2

Taskmaster Architecture

Chapter 2 describes the broad structure of a **Datacap Taskmaster** configuration that works principally with *Taskmaster Rulerunner Service* applications.

Chapter 2 covers these topics:

I. Datacap Taskmaster - Overview **2-2**

II. Document Hierarchies **2-6**

III. Rulesets and Rules **2-8**

IV. Task Profiles **2-10**

V. Fingerprints **2-11**

VI. Workflows **2-13**

✓ ***Important!*** The *Taskmaster Administrator's Guide* refers regularly to the concepts and terminology introduced in this chapter. Be sure to familiarize yourself with it contents before you proceed to other chapters.

Chapter 4 looks more closely at examines the nature of a Taskmaster Rulerunner Service application.

This *Taskmaster Administrator's Guide* also relies on the *1040EZ* training application for examples and illustrations.

Datacap Taskmaster - Overview

The chart on the following page highlights relationships between the two principal components of **Datacap Taskmaster:**

- At the top, Datacap Studio designs and develops a *Taskmaster* Rulerunner *Service* application.
- On the lower level, Taskmaster Client prepares workflows and runs the application.
- In between, the Rulerunner Service funnels information from Datacap Studio to the Taskmaster Client to be sure that rulerunner tasks process the contents of each batch, according to the requirements of rules established in Datacap Studio.

Datacap Studio is responsible for:

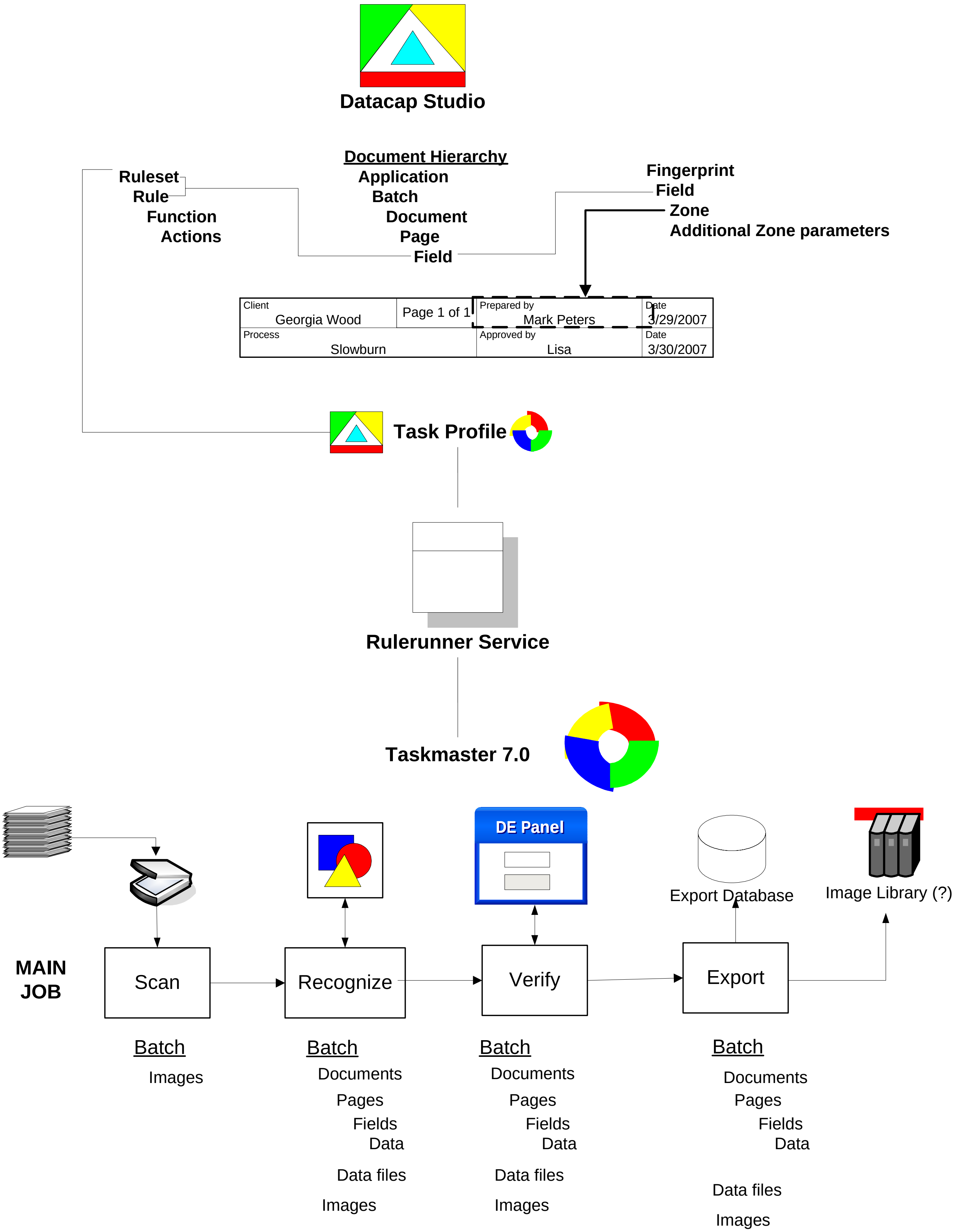
- ♦ Assembling and maintaining an application's Document Hierarchy (Page 6)
- ♦ Creating fingerprints for ***source*** pages, and zoning their critical fields (Page 11)
- ♦ Defining the Rulesets and rules that govern the way tasks process objects at four levels of the Document Hierarchy: **Batch, Document**, **Page** and **Field**. (Page 6).

Taskmaster Client sets up and runs components of an application's workflows, jobs and tasks – and incorporates specifications of both the Document Hierarchy and of the Ruleset Hierarchies.

The chart shows the tasks of a sample application's Main job – the job that takes care of a form from the moment it enters the Scanning queue right up until the data on a ***source*** page have been verified, validated and exported. (***Important!*** A ***source*** page contains fields with data. A typical Recognize task identifies a ***source*** page, locates its fields, and recognizes any data in the fields.)

