



**TaskAide  
Administration Console**

# Table of Contents

<a href="#"><u>Introduction</u></a>	3
<a href="#"><u>Accessing the Administration Console</u></a>	4
<a href="#"><u>Navigation Menu</u></a>	5
<a href="#"><u>Working with Users</u></a>	6
<a href="#"><u>Working with Custom User Properties</u></a>	7
<a href="#"><u>Working with Roles</u></a>	8
<a href="#"><u>Working with Flow Definitions</u></a>	9
<a href="#"><u>Using the Task Editor</u></a>	11
<a href="#"><u>Working With Flow Instances</u></a>	19
<a href="#"><u>E-Mail Notifications</u></a>	21
<a href="#"><u>Scripting Reference</u></a>	23
<a href="#"><u>Scripting Examples</u></a>	32

## Introduction

TaskAide is a software package offered by LaserVault for use with our LaserVault Imaging system. LVI must be installed for TaskAide to function. TaskAide expands the functionality of the LVI system by allowing you to route your documents through a business process in a digital format. There are many benefits to using TaskAide, most notably, the ability to track documents as they are worked through the business process. TaskAide can also allow you to examine and improve the efficiency of your business processes.

The TaskAide system is driven by workflows. In TaskAide, a workflow is made up of a series of tasks linked by transitions. Tasks can be scripted or manual. Manual tasks can include both data input fields and decision options. A scripted task uses VBScript to evaluate data and determine which task to move to next.

Each workflow and task allows the ability to send email notifications when various events occur. This allows you to notify either the assigned user that there is new work to be completed, or any other user or role that you wish to send a notification at any part of the workflow.

## Accessing the Administration Console

To access the TaskAide Admin application, browse to the URL representing the virtual directory that was setup for TaskAide at the time of installation.

For Example:

`http://[servername]/TaskAide/TaskAideAdmin.asp`

Before the Admin Console loads, a small version-checking, ActiveX control will need to load and check the version of the Admin Console files on your pc. In order for the version checker to load, you must either add your TaskAide site as a "trusted site" in your browser configuration, or allow the operation of ActiveX controls in your browser configuration. If you are unsure of the proper configuration, in regards to security, feel free to contact our Technical Support department for the best solution, as these settings are dependant on your internet connectivity and network configuration.

Once the Admin Console has loaded, you will be presented with a login screen. Enter the user name and password for the admin user you created when installing task aide.

# Navigation Menu

You will find a navigation menu on the left side of the screen after logging in.

The following functions can be accessed from the navigation menu:

## **TaskAide System Statistics:**

Displays a quick summary of TaskAide system information.

## **Work With Users:**

Displays options for adding and managing user accounts in TaskAide.

## **Work With User Properties:**

Displays options for managing custom user property fields.

## **Work With Roles:**

Displays Options for adding and managing roles in TaskAide. A Role in TaskAide is a group of one or more users that perform a certain function. Examples of roles might be Ap Clerk, Controller, Department Manager etc.

## **Work With Flow Definitions:**

Flow definitions are the heart of TaskAide this is where you define how a document is routed or flows through your various business processes. From this menu you can define new and edit existing flows.

## **Work With Flow Instances:**

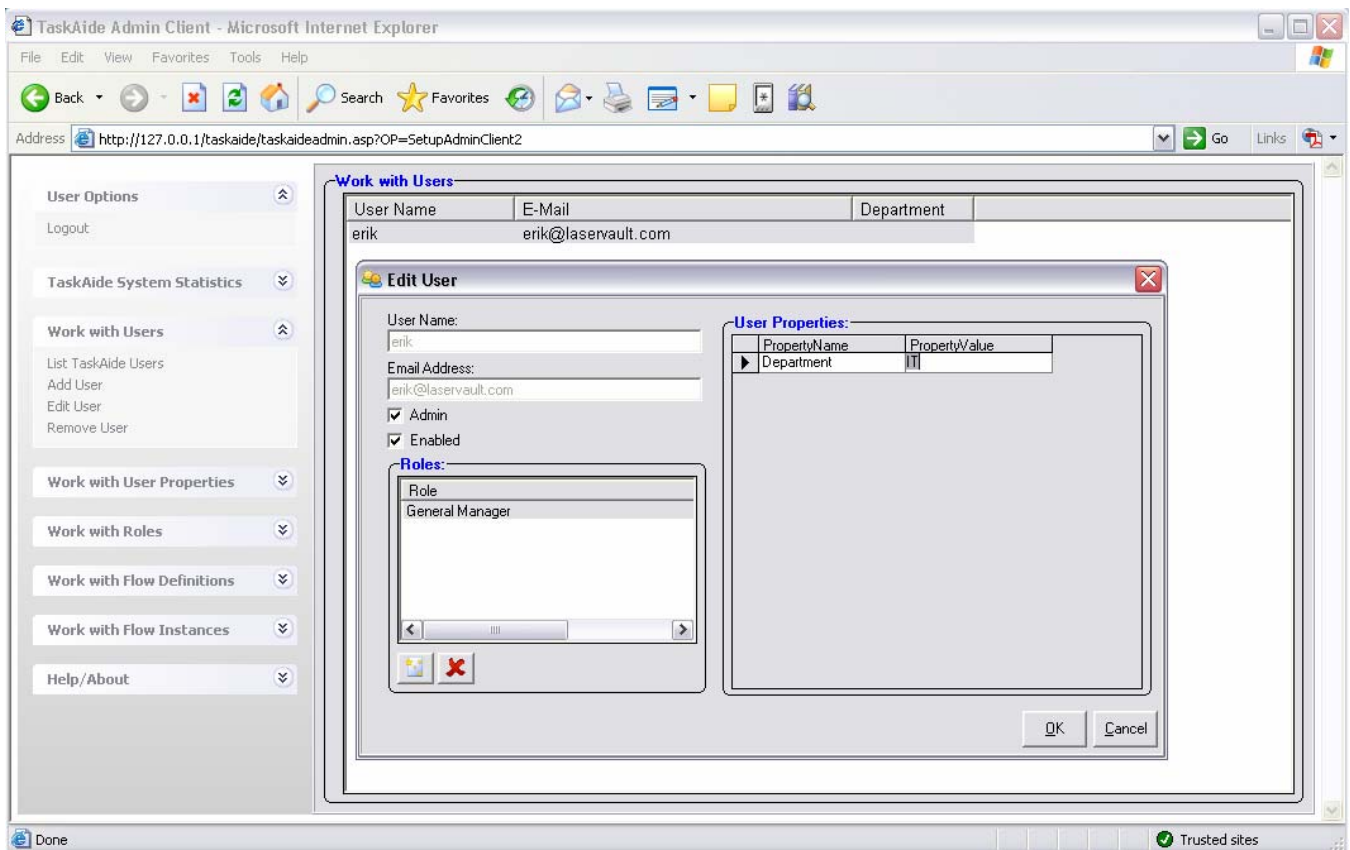
Flow instances represent the actual items going through the work flow system. Where you would define a work flow to process invoices, each invoice going through the system would be represented by a flow instance. From this menu you can search for flow instances (documents) and see the status or track the path the document traveled through the flow.

# Working with Users

The Work with Users screen allows you to manipulate the user accounts. It is important to note that TaskAide itself does not store the user accounts, but TaskAide rather uses the account information from the LaserVault Imaging system. TaskAide stores some additional information for its own use, but the actual user account resides in the LVI system.

