screen

Configuration of Job Pools

Sample Configuration of Job Pools - Correct

Sample Configuration of Job Pools - Incorrect

Job Pools - Recommendations

Chapter 3: JS Processing Model

JS Processing Model

JS Processing Model - Retry

JS Central Configuration

Job Executor and Scheduler User Interface

Scheduled Jobs

Jobs Currently Executed

Synchronization Queues Status

XML Messages Screen

Execution Log

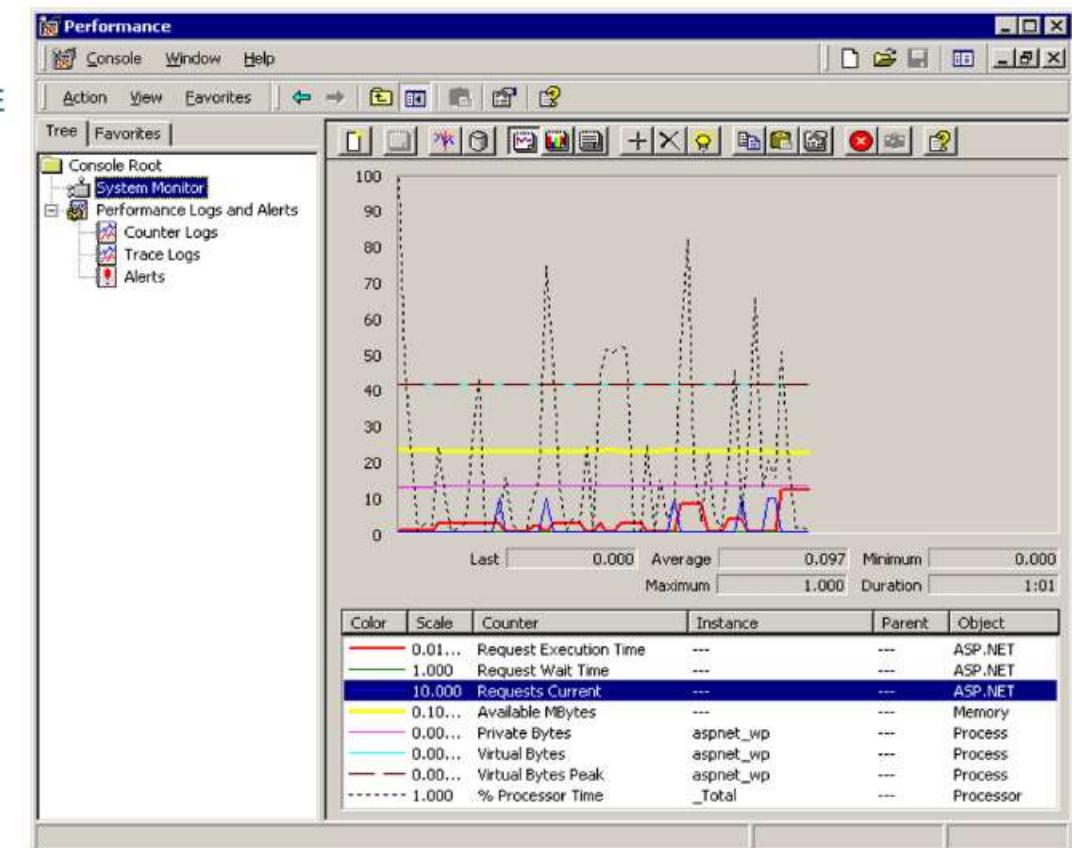
Immediate Jobs History

Logging

Performance Counters

Performance Counters

- ▶ DELMIA Apriso offers a choice of several performance counters that can be used in the Windows Performance Monitor to observe the performance of JE
- ▶ These counters can help users identify bottlenecks, reduce inefficiency, and boost productivity
- ▶ Examples of performance counter values for Background Processing functionality are published by DELMIA Apriso:
 - Scheduler Active Executor Threads
 - Scheduler Job Execution Duration
 - Scheduler Job Executions
 - Scheduler Job Executions/sec
 - Scheduler Job Executions (successful)
 - Scheduler new jobs
 - Executor Job Execution Duration
 - Executor Successful Job Executions/sec
 - Executor new jobs
 - Executor Active Executor Threads



JE Processing Model - Double Retry Mechanism

JE Processing Model - Finishing

JE Self-Monitoring

JE Central Configuration

Job Pool Configuration Screen

Configuration of Job Pools

Sample Configuration of Job Pools - Correct

Sample Configuration of Job Pools - Incorrect

Job Pools - Recommendations

▼ Chapter 3: JS Processing Model

JS Processing Model

JS Processing Model - Retry

JS Central Configuration

Job Executor and Scheduler User Interface

Scheduled Jobs

Jobs Currently Executed

Synchronization Queues Status

XML Messages Screen

Execution Log

Immediate Jobs History

Logging

Performance Counters

Troubleshooting

Troubleshooting

How do I know whether JE has stopped running or is just running slowly?

- ▶ Check logs - if there is a new activity then it is running. If it has stopped, then restart the JE service. If it is doing something then check the Immediate Jobs - a job may be blocking the synchronization queue if it is set to retry 5 times and wait 5 minutes between retries. This effectively blocks the queue for 25 minutes

How do I remove a job from the queue if its status is 'running' ?

- ▶ Stop the JE service and then remove the job. Do it either via the Immediate Jobs screen or directly in the database. It has to be agreed with the client so that they resubmit the job again

If I have multiple JE instances running on my server (not very frequent), how do I know which one has the job that is causing trouble?

- ▶ The Job Pools screen shows the relationship between Job Pools and JE instances (installing additional JE instances is rare and requires DELMIA Apriso Support help)

[Job Executor and Scheduler User Interface](#)[JE Self-Monitoring](#)[JE Central Configuration](#)[Job Pool Configuration Screen](#)[Configuration of Job Pools](#)[Sample Configuration of Job Pools - Correct](#)[Sample Configuration of Job Pools - Incorrect](#)[Job Pools - Recommendations](#)**▼ Chapter 3: JS Processing Model**[JS Processing Model](#)[JS Processing Model - Retry](#)[JS Central Configuration](#)[Job Executor and Scheduler User Interface](#)[Scheduled jobs](#)[Jobs Currently Executed](#)[Synchronization Queues Status](#)[XML Messages Screen](#)[Execution Log](#)[Immediate Jobs History](#)[Logging](#)[Performance Counters](#)[Troubleshooting](#)[LAB 1: Create a User Defined Job](#)[LAB 1: Create a User Defined Job](#)

LAB 1: Create a User Defined Job

Task:

- ▶ **Create a job**
- ▶ **Add a job action**
- ▶ **Run the job**
- ▶ **Schedule the job**

What you will learn:

- ▶ **How to create a job,**
- ▶ **How to run it manually**
- ▶ **How to schedule it to run on a re-occurring basis**

Training environment:

- ▶ In case of any technical problems, please contact DELMIA.Apriso.training@3ds.com



Remember to use the following to login and name Screens thorough this entire training:

- TRN<yourinitials> if your are an external self-paced learner
- TRN<yourtrigram> if you are a 3DS employee self-paced learner