Note, too, the one-to-one relationships between the levels of the Document Hierarchy (at the top) and the **Data** categories listed below the four tasks. This correspondence stipulates how and when the application retrieves, evaluates and exports **data** about batches, documents, pages – and the **fields** on every ***source*** page.

✓ Chapter 4 builds on the information in this chapter to explore all aspects of a real application – the *1040EZ* training application – and to demonstrate its operations in a friendly, hands-on mode.



Client Georgia Wood	Page 1 of 1	Prepared by Mark Peters	Date 3/29/2007
Process Slowburn		Approved by Lisa	Date 3/30/2007

Datacap Studio and Taskmaster Client

Datacap Studio occupies a central role in an application's setup and operations.

✓ The basic diagram on the next page highlights the essential components that you manage with Datacap Studio.

At the top:

- **Datacap Studio** prepares and manages a **Document Hierarchy** with four processing levels – **Batch**, **Document, Page** and **Field.** The Document Hierarchy clearly identifies and distinguishes **objects** at each level. Each object has **properties** that are just as easy to identify and evaluate. At the top of the Document Hierarchy, an inactive **Application** object identifies the application.
- Datacap Studio is responsible for defining an application's **fingerprint(s).** A fingerprint is a pair of files that represent a single ***source* page** – a page with data that is to be recognized, verified, validated and exported.

  One member of the pair is an Image file (.tif);

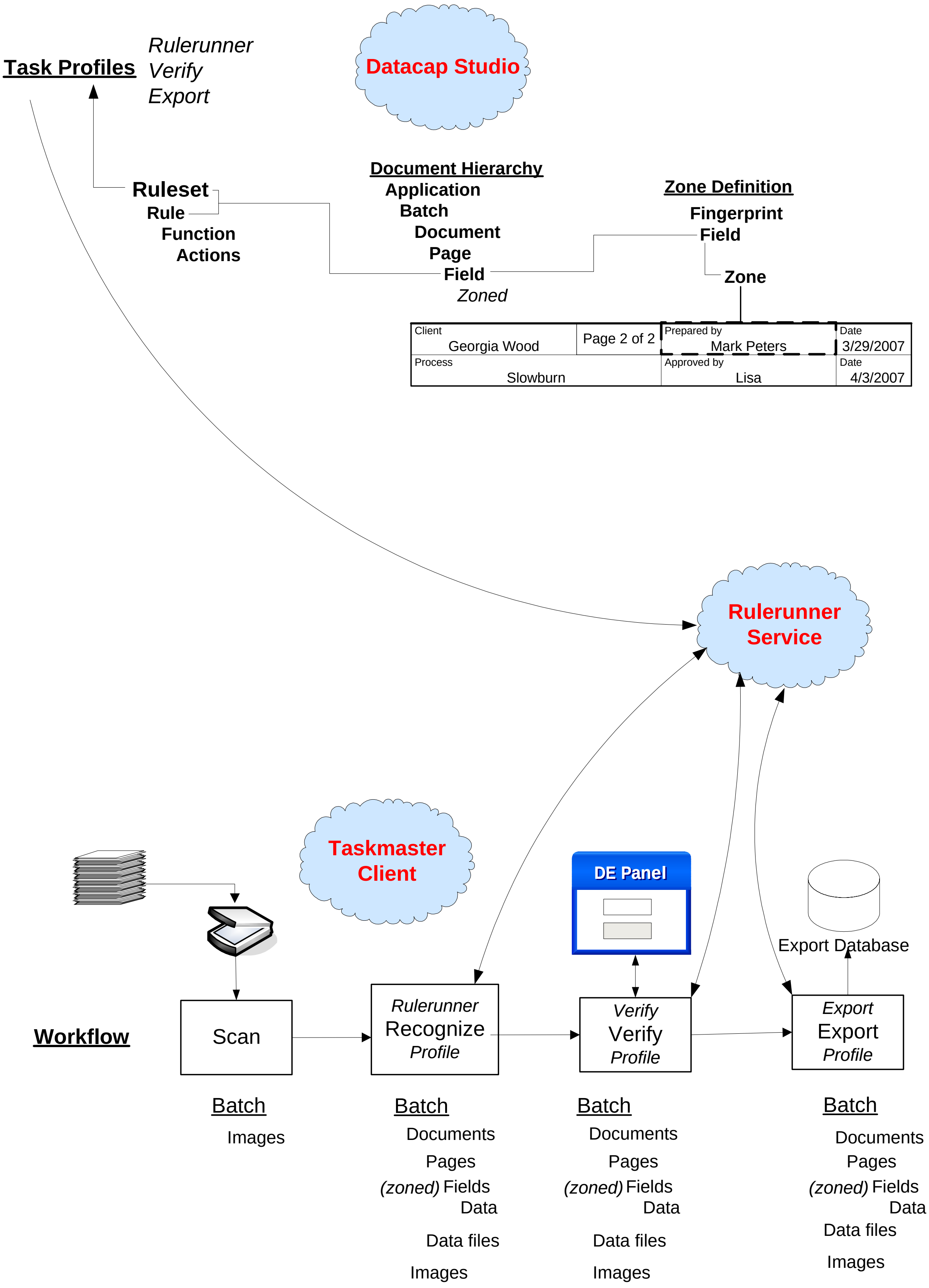
  The other is a Recognition file (.cco).

  Zoning identifies and locates fields and blocks of fields on the Image file and supplies this information to the Recognition file. This procedure can also assign special recognition criteria to individual fields.

  The illustration highlights just one zoned **Field** object in the fingerprint: *Prepared by*. (The Document Hierarchy includes this **Field** object and accumulates values about the field and its properties.)
- A **Rule** consists of one or more **functions**, and a function is made up of one or more **actions**.

