6 Workflows, Jobs and Tasks

Chapter 6 outlines the steps you take to construct a Workflow Hierarchy made up of a *Taskmaster Rules* workflow, its jobs, and the tasks assigned to each job.

Chapter 6 explores these topics:

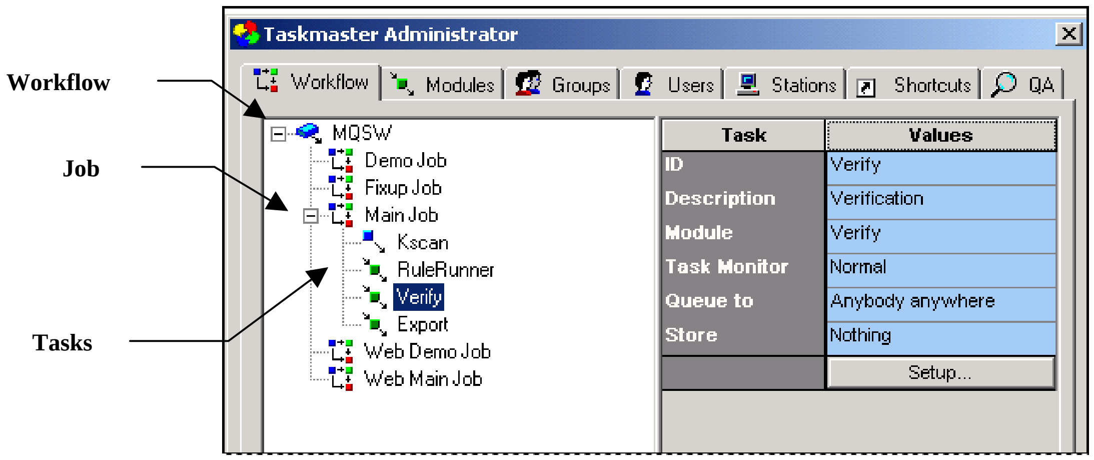
This chapter examines the structure and components of a Workflow Hierarchy, and shows you how to define a workflow; its jobs; and the tasks that are part of a job. Upcoming chapters concentrate on the definition and operation of tasks in these categories:

- Scan tasks – Chapter 7
- RuleRunner tasks – Chapter 8
- Verify tasks – Chapter 9
- FixUp tasks – Chapter 10.

Introduction

The *Taskmaster Wizard Guide* shows you how to set up an entirely new *Taskmaster* application, using the **Taskmaster Application Wizard.**

A prominent feature of the application is a Workflow Hierarchy with a structure similar to that of the fictional *MQSW* application:



Task	Values
ID	Verify
Description	Verification
Module	Verify
Task Monitor	Normal
Queue to	Anybody anywhere
Store	Nothing

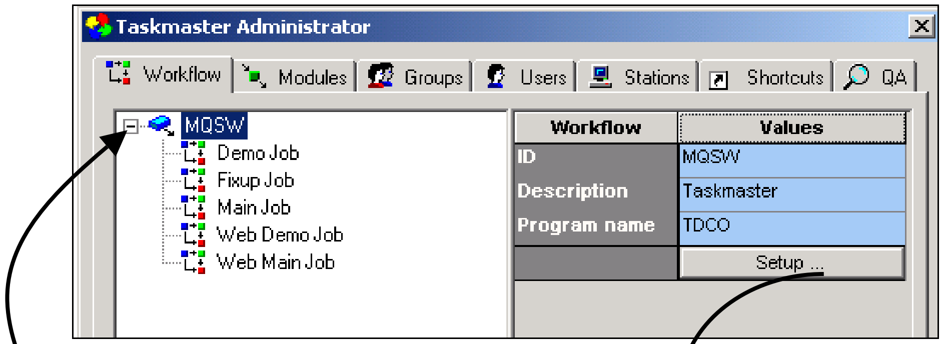
Taskmaster Administrator – *Workflow tab*

In the example above:

MQSW identifies the **workflow**. If you highlight this title, the *Workflow* tab's right side lists the workflow's properties ***and*** gives you access to this workflow's Document Hierarchy (illustrated on the next page.) Objects of the Document Hierarchy – and the rules that are applied to them – determine how the workflow's jobs and their tasks create and process batches, documents, pages, fields and data (Chapter 3 investigates the Document Hierarchy.)

Main Job designates a fully-equipped **job** that scans paper and generates Image files - then reads, verifies, validates and exports the data on a ***source*** page. *Demo* jobs can be used for testing, training and demonstration. The *Web* jobs are ready to run in a *Taskmaster Web* environment. The *FixUp Job* reviews and repairs problem batches.

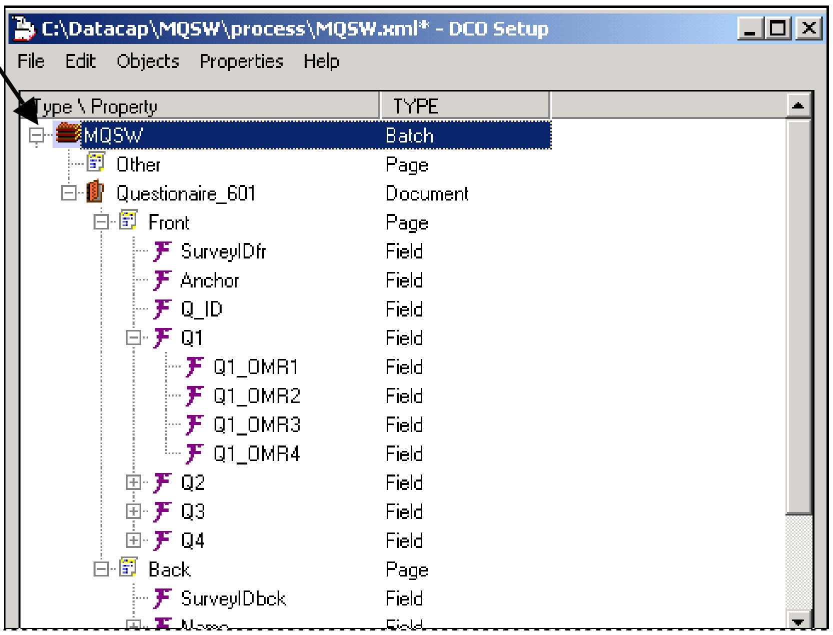
Kscan is the opening **task** of the Main job. This operator-intensive task creates a new batch, scans the paper in its tray, generates an Image file for each page, and places the images in the batch. The *RuleRunner* task first cleans and de-skews images. It then organizes the batch into a series of documents and pages, recognizes the data in the ***source*** page of each document, and adds these values to a Data file for the page. Data Entry operators use the *Verify* task to review and, if necessary, correct a ***source*** page's data. The *Export* task adds the verified data to an Export file or database.



- MQSW
  - Demo Job
  - Fixup Job
  - Main Job
  - Web Demo Job
  - Web Main Job

Workflow	Values
ID	MQSW
Description	Taskmaster
Program name	TDCO

Taskmaster Administrator – *Workflow tab*



Type \ Property	TYPE
MQSW	Batch
Other	Page
Questionaire_601	Document
Front	Page
SurveyIDfr	Field
Anchor	Field
Q_ID	Field
Q1	Field
Q1_OMR1	Field
Q1_OMR2	Field
Q1_OMR3	Field
Q1_OMR4	Field
Q2	Field
Q3	Field
Q4	Field
Back	Page
SurveyIDbck	Field
Name	Field

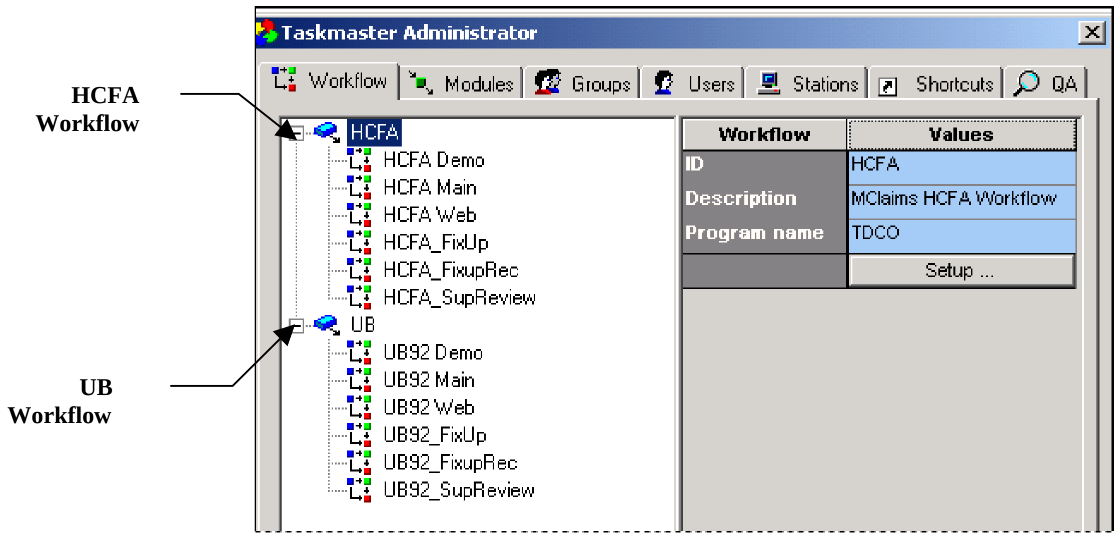
Document Hierarchy Setup Window

This chapter of the *Guide to Taskmaster Rules* summarizes the steps you take define a Workflow Hierarchy, its jobs and their tasks. Succeeding chapters review the setup and operation of tasks in each category.

Workflow Definition

Most applications employ a single **Workflow Hierarchy** linked to a single **Document Hierarchy.**

Other applications use multiple hierarchies. *Taskmaster for Medical Claims*, for example, employs one Workflow Hierarchy to process *HCFA-1500* health claims – and another to handle *UB-92* claims.

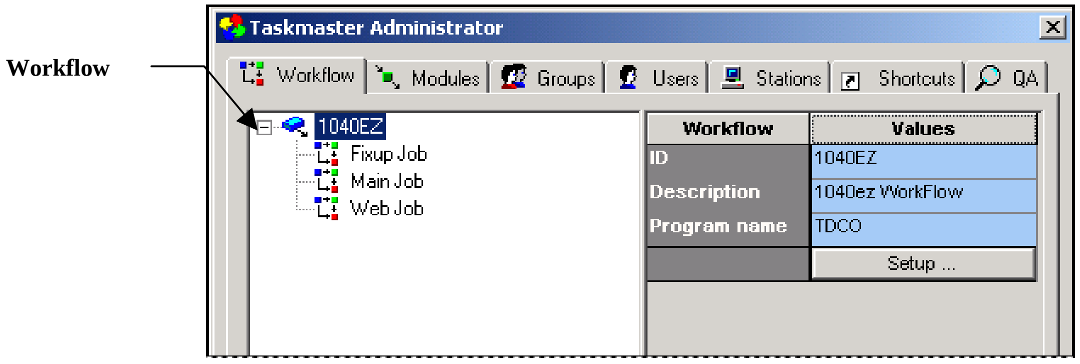


- HCFA
  - HCFA Demo
  - HCFA Main
  - HCFA Web
  - HCFA_FixUp
  - HCFA_FixupRec
  - HCFA_SupReview
- UB
  - UB92 Demo
  - UB92 Main
  - UB92 Web
  - UB92_FixUp
  - UB92_FixupRec
  - UB92_SupReview

Workflow	Values
ID	HCFA
Description	MClaims HCFA Workflow
Program name	TDCO

Medical Claims Taskmaster Administrator – *Workflow tab*

In contrast, the *1040EZ* application has just one Workflow Hierarchy – but you can add a new workflow without difficulty or hesitation. ***Remember:*** *1040EZ* is indestructible, and welcomes trials and errors of every kind.



- 1040EZ
  - Fixup Job
  - Main Job
  - Web Job

Workflow	Values
ID	1040EZ
Description	1040ez WorkFlow
Program name	TDCO

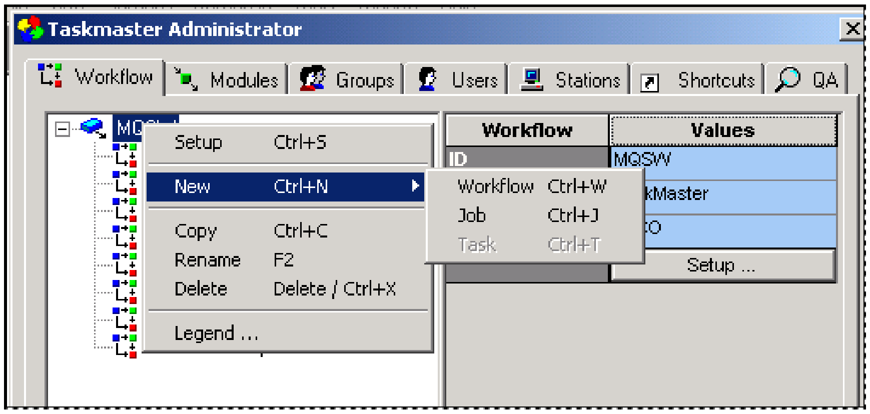
1040EZ Taskmaster Administrator – *Workflow tab*

Remember! The Taskmaster Application Wizard automatically provides a new application with a Workflow Hierarchy. For details, see the *Application Wizard Guide.*

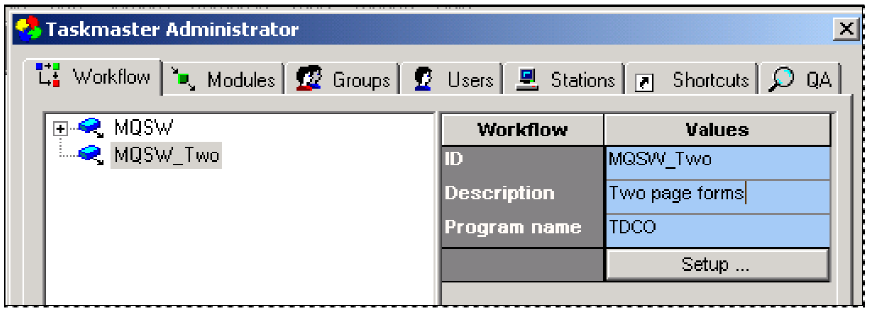
How to Add A Workflow

The basic procedure for providing an application with a second **workflow** component is simple. Here's how you would add a workflow to the *MQSW* Workflow Hierarchy:

Step	Action
1.	Right-click on the title of the existing workflow – *MQSW*, in this case. (Do ***not*** use the Add button at the bottom of the *Workflow* tab.)