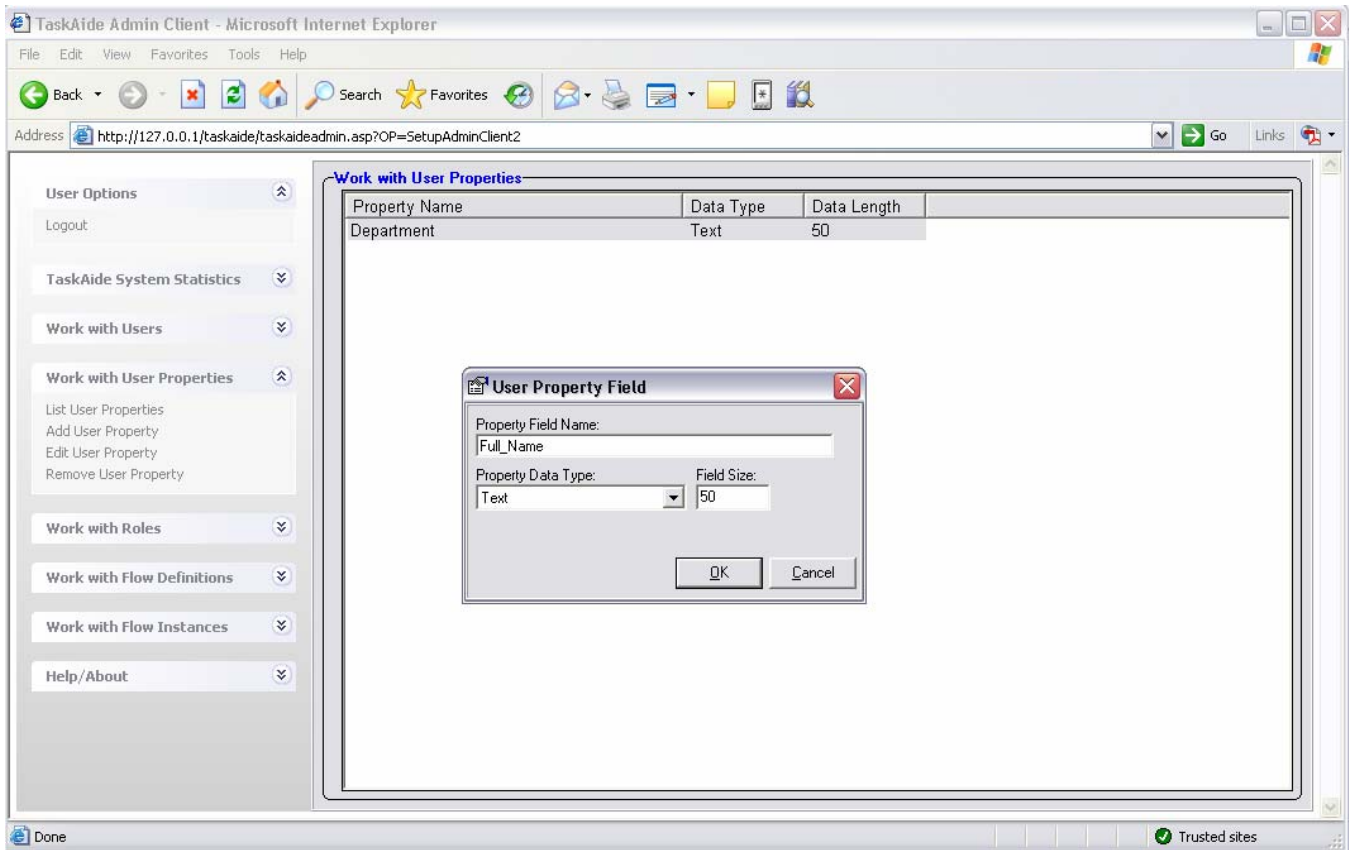
Click the List TaskAide Users link to view all users defined in TaskAide

To add a user to TaskAide, click the Add User link. This will present a screen from which you can select a user account. The user account must have been previously defined in the LaserVault Imaging System. Click the Edit User link to edit the properties of a user account. From this screen you can set the admin and enabled options, define what roles a user is a member of, and enter any custom use property values. If you have defined custom user properties, you can edit the values by typing in the "PropertyValue" column.



# Working with Custom User Properties

The user properties screen allows you to define custom user properties. This is basically adding fields to the users table in the TaskAide database. When adding a user property, enter the field name, data type and length for text fields. Once a user property has been added, you can edit the user accounts and enter a value for that property.



The user properties can be used later in scripted tasks doing such things as assigning work based on department etc.

# Working with Roles

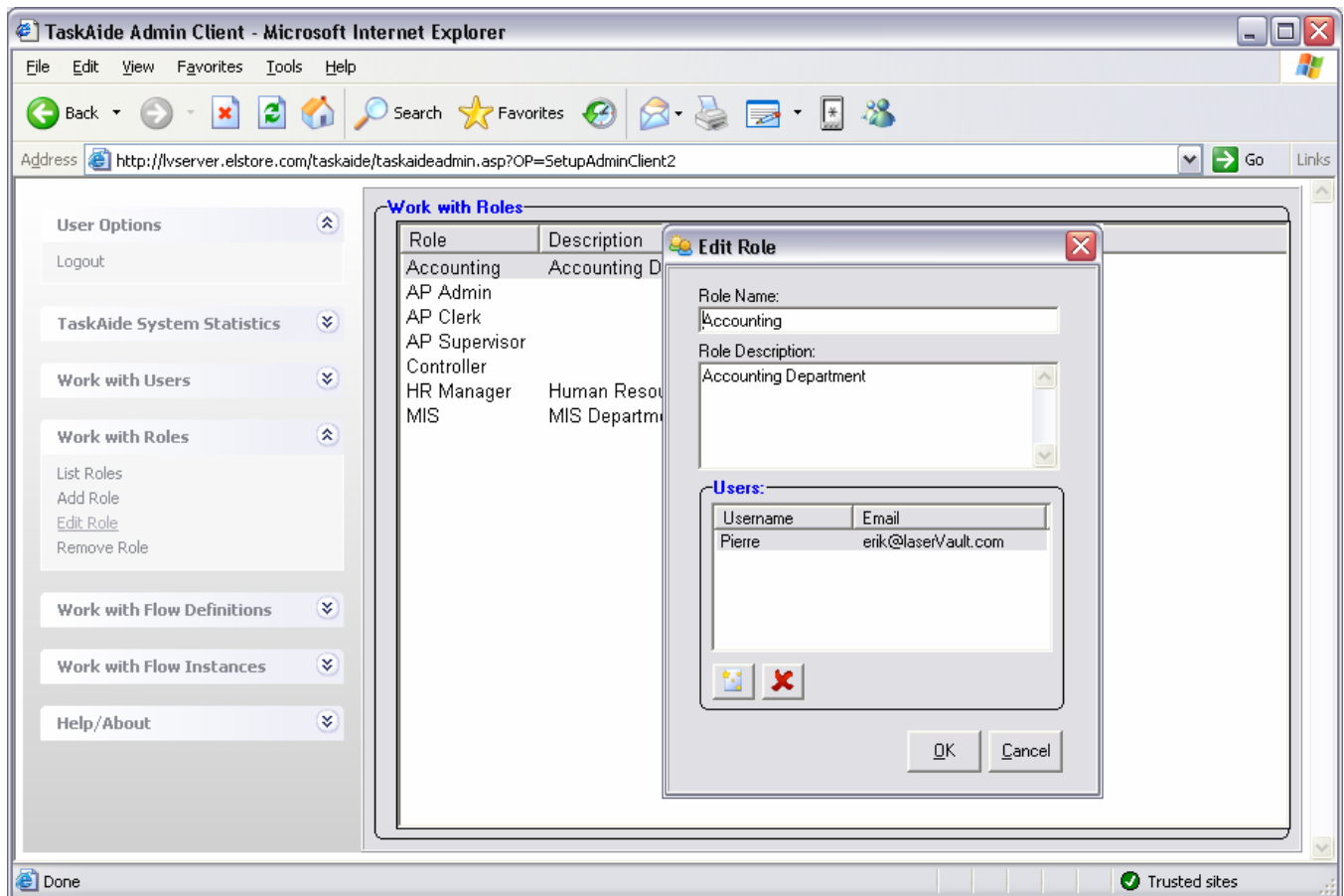
A Role in TaskAide is simply a grouping of user accounts. You may group these users in any logical fashion that fits your needs, and users may be members of unlimited Roles.

Roles are used to define which users perform which functions in TaskAide. Examples of roles you might set up would be AP Clerks, Controllers, Accounting, Managers, Department Manager.

Users who are part of the AP Clerks role might do data entry for invoices. Controllers might do approvals or denials for purchase orders etc.

Later in the documentation you will see how certain tasks can be assigned to certain roles.