  In addition, a rule belongs to a particular **Ruleset.** The resulting **Ruleset/Rule combination** is **bound** to a specific object of the Document Hierarchy – in this case, to the *Prepared by* **Field** object of the Document Hierarchy. An application's *Rulerunner* tasks (Recognize, Verify and Export tasks, for example) operate in response to the rules of a Ruleset.
- Datacap Studio assigns specific Rulesets (complete with their rules) to individual **Task Profiles**. During the set up of a workflow's **rulerunnner tasks**, a Task Profile is associated with each rulerunner task.

At the bottom of the chart, the application's **Taskmaster Client** assembles the tasks that result in a workflow. In this basic example, the opening Scan task creates a new batch, processes paper using a physical scanner, and adds the resulting Image files to the batch. Because the Scan task is ***not*** a rulerunner task, it is not linked to one of Datacap Studio's Task Profiles.



Client Georgia Wood	Page 2 of 2	Prepared by Mark Peters	Date 3/29/2007
Process Slowburn		Approved by Lisa	Date 4/3/2007

✓ In this example, the workflow's other tasks – Recognize, Verify, and Export – ***are*** rulerunner tasks: each responds to the rules that belong to the Rulesets assigned to a Task Profile. The core procedure that Taskmaster Client uses to set up a rulerunner task involves associating the correct Task Profile to a Task Definition.

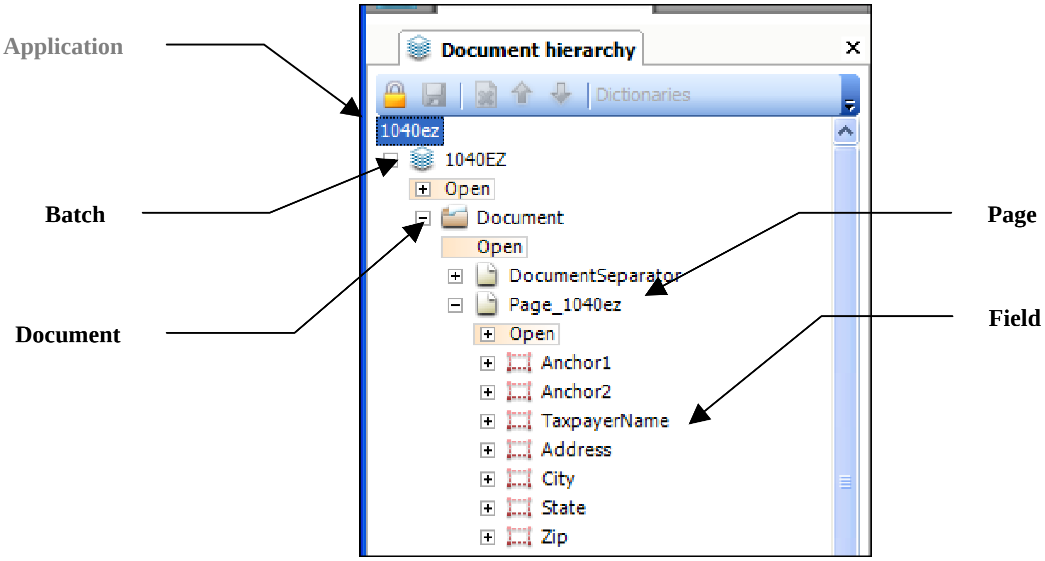
Rulerunner Service is the management entity in the middle. Rulerunner Service determines how information in the many elements of Datacap Studio that result in a Task Profile move to the correct Taskmaster Client task. This ensures that the task correctly processes a batch, its documents and pages, and the data the pages contain.

Document Hierarchies

A Document Hierarchy is the core component of the *Taskmaster Rulerunner Service* application itself. (Chapter 5 of the *Guide to Datacap Studio* provides detailed explanations of the Document Hierarchy.)

✓ The Document Hierarchy is an XML file that typically resides in the application's **Process** directory. By default, its structure and contents appear in this tab as soon as you open Datacap Studio.

Here is an opening view of the *1040EZ* application's Document Hierarchy (*1040.xml*).



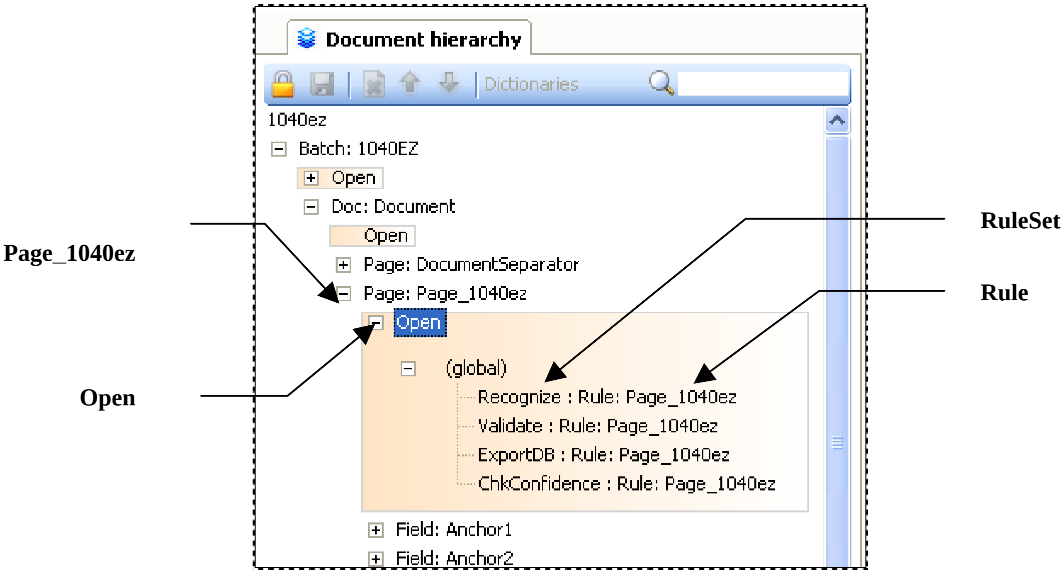
Survey DStudio - *Rulemanager Page*
Document Hierarchy tab

This illustration above lists objects at four levels:

- **Application**. ***Very important!*** This level is ***not*** a true member of the tree because it contains only an Application ID rather than an object name, and therefore cannot be the parent of objects at any other levels. Still, the **Application** does have important properties.