Step	Action
2.	Select **Workflow** from the **New** options (or press the Ctrl + W keyboard combination.)
3.	Enter the workflow's **ID** in the *Workflow* tab of the ***Taskmaster Administrator.***

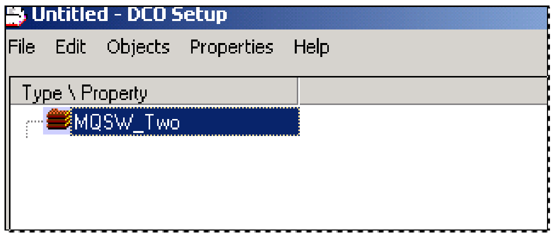


Workflow	Values
ID	MQSW_Two
Description	Two page forms
Program name	TDCO

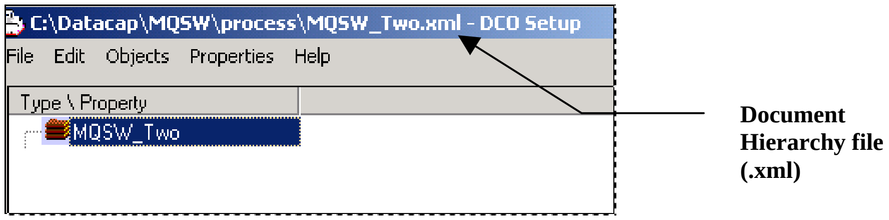
Step	Action
4.	Include a brief but important **Description** of the workflow's scope and focus.
5.	Select *TDCO* from the **Program Name** drop-down list. ***Alert!*** This is a ***required*** property of any *Taskmaster Rules* workflow.
6.	Press the Apply button at the bottom of the tab.

✓ This is only the first stage of Workflow Definition. Although the obvious properties of the new workflow are now in place, it does not yet have a Document Hierarchy. To add this ***essential*** property:

Step	Action
1.	Highlight the new Workflow ID.
2.	Click on the *Workflow* tab's Setup button to access the ***Document Hierarchy Setup*** window.



Step	Action
3.	Note that the window displays a default **Batch** object – and that the object's title is ***the same as*** the workflow's ID. Note, too, that the Document Hierarchy itself is not yet part of a file (.xml) and is therefore *Untitled.*
4.	Use the **File** menu's **Save** item to establish a file for the Document Hierarchy and add it to the application's **Process** directory.


C:\Datacap\MQSW\process\MQSW_Two.xml - DCO Setup
File Edit Objects Properties Help
Type \ Property
MQSW_Two
Document Hierarchy file (.xml)

You are now free to expand the Document Hierarchy's contents in line with the discussions and instructions in Chapter 3.

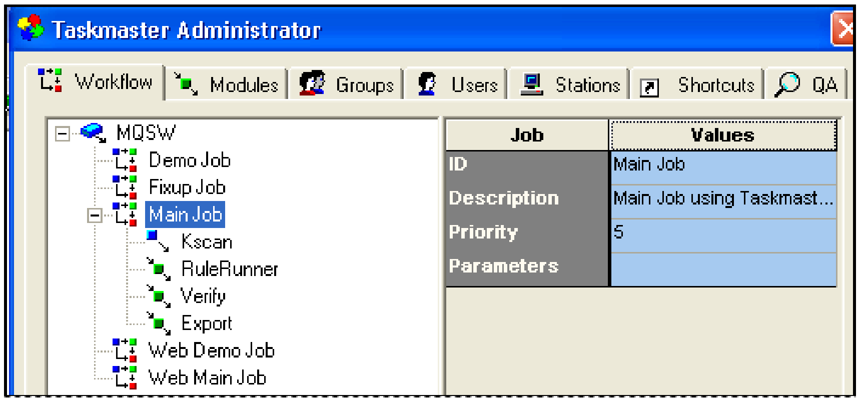
✓ Chapter 6 of the *Taskmaster Windows & Dialogs Reference* examines all aspects of the ***Taskmaster Administrator's*** *Workflow* tab. This text is available as electronic PDF documents and in printed format.

Job Definitions

Jobs occupy the second level of a Workflow Hierarchy, directly below the workflow itself. As a result:

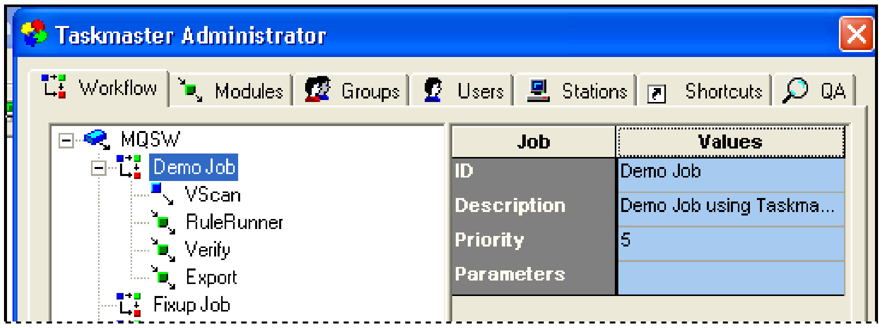
> All jobs that belong to the workflow – and the tasks that belong to each job – have a direct and immediate association with the workflow's **Document Hierarchy**, its objects and their properties, and the rules that are bound to these objects.

As Page 2 explains, a "new" application such as *MQSW* comes equipped with various jobs – each with its own objective and tasks to meet that objective. Below are elements of the Main Job, and its closely related Demo Job.



Job	Values
ID	Main Job
Description	Main Job using Taskmast...
Priority	5
Parameters	

Taskmaster Administrator – *Workflow tab*
<u>MQSW Main Job</u>



Job	Values
ID	Demo Job
Description	Demo Job using Taskma...
Priority	5
Parameters	

Taskmaster Administrator – *Workflow tab*
<u>MQSW Demo Job</u>

Two properties on the tab's right-hand side deserve your attention:

- ♦ The value in the **Priority** field determines the processing rank of batches processed by the job's tasks: "5" is the default value. ("1" is High; "10" is low.)
- ♦ Job Definitions do ***not*** usually include a **Parameters** value.

How to Add a Job to a Workflow

A **workflow** needs at least one **job.**

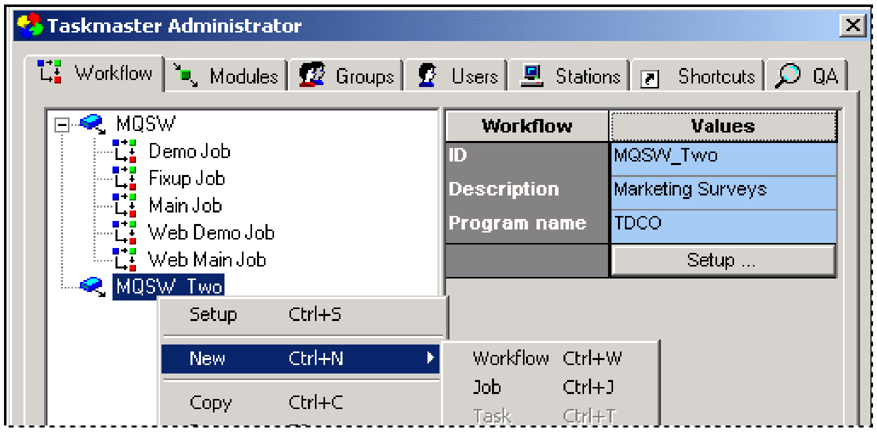
The new *MQSW_Two* workflow has been set up to process a second type of form:

- The original *MQSW* workflow takes care of customer satisfaction responses.
- The *MQSW_Two* workflow deals with marketing surveys.

At this point, the *MQSW* Workflow Hierarchy is fully configured. The new *MQSW_Two*, however, has a Document Hierarchy (Page 2) but little else. To start off, it needs a set of jobs comparable to those of the *MQSW* workflow.

To add a job to a workflow, take these steps:

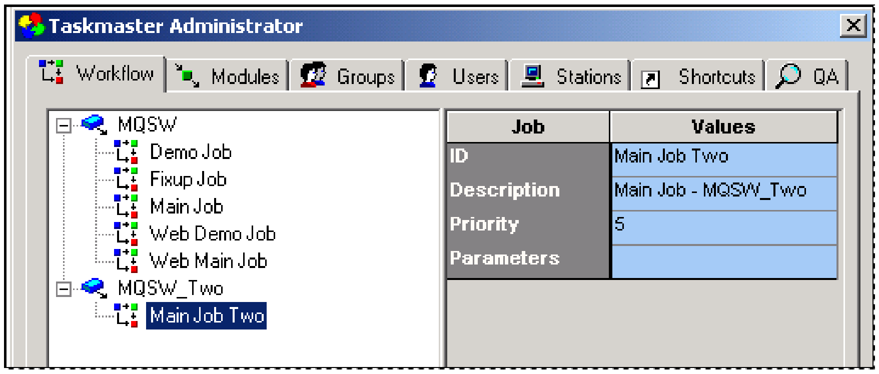
Step	Action
1.	In your ***Taskmaster Administrator's*** *Workflow* tab, right-click on the Workflow ID of the workflow to which you are adding the job.
2.	Select **New** and **Job** from the options, or press the Ctrl + N/Ctrl + J keyboard sequence. (Alternatively, you can click on the Add button at the bottom of the *Workflow* tab.)



MQSW
Demo Job
Fixup Job
Main Job
Web Demo Job
Web Main Job
MQSW_Two

Workflow	Values
ID	MQSW_Two
Description	Marketing Surveys
Program name	TDCO

3.	Enter the job's title in the rectangle *Taskmaster* provides.



MQSW
Demo Job
Fixup Job
Main Job
Web Demo Job
Web Main Job
MQSW_Two
Main Job Two

Job	Values
ID	Main Job Two
Description	Main Job - MQSW_Two
Priority	5
Parameters	

To Add a Job to a Workflow (continued)

Step	Action
4.	Press the Apply button at the bottom of the tab.
5.	On the right-hand side of the tab, add a brief but important **Description** of the new job.
6.	Confirm or change the **Priority** setting.
7.	Click on the Apply button again, and on the Done button.

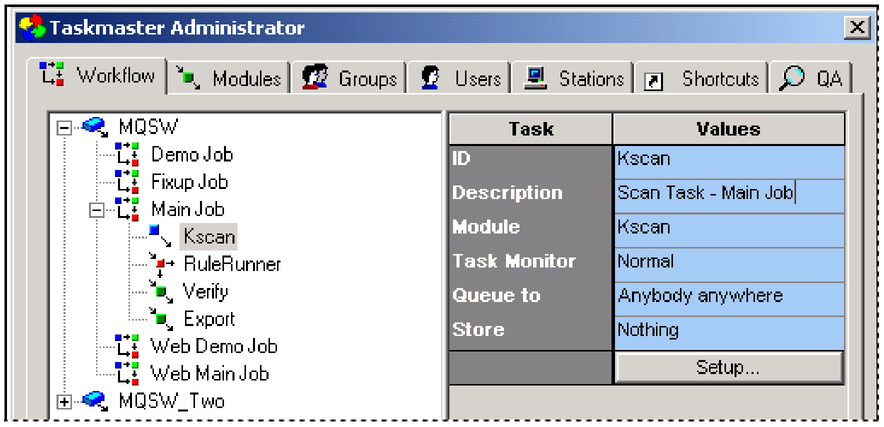
✓ ***Remember:*** the new job and, eventually, its tasks inherit the objects of the workflow's Document Hierarchy and their properties.

Task Definitions

All tasks of a *Taskmaster* application:

- Occupy the lowest level of a Workflow Hierarchy that consists of workflow, job and task.
- Inherit all aspects of the workflow's association with its Document Hierarchy.
- Are designed, assembled and tested as *Batch Pilot* **Task Projects -** each with a ***setup*** form and a ***runtime*** form…and each linked to the workflow's Document Hierarchy.
- Are defined as *Taskmaster* components that rely on Task Modules to connect them to their respective *Batch Pilot* projects.
- Respond to specifications in a ***Task Settings*** dialog.
- Operate in a job/task sequence that is established in the *Workflow* tab of the ***Taskmaster Administrator***, in response to the interaction of qualified users – operators, Administrators and Supervisors.

As an example, all tasks of the *MQSW* workflow's *Main Job* share the characteristics outlined above.



Task	Values
ID	Kscan
Description	Scan Task - Main Job
Module	Kscan
Task Monitor	Normal
Queue to	Anybody anywhere
Store	Nothing

Taskmaster Administrator – *Workflow tab*
<u>kScan Task Definition</u>

There are, of course, differences among the tasks. The most obvious are ***functional***: at one end, the kScan task scans paper; at the other end, the Export task sends verified and validated data to a file or database.

Less obvious but just as important differences involve the tasks' incorporation of previously defined RuleSets, rules, and actions (Chapter 5). And, as Chapter 7 explains, a task such as kScan operates in response to the numerous and varied settings that you select from the tabs of the ***Kscan Setup*** dialog. The RuleRunner and Export tasks, in contrast, are driven by the RuleSets you've defined and bound to individual objects of the Document Hierarchy (Chapter 8).