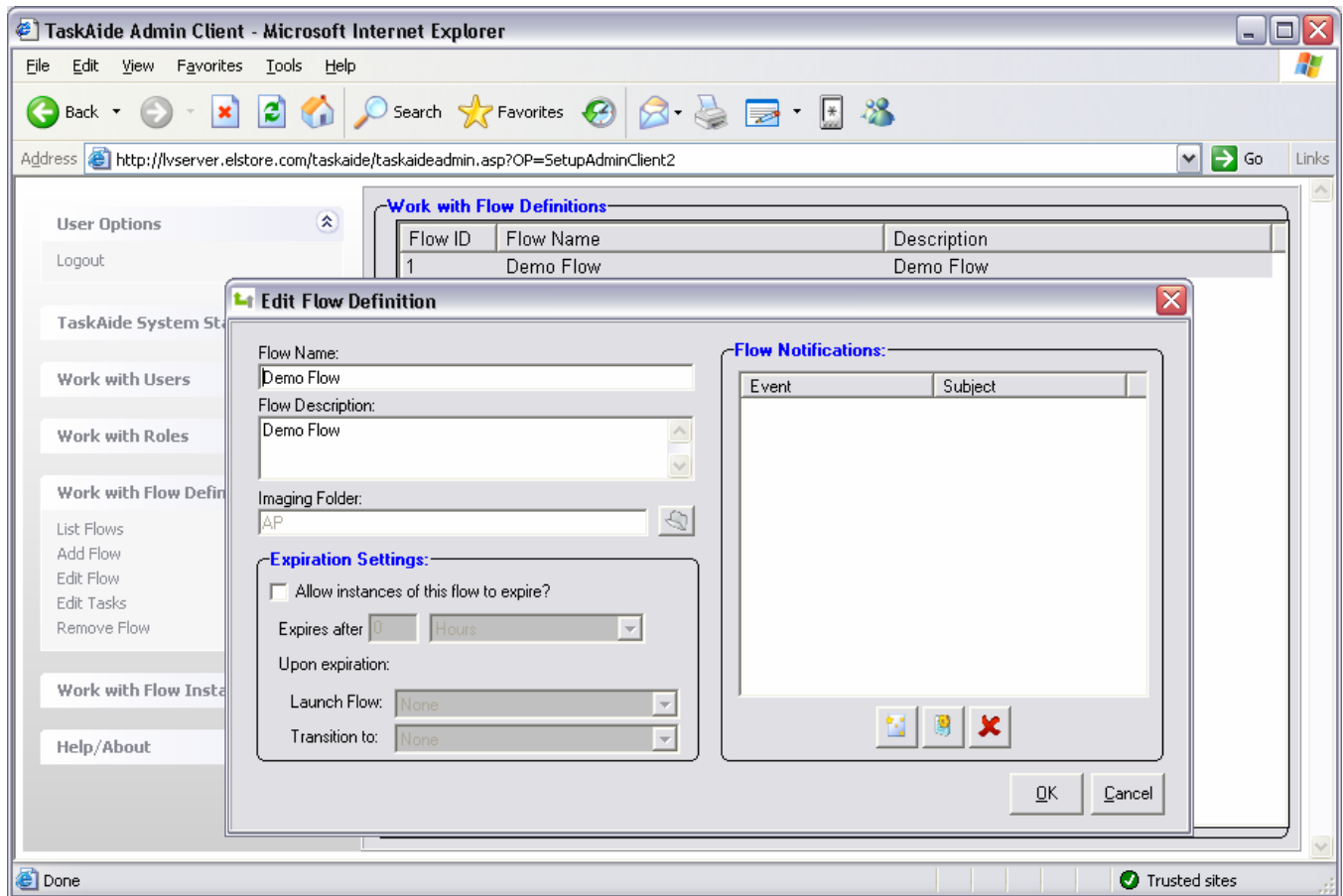
The Work with Roles screen allows you to add, edit, and remove the roles that you have defined in your TaskAide system. You may also assign users to the roles from this interface.



# Working with Flow Definitions

Defining a flow is done in two parts. First the basic properties of the flow are defined such as the flow name, which imaging folder the flow will be associated with, notification events etc.

The second part of defining a flow is to define the tasks or steps that each flow must go through. This will be covered in more depth later under the “Task Editor” section.



A few of the important pieces of information when setting up the flow definition:

## Imaging Folder:

Note that a flow definition must be associated with a specific Imaging Folder. Because of the level of integration between TaskAide and LaserVault Imaging, the custom field definitions play an important part in flow configuration and require the association of a flow definition with one LaserVault Imaging Folder.

## Expiration Settings:

These settings allow the flow to be defined with an expiration timer, and various actions taken if the flow goes beyond the designated time frame. You have options to launch a new flow when this one expires, or transition to a specified task within this flow.

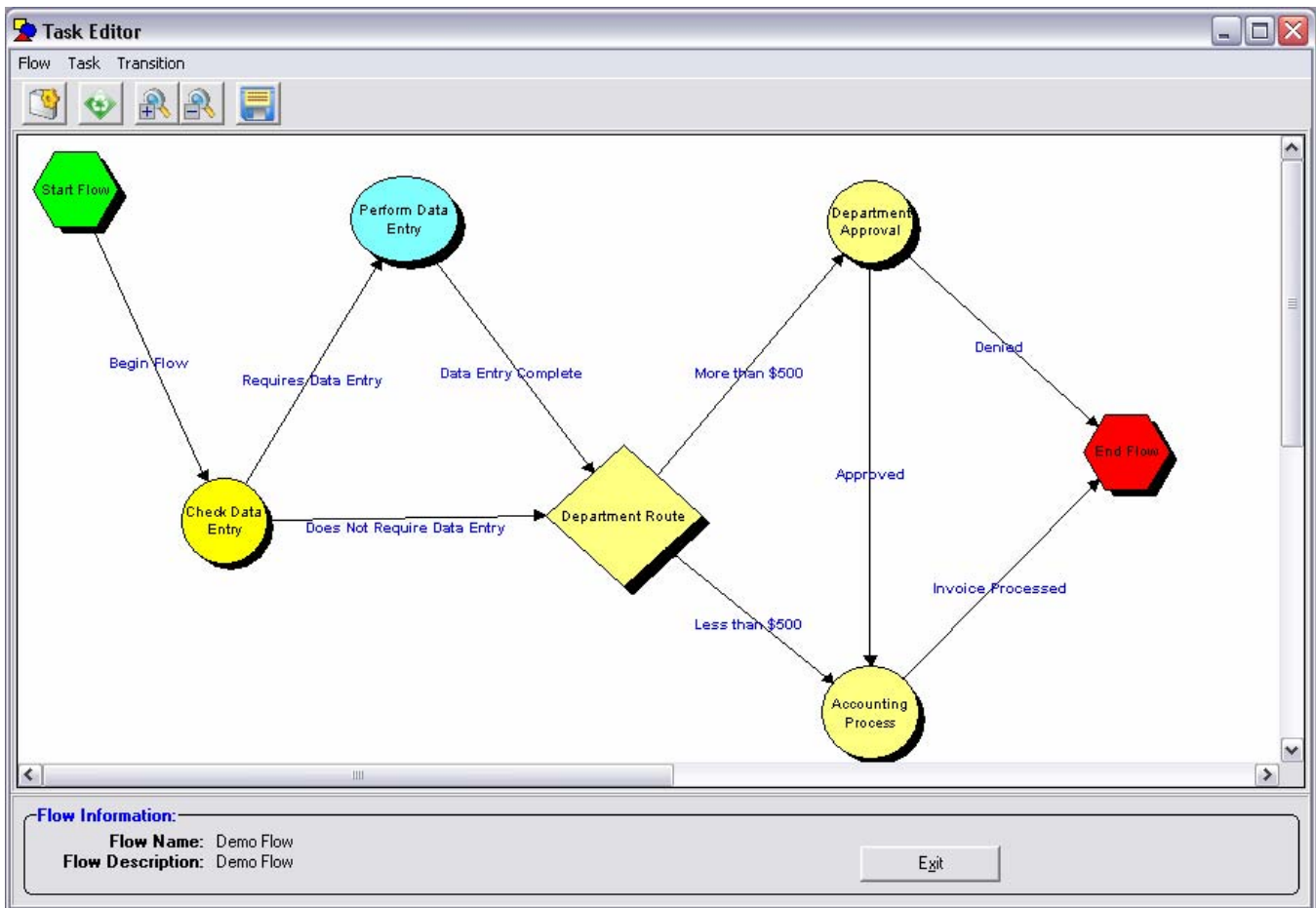
## Flow Notifications:

This functionality allows for email notifications to be sent from the TaskAide system upon various events along the flow's lifecycle. This is covered in more detail in the Notifications section.

After creating a work flow definition, you will need to open the LaserVault Imaging Management Console and configure a launch event. This will define how a flow is launched for a particular document or imaging record. See the LaserVault Imaging Management Console documentation for help on how to set up the launch.