- **Batch**. The single object at this level represents the application's foremost processing entity: it is the parent of objects at every other level. The **Batch** object's name – *1040EZ,* in this case – is also the name of the Taskmaster workflow that creates and processes a batch and its contents. A batch can hold an object at any of the lower levels, including documents, pages and fields.
- **Document**. A typical application uses a *Rulerunner* task to organize a batch into a series of documents, and assign pages to each document. The **Document** object (*Document*, above) is the parent to one or more **Page** objects that represent the pages that will be part of each processed document. A document can also be the parent of field objects.
- **Page**. A **Page** object usually corresponds to a single document image. It may have different Page Types. Most fields exist at the **Page** level.
- **Field**. Data that has been recognized, manually entered, or created by rules is usually stored in **Field** objects. Although most commonly on the **Page** level, fields can be at the **Batch** or **Document** level. In addition, "sub-fields" can be stored in other fields.

✓ ***Very important!*** The boxed and shaded **Open** container holds Rulesets and rules tied to a particular object. Here is an example from the *1040EZ* application:



```
1040ez
⊟ Batch: 1040EZ
    ⊞ Open
    ⊟ Doc: Document
        Open
        ⊞ Page: DocumentSeparator
        ⊟ Page: Page_1040ez
            ⊟ Open
                ⊟ (global)
                    Recognize : Rule: Page_1040ez
                    Validate : Rule: Page_1040ez
                    ExportDB : Rule: Page_1040ez
                    ChkConfidence : Rule: Page_1040ez
            ⊞ Field: Anchor1
            ⊞ Field: Anchor2
```

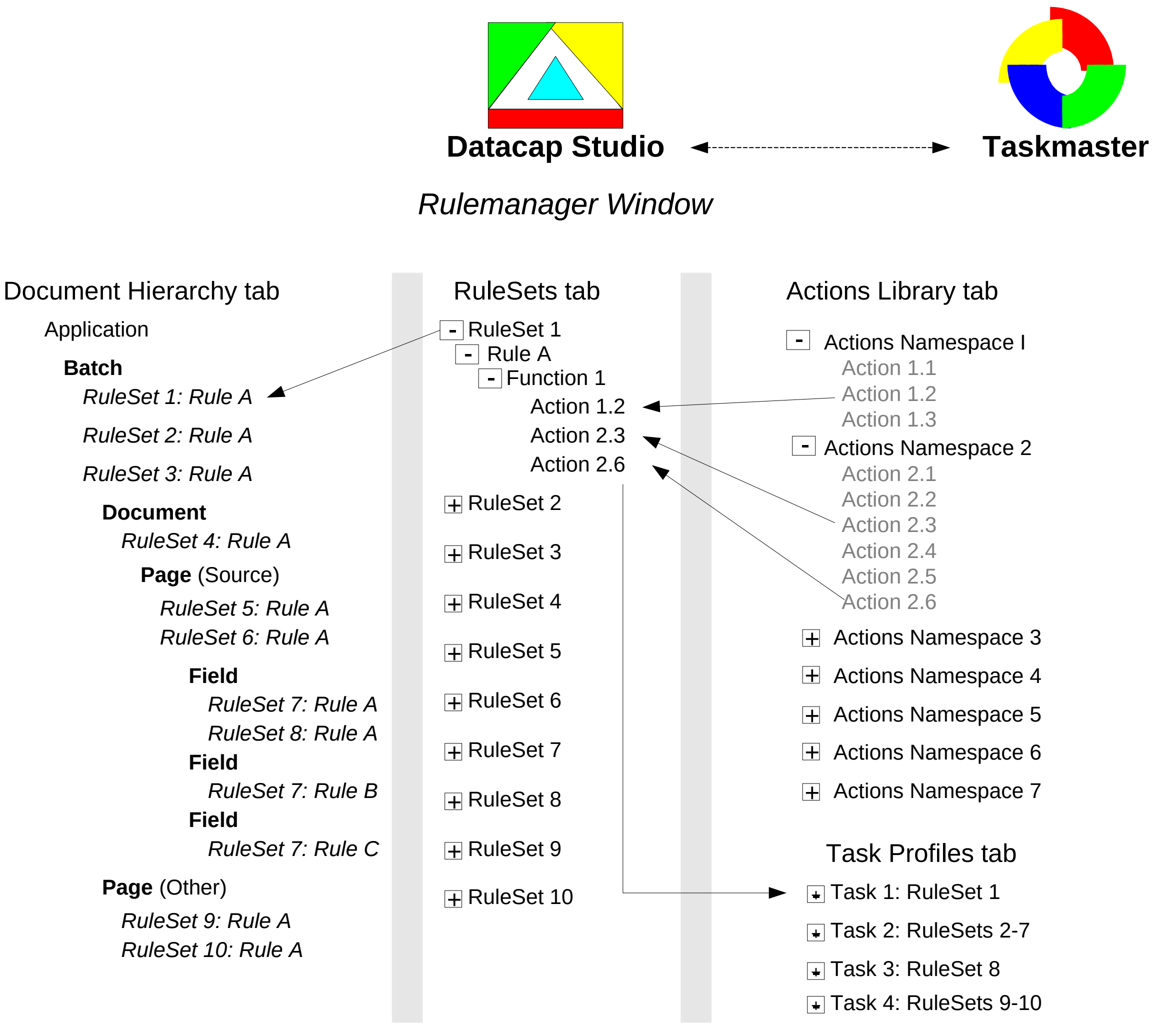
The **Ruleset/Rule combinations** illustrated above determine how a task processes the page and its data when the page is **Open** and available.