20 min

ID	Name	Description	Job Type	Enabled	Status	Last Result	Last Started On	Last Finished On	Last update On
10000001	ALUIC_Calculate	ALUIC_Calculate	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	2 - Failure	1/2/2017 11:30:02	1/2/2017 11:30:02 A	1/2/2017 11:30:02 AM
10000002	APR_BNC_AutoH	Auto Machine Shift	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:30:03	1/2/2017 11:30:01 A	1/2/2017 11:30:01 AM
10000003	APR_CMT_FUTUR	Condimentum Futur	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:39:08	1/2/2017 11:39:08 A	1/2/2017 11:39:08 AM
10000004	APR_HNT_Rule01	Schedule for APR	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 01:24:00	1/2/2017 01:24:00 A	1/2/2017 01:24:00 AM
10000005	APR_HNT_Rule02	Schedule for APR	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:36:00	1/2/2017 11:36:00 A	1/2/2017 11:36:00 AM
10000006	APR_HNT_Rule03	Call APR_PRINTAU	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:00:00 P	1/2/2017 11:00:00 P	1/2/2017 11:00:00 PM
10000007	APR_MPL_Populate	APR_MPL_Populate	1 - User	<input type="checkbox"/>	0 - New	1 - Success	2/6/2018 12:00:00 P	2/6/2018 12:00:00 P	12:40:00 PM
10000008	DELETE_APF_FULL_ID	Delete full info ID	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:00:18	1/2/2017 11:00:13 A	1/2/2017 11:00:13 AM
10000009	DELETE_APF_CAL	Call Archiving Proc	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:00:24	1/2/2017 11:00:24 A	1/2/2017 11:00:24 AM
10000010	DELETE_TRANSA	Delete transaction	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:00:03	1/2/2017 11:00:09 A	1/2/2017 11:00:09 AM
10000011	KPI_APB_Available	Equipment Available	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:30:08	1/2/2017 11:30:02 A	1/2/2017 11:30:02 AM
10000012	KPI_APB_Electricity	Electricity Counter	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:35:08	1/2/2017 11:35:00 A	1/2/2017 11:35:00 AM
10000013	KPI_APB_GasCon	Gas Consumption	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:35:09	1/2/2017 11:35:00 A	1/2/2017 11:35:00 AM
10000014	KPI_APB_HTBF	Mean Time Before	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:00:01	1/2/2017 11:00:01 A	1/2/2017 11:00:01 AM
10000015	KPI_APB_MTTR	Mean Time To Re	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	2 - Failure	1/2/2017 11:00:01	1/2/2017 11:00:01 A	1/2/2017 11:00:02 AM
10000016	KPI_APB_OEE	Overall Equipment	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:30:08	1/2/2017 11:30:09 A	1/2/2017 11:30:09 AM
10000017	KPI_APB_Perfom	Equipment Perform	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:30:32	1/2/2017 11:30:32 A	1/2/2017 11:30:32 AM
10000018	KPI_APB_Quality	Equipment Quality	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:30:04	1/2/2017 11:30:00 A	1/2/2017 11:30:00 AM
10000019	KPI_APB_Retainance	Retainance	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:38:08	1/2/2017 11:38:08 A	1/2/2017 11:38:08 AM
10000020	KPI_APB_Notifications	Notifications	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:39:00	1/2/2017 11:39:00 A	1/2/2017 11:39:00 AM
10000021	OHOpenEndAlert	Close old open End	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 12:00:00	1/2/2017 12:00:00 A	1/2/2017 12:00:00 AM
10000022	Hact HPI_DIM_D	Hact HPI DIM_D	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 09:01:08	1/2/2017 09:01:08 A	1/2/2017 09:01:08 AM
10000023	Process_QAP_Tee		1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 01:05:09	1/2/2017 01:05:22 A	1/2/2017 01:05:22 AM
10000024	Process_Plant_MP1	Process Plant MP1	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:25:25	1/2/2017 11:27:05 A	1/2/2017 11:27:05 AM
10000025	QHS_Inspection_3	QHS Inspection 3	1 - User	<input type="checkbox"/>	0 - New	1 - Success	10/12/2018 02:21:00	10/12/2018 02:21:00	10/12/2018 02:21:00
10000026	TRN_Task	TRN_Task	1 - User	<input checked="" type="checkbox"/>	1 - Recurrent	2 - Failure	1/2/2017 11:25:06	1/2/2017 11:25:00 A	1/2/2017 11:25:00 AM
10000027	UPDATE_UPDATI	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	12/31/2016 06:00:00	12/31/2016 06:00:15	12/31/2016 06:00:15	

- [JE Central Configuration](#)
- [Job Pool Configuration Screen](#)
- [Configuration of Job Pools](#)
- [Sample Configuration of Job Pools - Correct](#)
- [Sample Configuration of Job Pools - Incorrect](#)
- [Job Pools - Recommendations](#)
- [Chapter 3: JS Processing Model](#)
 - [JS Processing Model](#)
 - [JS Processing Model - Retry](#)
 - [JS Central Configuration](#)
 - [Job Executor and Scheduler User Interface](#)
 - [Scheduled Jobs](#)
 - [Jobs Currently Executed](#)
 - [Synchronization Queues Status](#)
 - [XML Messages Screen](#)
 - [Execution Log](#)
 - [Immediate Jobs History](#)
 - [Logging](#)
 - [Performance Counters](#)
 - [Troubleshooting](#)
- [LAB 1: Create a User Defined Job](#)
- [LAB 1: Create a User Defined Job](#)
- [LAB 1: Create TRNXX_Job](#)

LAB 1: Create TRNXX_Job

- ▶ Go to Desktop Client and open the Job Scheduler: **Scheduled Jobs M&M Screen**
- ▶ Create a new job called **TRNXX_Job**