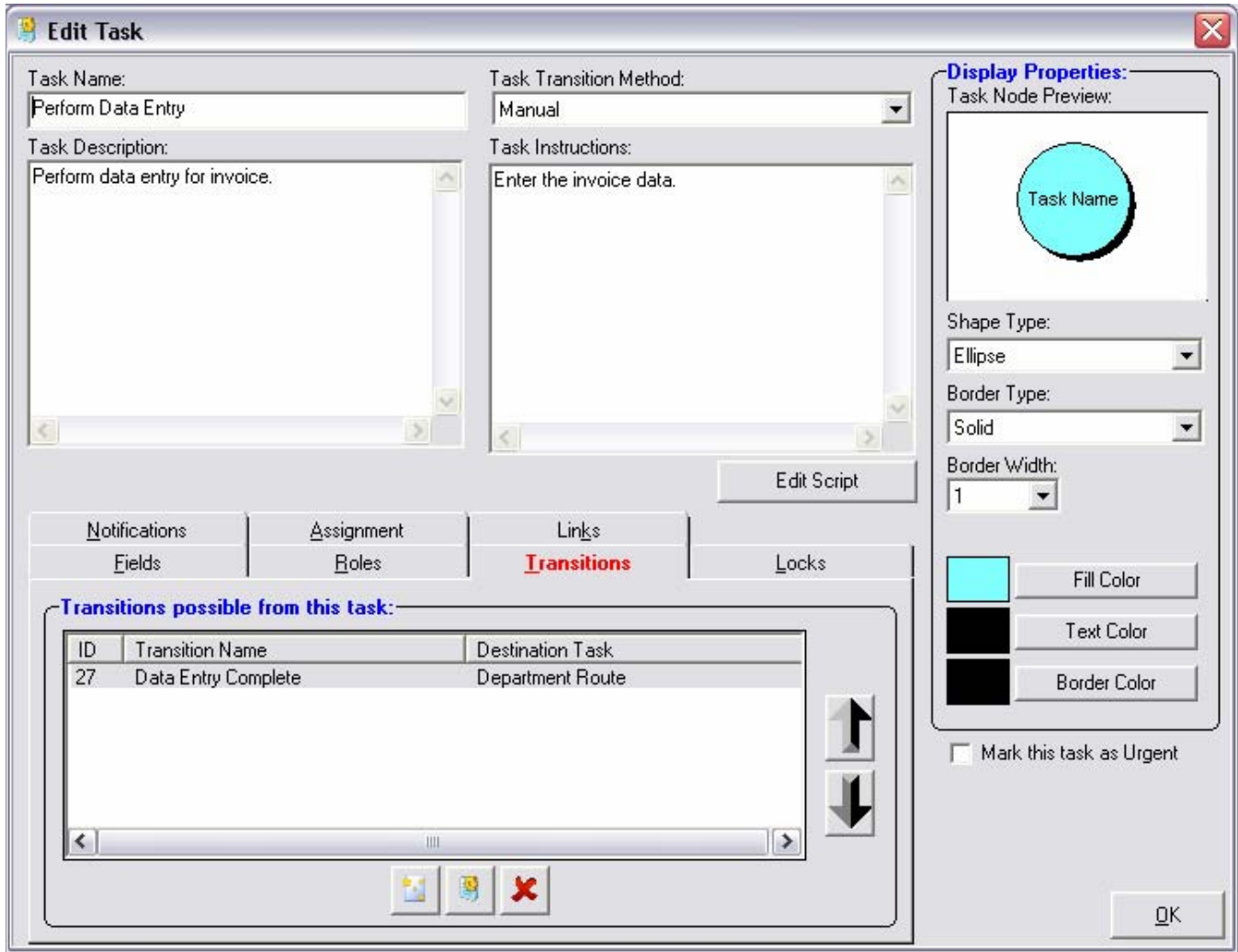
# Using the Task Editor

The task editor is used to define the individual steps or tasks that must be performed in a work flow. Tasks might include, data entry, routing based on amounts, approving or denying etc.



When a new flow is started the task editor will only contain 2 tasks – Start Flow and End Flow. These are the system defined tasks that mark the beginning and ending of the flow.

To add a new task to the flow, either select the “Add Task” option from the task menu or right-click on an empty area in the flow layout and select “Add Task”. You will be presented with a Task Properties screen that allows you to set various attributes for this task.



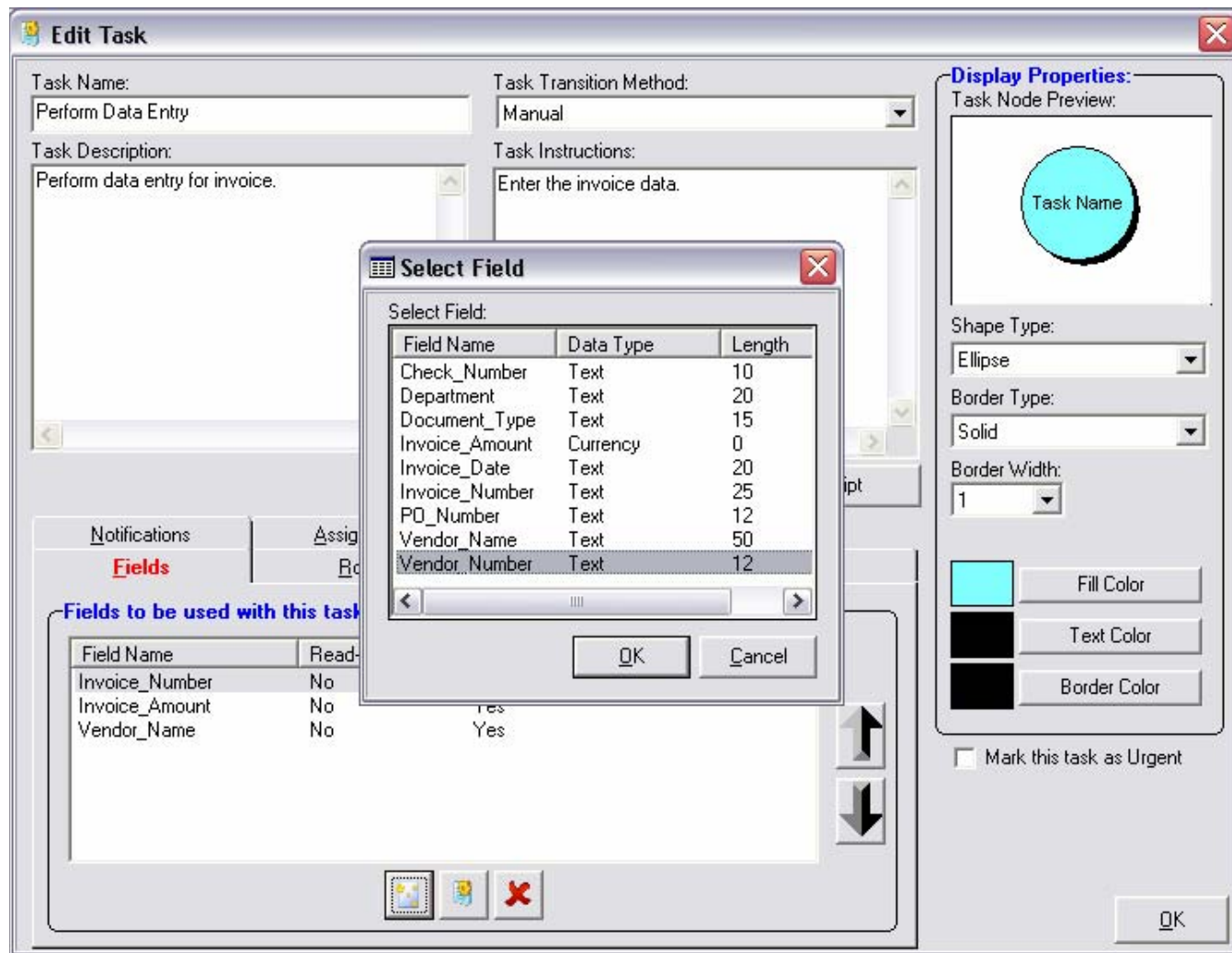
There are 2 types of tasks that can be defined – Manual and Scripted tasks.

A manual task is one that requires end user input such as data entry or any other task that cannot be accomplished automatically through scripting.

A scripted task is one that uses VBScript to perform actions and move on to the next task in the flow. Scripted tasks will be covered in more depth later.

The following sections can be defined for manual tasks

### Fields:



For a manual task you can select fields defined in the imaging system for the end user to view or enter data into. Click the add button to bring up a list of available fields (fields that have been defined for the associated folder in the imaging system).

After selecting a field you will see a dialog where you can choose options for the field.

**Add Field**

**Display Settings:**

Display Name: Vendor Number

Display Size: 0

**Imaging Field Info:**

Field Name: Vendor\_Number

Field Type: Text

Allowed Values:

Field Length: 12

Lock Allowed Values: No

**Input Options:**

Field is Read Only

Field is Required

Line Item Entry

Line Item Rows: 0

Line Item Cols: 0

**Select List:**

Display select list with this field?

Connect String:

Select String:

OK Cancel

In the field properties dialog you can choose whether a field is read only (display only) or required for data input. You can give a field a display name. The display size applies to input fields. It determines how wide the input text box is for text, and numeric fields.

The line item entry options can be used to display multiple input fields for a single imaging field. The imaging field must be defined as a memo field. Each input field will be concatenated together and stored in the memo field separated by carriage return/line feed pairs. This can be used where you need multiple inputs for a single record such as accounting GL Codes that apply to an invoice.

The select list option allows you to pull field values from a database to use as choices for data input. The connect string is used to connect to the database (on ODBC connection string). The select list is a query that pulls the fields to populate the drop down list. Since databases vary widely from customer to customer contact tech support for assistance with this section if needed.

**Roles:**

This list defines which roles have access to work this task. For example if the task is “Data Entry” the AP Clerk role might be assigned to this task.

**Locks:**

When a task is assigned to a user or the user checks the task out to work it, the task is considered locked (only that user can access it). On the locks screen you can set expiration options. If the task is not completed with in a certain time frame you can check the task back in for someone else to pickup, or take a transition to another task.

You can set a lock expiration in increments of hours, days, weeks, or months.

There is also an option to not allow a user to check a task back in. With this option enabled, when a task is assigned to a user or they pick one from the list and check it out, they cannot check the task back in.

**Notifications:**

This functionality allows for email notifications to be sent from the TaskAide system upon various events pertaining to this task. This is covered in more detail in the Notifications section.