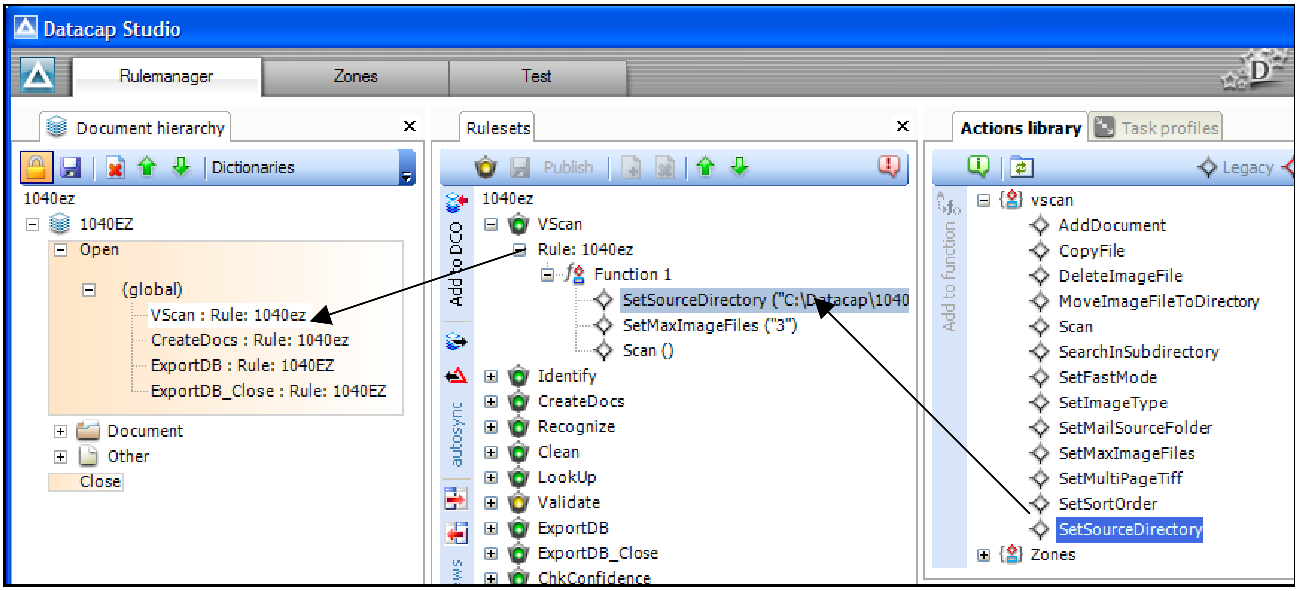
✓ This **Page** object also has a **Close** container. Any Ruleset/Rule combinations you place here will be carried out ***after*** all rules involving children of the page – its fields, in this case – have been carried out. Often, these rules close a database that was first opened by a Ruleset/Rule combination in the page's **Open** container.

Rulesets and Rules

You use tools in the tabs of Datacap Studio’s **Rulemanager Page** to:

- Assemble and modify your application’s **Document Hierarchy** – also known as its ***setup*** DCO.
- Define the **Rulesets** that will guide individual *Rulerunner* tasks.
- Define each **rule** that is part of a **Ruleset** – by assigning one or more **actions** to a function, and one or more **functions** to the **rule**.
- Bind a completed **Ruleset/Rule** combination to an object of the Document Hierarchy.
- Assign the Rulesets that contain these rules to *Taskmaster RRS* **Task Profiles**.





1040EZ DStudio: Rulemanager Page

Above, on the left, the *Document Hierarchy* tab displays data of two kinds:

- Identities of objects at each level of the Document Hierarchy: **Batch**, **Document**, **Page** and **Field**.
- Ruleset/Rule combination(s) assigned to an object of the Document Hierarchy – in this case, to the *1040EZ* **Batch** object.

The *Rulesets* tab in the middle of the example lists:

- Rulesets of the application
- Rules that belong to a Ruleset: **Rule:1040ez**, for example, is a member of the **VScan** Ruleset.
- Functions within a rule: **Function1** of **Rule:1040ez**,
- Action(s) that are the function’s code:

```
SetSourceDirectory(“C:\Datacap\1040EZ\images”)
SetMaximumFules(“3”)
Scan()
```

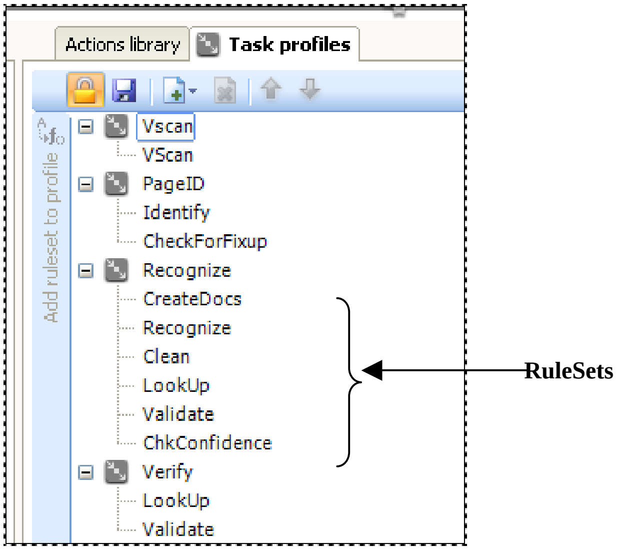
Tools of the *Rulesets* tab construct Rulesets and rules, and link them to objects of the Document Hierarchy.

The *Actions Library* tab on the right lists available Actions categories, and the actions that belong to each category. Tools of this tab assign actions to the function(s) of a rule.

The *Task Profiles* tab, which is behind the *Actions Library* tab, lists Rulesets that will be applied by a specific *rulerunnner* task when the task runs. You use this tab to assign Rulesets to a task after the task’s has set up by the ***Taskmaster Administrator*** .

Task Profiles

A **Task Profile** consists of Rulesets. When Rulerunner Service (RRS) is asked to run rules, it is told what Task Profile to run when carried out by a *Taskmaster RRS* task. An application's Task Profiles are listed in Datacap Studio's *Task Profiles* tab.