☛ The following sections focus on the steps the *MQSW* Administrator would take to add a Valildate task to the Main job of the MQSW workflow, just after the RuleRunner task. The new task needs a Task Project, a Task Module and a Task Identity.

How to Construct a Task Project

A **Task Project** is a task's foundation. It links the task directly to the application's Document Hierarchy (Chapter 3); provides the task with ***setup*** and ***runtime*** forms; and contains the task's software.

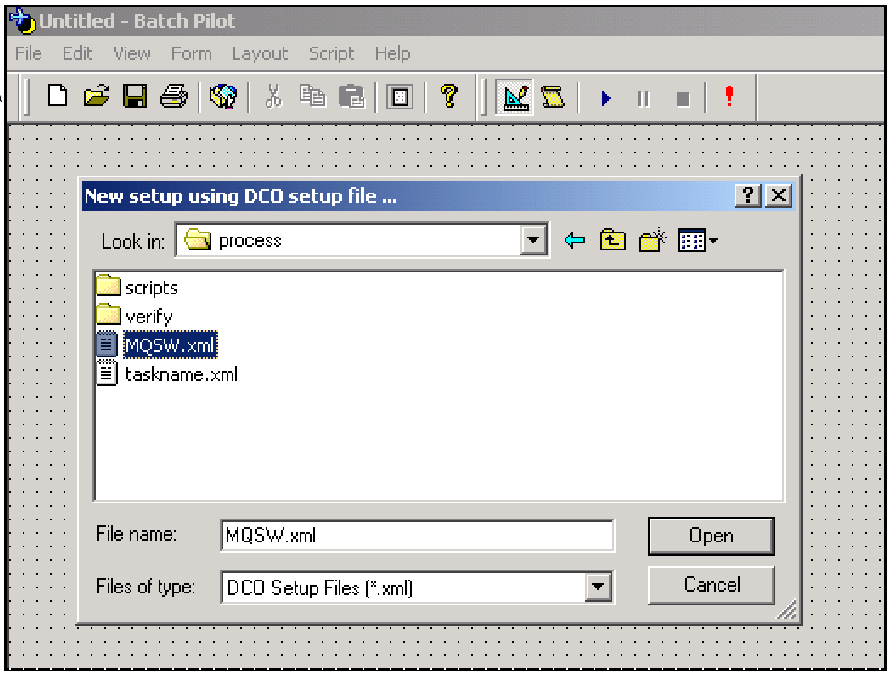
The Task Project is also a file **- <task>.bpp** – that usually resides in the application's **Process** directory.

✓ Because you ***cannot*** define a task until its Task Project is in place, this section is a broad review of the steps you take to assemble a project. Upcoming chapters describe the setup of Task Projects for specific types of tasks; Chapter 4 of the *Guide to Batch Pilot* explores the components and basic structure of a Task Project.

To define a Task Project, take these steps:

Step	Action
1.	Access your *Batch Pilot* workshop by selecting **Datacap Taskmaster** from your Windows Start button's list of **Programs.** Double-click on the **Batch Pilot** icon in the **Batch Pilot** folder to open the ***Batch Pilot Window.***
2.	Select **New Project** from the window's **File** menu. When *Batch Pilot* asks you to identify your application's Document Hierarchy file (.xml), use the ***Open File*** dialog to locate and open the file – *MQSW.xml*, in the example.

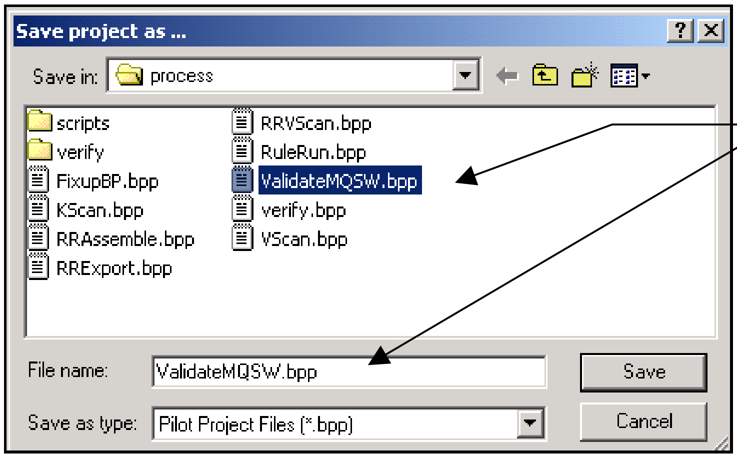
Batch Pilot Window


Untitled - Batch Pilot
File Edit View Form Layout Script Help
New setup using DCO setup file ...
Look in: process
scripts
verify
MQSW.xml
taskname.xml
File name: MQSW.xml
Files of type: DCO Setup Files (*.xml)
Open
Cancel

To Define a Task Project (continued)

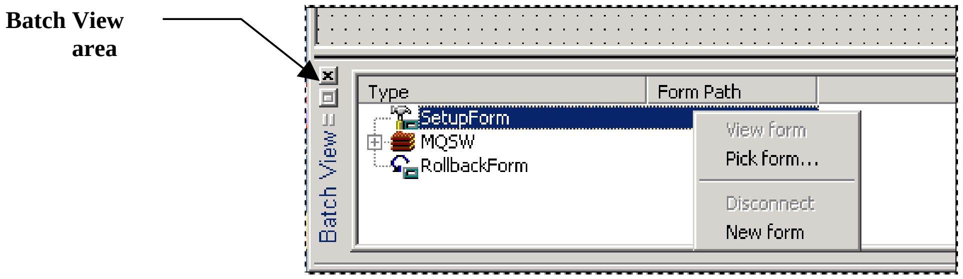
Step **Action**

3. Use the **File** menu's **Save As** item to save the Task Project as a **B**atch **P**ilot **P**roject file (.bpp) in your application's **Process** directory.



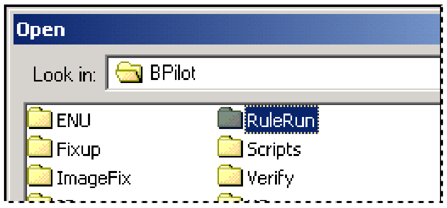
4. Close the project and the ***Batch Pilot Window***, then re-open both.

5. Select **Setup Tree** from the **View** menu to display objects at each level of the Document Hierarchy in the **Batch View** area at the bottom of the window.



6. Right-click on the **SetupForm** listing and select **Pick form** from the options. (This gives you a chance to select the "stock" form that will be the basis for this task's ***Setup*** dialog. The Task Project in this example is using stock *RuleRunner **setup*** and ***runtime*** forms. For more about these forms, see Chapter 8.)

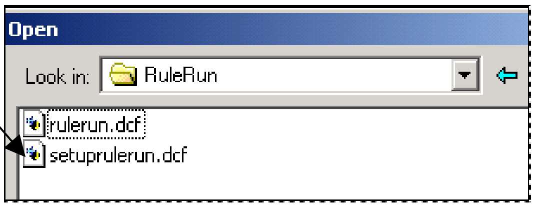
7. Open the **Datacap** directory's **BPilot** folder, and the **RuleRun** sub-folder (in this example.)



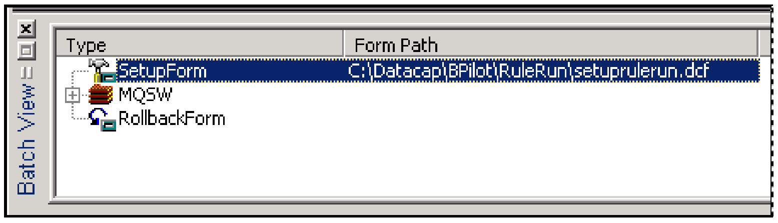
To Define a Task Project (continued)

Step	Action
8.	Select the **setupprulerun.dcf** file and press the ***Open File*** dialog's OK button.

Select this file.

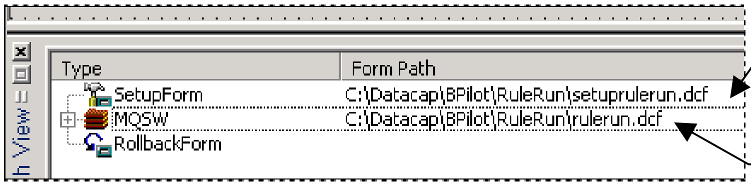


9.	Confirm that the **Form Path** column displays the name and path of the stock ***setup*** form.



Type	Form Path
SetupForm	C:\Datacap\BPilot\RuleRun\setuprulerun.dcf
MQSW	
RollbackForm	

10.	Right-click on the **Batch** object (*MQSW*, in the example) and repeat Steps 7-9 – this time, selecting **rulerun.dcf** to serve as the task's ***runtime*** dialog.



Type	Form Path
SetupForm	C:\Datacap\BPilot\RuleRun\setuprulerun.dcf
MQSW	C:\Datacap\BPilot\RuleRun\rulerun.dcf
RollbackForm	

11.	Save the new Task Project - again.

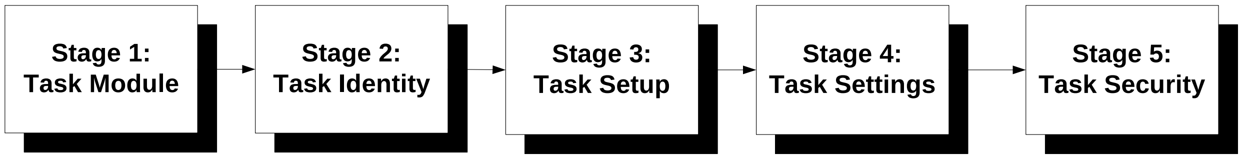
Additional Considerations

The Taskmaster Application Wizard (Chapter 2) automatically places skeletal Task Projects (.bpp) in an application's **Process** directory. Although the Task Modules you define (see the next page) can connect a Task Definition to one these projects, be sure to examine the project and its components before you take this step.

You can also construct Task Projects ***and*** their forms from scratch. Chapter 3 and Chapter 4 of the *Guide to Batch Pilot* describes this process.

How to Define a Task

Task Definition takes place in five stages:

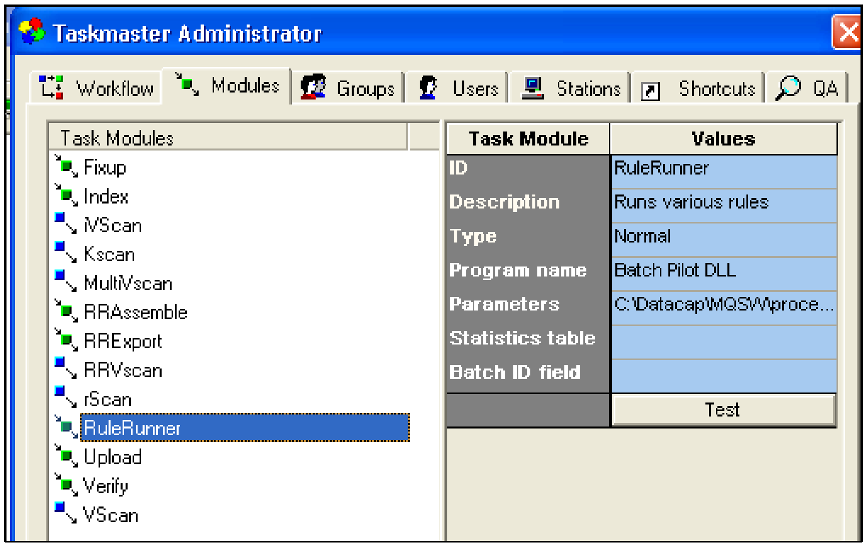


The following sections briefly review each stage of Task Definition, and refer to documentation that provides more extensive explanations.

Stage 1: Defining the Task Module

A Task Module forms the link between a task – a processing component of a job - and the Task Project that gives the task its structure and dynamics (Page 11). As a result, a task ***cannot*** operate without a Task Module – and you cannot define the task without designating its Task Module.

✓ Just as the Taskmaster Application Wizard automatically provides a workflow with jobs and tasks (Chapter 2), it also supplies Task Modules for the tasks. You'll find settings for these modules in the *Modules* tab of the ***Taskmaster Administrator***: For example:



Task Modules
- Fixup
- Index
- MScan
- Kscan
- MultiMscan
- RRAssemble
- RRExport
- RRMscan
- rScan
- RuleRunner
- Upload
- Verify
- VScan

Task Module	Values
ID	RuleRunner
Description	Runs various rules
Type	Normal
Program name	Batch Pilot DLL
Parameters	C:\Datacap\MQSW\proce...
Statistics table	
Batch ID field	

MQSW Taskmaster Administrator – *Modules tab*

When reviewing or defining a Task Module, pay close attention to these key properties on the right-hand side of the tab:

- **ID** is the unique name assigned to the module.
- **Type** indicates any special role a task with this module may have. The field is a drop-down list with four values:

  Normal: the task processes batches and their contents without exceptional procedures.

  Batch creation: the task can generate new batches. (This setting applies to the various Scan modules – see Chapter 7.)

  Job router: the task can "route" batches from the current ***parent*** job to a ***child*** job and, often, back to the ***parent*** job. Modules with this setting are often assigned to tasks such as RuleRunner and Verify that branch problem batches to a FixUp job (Chapter 10).

  Batch creation and router: a task with this module can create ***and*** route batches.
- **Program Name** designates the site of the programming code responsible for tasks covered by this Task Module. Because most tasks have been assembled in the *Batch Pilot* workshop, this property's value is typically **Batch Pilot DLL.**
- **Parameters** specifies the name and path of the *Batch Pilot* Task Project (.bpp) responsible for the setup and operation of tasks using this Task Module. The setting for the MQSW RuleRunner module, as an example, is **C:\Datacap\MQSW\Process\RuleRun.bpp**.