**Task Assignment:**

Task assignment can used to assign a task to a specific user account within a role. If the task is not assigned to a specific individual then any member of the role can check the task out and work it. Tasks can be assigned to users based on the following methods:

Don't assign – users must check the task out from their list.

Assign to the user with the least work load for this task.

Assign to the user with the least work load for this flow.

Assign to the user with the least work over all.

Assign to users in a round robin method.

Assign to users randomly.

Assign to users using a script. This will be covered in more depth under the scripting section.

**Links:**

Under the links section you can define http or web links to be displayed on the data entry screen for manual tasks. To add a link, press the add button and enter the link title and URL.

**Transitions:**

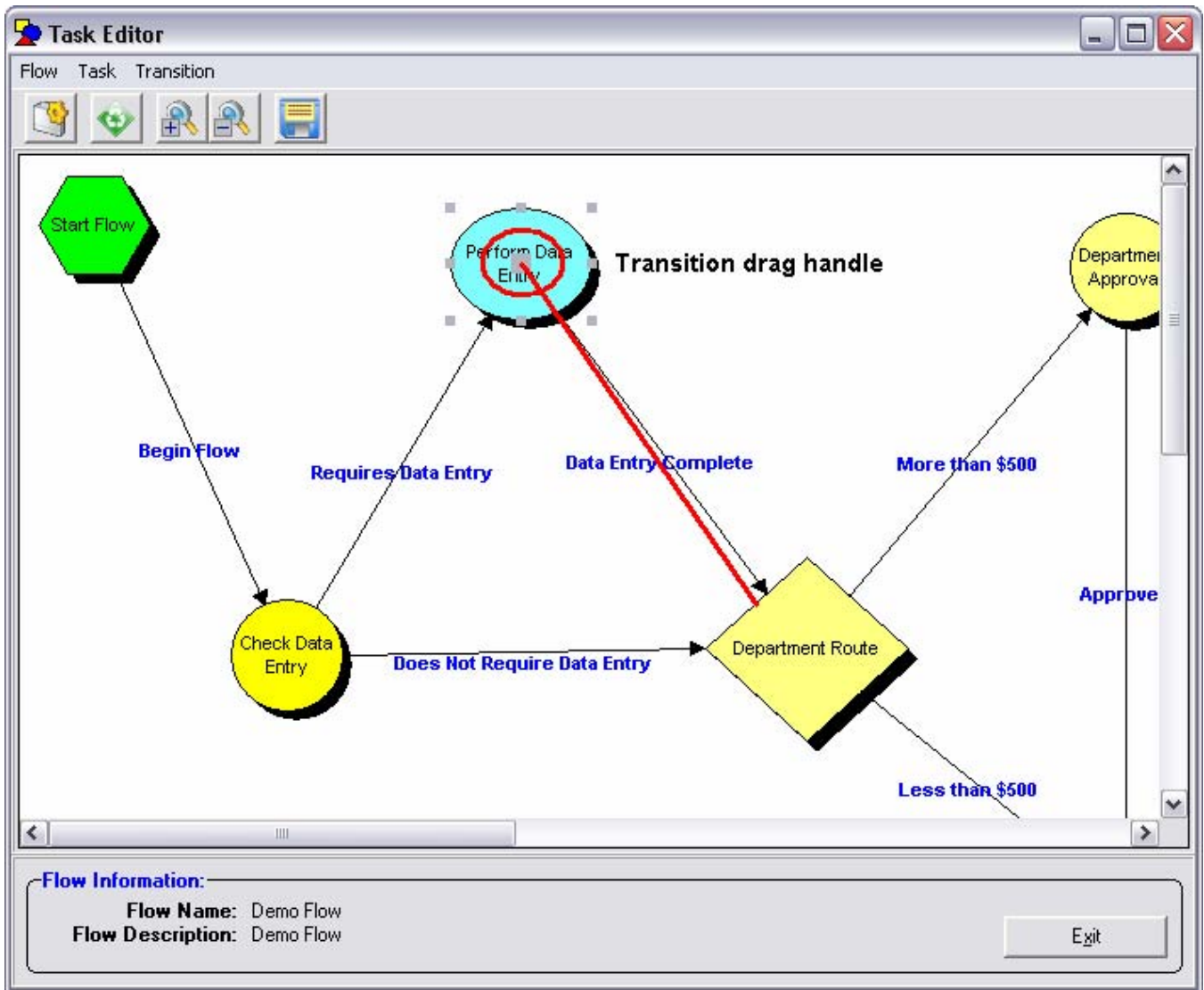
Transitions are the defined paths to get from one task to another. For manual tasks transitions show up in the TaskAide client as option that the end user can take to complete the task.

Examples of transition options might be “Approve” or “Deny” for a purchase order.

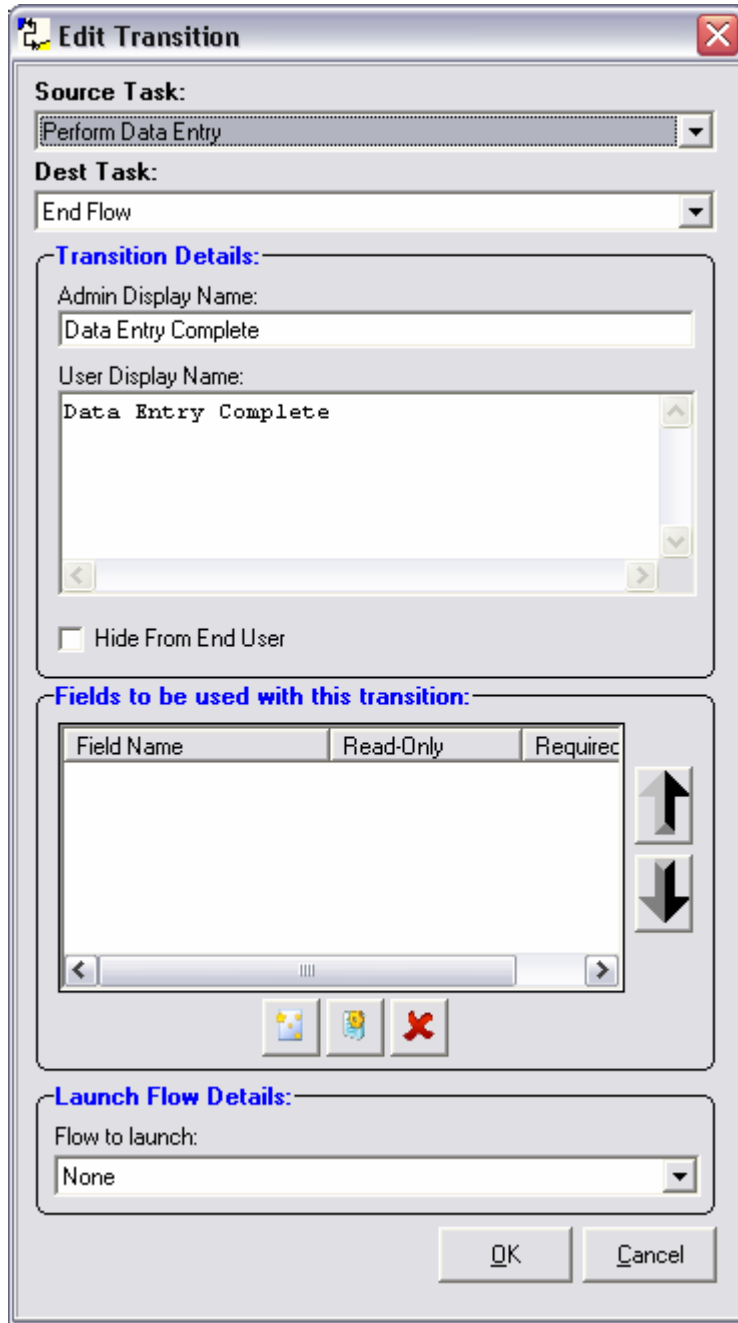
You can control what order the transitions are displayed to the end user in by using the up and down arrow buttons to move the transition.

Transitions are also used in scripted tasks. A scripted task could be defined to look at an invoice amount and then based on the amount route it for approval if it's over a certain amount, or route it for payment if it's under a certain amount. See the scripting section for more information on scripted tasks and transitions.

You can add a transition here in the task properties, or from the task editor you can add a transition by drawing a line between 2 tasks. To draw a transition between 2 tasks, click the center of the task icon and drag the mouse to the destination task. The mouse icon will change when you are over the “transition drag” handle (the square in the center of the task shape)



The transition properties dialog will be displayed.



The Admin Display Name is shown in the task editor. The User Display Name is shown in the task aide client for manual tasks.

The “Hide From End User” option can be used for timeout transitions. When you set up a timeout on the task and want it to take a transition you can hide this transition from the end user so it’s not available as a choice for them to select.

Data entry or display fields can also be defined at the transition level. When fields are defined here they apply to this transition. The purpose of this is for cases where data entry is required based on which option is selected. An example would be a case where a person can Approve or Deny a purchase order, you could have a "Reason Denied" field that the person must enter if they choose the "Deny" option.