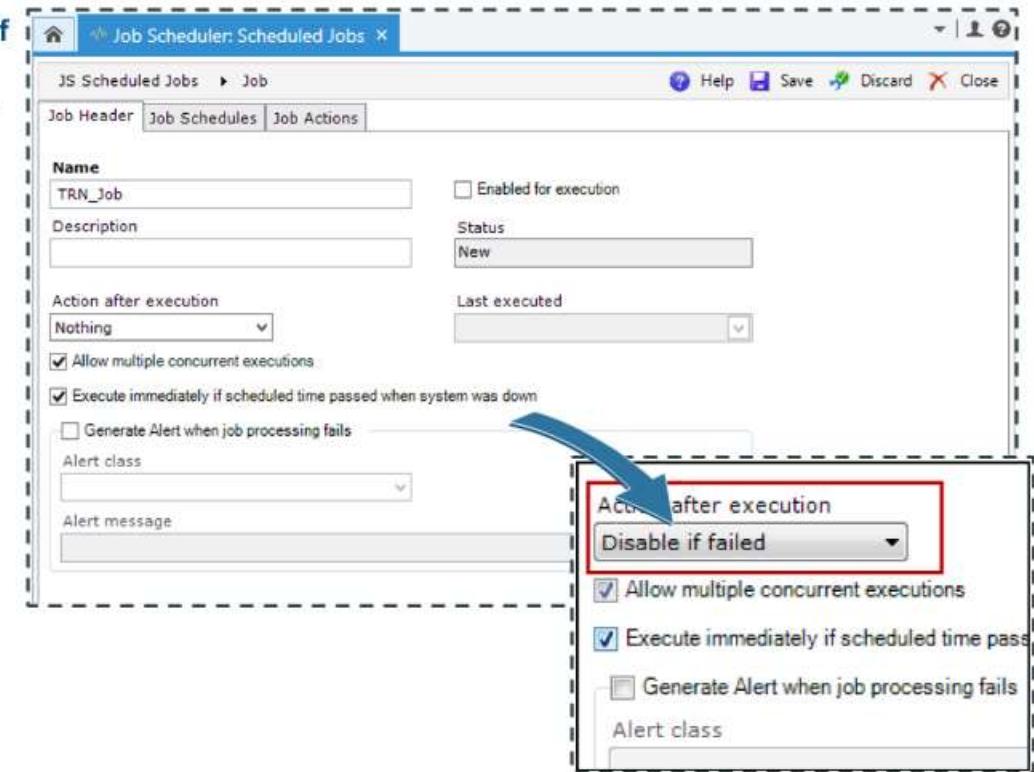
The screenshot shows the desktop client interface with a navigation bar at the top. The 'Browse' tab is selected. The left sidebar contains links for various system components like 3D Experience Widgets Menu, Administration, Monitoring Tools, Manufacturing Intelligence, Services Library, and Inventory Coverage. On the right, there's a list of links under 'Job Scheduler' such as Maintenance Cockpit, Dispatching Board, Szymon Application, Transaction History, Background Jobs, Electronic Signatures, EMR Monitoring, EPC/RFID Cockpit, and Genealogy. A red box highlights the 'Job Scheduler: Scheduled Jobs' link. Below the navigation bar, a table titled 'JS Scheduled Jobs' lists several scheduled jobs with columns for ID, Name, Description, Job Type, Enabled, Status, Last Result, Last Started On, Last Finished On, and Last Update On. One row in the table is highlighted with a red box around the 'Name' column, indicating where to enter the job name.

ID	Name	Description	Job Type	Enabled	Status	Last Result	Last Started On	Last Finished On	Last Update On
10000000	APR_MPI_Populat	APR_MPI_Populat	1 - User	<input type="checkbox"/>	0 - New	2 - Failure	7/20/2015 06:00	7/20/2015 06:01	7/20/2015 06:2
10000000	APR_CNT_FUTUR	Containment Futa	1 - User	<input type="checkbox"/>	0 - New	1 - Success	7/20/2015 06:26	7/20/2015 06:26	7/20/2015 06:2
10000000	DELETE_KPI_CAL	Call Archiving Pro	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	3/5/2016 05:00	3/5/2016 05:00	3/21/2016 02:10
10000000	DELETE_FULL_JO	Delete full Job His	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	3/5/2016 05:00	3/5/2016 05:00	3/21/2016 02:10
10000000	DELETE_TRANSA	Delete Transaction	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	3/5/2016 05:00	3/5/2016 05:00	3/21/2016 02:10
10000000	ALLWC_Calculate	ALLWC_Calculate	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/1/2016 07:00	1/1/2016 07:00	3/21/2016 02:12
10000000	KPI_APP_Quality	Equipment Qualit	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	2 - Failure	1/1/2016 07:00	1/1/2016 07:00	3/21/2016 02:12

- Configuration of Job Pools
- Sample Configuration of Job Pools - Correct
- Sample Configuration of Job Pools - Incorrect
- Job Pools - Recommendations
- Chapter 3: JS Processing Model
 - JS Processing Model
 - JS Processing Model - Retry
 - JS Central Configuration
 - Job Executor and Scheduler User Interface
 - Scheduled jobs
 - Jobs Currently Executed
 - Synchronization Queues Status
 - XML Messages Screen
 - Execution Log
 - Immediate Jobs History
 - Logging
 - Performance Counters
 - Troubleshooting
- LAB 1: Create a User Defined Job
- LAB 1: Create a User Defined Job
- LAB 1: Create TRNXX_job
- LAB 1: Set Action After Execution**
- LAB 1: Create TRNXX_jobAction

LAB 1: Set Action After Execution

- ▶ Set the Action after Execution setting to **Disable if failed**
- ▶ Save your changes, but do not close this window



Search... 

Configuration of Job Pools

Sample Configuration of Job Pools - Correct

Sample Configuration of Job Pools - Incorrect

Job Pools - Recommendations

Chapter 3: JS Processing Model

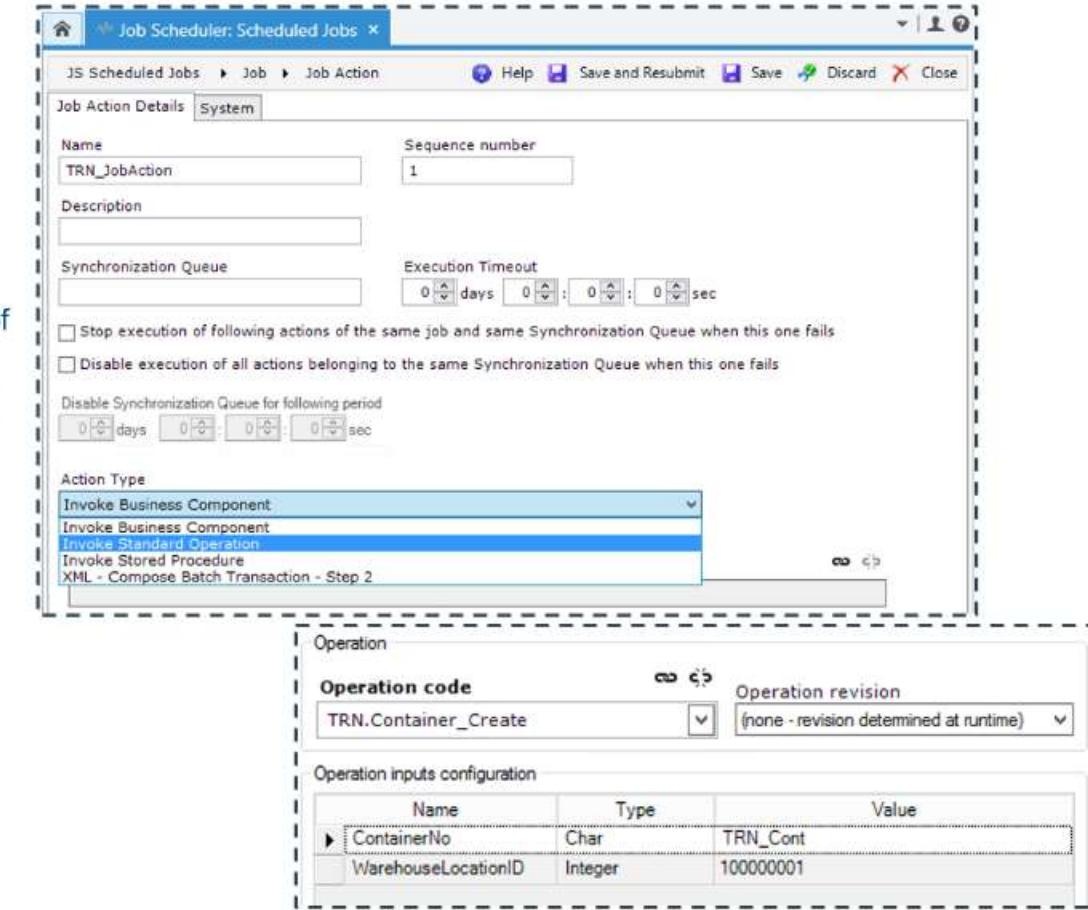
- JS Processing Model
- JS Processing Model - Retry
- JS Central Configuration
- Job Executor and Scheduler User Interface
- Scheduled jobs
- Jobs Currently Executed
- Synchronization Queues Status
- XML Messages Screen
- Execution Log
- Immediate Jobs History
- Logging
- Performance Counters
- Troubleshooting
- LAB 1: Create a User Defined Job
- LAB 1: Create a User Defined Job
- LAB 1: Create TRNXX_Job
- LAB 1: Set Action After Execution