✓ For a complete review of Task Modules, see Chapter 3 of the *Taskmaster Workflow Development Manual* and Chapter 6 of the *Taskmaster Windows & Dialogs Reference*

How to Define a New Task Module

The list of Task Modules on the previous page is extensive but does not include a module for the new Validate task (MQValidate) that will check data previously recognized by the RuleRunner task.

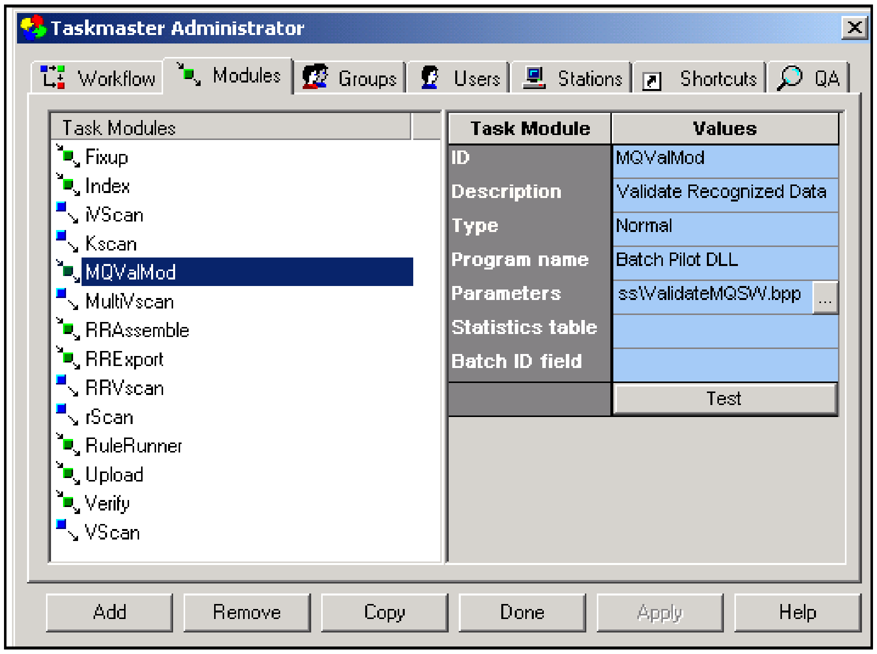
The task can carry out its work only if the module's **Parameters** setting contains the name and path of a previously assembled *Batch Pilot* Task Project – in this case, the *MQSWValidate* Task Project (Page 11).

If the Validate task can divert a "problem" batch to a FixUp job for review and repair, you would assign *Job Router* as the value of the module's **Type** property (Chapter 10). For Scan tasks, this Task Module value is usually *Batch Creation*; for other modules, it is *Normal.*

To define a new module, take the steps on the next page.

To Define a Task Module

Step	Action
1.	Be sure the applicable Task Project (.bpp) is complete and available (Page 11).
2.	Open the *Modules* tab of your ***Taskmaster Administrator***.
3.	Press the Add button at the bottom of the tab.
4.	Enter the new module's **ID** and add a brief but important **Description.**

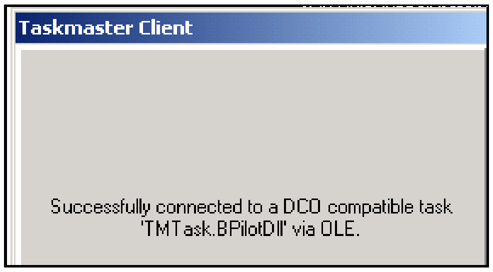


Task Module	Values
ID	MQValMod
Description	Validate Recognized Data
Type	Normal
Program name	Batch Pilot DLL
Parameters	ss\ValidateMQSW.bpp
Statistics table	
Batch ID field	

Step	Action
5.	Specify the module's **Type**. (Because, in this case, tasks using the MQValMod module will issue Error Messages if there is a problem, the Administrator selected *Normal* as the **Type** property's value.)
6.	Select *Batch Pilot DLL* from the **Program Name** drop-down list.
7.	Enter the name and path of the applicable Task Project file (.bpp) in the **Parameters** field.
8.	Click on the Apply button at the bottom of the tab to add the module to the list on the left.

To Define a Task Module (continued)

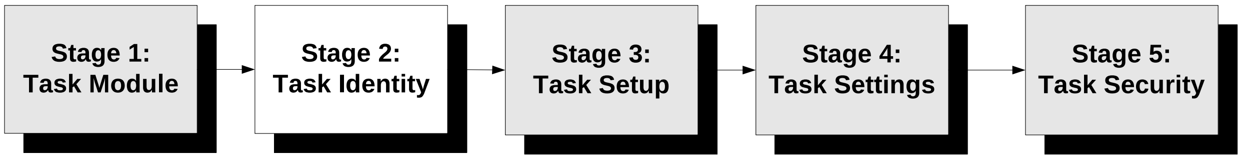
Step	Action
9.	Click on the Test button to verify the module's connection to the Task Project. If the test is successful, you will receive this message:


Taskmaster Client

Successfully connected to a DCO compatible task 'TMTask.BPilotDll' via OLE.

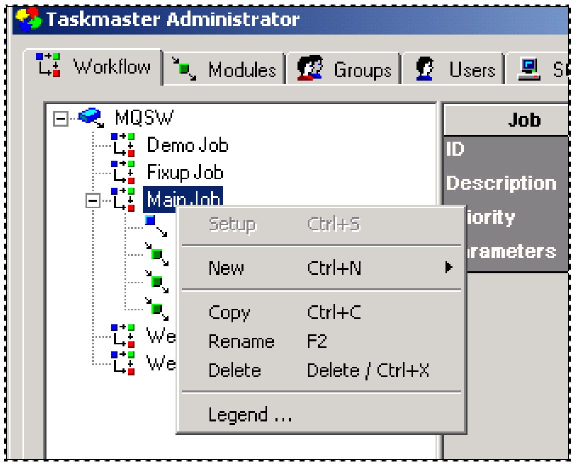
Stage 2: Task Identity

Stage 2 provides the new task with unique identifying characteristics, and assigns the appropriate Task Module.



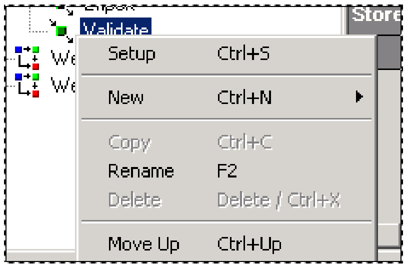
You'll use the *Workflow* tab of the ***Taskmaster Administrator*** as you take these steps:

Step	Action
1.	Right-click on the Job ID of the job to which the task will belong.



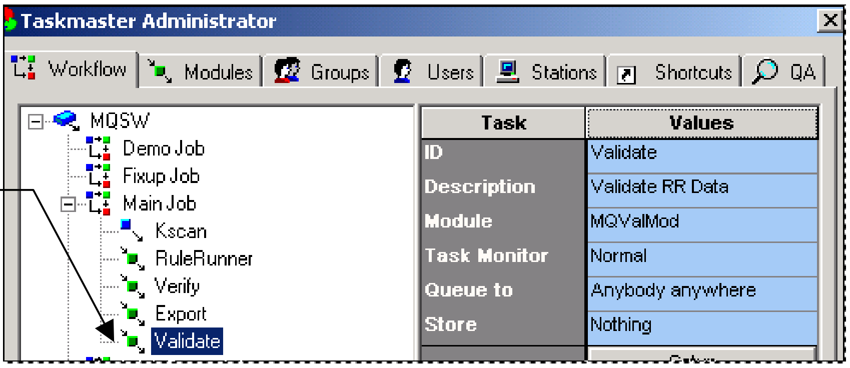
To Provide a Task with its Task Identity (continued)

Step	Action
2.	Select **New** and **Task** from the options you see– or use the Ctrl + N, Ctrl + T keyboard combinations.



3. In the open rectangle, enter a ***unique*** Task ID.

Task ID



Task	Values
ID	Validate
Description	Validate RR Data
Module	MQValMod
Task Monitor	Normal
Queue to	Anybody anywhere
Store	Nothing

4. Add a brief but important **Description** of the new task.

5. Use the **Module** drop-down list to assign the correct Task Module (Page 14).

6. Press the Apply button at the bottom of the tab to add the task's identity to the list on the left-hand side.

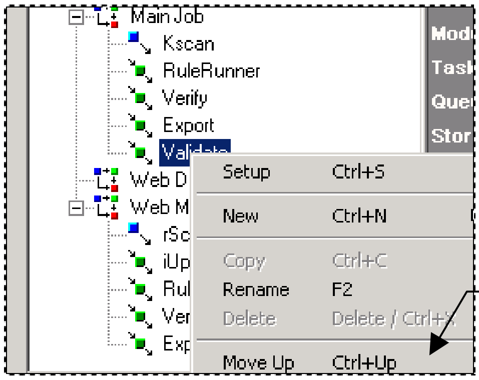
7. Select a **Task Monitor** location from the drop-down list (*Normal* is the default setting).

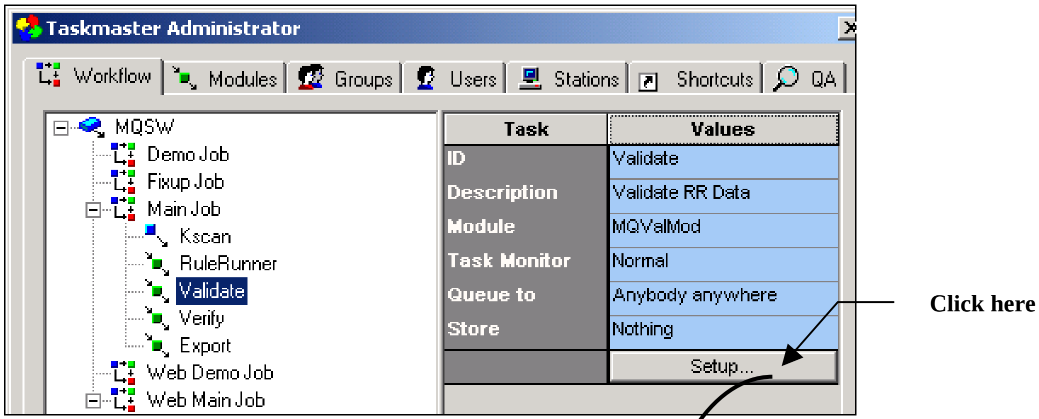
8. If this is a new task in a job with a group of tasks already in place, you can right-click on the Task ID to move the task up or down within the job's components.

9. If applicable, use the drop-down lists to add **Queue to** and **Store** parameters, according to the guidelines on Page 20.

10. Review the Task ID settings.

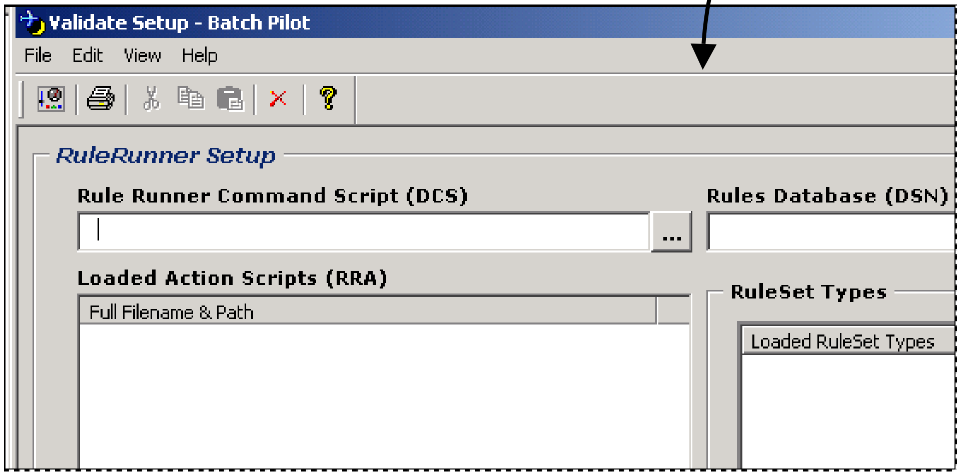
11. Click on the Setup button as a test: if the applicable ***Task Setup*** dialog appears, Stage 2 is complete and correct – and you're ready for Stage 3!





Task	Values
ID	Validate
Description	Validate RR Data
Module	MQValMod
Task Monitor	Normal
Queue to	Anybody anywhere
Store	Nothing

MQSW Taskmaster Administrator – *Workflow tab*
<u>Validate Task Definition</u>



Validate Task Setup dialog

The resulting **Job/Task Combination** – **Main.Validate**, in this case - is a fundamental processing unit of this *Taskmaster* application. For details, see upcoming chapters that examine task in each major category – *Scanning*, *RuleRunner* (Image Management, Recognition, Verification, Validation and Export), *Verify* and *FixUp*.

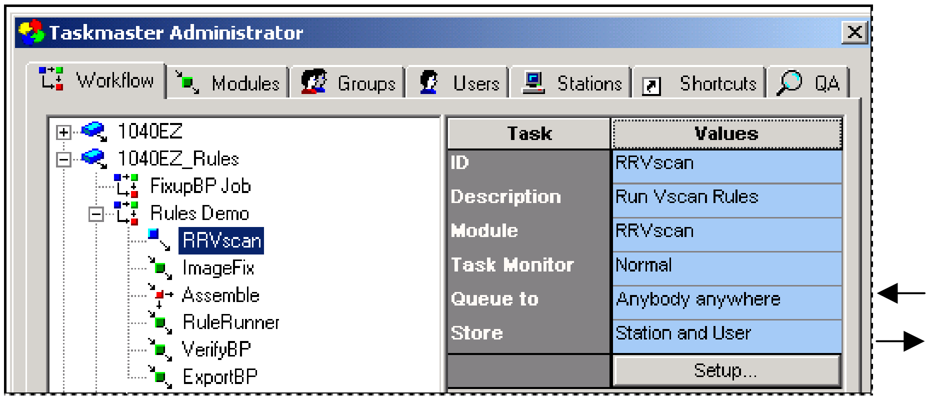
“Queue to” and “Store” Properties

The **Store** and **Queue to** properties of a Task Definition determine which operators and workstations can process individual batches. (*Taskmaster’s* User Definitions and Station Definitions govern the ability of individual operators and workstations to initiate specific Job/Task Combinations.)