Rulemanager Page – *Task Profiles tab*

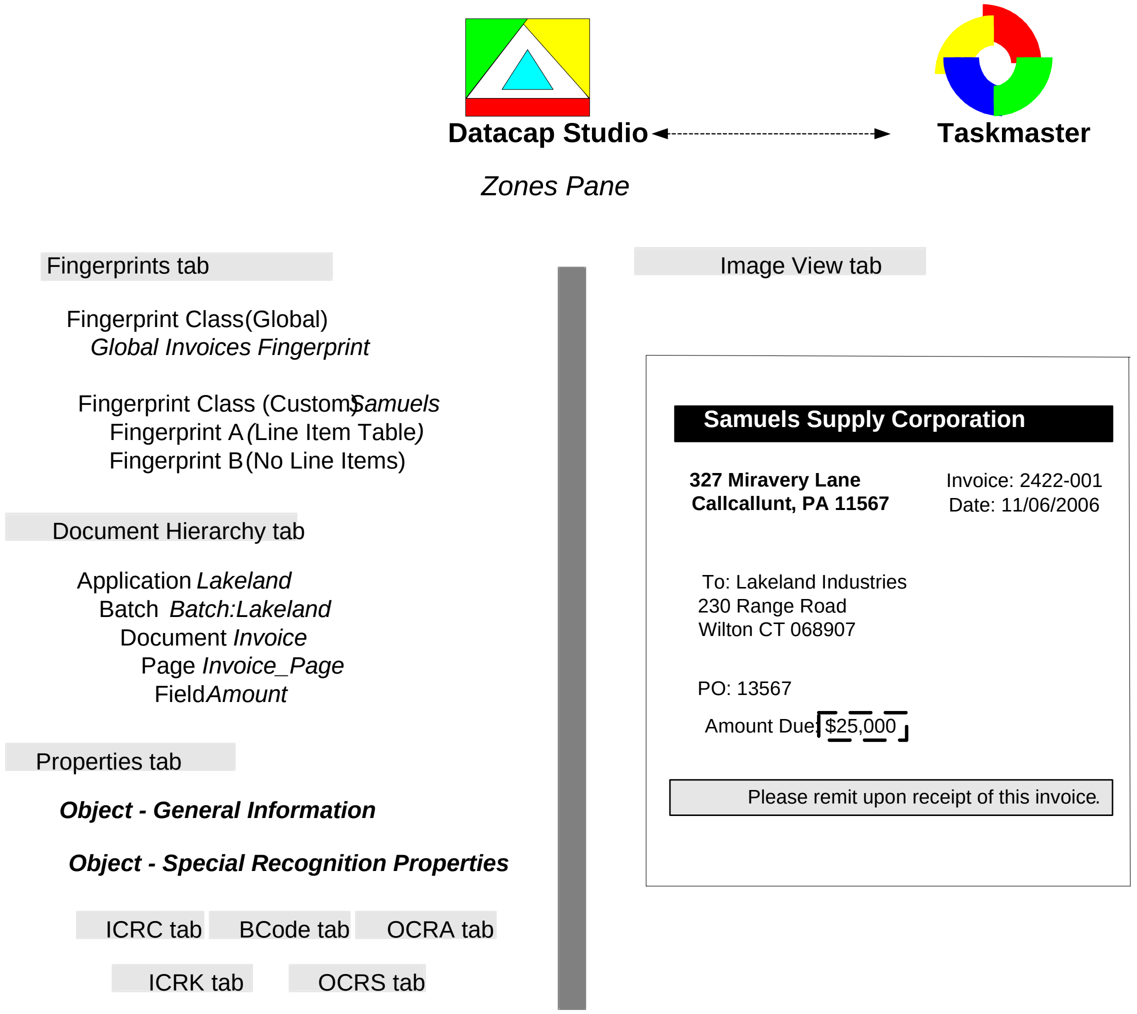
Above are the Rulesets that are part of four Task Profiles of the *1040EZ* application.

Important(1): When you assign a Ruleset to a Task Profile, you assign all its rules as well.

Important(2): The order in which the Rulesets appear in the Task Profile is the order in which the task will apply them when processing a batch and its contents.

Fingerprints

Datacap Studio’s **Zones Page** has this structure:



Fingerprint Class(Global)
Global Invoices Fingerprint

Fingerprint Class (Custom) *Samuels*
Fingerprint A *(*Line Item Table*)*
Fingerprint B (No Line Items)

Application *Lakeland*
Batch *Batch:Lakeland*
Document *Invoice*
Page *Invoice_Page*
Field *Amount*

Object - General Information

Object - Special Recognition Properties

Samuels Supply Corporation

327 Miravery Lane
Callcallunt, PA 11567

Invoice: 2422-001
Date: 11/06/2006

To: Lakeland Industries
230 Range Road
Wilton CT 068907

PO: 13567

Amount Due: $25,000

Please remit upon receipt of this invoice.

This basic illustration highlights the page’s principal tabs:

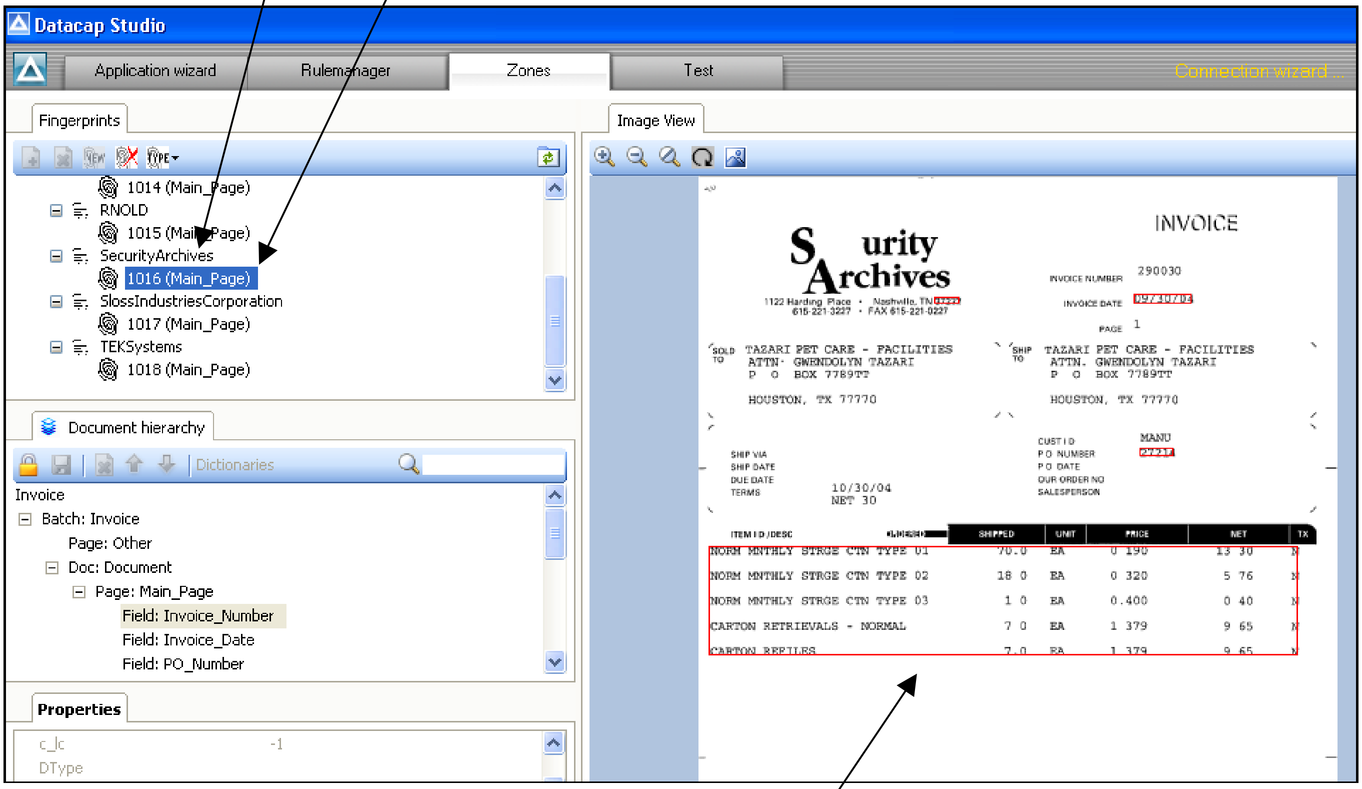
- On the left, the *Fingerprints* tab modifies existing fingerprints and sets up new Fingerprint Classes and their fingerprints.
- The *Document Hierarchy* tab gives access to the **Fields used by the application.** From here you can zone the fields on a fingerprint image – and assign recognition properties to individual **Field**s.
- The *Image View* tab displays the image of a fingerprint that you have selected from the *Fingerprints* tab. The image in this tab serves as the foundation for zoning procedures. The *Image View* tab also gives you a chance to enhance a new fingerprint’s image before you assign it to the fingerprint.

- ✓ The chart on the previous page is important from another perspective: it shows an invoice without a **Line Item Table**. As a result, the fingerprint is easier to set up, and recognition procedures are less complicated than those involving a fingerprint such as the actual fingerprint depicted below.

  This chapter describes how Datacap Studio's **Zones Page** sets up and works with fingerprints of both kinds.

Fingerprint Class

Fingerprint



Line Item Table

- ✓ ***Very important!*** Datacap Studio establishes the recognition parameters and procedures that *Rulerunner* tasks use to:
  - ♦ Identify the pages in a batch.
  - ♦ Locate fields and blocks of fields on each page.
  - ♦ Match the page to a pre-defined fingerprint.
  - ♦ Recognize each field's data.

Workflows

The *Workflow* tab of an application's ***Taskmaster Administrator*** manages the application's **Workflow Hierarchies**.

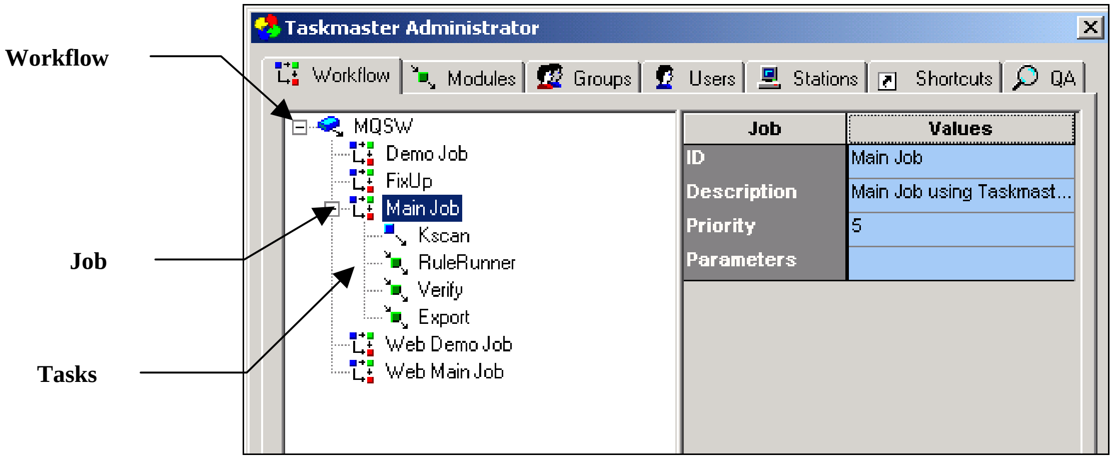
A Workflow Hierarchy has this structure:

Workflow

Job

Task

In the illustration below, *MQSW* identifies a **workflow**, and *Main Job* is one of five **jobs** that are part of the *MQSW* workflow. The *Main Job*, in turn, relies on four **tasks** to meet its processing objectives.