Finally you have the option to start a whole new flow when a certain transition is taken.

## Working with Flow Instances

While a Flow Definition is the path that a particular document will take through a predefined process, a Flow Instance is one unique document traveling that path. So a Flow Instance is the record of a particular document going through a Flow Definition. For example you might create a flow definition to route purchase orders. An individual purchase order traveling through that flow would be represented by a flow instance.

An interface is provided to search the documents that have already gone through the system or are currently traveling through the system.

The screenshot shows a dialog box titled "Search Flow Instances". It features a search icon and a close button in the title bar. The main content area is divided into several sections:

- Search:** A "Flow:" dropdown menu.
- Flow Instance:** A group of five radio buttons: "Currently Active Flow Instances" (selected), "Errored Flow Instances", "Completed Flow Instances", "Pending Flow Instances", and "Urgent Flow Instances".
- Date Field:** A dropdown menu currently showing "\*None".
- Start Date:** and **End Date:** text input fields.
- Imaging Fields:** Three rows, each consisting of a dropdown menu and a text input field.

At the bottom of the dialog are "OK" and "Cancel" buttons.

To execute the search, select the flow definition you wish to search and which category of Flow Instances that you wish to search.

Active flow instances are flows that are not yet complete (have not reached their end point).

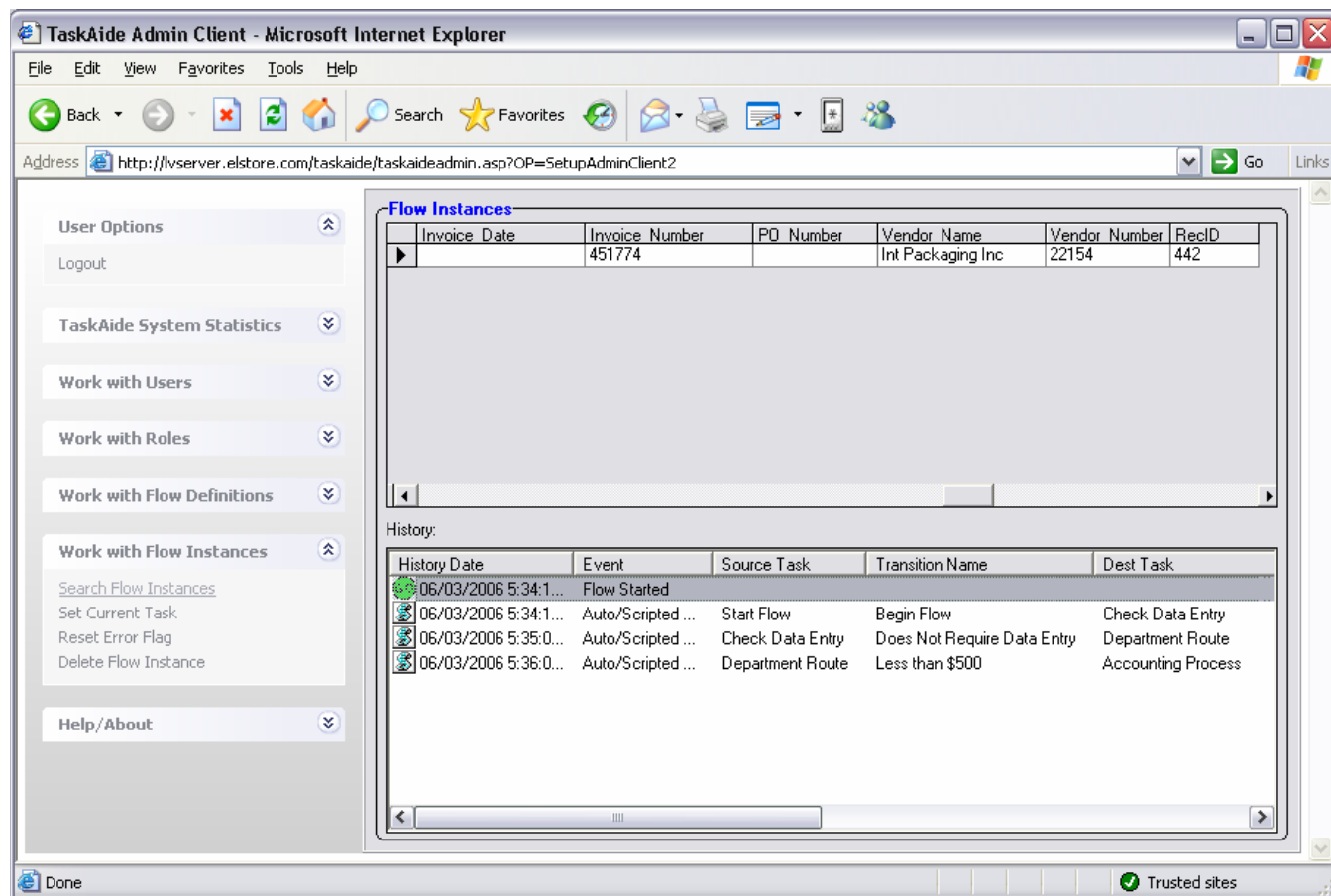
Errored flows instances are flows where an error has occurred.

Completed flow instances are flows that have gone all the way through the flow to completion.

Pending and Urgent flow instances are flows that have had their status set to pending or urgent.

To do a more specific search, use the date field to narrow by date range, and use the imaging fields to narrow by field values in the imaging system. For example invoice number, po number etc.

The results for each flow instance found will be displayed in the top pane of the window. Clicking on a flow instance in the top will display the history for that flow. (What steps or tasks it's gone through in the system.)



A flow instance can be deleted from the system by selecting it in the top list, then selecting the "Delete Flow Instance" option on the left menu.

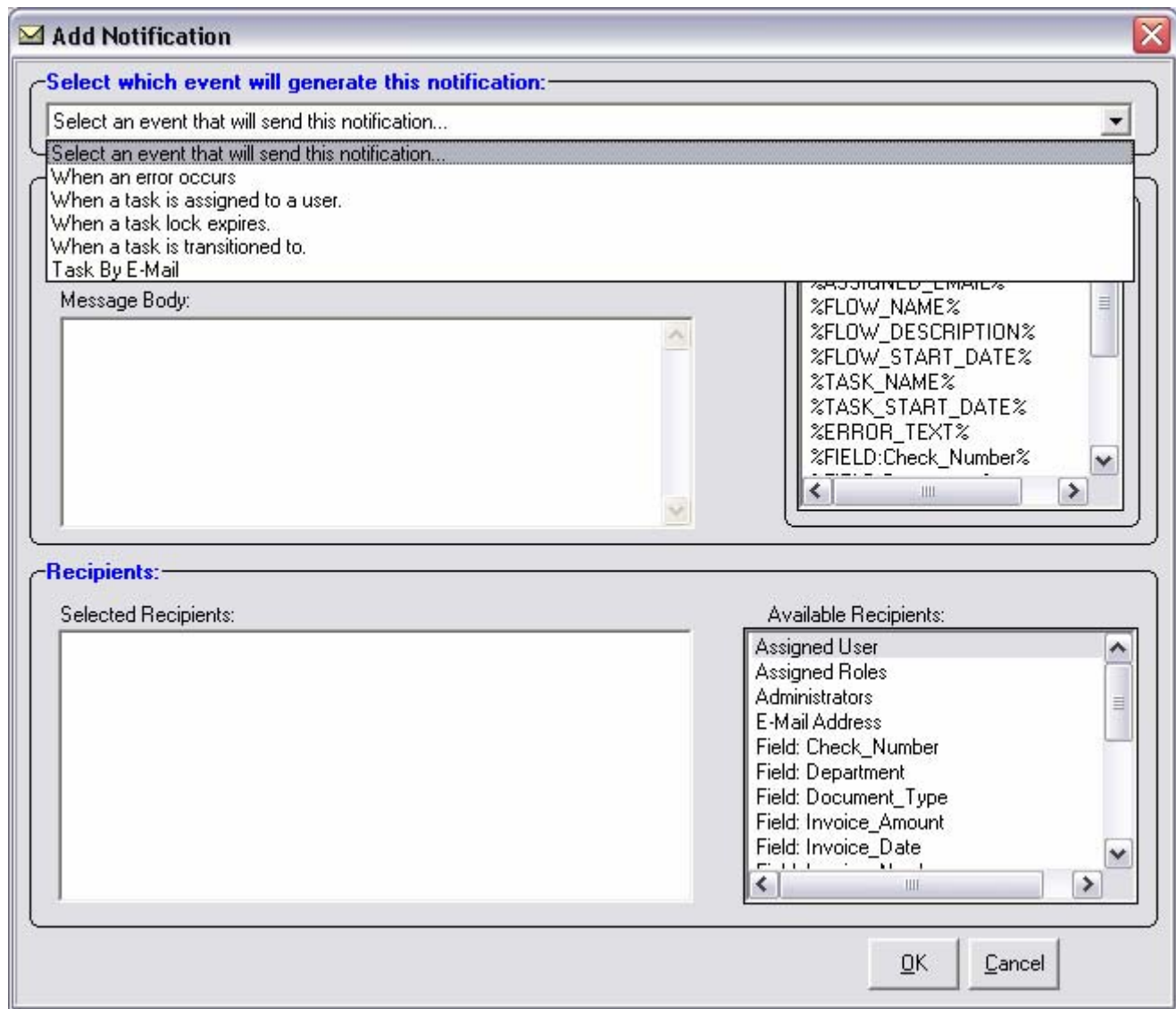
You can also reset the error flag or manually set the flow to a specific task in the work flow.