LAB 1: Create TRNXX_JobAction

LAB 1: Create TRNXX_JobAction

- ▶ Select the Job Actions tab
- ▶ Create a new Job Action
- ▶ Name your Job Action **TRNXX_JobAction**
- ▶ Set the Action Type to **Invoke Standard Operation**
- ▶ Assign the Standard Operation **TRN.Container_Create** to the Operation Code
- ▶ Add the Value **TRNXX_Cont** to the Input Name of ContainerNo
- ▶ Add the Value **100000001** to the Input Name of WarehouseLocationID

This Operation will create a new Container named **TRNXX_Cont**.



The screenshot shows two windows of the Job Scheduler interface. The top window is titled 'Job Scheduler: Scheduled Jobs' and displays the 'Job Action Details' tab. It shows a 'Name' field set to 'TRN_JobAction' and a 'Sequence number' field set to '1'. Below these are fields for 'Description', 'Synchronization Queue', and 'Execution Timeout' (set to 0 days, 0 hours, 0 minutes, 0 seconds). There are also two checkboxes: 'Stop execution of following actions of the same job and same Synchronization Queue when this one fails' and 'Disable execution of all actions belonging to the same Synchronization Queue when this one fails'. The bottom window is titled 'Operation' and shows the 'Operation code' field set to 'TRN.Container_Create' and the 'Operation revision' field set to '(none - revision determined at runtime)'. Under 'Operation inputs configuration', there is a table with two rows:

Name	Type	Value
ContainerNo	Char	TRN_Cont
WarehouseLocationID	Integer	100000001

- Search... 
- Sample Configuration of Job Pools - Correct
- Sample Configuration of Job Pools - Incorrect
- Job Pools - Recommendations
- Chapter 3: JS Processing Model
- JS Processing Model
 - JS Processing Model - Retry
 - JS Central Configuration
 - Job Executor and Scheduler User Interface
 - Scheduled Jobs
 - Jobs Currently Executed
 - Synchronization Queues Status
 - XML Messages Screen
 - Execution Log
 - Immediate Jobs History
 - Logging
 - Performance Counters
 - Troubleshooting
- LAB 1: Create a User Defined Job
- LAB 1: Create a User Defined Job
- LAB 1: Create TRNXX_Job
- LAB 1: Set Action After Execution
- LAB 1: Create TRNXX_JobAction
- LAB 1: Assign the Employee

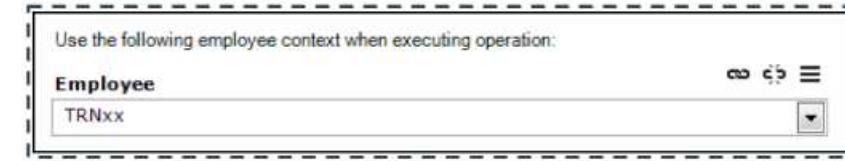
LAB 1: Assign the Employee

- ▶ Assign the **Employee Admin** (yourself)
- ▶ Save the changes
- ▶ Close both, the Job Action Details screen and the Job Actions screen so that you are back to the list of all Jobs in the system

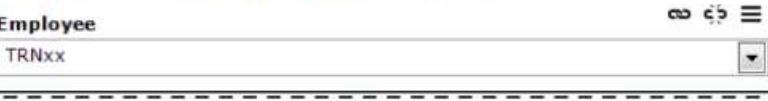


Employee

TRNXX

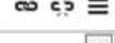


Use the following employee context when executing operation:



Employee

TRNXX



Search...	
Correct	
Sample Configuration of Job Pools - Incorrect	
Job Pools - Recommendations	
▼ Chapter 3: JS Processing Model	
JS Processing Model	
JS Processing Model - Retry	
JS Central Configuration	
Job Executor and Scheduler User Interface	
Scheduled Jobs	
Jobs Currently Executed	
Synchronization Queues Status	
XML Messages Screen	
Execution Log	
Immediate Jobs History	
Logging	
Performance Counters	
Troubleshooting	
LAB 1: Create a User Defined Job	
LAB 1: Create a User Defined Job	
LAB 1: Create TRNXX_Job	
LAB 1: Set Action After Execution	
LAB 1: Create TRNXX_JobAction	
LAB 1: Assign the Employee	
LAB 1: Run Your Job	

LAB 1: Run Your Job

- ▶ Find your job in the list of Jobs and select it
- ▶ Enable your job
- ▶ Run your job

The figure consists of two screenshots of a software interface titled "JS Scheduled Jobs". Both screenshots show a table with columns: ID, Name, Description, Job Type, Enabled, Status, Last Run, and Last Started On.

Screenshot 1 (Top): The "Enabled" column for all jobs is checked (checked boxes). The "Status" column shows various states: Success, New, and Scheduled. The "Last Run" column shows the date and time of the last successful run. The "Last Started On" column shows the date and time of the last start.

ID	Name	Description	Job Type	Enabled	Status	Last Run	Last Started On
10000000	APR_MPI_Populat	APR_MPI_Populat	1 - User	<input checked="" type="checkbox"/>	1 - Schedu	1 - Success	8/19/2015 03:00
10000000	APR_CNT_FUTUR	Containment Futu	1 - User	<input checked="" type="checkbox"/>	1 - Schedu	1 - Success	8/19/2015 03:49
10000000	MaintenanceSche		1 - User	<input checked="" type="checkbox"/>	1 - Schedu	1 - Success	8/19/2015 03:48
10000000	Plant MPI DIM_D	Plant MPI DIM_D	1 - User	<input checked="" type="checkbox"/>	1 - Schedu	1 - Success	8/19/2015 12:01
10000000	APR_MNT_RunSc	Schedule for APR	1 - User	<input type="checkbox"/>	0 - New		
10000000	APR_MNT_RunSc	Call APR_MNT.Ru	1 - User	<input type="checkbox"/>	0 - New		
10000000	APR_MNT_RunSc	Schedule for APR	1 - User	<input type="checkbox"/>	0 - New		
10000000	Process Plant MPI	Process Plant MPI	1 - User	<input type="checkbox"/>	0 - New		
10000000	Notifications		1 - User	<input checked="" type="checkbox"/>	1 - Schedu	1 - Success	8/19/2015 03:49
10000001	TRNxx_Job		1 - User	<input type="checkbox"/>	0 - New		

Screenshot 2 (Bottom): The "Enabled" column for all jobs is checked. The "Status" column shows various states. The "Last Run" column shows the date and time of the last successful run. The "Last Started On" column shows the date and time of the last start. In this screenshot, the "Run" button (highlighted with a blue box) has been clicked for the "TRNxx_Job" entry, which now has a status of "3 - Finished" in the "Last Run" column.