MQSW
- Demo Job
- FixUp
- Main Job
  - Kscan
  - RuleRunner
  - Verify
  - Export
- Web Demo Job
- Web Main Job

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

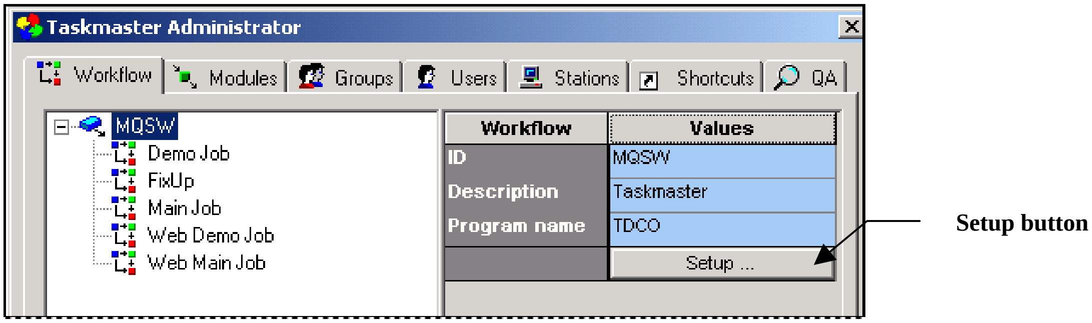
MQSW Taskmaster Administrator – *Workflow tab*

Workflows

A **workflow** component – *MQSW*, in this case - occupies the top level. If you highlight the Workflow ID, most of its properties and their values show up on the right-hand side (illustrated on the next page.)

The "TDCO" **Program Name** specifies that this hierarchy and its members refer to a Document Hierarchy for ***setup*** and ***runtime*** parameters.

Clicking on the Setup button accesses the application's Document Hierarchy – **MQSW.xml** (Page 6). ***Remember:*** This is a workflow's ***critical*** property.



Workflow	Values
ID	MQSW
Description	Taskmaster
Program name	TDCO
	Setup ...

Setup button

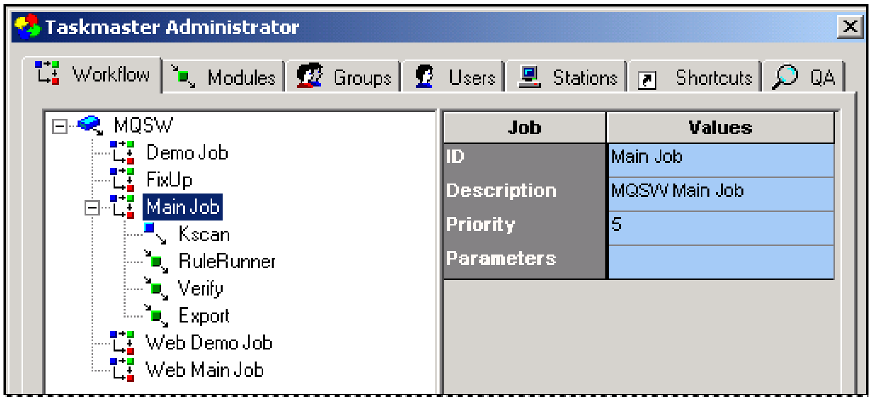
Taskmaster Administrator – *Workflow tab*

Jobs

A typical *Taskmaster* workflow has multiple jobs. Using the *MQSW* workflow as an example:

> *Main Job* is the application's workhorse. Its four tasks scan paper questionnaires; create images and add them to a processing batch; locate, recognize, verify and validate the data on each *Front* or *Back* ***source*** page; and export the data to a file or database.
>
> *Demo* scans images rather than paper, and is a powerful training tool.
>
> *FixUp* corrects problems identified by the Main Job's Rulerunner or Verify task.
>
> *Web Job* uses the Internet to scan and verify paper forms from remote locations.
>
> *Web Demo Job* uses the Internet to scan and verify images from remote locations.

When you highlight a Job ID on the tab's left side, values of its properties occupy the right side.



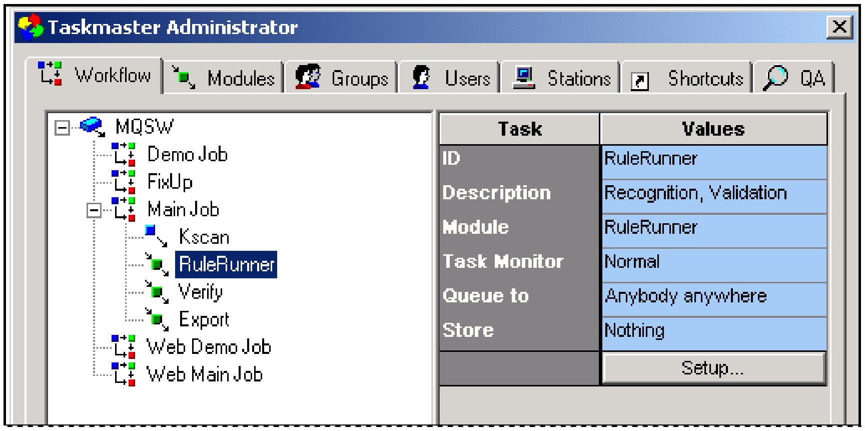
Job	Values
ID	Main Job
Description	MQSW Main Job
Priority	5
Parameters	

Properties of the MQSW Main Job

Tasks

A task is a set of highly focused procedures that help meet a job's overall objectives. Together, a task and the job to which it belongs form a **Job/Task Combination**: a Job/Task Combination is the ***principal operating entity*** of a *Taskmaster* application.

For example, a *Rulerunner* task is part of the *Main Job*; the resulting Job/Task Combination is referred to as `Main Job.Rulerunner`. When you highlight the task's ID on the left-hand side of the *Workflow* tab, the task's properties and their values appear on the right. Clicking on the Setup button accesses the task's ***Setup*** dialog.



Task	Values
ID	RuleRunner
Description	Recognition, Validation
Module	RuleRunner
Task Monitor	Normal
Queue to	Anybody anywhere
Store	Nothing

Properties of the Rulerunner Task

✓ Chapter 5 describes workflow components and shows you how to set up workflows, jobs and tasks.