To see details of the history entry, double click an entry in the bottom list.

# E-Mail Notifications

The TaskAide system can be configured to send e-mail notifications based on certain events in the system. Notifications can also be sent with links in the e-mail body to choose a certain transition such as approve or deny.

Notifications can be defined at the flow definition level or at the individual task level.



E-mail notifications can be sent when the following events occur.

For tasks:

- When an error occurs.
- When a task is assigned to a user.
- When a task lock expires.
- When a task is transitioned to.

Task by E-Mail – This is a special type of notification that will send an e-mail to the assigned user and give them links to select a transition directly from their e-mail if there is no data entry required (no fields defined for data entry).

For Flow Definitions the following notifications can be sent:

When a flow expires.

When a flow is started.

When a flow is completed.

When a task is transitioned to.

When a task is assigned to a user.

When a task lock expires.

When a task is transitioned to.

The subject and body of the e-mail can contain replaceable values. Just drag and drop the values from the top right on to the subject or message body.

For recipients you can select from the options on the lower right and send e-mails to:

The assigned user.

Each member of the assigned role.

Administrators.

A specific e-mail address, or you can use the value from an imaging field as the e-mail address. (The imaging field must contain the e-mail address).

The Task By E-Mail is a special type of notification and does not allow recipients to be selected. It is automatically sent to the assigned user for the task.

# Scripting Reference

The scripting engine used in TaskAide is VBScript. You can use intrinsic objects in VBscript or create references to ActiveX objects such as the ADO data connection objects, File System Object etc.

For an in-depth reference on VBScript see the Microsoft web site.

TaskAide passes several objects in to the script that can be used from within the script.

## **TASK**

The task object contains properties relating to the current task where the script is executing. In scripted tasks it's used to set the next transition for the system to take. In user assignment scripts it's used to set the current user account. (IE assign the task to the user.)

It has the following properties:

### **TaskName**

Returns the name of the current task.

### **TaskID**

Returns the system ID number for the current task.

### **FlowName**

Returns the name of the flow definition that the current task is a member of.

### **FlowDefinitionID**

Returns the system ID number for the flow definition.

### **ImagingFolderName**

Returns the name of the imaging folder this flow is defined for.

### **FlowInstanceID**

Returns the system ID number for the flow instance.

### **CurrentUser**

Used to assign a task to a user. This is only used in a "User Assignment" script. For a scripted task it contains the value "SYSTEM".

Example:

```
Task.CurrentUser = "joe"
```

This will assign the current task to user account "joe".

### **SelectedTransitionID**

Used to tell the system which transition to take to the next task. Set this to the transition ID number as listed in the task properties.

Example:

```
Task.SelectedTransitionID = 35
```

This will cause the system to select transition ID 35 when the script finishes executing. If this value is left at 0, then the scripting engine will not take any transition and will leave the flow at the current task. This will cause the script to execute again next time the transition daemon is run. Setting the SelectedTransitionID=0 can be useful in cases where you need the flow to wait at this task until a certain condition is met.

### **LastWorkedBy**

This property returns a Taskaide user object that represents the user that worked the last manual task in this flow. This can be used after an approval/denial task to get which actual user performed the approval or denial. You can then update a database field or do other work based on the user account.

Example:

```
ImagingRecord("ApprovedBy") = Task.LastWorkedBy.UserName
```

## **TASKERROR**

This object is used to flag an error. Setting the error property on this object will cause the flow to enter an error state.

The object can also be used for debugging scripts by using it's logging methods.

It has the following properties and methods:

### **ErrorFlag**

Set this to True to set the flow to an error state.

### **ErrorDescription**

Use this to set a descriptive error message.

### **LogFile**

Use this to specify a log file to write log entries to. By default it is set to [TACore.DLL path]\TAScript.log. (IE: C:\Program Files\ESCCCommonFiles\TAScript.log)

You can change this to have different tasks write to different log files for debugging purposes.

### **WriteLog(LogEntry)**

Use this method to output an entry to the log file. This can be used to log progress through a script or to output values from a script for debugging purposes.

Example:

```
TaskError.WriteLog "Checking Value X = " & X
```

## **EMAIL**

The e-mail object can be used to send basic e-mails from a script. This object adds e-mail messages to the Task Aide Mail queue.

### **Email\_Add(Recipient, Subject, MessageBody)**

Use this method to send an e-mail.

For the Recipient parameter, you can enter the following values:

“Admins” – sends the message to any user marked as an admin in TaskAide

“Role:[RoleName]” – Sends the message to all members of a role.

“User:[UserName]” – Sends the message to a specific user account.

“[UserAccount]” – Sends the message to a specific user account.

“Field:[Field Name]” – Sends the message to the e-mail address or user name contained in the specified imaging field.

“[specific e-mail address]” – Sends the message to specific email address such as

“[someone@domain.com](mailto:someone@domain.com)”

The “Subject” parameter is a test value for the message subject.

The “MessageBody” parameter is the text of the message body. HTML tags can be used here to format the message.

Mail message are submitted to the mail queue after the script completes.

Examples:

```
Email.Email_Add "joe@company.com", "Here is a test message", "Please read this test message."
```

```
Email.Email_Add "ROLE:General Manager", "This is an e-mail notification", "Please reveiew the following.<BR>Item: 1..."
```

```
Email.Email_Add "User:jsmith", "Test Message", "Here is a test message. <B>Text in bold</B>"
```

```
Email.Email_Add "Field:AddedBy", "Test Message", "The request you added has been completed."
```

## **Email\_Clear()**

Use this method to clear any messages that were previously added.

## **IMAGINGRECORD**

The ImagingRecord object contains the field values from the imaging system for the record associated with the current flow.

This object is a Microsoft ADO RecordSet object containing a single record. This object is updateable meaning you can set field values and the imaging database will be updated after the script completes or when you call the Update method of the record set. For more information on Microsoft ADO RecordSet objects see the documentation at [www.microsoft.com](http://www.microsoft.com)

To get a value from the record set use parenthesis with the field name in quotes.

Example: Getting a value from the imaging record.

```
InvoiceAmt = ImagingRecord("INV_AMT")
```

Example: Sending an e-mail using a value from the imaging record.

```
Email_Add "joe@company.com", "Invoice Number " & _  
ImagingRecord("INV_Number") & " has been approved.", "Message text"
```

Example: Setting a value in the imaging record.

```
ImagingRecord("ApprovedBY") = "Bill"
```

You can call the update method here to immediately write the data to the database.

```
ImagingRecord.Update
```

Note: The system automatically calls the update method after the script completes.

## **DB**

The DB object contains references to Microsoft ADO Connection objects. It has properties representing the connection to the TaskAide database, the imaging master database, and the imaging folder database.

## **DBTaskAide**

Represents the connection to the TaskAide Database. ([workflow prefix]Master)

## **DBImaging**

Represents the connection to the Imaging Database. (IMEV\_Master)

## **DBImgingFolder**

Represents the connection to the Imaging Folder Database. (IMEV\_[FolderName])

These objects can be used to perform queries against databases. These are passed in to the script primarily as a convenience to save you the trouble of having to create the ADODB.Connection object yourself.

# Taskaide Core Objects

Taskaide Core Objects can be used from within your scripts. The objects are not passed in to the scripting engine by default. What this means is that in order to use them you will need to create an instance of the object before using it. To create an instance of the object, dim the variable, then use the CreateObject() function.

Example:

```
Dim taUser  
  
Set taUser = CreateObject("TACore.User")
```

The Taskaide Core Objects can be used for advanced scripting in cases where you need to load user or role data from the taskaide database. All core objects have a LastErrorCode and LastErrorDescription property that can be checked for errors after making a function call.

## User

### Properties:

`UserID_TaskAide` – Returns the TaskAide user ID for the user.  
`UserID_Imaging` – Returns the LaserVault Imaging UserID for the user.  
`UserName` – Users imaging account name.  
`EmailAddress` – User's e-mail address.  
`Roles` – RoleCollection object that contains each role the user is a member of.  
`UserProperties` – UserPropertiesCollection object that contains each user property and value defined for this user.

### Methods:

```
DB_LoadByUsername(DBConnection As TACore.DatabaseConnection, ByVal  
UserName As String, Optional ByVal IncludeRoles As Boolean, Optional  
ByVal RaiseError As Boolean = True)
```

This method loads the user's information into the object using the user name.