ID	Name	Description	Job Type	Enabled	Status	Last Run	Last Started On
10000000	APR_MPI_Populat	APR_MPI_Populat	1 - User	<input checked="" type="checkbox"/>	1 - Schedu	1 - Success	8/19/2015 03:00
10000000	APR_CNT_FUTUR	Containment Futu	1 - User	<input checked="" type="checkbox"/>	1 - Schedu	1 - Success	8/19/2015 03:49
10000000	MaintenanceSche		1 - User	<input checked="" type="checkbox"/>	1 - Schedu	1 - Success	8/19/2015 03:48
10000000	Plant MPI DIM_D	Plant MPI DIM_D	1 - User	<input checked="" type="checkbox"/>	1 - Schedu	1 - Success	8/19/2015 12:01
10000000	APR_MNT_RunSc	Schedule for APR	1 - User	<input type="checkbox"/>	0 - New		
10000000	APR_MNT_RunSc	Call APR_MNT.Ru	1 - User	<input type="checkbox"/>	0 - New		
10000000	APR_MNT_RunSc	Schedule for APR	1 - User	<input type="checkbox"/>	0 - New		
10000000	Process Plant MPI	Process Plant MPI	1 - User	<input type="checkbox"/>	0 - New		
10000000	Notifications		1 - User	<input checked="" type="checkbox"/>	1 - Schedu	1 - Success	8/19/2015 03:49
10000001	TRNxx_Job		1 - User	<input checked="" type="checkbox"/>	3 - Finished		

Sample Configuration of Job Pools -
Incorrect

Job Pools - Recommendations

Chapter 3: JS Processing Model

JS Processing Model

JS Processing Model - Retry

JS Central Configuration

Job Executor and Scheduler User
Interface

Scheduled jobs

Jobs Currently Executed

Synchronization Queues Status

XML Messages Screen

Execution Log

Immediate Jobs History

Logging

Performance Counters

Troubleshooting

LAB 1: Create a User Defined Job

LAB 1: Create a User Defined Job

LAB 1: Create TRNXX_Job

LAB 1: Set Action After Execution

LAB 1: Create TRNXX_JobAction

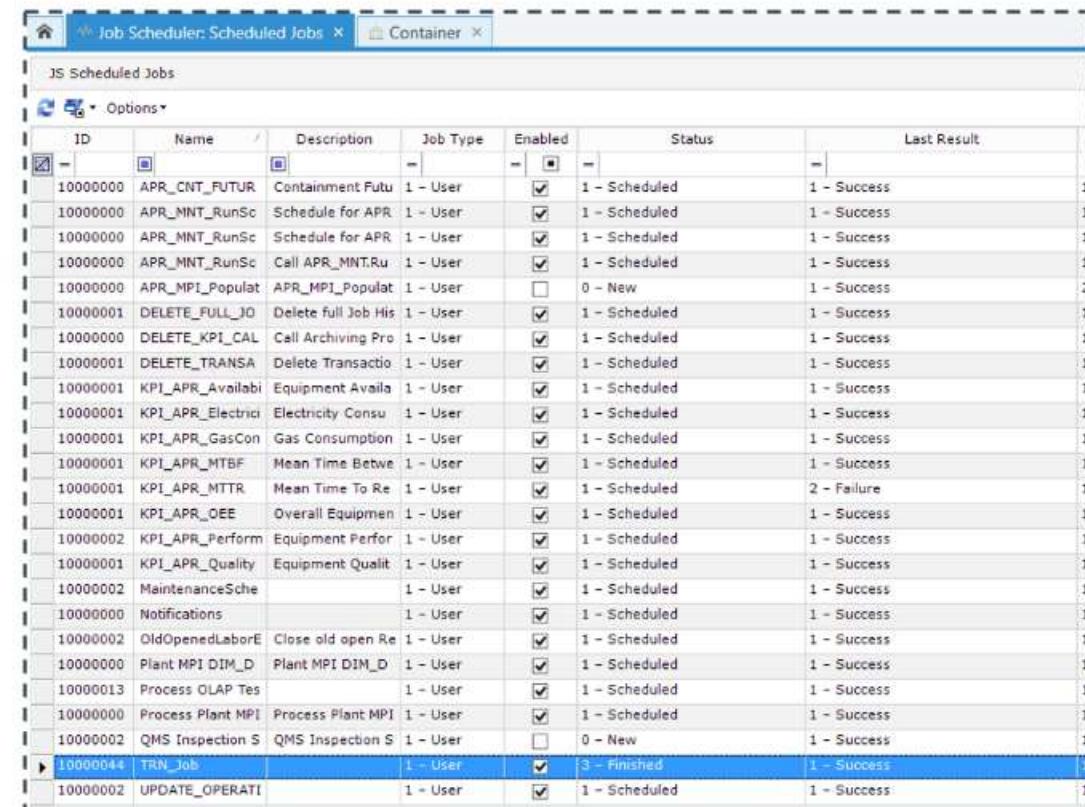
LAB 1: Assign the Employee

LAB 1: Run Your Job

LAB 1: Check Status of the Job

LAB 1: Check Status of the Job

- ▶ Confirm that your job has run by checking the status of the Job. The Last Result should be Success
- ▶ Open the **Container M&M** screen and check to see if your container has been created
 - ▶ Path to Screen: Administration -> Data Administration -> Container Maintenance -> Container