The **Store** property looks forward; the **Queue to** property looks back. In the first illustration below, the definition of the *1040EZ* application’s RRVScan task specifies *Station and User* as its **Store** value. The definition of the VerifyBP task in the second illustration indicates *Same Station same user* as its **Queue to** property.

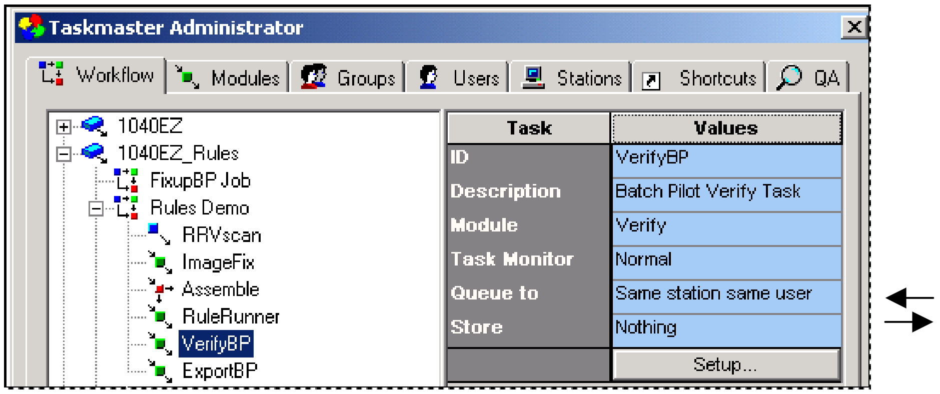
The **Store** property of the RRVScan task means that the batches it produces can only be processed by future tasks launched ***by*** the same operator ***from*** the same station. The **Queue to** property of the VerifyBP task (in this example) limits that task to batches previously processed by the same operator from the same station.

✓ Here, if the operator and workstation that are attempting to process batches in the VerifyBP task’s queue also created those batches by running the RRVScan task, the VerifyBP task can proceed. If there is a difference in either entity – operator or station – the task will not run.



Task	Values
ID	RRVscan
Description	Run Vscan Rules
Module	RRVscan
Task Monitor	Normal
Queue to	Anybody anywhere
Store	Station and User

Scan Task Definition



Task	Values
ID	VerifyBP
Description	Batch Pilot Verify Task
Module	Verify
Task Monitor	Normal
Queue to	Same station same user
Store	Nothing

Verification Task Definition

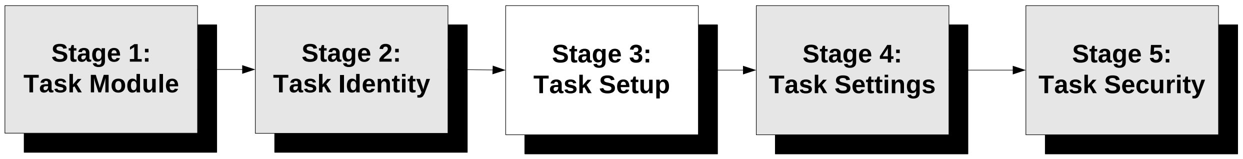
✓ The following table lists and explains these options.

Store… ➔	Queue to… 🡐	Explanation
“Nothing” or *“Station and user”*	*“Anybody anywhere”*	The *Anybody anywhere* value allows any operator and station to process batches in the task’s queue – and thus overrides the Station and User value assigned to an earlier task’s **Store** property.
“Station” or *“Station and user”*	*“Same station”*	The task can process only those batches previously processed from the same workstation.
“User” or *“Station and user”*	*“Same user”*	The task can process only those batches previously processed by the same operator.
“Station” or *“Station and user”*	*“Other station”*	The task can process only those batches previously processed from a different workstation.
“Station and user”	*“Same station same user”*	The task can process only those batches previously processed by the same operator, from the same workstation.
“Station and user”	*“Same station other user”*	The task can process only those batches previously processed by a different operator, from the same workstation.
“Station and user”	*“Other station same user”*	The task can process only those batches previously processed by the same operator, from a different workstation.
“Station and user”	*“Other station other user”*	The task can process only those batches previously processed by a different operator, from a different workstation.

☛ ***Important!*** An operator with *Job Monitor* security privileges can override these restrictions by double-clicking on a batch listing in the ***Job Monitor’s*** Batch Information table. (For a full explanation of security privileges, see Chapter 5 of the *Taskmaster Administrator’s Guide*. Chapter 5 of the *Taskmaster Windows & Dialogs Reference* explores all aspects of the ***Job Monitor***.)

Stage 3: Task Setup

In this stage, you click on the *Workflow* tab's Setup button to access the ***Setup*** dialog for the new task.

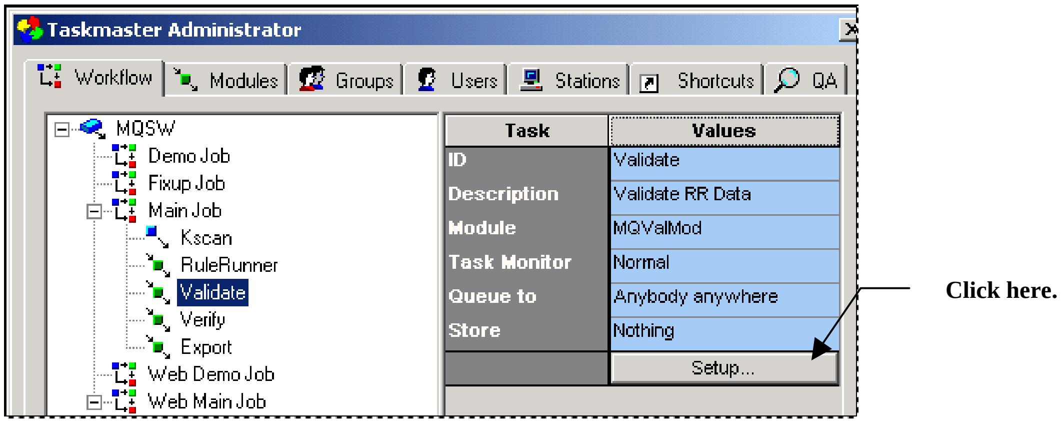


✓ The precise nature and content of a ***Setup*** dialog reflects the requirements of the underlying Task Project:

- Scan tasks that process paper combine the settings of a pre-configured software driver with the extensive settings you'll find in the tabs of the ***Setup*** dialog itself.
- Tasks that apply rules you've defined – tasks such as vScan, RuleRunner, Verify and Export – use the settings in the single tab of the ***RuleRunner Setup*** dialog. These settings specify the application's Rules database, RuleSet Types, and Actions files.

Chapters 7-10 examine the setup criteria of core *Taskmaster Rules* tasks. The steps below show you how to assign these criteria to a task.

Step	Action
1.	Open the *Workflow* tab of your **Taskmaster Administrator.**
2.	Highlight the Task ID of the applicable task.

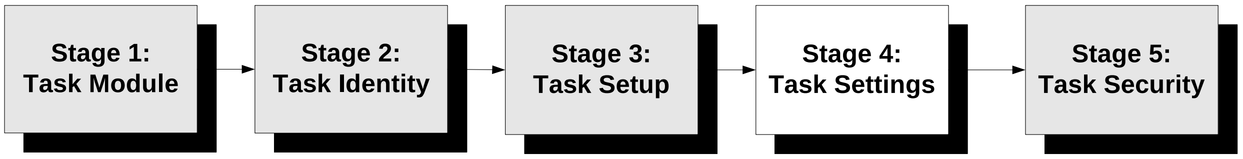


Task	Values
ID	Validate
Description	Validate RR Data
Module	MQValMod
Task Monitor	Normal
Queue to	Anybody anywhere
Store	Nothing

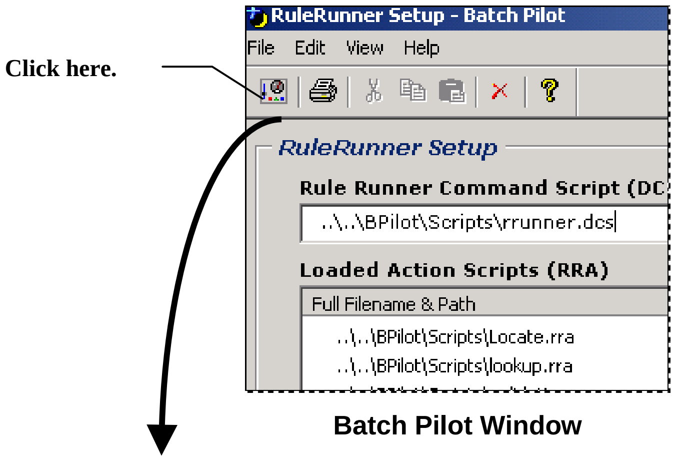
Step	Action
3.	Click on the Setup button to access the task's ***Setup*** dialog.
4.	Follow the guidelines and instructions in upcoming chapters as you provide the Task Definition with essential setup specifications.

Stage 4: Task Settings

This stage supplies a Task Definition with an additional group of specifications. These specifications reside in the ***Task Settings*** dialog.



To access this dialog, select **Task Settings** from the ***Batch Pilot Window's*** **File** menu after you have completed the Setup procedures in Stage 3 – or click on the window's **Task Settings** icon:



Rule Runner Command Script (DC

..\..\BPilot\Scripts\rrunner.dcs

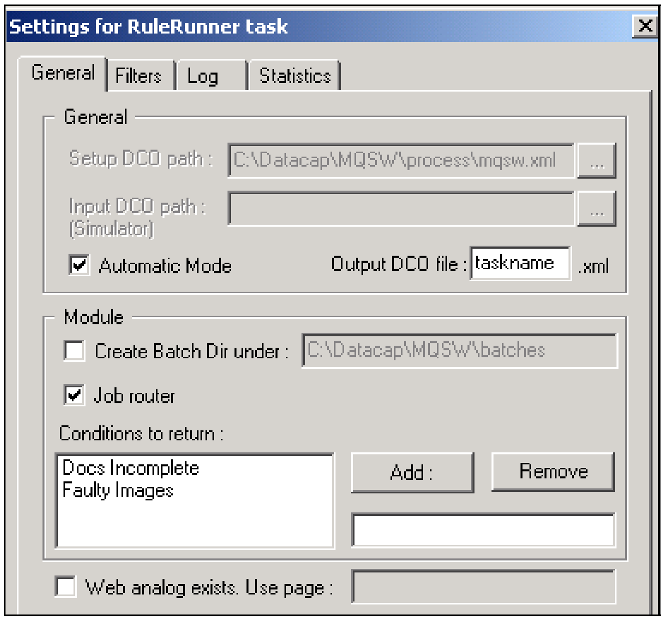
Loaded Action Scripts (RRA)

Full Filename & Path

..\..\BPilot\Scripts\Locate.rra

..\..\BPilot\Scripts\lookup.rra

Batch Pilot Window



General

Setup DCO path : C:\Datacap\MQSW\process\mqsw.xml

Input DCO path : (Simulator)

☑ Automatic Mode   Output DCO file : taskname .xml

Module

☐ Create Batch Dir under : C:\Datacap\MQSW\batches

☑ Job router

Conditions to return :

Docs Incomplete
Faulty Images

☐ Web analog exists. Use page :

Task Settings Window – *RuleRunner Task*

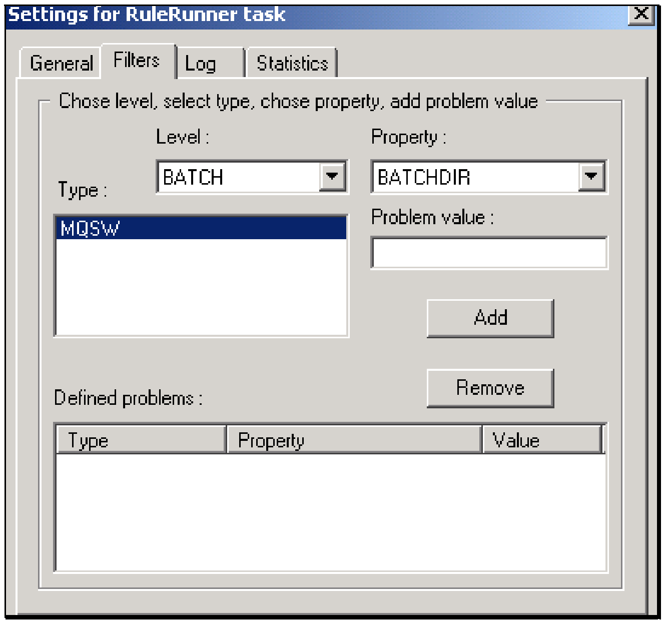
The tables which follow describe each tab of the ***Task Settings*** dialog:

General tab

Setting	Explanation
Setup DCO Path	The name and path of the underlying Task Project's Document Hierarchy. ***Alert!*** This value ***must*** also designate the Document Hierarchy associated with the ***Workflow*** Hierarchy of which this task is a part.
Input DCO Path	The name and path of a Document Hierarchy file (.xml) used for testing the Task Project.
Automatic Mode	A checkbox which, if selected, allows the task to run in the processing background, without operator participation. ***Alert!*** Do ***not*** check this option for tasks such as kScan or Verify, which rely on operator involvement.
Output DCO file	The name of the Page file (.xml) generated by the task. ***Alert!*** The default "taskname" value automatically combines the task's name with the **.xml** extension.
Module area	Settings in this area supplement the default characteristics that result when a Task Module is assigned to the task (Page 14).
Create Batch Dir under:	A checkbox which, if selected, designates the name and path of the application's **Batch Directory** in the accompanying field. This option is ***required*** if Task Definition has assigned a *Batch Creation* or *Job Router and Batch Creation* Task Module.
Job Router	A checkbox which, if selected, indicates that the Task Definition has assigned a Task Module with *Job Router* capabilities. If you select this option but do not select the **Create Batch Dir** checkbox, you can enter *string* values of conditions the task might encounter…conditions that would cause the task to divert the batch to a child job or take another other remedial action. For a complete explanation, see Chapter 10.
Add button	Adds a processing **Condition** you have entered in the data field below to the list of **Conditions to Run.**
Remove button	Deletes a processing **Condition** you have entered in the data field below.