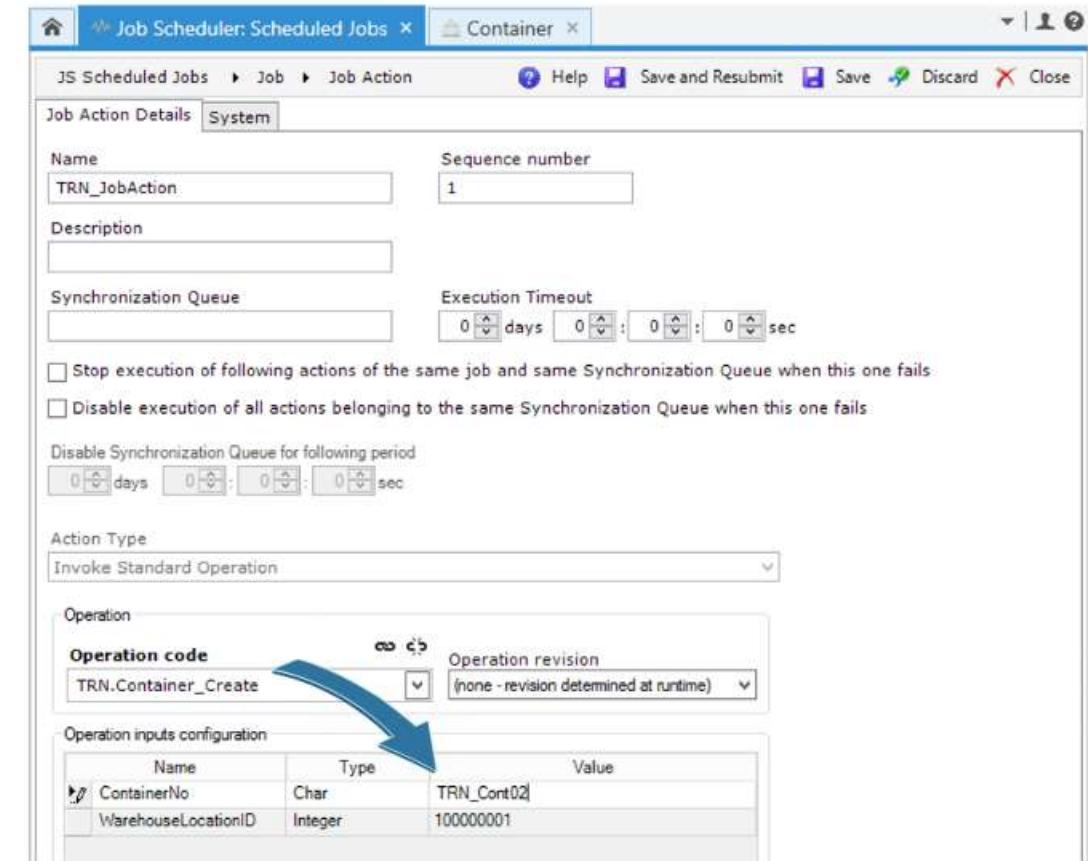
ID	Name	Description	Job Type	Enabled	Status	Last Result
10000000	APR_CNT_FUTUR	Containment Future	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success
10000000	APR_MNT_RunSc	Schedule for APR	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success
10000000	APR_MNT_RunSc	Schedule for APR	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success
10000000	APR_MNT_RunSc	Call APR_MNT.Ru	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success
10000000	APR_MPI_Populat	APR_MPI_Populat	1 - User	<input type="checkbox"/>	0 - New	1 - Success
10000001	DELETE_FULL_JO	Delete full Job His	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success
10000001	DELETE_KPI_CAL	Call Archiving Pro	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success
10000001	DELETE_TRANSA	Delete Transaction	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success
10000001	KPI_APP_Availabil	Equipment Availability	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success
10000001	KPI_APP_Electrici	Electricity Consumption	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success
10000001	KPI_APP_GasCon	Gas Consumption	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success
10000001	KPI_APP_MTBF	Mean Time Between Failures	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success
10000001	KPI_APP_MTTR	Mean Time To Repair	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	2 - Failure
10000001	KPI_APP_OEE	Overall Equipment Effectiveness	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success
10000002	KPI_APP_Perform	Equipment Performance	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success
10000001	KPI_APP_Quality	Equipment Quality	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success
10000002	MaintenanceSche	Maintenance Schedule	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success
10000000	Notifications	Notifications	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success
10000002	OldOpenedLaborE	Close old open Requests	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success
10000000	Plant MPI DIM_D	Plant MPI DIM_D	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success
10000013	Process OLAP Test	Process OLAP Test	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success
10000000	Process Plant MPI	Process Plant MPI	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success
10000002	QMS Inspection S	QMS Inspection S	1 - User	<input type="checkbox"/>	0 - New	1 - Success
10000044	TRN_JOB	TRN_JOB	1 - User	<input checked="" type="checkbox"/>	3 - Finished	1 - Success
10000002	UPDATE_OPERATI	Update Operator	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success

Search... 

- incorrect
- Job Pools - Recommendations
- ▼ Chapter 3: JS Processing Model
 - JS Processing Model
 - JS Processing Model - Retry
 - JS Central Configuration
 - Job Executor and Scheduler User Interface
 - Scheduled Jobs
 - Jobs Currently Executed
 - Synchronization Queues Status
 - XML Messages Screen
 - Execution Log
 - Immediate Jobs History
 - Logging
 - Performance Counters
 - Troubleshooting
 - LAB 1: Create a User Defined Job
 - LAB 1: Create a User Defined Job
 - LAB 1: Create TRNXX_Job
 - LAB 1: Set Action After Execution
 - LAB 1: Create TRNXX_JobAction
 - LAB 1: Assign the Employee
 - LAB 1: Run Your Job
 - LAB 1: Check Status of the Job
- LAB 1: Change Container Number

LAB 1: Change Container Number

- ▶ Return to the Job Scheduler M&M screen and open your Job
- ▶ Change your Container Number in the Job Actions from TRNXX_Cont to TRNXX_Cont02
- ▶ Save and Close your Job Action



The screenshot shows the 'Job Action Details' tab selected in the 'Job Action' section of the Job Scheduler. The job name is 'TRN_JobAction' and the sequence number is 1. The 'Action Type' is set to 'Invoke Standard Operation'. In the 'Operation' section, the operation code is 'TRN.Container_Create'. The 'Operation inputs configuration' table contains two entries:

Name	Type	Value
ContainerNo	Char	TRN_Cont02
WarehouseLocationID	Integer	100000001

Search...	
Job Pools - Recommendations	
Chapter 3: JS Processing Model	
JS Processing Model	
JS Processing Model - Retry	
JS Central Configuration	
Job Executor and Scheduler User Interface	
Scheduled jobs	
Jobs Currently Executed	
Synchronization Queues Status	
XML Messages Screen	
Execution Log	
Immediate Jobs History	
Logging	
Performance Counters	
Troubleshooting	
LAB 1: Create a User Defined Job	
LAB 1: Create a User Defined Job	
LAB 1: Create TRNXX_Job	
LAB 1: Set Action After Execution	
LAB 1: Create TRNXX_JobAction	
LAB 1: Assign the Employee	
LAB 1: Run Your Job	
LAB 1: Check Status of the Job	
LAB 1: Change Container Number	
LAB 1: Check Job Schedules	

LAB 1: Check Job Schedules

- ▶ Open the Job Schedules tab

You will notice that you have a record here that shows one execution. This was created when you manually ran the job.

The screenshot displays two windows related to job scheduling. The top window is a configuration dialog for an operation. It shows the 'Operation code' as 'TRN.Container_Create' and the 'Operation revision' as '(none - revision determined at runtime)'. Below this is a table for 'Operation inputs configuration' with two entries: 'ContainerNo' (Char type, value 'TRN_Cont02') and 'WarehouseLocationID' (Integer type, value '100000001'). A blue arrow points from this configuration to the second window below. The second window is titled 'Job Scheduler: Scheduled Jobs' and shows a single job entry in a table. The job details are: Name 'Executed manual', Description 'Executed per use', Frequency '1 - Immediately', Start Time '01/02/2017 11:15', and End Time '01/02/2017 11:15'.

Search... 

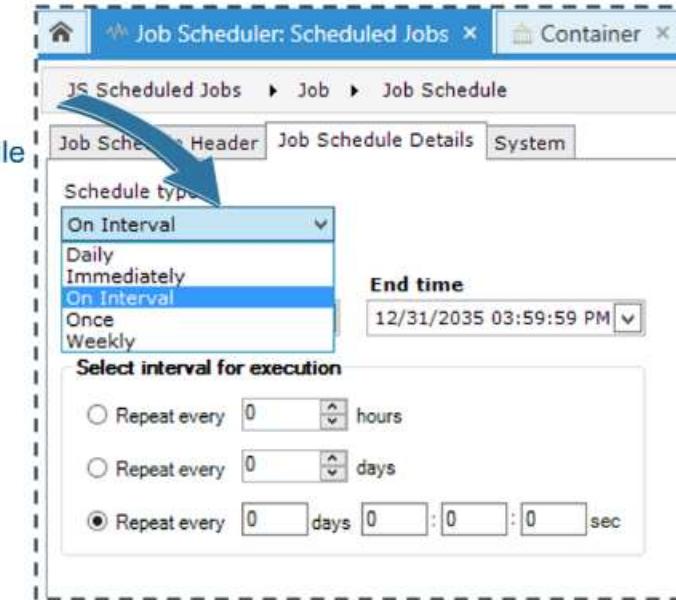
▼ Chapter 3: JS Processing Model

- JS Processing Model
- JS Processing Model - Retry
- JS Central Configuration
- Job Executor and Scheduler User Interface
- Scheduled Jobs
- Jobs Currently Executed
- Synchronization Queues Status
- XML Messages Screen
- Execution Log
- Immediate Jobs History
- Logging
- Performance Counters
- Troubleshooting
- LAB 1: Create a User Defined Job
- LAB 1: Create a User Defined Job
- LAB 1: Create TRNXX_Job
- LAB 1: Set Action After Execution
- LAB 1: Create TRNXX_JobAction
- LAB 1: Assign the Employee
- LAB 1: Run Your Job
- LAB 1: Check Status of the job
- LAB 1: Change Container Number
- LAB 1: Check Job Schedules

LAB 1: Add TRNXX_JobSchedule

LAB 1: Add TRNXX_JobSchedule

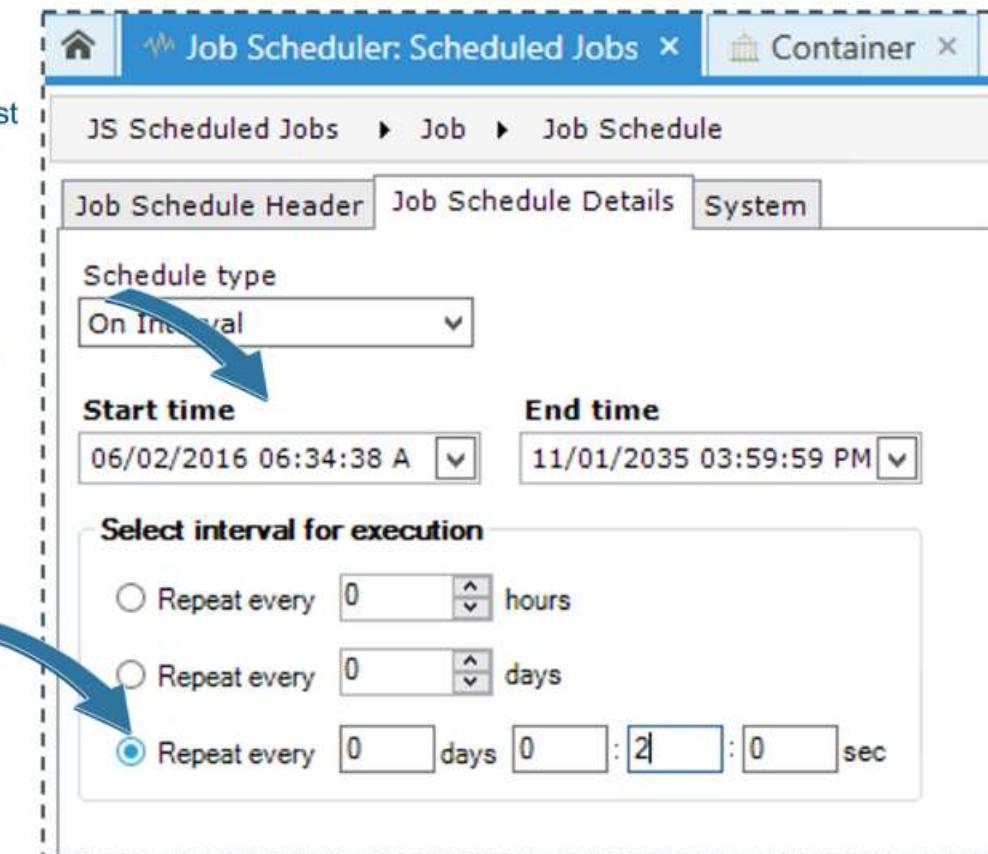
- ▶ Add a new Schedule by clicking on icon.
- ▶ Set the name of your Schedule to **TRNXX_JobSchedule**
- ▶ Open the Job Schedule Details and from Schedule Type drop-down menu tab choose **On Interval**



- Search... 
- JS Processing Model
 - JS Processing Model - Retry
 - JS Central Configuration
 - Job Executor and Scheduler User Interface
 - Scheduled jobs
 - Jobs Currently Executed
 - Synchronization Queues Status
 - XML Messages Screen
 - Execution Log
 - Immediate Jobs History
 - Logging
 - Performance Counters
 - Troubleshooting
 - LAB 1: Create a User Defined Job
 - LAB 1: Create a User Defined Job
 - LAB 1: Create TRNXX_Job
 - LAB 1: Set Action After Execution
 - LAB 1: Create TRNXX_JobAction
 - LAB 1: Assign the Employee
 - LAB 1: Run Your Job
 - LAB 1: Check Status of the Job
 - LAB 1: Change Container Number
 - LAB 1: Check Job Schedules
 - LAB 1: Add TRNXX_JobSchedule
 - LAB 1: Schedule Properties**

LAB 1: Schedule Properties

- ▶ Assign the following values to the Schedule Properties:
 - ▶ Start Time: pick any time that is in the past
 - ▶ End Time: pick any time that is at least 1 hour in the future
 - ▶ Set the seconds for the Repeat Every interval to 2 minutes
- ▶ Save your changes and close the properties screen so that you are looking at your job in the main Job Scheduler grid