Task Settings dialog - *General tab (continued)*

Setting	Explanation
Conditions to Return	A list of processing conditions that the task might encounter (Page 34). You can insert conditions ***only*** if you select the **Job Router** option and do not select the **Create Batch Dir** option. If the Task Definition assigns a Task Module of the *Job Router* type, this condition will appear with the task when you set up the Workflow Hierarchy.
Web analog exists. Use page:	A checkbox which, if selected, indicates that the task can be launched from a Taskmaster Client running in *client/server* or *serverless* mode – or can run on a Taskmaster Web Client with access to a *tmWeb* page. If you select this option, be sure to enter the Page ID in the accompanying field.
OK button	Saves current specifications and closes the ***Task Settings*** dialog.
Cancel	Closes the ***Task Settings*** dialog without saving new or modified specifications.
Apply	Saves current specifications but does not close the ***Task Settings*** dialog.


Settings for RuleRunner task

Chose level, select type, chose property, add problem value

Level : BATCH

Property : BATCHDIR

Type : MQSW

Problem value :

Defined problems :

Type	Property	Value

Task Settings dialog – *Filters tab*

Filters tab

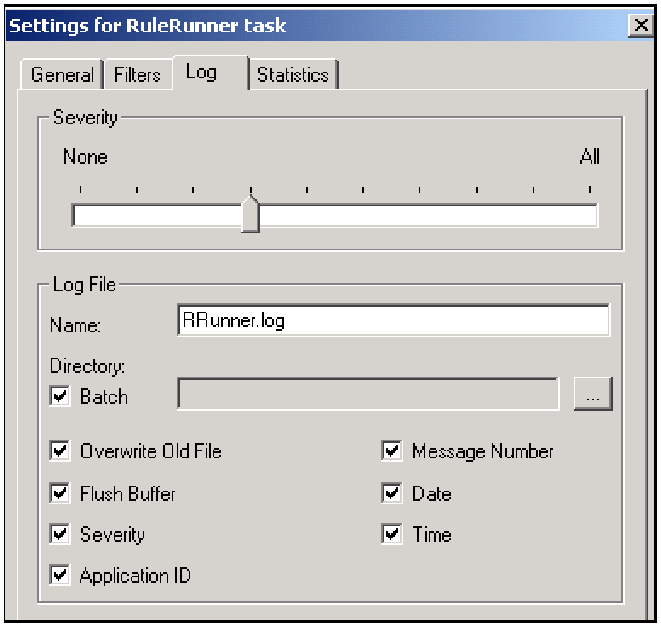
Settings in the *Filters* tab provide parameters the task can use to filter the contents of a batch ***before*** processing begins – or even to decide whether or not to process the batch itself. (Page 38 explains filters.)

Alert! If you use this tab to construct a filter, the tab will process ***only*** those objects of the Document Hierarchy that meet the filter's criteria. For example, a Verification task might only select **documents** with a processing **Status** of "146".

Setting	Explanation
Level	A drop-down list of Document Hierarchy levels: *Batch, Document, Page, Field.* ***Important!*** Although you can select any level, properties for that level will appear in the **Property** list to the right only you have bound a project form to that level of the Document Hierarchy in the Task Project's **Batch View** area. For details, see Chapter 3 of the *Guide to Batch Pilot.*
Type	Confirms the identity of the Document Hierarchy level you've chosen.
Property	A drop-down list of the properties of Document Hierarchy objects at the level to which you bound the project's ***Runtime*** or ***Setup*** form. There are three fundamental considerations: Although filtering involves one level of the Document Hierarchy, you can establish multiple criteria at that level. The properties of a particular level become available only after you bind a form to that level (see the *Guide to Batch Pilot*.) During Task Definition, an Administrator can review and modify the filtering parameter in the tab's **Defined Problems** area.
Problem Value	A value for the property you've selected from the **Property** drop-down list. The task will use this value as the basis for an Accept/Reject processing decision.

Log tab

Logs can be exceptionally helpful, especially when you're setting up and testing a project's tasks. The settings in this tab determine the nature of a log, its location (typically within a batch) and its contents.



Name: RRunner.log
Directory:
☑ Batch
☑ Overwrite Old File
☑ Flush Buffer
☑ Severity
☑ Application ID
☑ Message Number
☑ Date
☑ Time

Task Settings dialog – *Log tab*

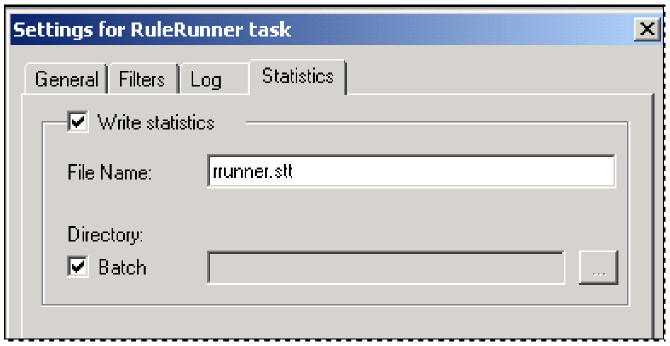
Setting	Explanation
Severity	This continuum measures the relative importance of problems the task is likely to encounter and report on in the Log file it provides for each batch. As the pointer moves to the right, severity drops while the scope and number of reported problems increases. If you move the pointer leftwards, only the more critical problems are considered and, steadily, the number of logged problems diminishes.
Log File	The name of the Log file that the task will add to a batch.
Directory	A check box which, if activated, directs *Taskmaster* to place Log files in the Batch Folders of the application's **Batches** directory. If you do not select this option, you ***must*** enter the name and path of the directory that is to hold Log files in the accompanying field.
Overwrite Old Files	A check box which, if activated, directs *Taskmaster* to replace an earlier Log file with the current file if they share a name and location.

Task Settings dialog - *Log tab (continued)*

Setting	Explanation
Flush Buffer	Activating this check box means that *Taskmaster* will continuously transfer messages from a mid-stream buffer to the log as processing continues. If you do ***not*** select this feature, *Taskmaster* will store messages in a buffer for a particular period. This alternative increases processing speed but compromises safety: if an error occurs, crashes, you may lose the messages in the buffer.
Severity	Adds a Severity Code to each message in the log.
Application ID	Adds an application code to each message.
Message Number	Includes a unique Message Number with each message in the log.
Date	Adds the Date.
Time	Adds the Time.

Statistics Tab

Settings in this tab authorize the task to generate a special Statistics file (.sts) when it completes its work with a batch.


Settings for RuleRunner task

General | Filters | Log | Statistics

☑ Write statistics

File Name: rrunner.stt

Directory:

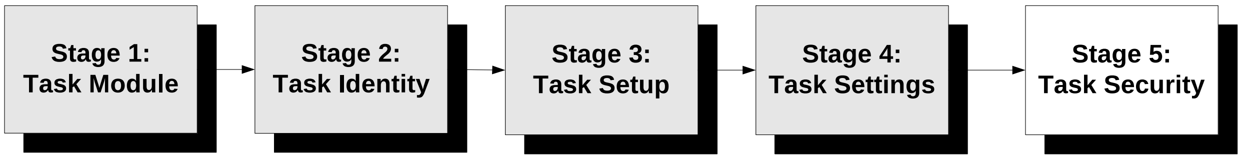
☑ Batch

Task Settings dialog – *Statistics tab*

Setting	Explanation
Write Statistics	A checkbox which, if activated, directs the task to compose a Statistics file with details of its operations with a batch.
File Name	The name and extension of the Statistics file.
Directory	A check box which, if activated, directs *Taskmaster* to place Log files in the Batch Folders of your application’s **Batches** directory. If you do not select this option, use the accompanying field to enter the name and path of the directory that is to hold Statistics files.

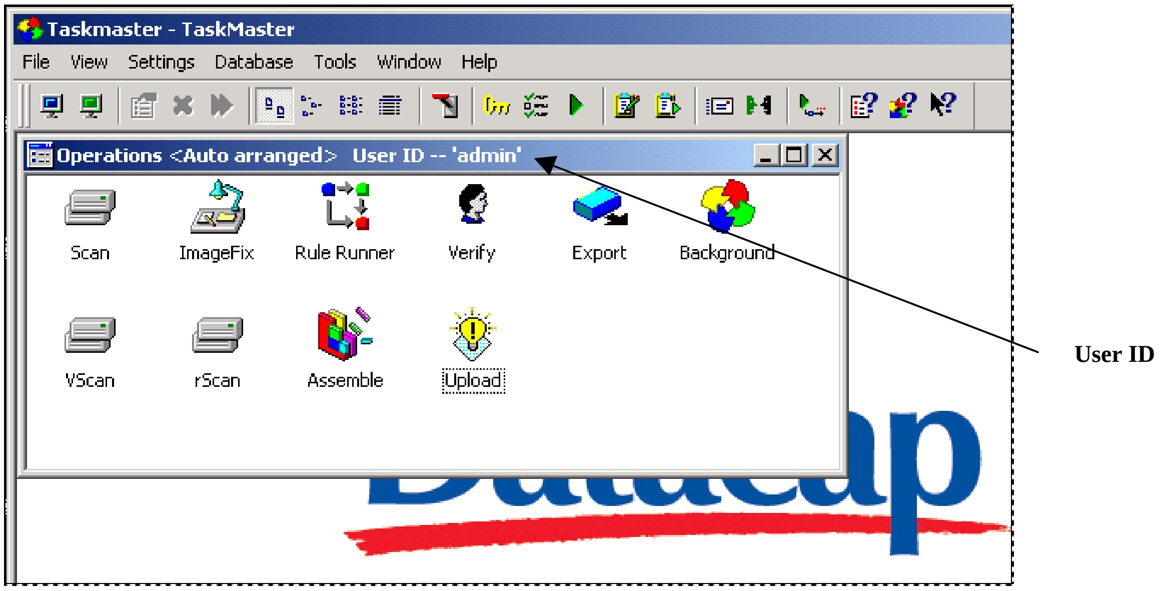
Stage 5: Task Security

The concluding stage of Task Definition determines which users and workstations can process a task – or, more accurately, a Job/Task Combination.



✓ Chapter 5 of the *Taskmaster Administrator's Guide* covers all aspects of Application Security. This topic summarizes the steps you take to limit access to a specific Job/Task Combination.

The ***Operations*** window of a *Taskmaster Rules* application opens automatically when a **user** - an operator, Supervisor or Administrator - signs on to the application.



Taskmaster Administrator – *with Operations window*

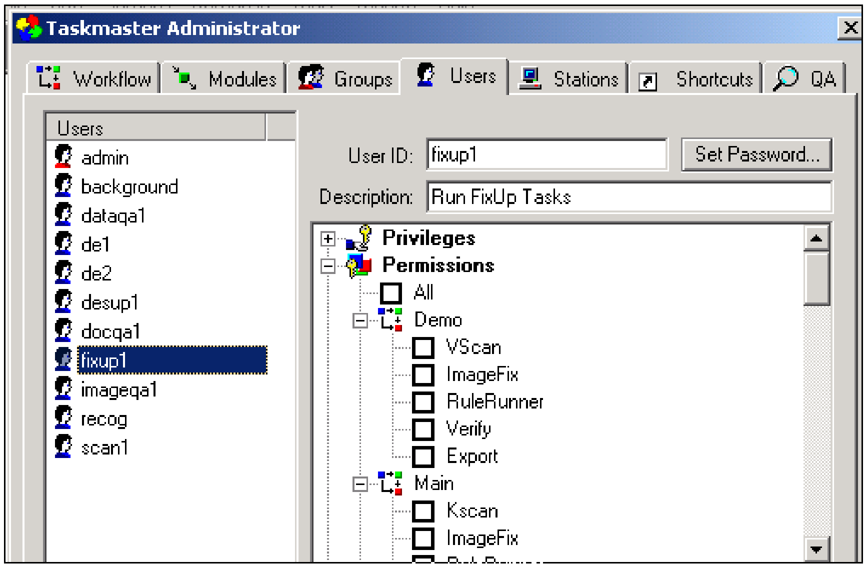
The ***Operations*** window contains a number of Job/Task shortcut icons; the mix of icons that actually appears in the window depends on the security parameters you've assigned to the current user; to the workstation he or she is operating; and to the individual Job/Task shortcut icons. The ***Operations*** window will not display an icon unless:

- The user has permission to run ***all*** Job/Task Combinations represented by the icon.
- The workstation also has permission to run the Job/Task Combinations represented by the icon.
- Job/Task Combinations covered by the icon are those authorized for ***both*** the current user and his or her workstation.

After your definitions of jobs and tasks are complete, you'll take the steps outlined below to establish Job/Task Security parameters.

Job/Task Security

Step	Action
1.	Open the *Users* tab of your application's ***Taskmaster Administrator.***
2.	Construct a User Definition for the individual who will be running the task, or scroll down the **Users** list for a user you've already defined. (***Alert!*** The Taskmaster Application Wizard described in Chapter 2 automatically installs a default set of users.)