Job Executor and Scheduler User Interface

Scheduled jobs

Jobs Currently Executed

Synchronization Queues Status

XML Messages Screen

Execution Log

Immediate Jobs History

Logging

Performance Counters

Troubleshooting

LAB 1: Create a User Defined Job

LAB 1: Create a User Defined Job

LAB 1: Create TRNXX_Job

LAB 1: Set Action After Execution

LAB 1: Create TRNXX_JobAction

LAB 1: Assign the Employee

LAB 1: Run Your Job

LAB 1: Check Status of the Job

LAB 1: Change Container Number

LAB 1: Check Job Schedules

LAB 1: Add TRNXX_JobSchedule

LAB 1: Schedule Properties

LAB 1: Check Results

End of LAB 1

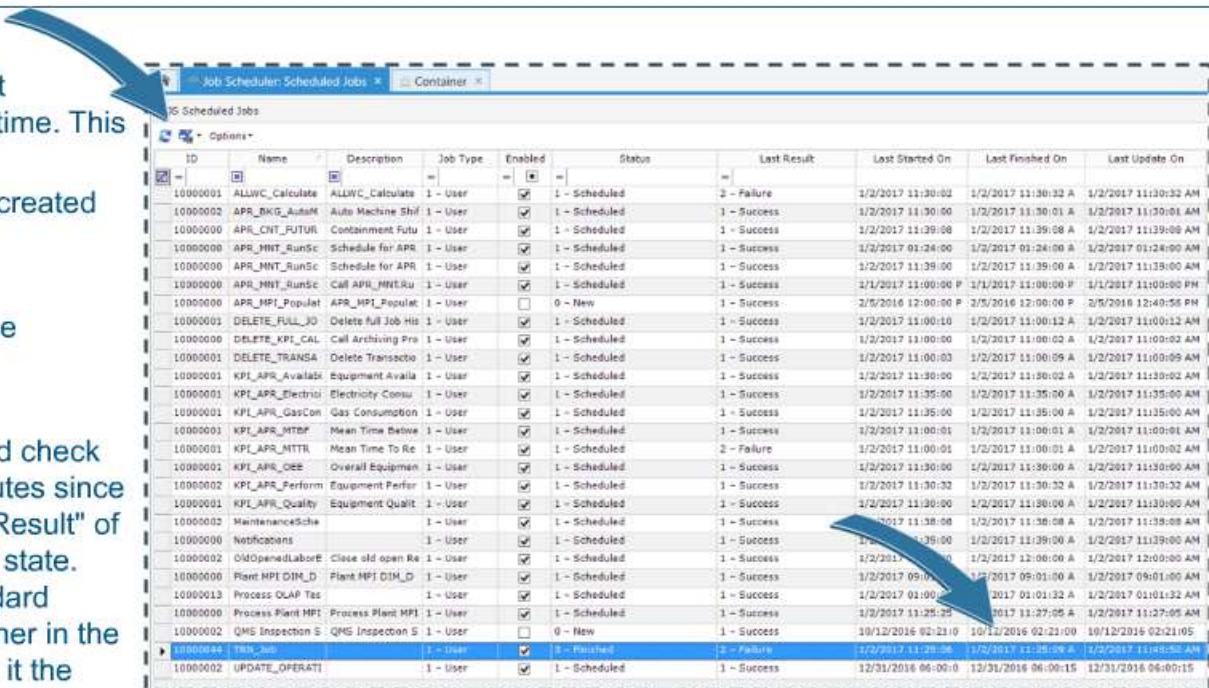
End of Course

LAB 1: Check Results

- ▶ Refresh the screen until you see the Last Finished On field updated to the current time. This means the job has finished running
- ▶ Check to make sure your container was created (TRNXX_Cont02)

Now wait two minutes - this is the interval time scheduled for the job.

- ▶ Go back to the Job Scheduler screen and check your Job. If it has been more than 2 minutes since it last ran, your job should show a "Last Result" of "Failure" and it should be in a "Disabled" state. This is because the Job is calling a standard operation that is trying to create a container in the database that already exists (we created it the first time the job ran)



ID	Name	Description	Job Type	Enabled	Status	Last Result	Last Started On	Last Finished On	Last Update On
10000001	ALLWC_Calculate	ALLWC_Calculate	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	2 - Failure	1/2/2017 11:20:02	1/2/2017 11:30:32 A	1/2/2017 11:30:32 AM
10000002	APR_BKG_AutoM	Auto Machine Shift	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:30:01	1/2/2017 11:30:01 A	1/2/2017 11:30:01 AM
10000003	APR_CNT_FUTUR	Containment Futu	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:39:08	1/2/2017 11:39:08 A	1/2/2017 11:39:08 AM
10000004	APR_MNT_RunSc	Schedule for APR	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 01:24:00	1/2/2017 01:24:00 A	1/2/2017 01:24:00 AM
10000005	APR_MNT_RunSc	Schedule for APR	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:39:00	1/2/2017 11:39:00 A	1/2/2017 11:39:00 AM
10000006	APR_MNT_RunSc	Call API_MNTRU	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/1/2017 11:00:00 P	1/1/2017 11:00:00 P	1/1/2017 11:00:00 PM
10000007	APR_MTF_Popula	APR_MTF_Popula	1 - User	<input type="checkbox"/>	0 - New	1 - Success	2/5/2016 12:00:00 P	2/5/2016 12:00:00 P	2/5/2016 12:40:56 PM
10000008	DELETE_FULL_JO	Delete full Job His	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:00:10	1/2/2017 11:00:12 A	1/2/2017 11:00:12 AM
10000009	DELETE_XPT_CAL	call Archiving Pro	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:00:01	1/2/2017 11:00:02 A	1/2/2017 11:00:02 AM
10000010	DELETE_TRANSFA	Delete Transactio	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:00:03	1/2/2017 11:00:09 A	1/2/2017 11:00:09 AM
10000011	KPI_APR_Availab	Equipment Availa	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:30:00	1/2/2017 11:30:02 A	1/2/2017 11:30:02 AM
10000012	KPI_APR_Electro	Electricity Consu	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:35:00	1/2/2017 11:35:00 A	1/2/2017 11:35:00 AM
10000013	KPI_APR_GasCon	Gas Consumption	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:35:00	1/2/2017 11:35:00 A	1/2/2017 11:35:00 AM
10000014	KPI_APR_MTBF	Mean Time Before	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:00:01	1/2/2017 11:00:01 A	1/2/2017 11:00:01 AM
10000015	KPI_APR_MTTR	Mean Time To Re	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	2 - Failure	1/2/2017 11:00:01	1/2/2017 11:00:02 A	1/2/2017 11:00:02 AM
10000016	KPI_APR_OE	Overall Equipment	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:30:00	1/2/2017 11:30:00 A	1/2/2017 11:30:00 AM
10000017	KPI_APR_Perform	Equipment Perfis	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:30:32	1/2/2017 11:30:32 A	1/2/2017 11:30:32 AM
10000018	KPI_APR_Quality	Equipment Qualit	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:30:00	1/2/2017 11:30:00 A	1/2/2017 11:30:00 AM
10000019	MaintenanceSche	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:35:00	1/2/2017 11:35:00 A	1/2/2017 11:35:00 AM	
10000020	Nothelpoens	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:35:00	1/2/2017 11:35:00 A	1/2/2017 11:35:00 AM	
10000021	OldOpenedLaborE	Close old open Re	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:30:00	1/2/2017 12:00:00 A	1/2/2017 12:00:00 AM
10000022	Plant MPI DIM_D	Plant MPI DIM_D	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 09:00	1/2/2017 09:01:00 A	1/2/2017 09:01:00 AM
10000023	Process OLAP Tas	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 01:00:00	2017 01:01:32 A	1/2/2017 01:32 AM	
10000024	Process Plant MPI	Process Plant MPI	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 11:25:25	2017 11:27:05 A	1/2/2017 11:27:05 AM
10000025	QMS Inspection S	QMS Inspection S	1 - User	<input type="checkbox"/>	0 - New	1 - Success	10/12/2016 02:21:00	10/12/2016 02:21:05	10/12/2016 02:21:05
10000026	TRNXX_MBD	1 - User	<input checked="" type="checkbox"/>	1 - Enabled	2 - Failed	2 - Failure	1/2/2017 11:28:56	1/2/2017 11:30:00 A	1/2/2017 11:30:00 AM
10000027	UPDATE_OPERATI	1 - User	<input checked="" type="checkbox"/>	1 - Scheduled	1 - Success	1/2/2017 06:00:00	1/2/2017 06:00:15	1/2/2017 06:00:15	