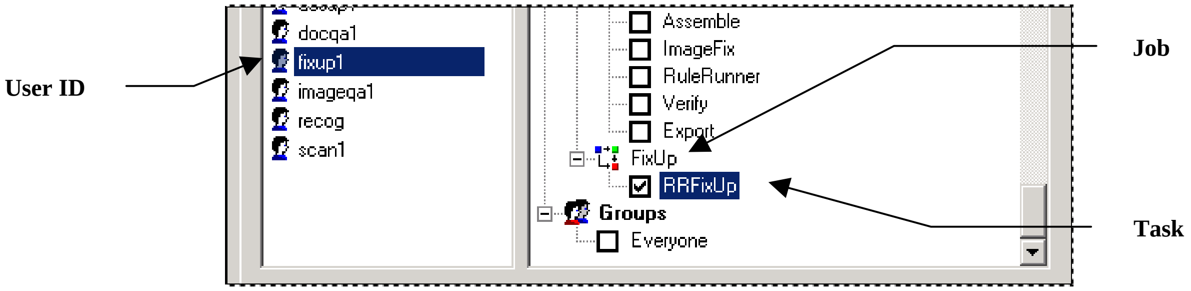


Users
admin
background
dataqa1
de1
de2
desup1
docqa1
fixup1
imageqa1
recog
scan1

User ID: fixup1
Description: Run FixUp Tasks

Privileges
Permissions
All
Demo
VScan
ImageFix
RuleRunner
Verify
Export
Main
Kscan
ImageFix

Step	Action
3.	Highlight the User ID and scroll through the **Permissions** list on the tab's right-hand side until you reach a job and task that this user can run.



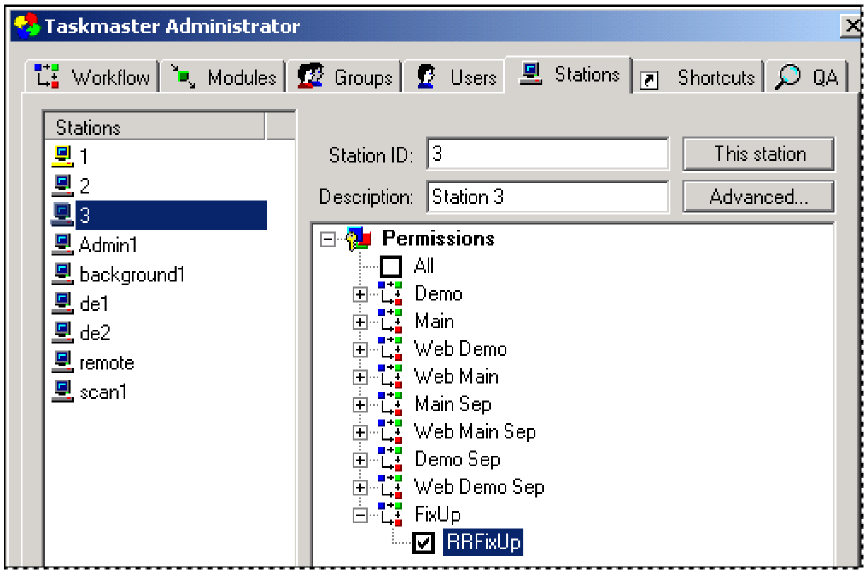
docqa1
fixup1
imageqa1
recog
scan1

Assemble
ImageFix
RuleRunner
Verify
Export
FixUp
RRFixUp ✓
Groups
Everyone

Step	Action
4.	Repeat Step #3 for any other Job/Task Combinations this user is authorized to run. (Note that the *Admin* user probably has permission to run **All** jobs and tasks!)
5.	Press the Apply button at the bottom of the ***Taskmaster Administrator.***

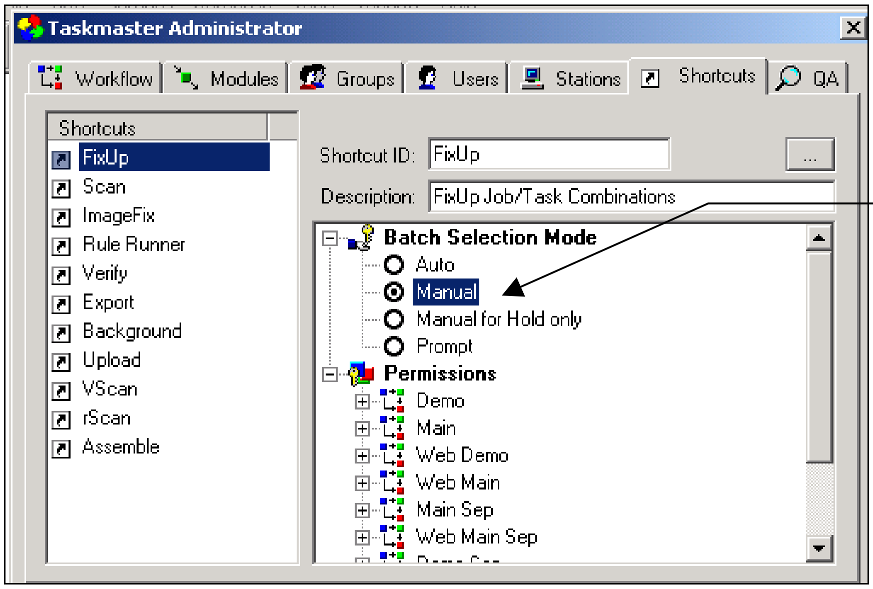
Job/Task Security (continued)

Step	Action
6.	Move to the ***Taskmaster Administrator's*** *Stations* tab.
7.	From the **Stations** list, select the Station ID of a workstation you're authorizing to run this Job/Task Combination (or construct a new Station Definition.)



Station ID: 3
Description: Station 3

Step	Action
8.	Scroll through the **Permissions** list on the right to find the Job/Task Combination(s) that this station can run. ***Important!*** Together, this step and Step #3 allow a user and a workstation to launch the same task.
9.	All that's needed now is a Job/Task shortcut icon in the ***Operations*** window. In this example, however, there is no such icon. To remedy this problem, begin by accessing the ***Taskmaster Administrator's*** *Shortcuts* tab (illustrated on the next page.)
10.	Press the Add button at the bottom of the ***Taskmaster Administrator.*** This stop will activate the **Shortcut ID** and **Description** fields on the tab's right-and side – and "gray out" the other fields.
11.	Enter a unique ID and a brief ***but important*** description of this Job/Task shortcut icon.
12.	Press the Apply button at the bottom of the ***Taskmaster Administrator.***
13.	Confirm that the Shortcut ID is now part of the **Shortcuts** list on the left; highlight it and that all fields are active.
14.	Select the appropriate **Batch Selection Mode.** (An operator-intensive task such as FixUp probably uses *Manual* or *Manual for Hold Only*.)



Shortcuts: FixUp, Scan, ImageFix, Rule Runner, Verify, Export, Background, Upload, VScan, rScan, Assemble

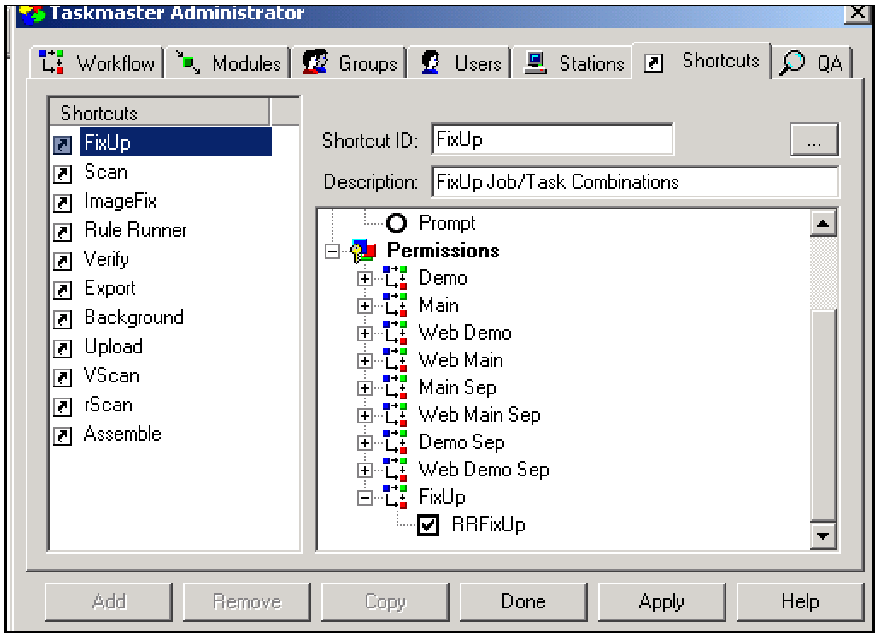
Shortcut ID: FixUp

Description: FixUp Job/Task Combinations

Batch Selection Mode
- Auto
- [✓ Manual]
- Manual for Hold only
- Prompt

Permissions
- Demo
- Main
- Web Demo
- Web Main
- Main Sep
- Web Main Sep

Taskmaster Administrator – *Shortcuts tab*



Shortcuts: FixUp, Scan, ImageFix, Rule Runner, Verify, Export, Background, Upload, VScan, rScan, Assemble

Shortcut ID: FixUp

Description: FixUp Job/Task Combinations

- Prompt

Permissions
- Demo
- Main
- Web Demo
- Web Main
- Main Sep
- Web Main Sep
- Demo Sep
- Web Demo Sep
- FixUp
  - [✓] RRFixUp

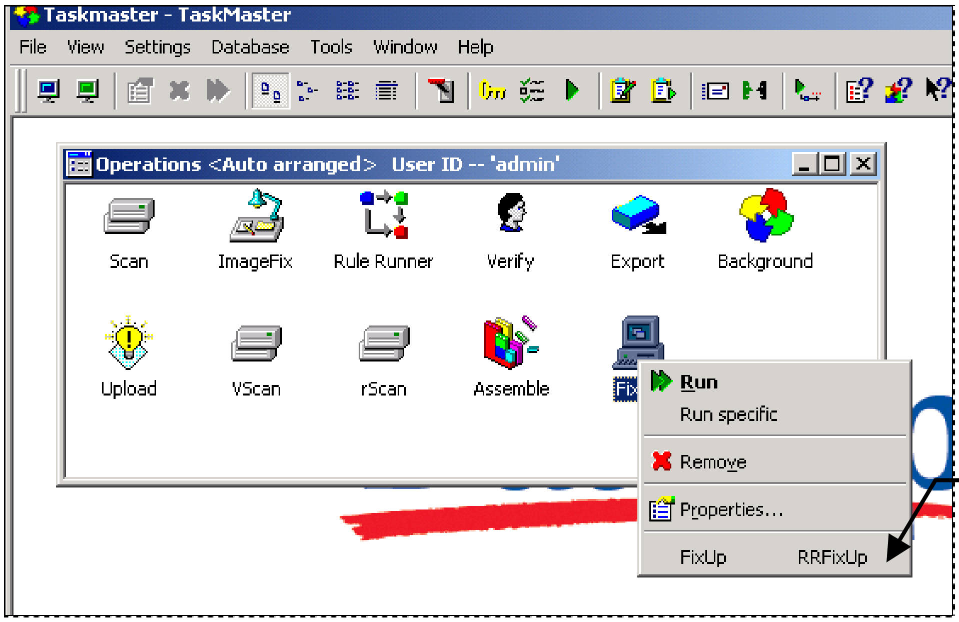
Taskmaster Administrator – *Shortcuts tab*

Job/Task Security (continued)

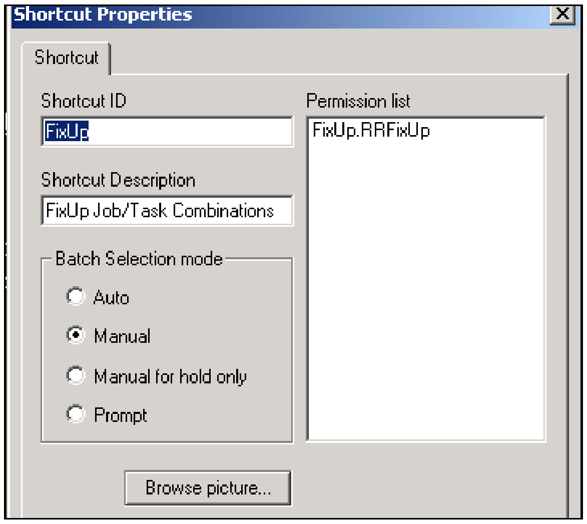
Step	Action
15.	Scroll through the **Permissions** list until you can authorize this Job/Task shortcut to carry out the same job and task - or jobs and tasks! - you've assigned to the user and workstation.
16.	Click on the Apply button.

Job/Task Security (continued)

Step	Action
17.	Close the ***Taskmaster Administrator.***
18.	Open the ***Operations*** window.
19.	Conform that this window now features the Job/Task shortcut icon you've just defined.
20.	Right-click on the icon: be sure the job and task are listed. (For more information, open the icon's **Properties** dialog.)



Job and task



Shortcut ID: FixUp

Permission list: FixUp.RRFixUp

Shortcut Description: FixUp Job/Task Combinations

Batch Selection mode: ( ) Auto [✓ Manual] ( ) Manual for hold only ( ) Prompt

Shortcut Properties

✓ ***Remember!*** The previous pages only summarize a very important topic: Application Security. For complete explanations, please refer to following documentation:

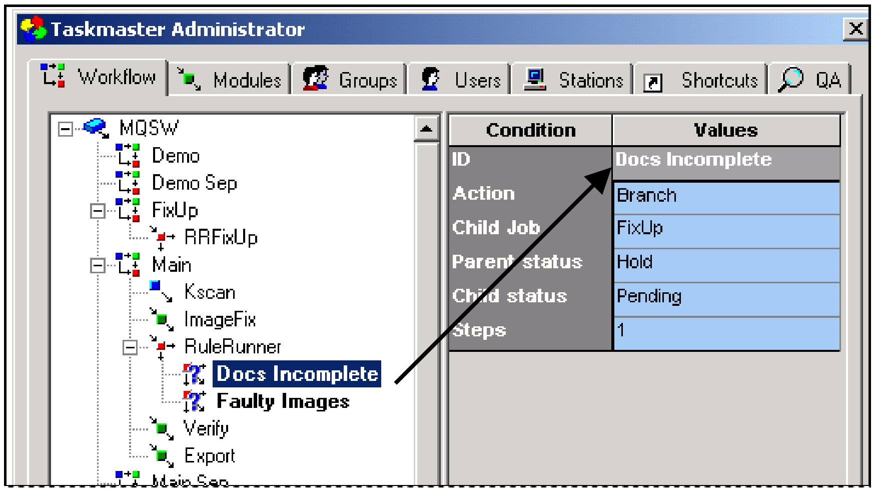
- *Taskmaster Administrator's Guide*, Chapter 5.
- *Taskmaster Windows & Dialogs Reference*, Chapter 6.
- *Guide to Taskmaster Web*, Chapter 2 and Chapter 3.

Special Task Setup Procedures

Batch Routing

Batch Routing occurs when a batch confronts a processing condition that diverts the batch from a task of a ***parent*** job to a ***child*** job for review and possible correction. After this remedial step, the batch returns to the initiating task of the ***parent*** job.

Because the Batch Routing decision is made by a task in the ***parent*** job, configuration of the mechanism involves the setup of that task. In the illustration below, the ***parent*** Main job's RuleRunner task branches any batch with a *Docs Incomplete* condition to the ***child*** FixUp job for further attention. Settings governing the *Docs Incomplete* condition are on the right.



Condition	Values
ID	Docs Incomplete
Action	Branch
Child Job	FixUp
Parent status	Hold
Child status	Pending
Steps	1

Taskmaster Administrator – *Workflow tab*
Batch Routing Parameters: RuleRunner Task

The configuration of a Batch Routing procedure occurs in six steps.

Step 1: Parent and Child Job Definitions

This stage use the *Workflow* tab of the *Taskmaster Administrator* to construct a Workflow Hierarchy made up of a ***parent*** job that includes one or more tasks with Batch Routing capabilities and one or more ***child*** jobs that can process a routed batch.

In the illustration above, the Main job of the MQSW workflow is the ***parent*** and the FixUp job is the ***child*** job. Details of this relationship are spelled out in Step 4 when you specify the condition that links the diverting task to the ***child*** job and identify the routing action that the task will take. For now, be sure that you have a workflow with both jobs.

Step 2: Task Module Definition

As Page 14 explains, every Task Definition must designate a Task Module.

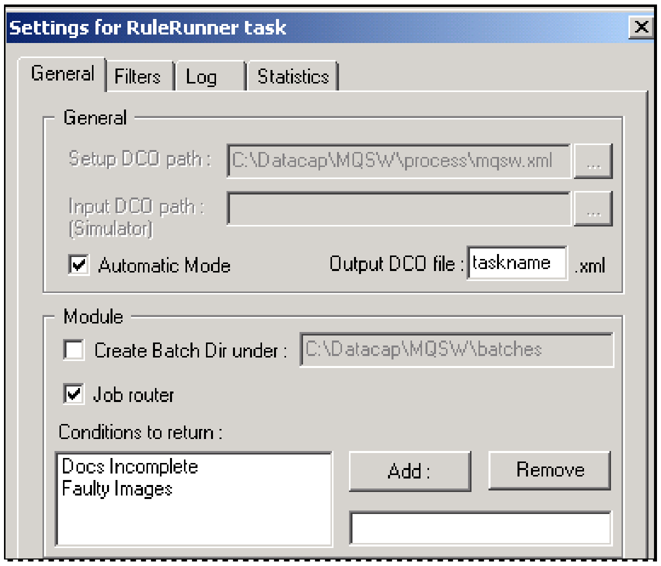
The module of a task that can route a batch to a ***child*** job must include *Job Router* as its Type property - as part of its definition:

Step 3: Task Definition – Batch Routing Task Settings

The *General* tab of the *Task Settings* dialog (Page 23) has two specifications that are essential for the Task Definition of any task with Batch Routing capabilities

Be sure to check the **Job Router** option in the **Module** area. This reinforces a comparable specification of the Task Module definition.

Enter one or more processing conditions in the **Conditions to Return** field. The illustration below lists the two existing conditions for the RuleRunner task. A condition you specify here will show up as a property of the task in Step 4.


Settings for RuleRunner task

Setup DCO path : C:\Datacap\MQSW\process\mqsw.xml

Input DCO path : (Simulator)

[✓] Automatic Mode

Output DCO file : taskname .xml

[ ] Create Batch Dir under : C:\Datacap\MQSW\batches

[✓] Job router

Conditions to return :

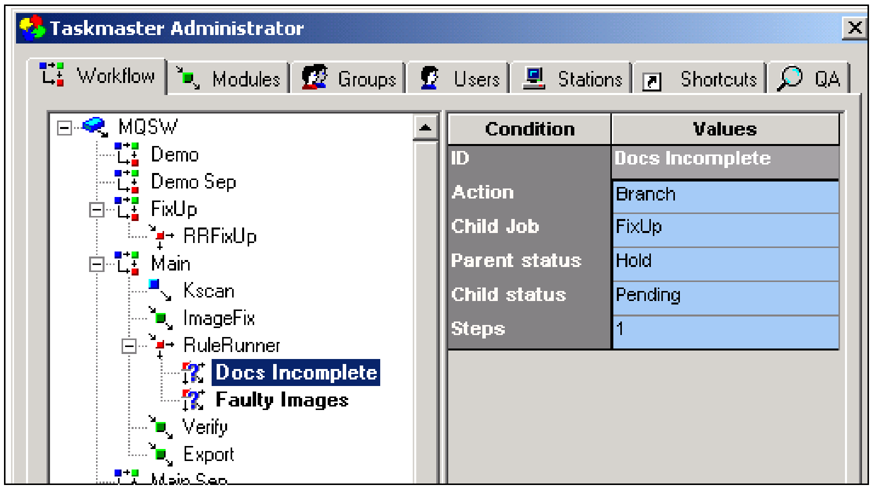
Docs Incomplete
Faulty Images

Task Settings dialog – *General tab*
<u>RuleRunner Task</u>

Step 4: Task Definition – Condition and Routing Action Parameters

Step 4 links the task of the ***parent*** job to a ***child*** job when a batch confronts a processing condition designated in Step 3. This stage also assigns the routing action that the task will take in response to the condition you select.

In the example below, the Task Definition lists the two conditions added in Step 3. The tab's right side, however, lists the parameters for the *Docs Incomplete* condition.



Condition	Values
ID	Docs Incomplete
Action	Branch
Child Job	FixUp
Parent status	Hold
Child status	Pending
Steps	1

Task Master Administrator – *Workflow tab*
RuleRunner Task Definition

A condition's parameters include:

ID: the condition's name (see Step 5).

Action: a drop-down list of alternative routing actions that the task can take when it confronts the processing condition. For explanations of each routing action, see Page 34.

Child job: a drop-down list of jobs other than the current job that are part of the workflow. You selection of a job from this list immediately establishes the ***child-parent*** job relationship.

Parent Status: two processing statuses – *Pending* and *Hold* - that can be assigned to the batch as a member of the ***parent*** job's queue.

Hold is usually the choice if the diverting task is operator-intensive. This status means that the batch can only be started up by this operator when it re-joins the ***parent*** job's workflow.

Pending allows the batch to return automatically to the ***parent*** job's workflow when it is ready.

Child Status: the processing status assigned to the batch when it joins the queue of the ***child*** job's first task. *Pending* is usually assigned.

Steps: The task in the ***parent*** job to which the batch returns after processing by the ***child*** job:

Blank (no value) prevents the ***child*** job from returning the batch to the parent.

Zero (0) returns the batch to the originating task in the ***parent*** job.

Positive (+1, for example) forwards the batch to a task further along in the ***parent*** job.

Negative (-1, for example) returns the batch to task earlier in the job.

Alert! In the case of a *Jump* action, this setting determines which task in the *current* job will be next to process the batch.

Step 5. Batch Routing Rule Definitions

Tasks of a *Taskmaster Rules* application respond to rules *you* define. A task such as Assemble or RuleRunner can divert problem batches to a child job *only* if the rules are complete and have been linked directly to the task in Step 6.

For more about the rules involving Batch Routing and the actions they contain, see Chapter 10.

Stage 6: Task Definition – Rules and Actions Specifications

In Stage 1, you defined the rules that will determine how and when Batch Routing occurs. This stage actually assigns the rules to the diverting task – to the RuleRunner task, in this example.

When you select the Task ID from the list of the ***parent*** job's tasks in the *Workflow* tab, and click on the Setup button, the ***Task Setup*** dialog will appear:

Batch Routing relies on three sets of specifications in this dialog.

First, be sure that the RuleSet Type you worked with in Step 5 is part of the RuleSet Type list, and that you have selected its checkbox.

In the **RuleRunner Command Script** field, enter a name and path similar to the specification in the illustration. (Among its other attributes, this script contains RuleRunner actions such as **Task_NumberOf Splits** and **Task_RaiseCondition**.)

Check that the **Loaded Actions Scripts (RRA)** field lists the Actions files that are part of the **CheckForFixup** RuleSet Type's Actions Library (see Step 5).

Action Drop-down List

The **Action** drop-down list (Step 4) has six routing actions:

None. This is the default, temporary setting you can use as you assemble a condition's properties: it makes certain that your selection of a condition has no immediate impact on the processing of a batch.

Branch. This action diverts the batch that the task is processing from the ***parent*** job to the task(s) of a ***child*** job. When the ***child*** job is finished, the batch returns to the ***parent*** job. ***Alert!*** A workflow can contain a ***child*** job hierarchy in which a task in a ***child*** job branches to a job on a lower tier. In this situation, the first ***child*** job becomes the parent to the second ***child*** job.

Most Batch Routing procedures employ the *Branch* action.

Jump. This action is a processing shortcut that skips one or more tasks in the workflow, and places the batch in a more advanced position, or in an earlier position. The *jump* action uses a value in the condition's **Steps** property to indicate the next task, and does not involve the use of a ***child*** job.

Split: The *Split* action splits one or more documents from the current batch. The task in the ***parent*** job creates a sub-batch, adds the document(s) to it and sends the sub-batch to a ***child*** job for special processing. While the ***child*** job takes care of the sub-batch, the ***parent*** job processes the remaining documents in the batch. The ***child*** job does *not* return the sub-batch to the parent job when processing is complete.

Stop: This action terminates *Taskmaster's* processing of the batch.

Hub: The Hub action combines features of the Branch and Split actions. Hub splits documents from the batch, adds them to sub-batches, and delivers the sub-batches to various child jobs. When the ***child*** jobs have processed the sub-batches, they return to the ***parent*** jobs.

✓ Chapter 10 explores all aspects of Job Routing procedures.

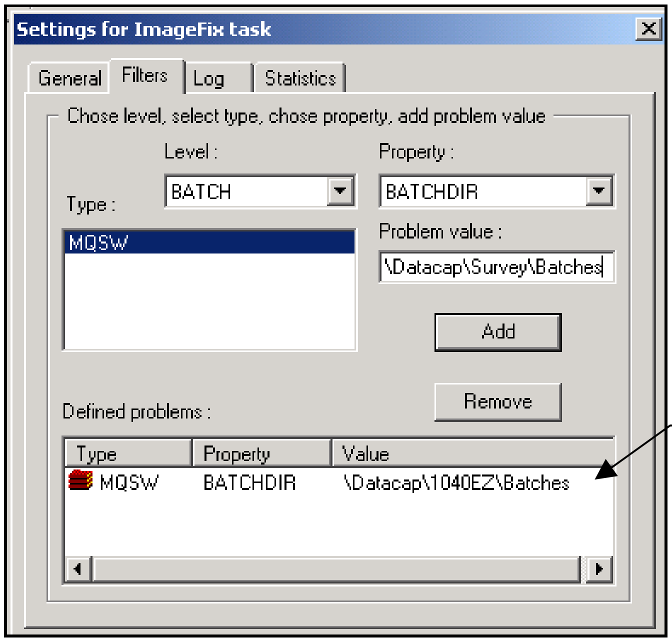
Filters

Together, the *Filters* tab of the ***Task Settings*** dialog and a Task Project's binding mechanism can design an efficient sieve for the project's task.

Filtering works at only one level of the Document Hierarchy at a time, and that's where your binding skills come in. (For an explanation of Form Binding, see Chapter 3 of the *Guide to Batch Pilot*.)

To construct a filter, open a task's ***Task Settings*** dialog (Page 23) and go right to the *Filters* tab (illustrated on the next page.)

Continued on the next page ⟶


Settings for ImageFix task

General | Filters | Log | Statistics

Chose level, select type, chose property, add problem value

Level : BATCH

Property : BATCHDIR

Type : MQSW

Problem value : \Datacap\Survey\Batches

Add

Remove

Defined problems :

Type	Property	Value
MQSW	BATCHDIR	\Datacap\1040EZ\Batches

Task Settings dialog – *Filters tab*

✓ In this example, the filtering mechanism has been bound to the **Batch** object of the MQSW Document Hierarchy. The result is a list of filtering **Properties** available at this level.

The Administrator has selected *BATCHDIR* as a possible problem, and specified the Batches directory of another application in the **Defined Problems** area. Therefore, if batches mistakenly flow from the **Batches** directory of the *1040EZ* application, the ImageFix task will halt its activities until the problem is resolved.