Example:

```
Dim taUser  
Set taUser = CreateObject("TACore.User")  
taUser.DB_LoadByUsername DB.DBTaskAide, "joe", True
```

This would load the properties for user account "joe" and also load the roles the user is a member of.

```
DB_Load(DBConnection As TACore.DatabaseConnection, Optional ByVal  
IncludeRoles As Boolean, Optional ByVal RaiseError As Boolean = True)
```

This method will load a user's account details based on the task aide user id.

Example:

```
taUser.UserID_TaskAide = 1  
taUser.DB_Load DB.DBTaskAide, True
```

## **UserCollection**

### **Properties:**

Count – Number of items in the collection.

### **Methods:**

Item([index, or user name]) – Default method that returns the selected item

Example:

```
Set User = UserCollection.Item("joe")  
Set User = UserCollection("joe")
```

```
Set User = UserCollection.Item(1)  
Set User = UserCollection(1)
```

```
DB_LoadAllTaskAideUsers(DBConnection As TACore.DatabaseConnection,  
Optional ByVal IncludeRoles As Boolean, Optional ByVal RaiseError As  
Boolean = True)
```

This method loads all users defined in TaskAide. For the DBConnection parameter you can use the DB object passed in to the script.

Example:

```
Dim taUsers
```

```
Set taUsers = CreateObject("TACORE.UserCollection")
```

```
taUsers.DB_LoadAllTaskAideUsers DB, True
```

```
DB_LoadAdmins(DBConnection As TACore.DatabaseConnection)
```

This method loads all users marked as an admin in the TaskAide system.

Example:

```
Dim taUsers  
Set taUsers = CreateObject("TACore.UserCollection")  
taUsers.DB_LoadAdmins DB
```

## **UserProperty**

This object represents on user property associated with a user.

### **Properties:**

`PropertyName` – Name of the user property.  
`PropertyValue` – Value of the user property.

## **UserPropertyCollection**

This is a collection of user property objects.

### **Properties:**

`Count` – Number of items in the collection

### **Methods:**

`Item([Index or property name])` – Returns an item in the collection.

Example:

```
Dim taUserProp  
  
Set taUserProp = UserProperties.Item("Department")  
  
Exists(PropertyName as String) as Boolean
```

Returns true if a property with the given name exists in the collection.

## **Role**

The role object represents a defined role in the system.

### **Properties:**

`RoleName` – Name of the role.

`RoleDescription` – Description of the Role

`Users` – `UserCollection` containing all the users in the role.

## **RoleCollection:**

This object represents a list of Role objects.

### **Properties:**

`Count` – Number of items in the collection.

### **Methods:**

`Item([Index or Role Name])` – Returns the specified role from the collection.

### **Example:**

```
Dim taRole
```

```
Set taRole = Roles.Item("General Manager")
```

```
Set taRole = Roles.Item(1)
```

```
DB_LoadAllRoles(DBConnection As TACore.DatabaseConnection, Optional  
ByVal IncludeUsers As Boolean, Optional ByVal RaiseError As Boolean =  
True)
```

This function will load all roles defined in the system. You can pass the DB object for the `DBConnection` parameter.

### **Example:**

```
Dim taRoles
```

```
Set taRoles = CreateObject("TACore.RoleCollection")
```

```
taRoles.DB_LoadAllRoles DB, True
```

## Scripting Examples

The example below shows how to route an invoice based on the invoice amount.

The screenshot shows the 'Edit Task' dialog box with the following configuration:

- Task Name:** Department Route
- Task Transition Method:** Scripted
- Task Description:** Route to the appropriate department.
- Task Instructions:**

```
If ImagingRecord("Invoice_Amount") > 500 Then
  Task.SelectedTransitionID=2
Else
  Task.SelectedTransitionID=3
End If
```
- Display Properties:**
  - Task Node Preview: A yellow diamond shape labeled 'Task Name'.
  - Shape Type: Losange
  - Border Type: Solid
  - Border Width: 0
  - Fill Color: Yellow
  - Text Color: Black
  - Border Color: Black
  - Mark this task as Urgent
- Transitions possible from this task:**

ID	Transition Name	Destination Task
3	Less than \$500	Accounting Process
2	More than \$500	Department Approval

We've created a task called "Department Route" and selected the "Scripted" transition method.

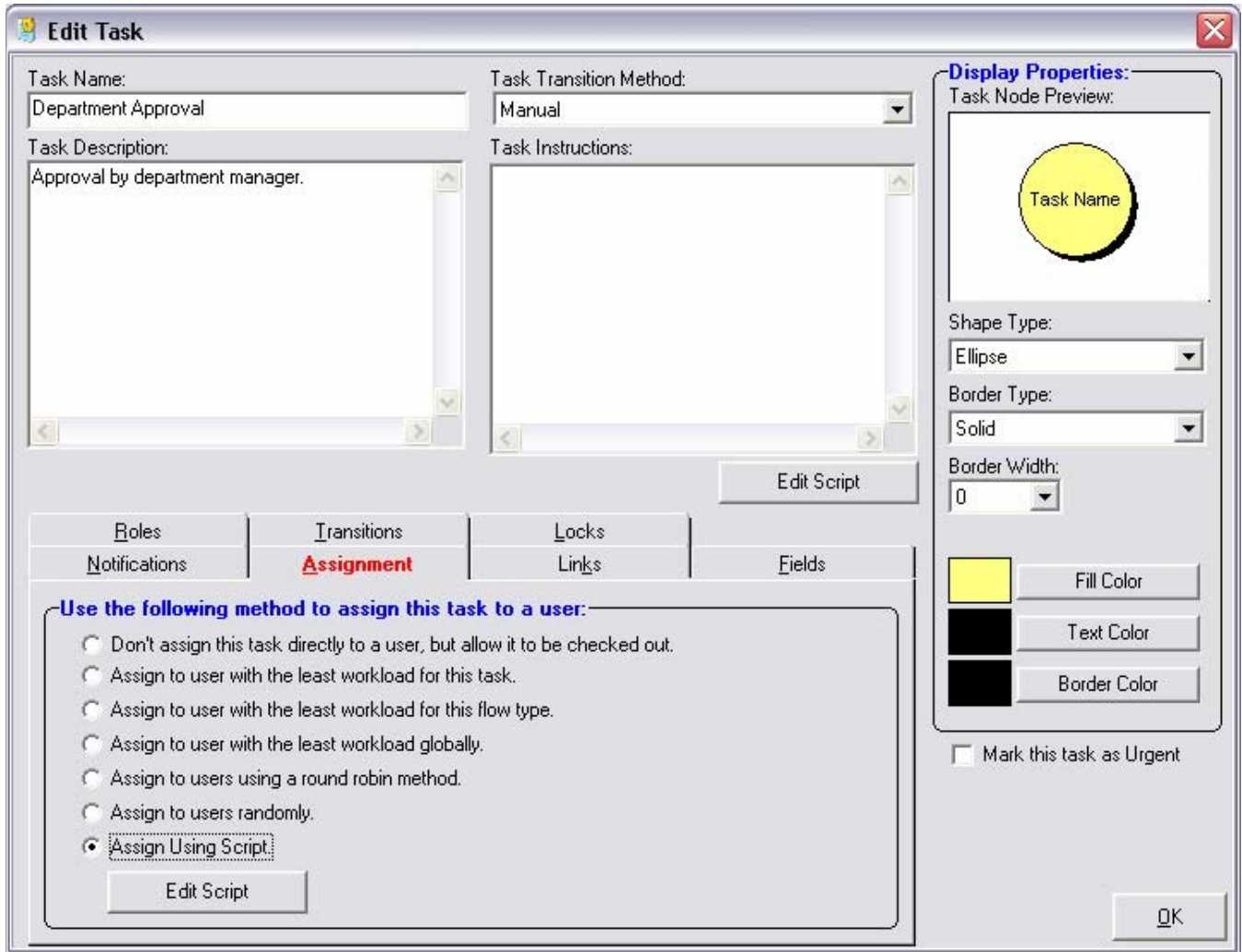
For the instructions we've entered the following VBScript. Note: you can click the "Edit Script" button to bring up a full screen window for editing the script.

```
If ImagingRecord("Invoice_Amount") > 500 Then
  Task.SelectedTransitionID=2
Else
  Task.SelectedTransitionID=3
End If
```

In this example we are using the "Invoice\_Amount" field from the imaging system to determine where to go next. You can see in the screen shot we have to 2 transitions defined.

The ID number is what's used by the system to determine which transition we take when we set TASK.SelectedTransitionID=[transition id]

The next example shows how to assign a task to a user using script.



In the task properties we have set the user assignment method to “Assign Using Script”. Click the “Edit Script” button on the Assignment pane to edit the user assignment script.

In our example we have the following script defined:

```
If ImagingRecord("State") = "TX" Then
    Task.CurrentUser = "Joe"
Else
    Task.CurrentUser = "Pam"
End IF
```

In this example we are assigning the task based on the “State” field from the imaging system. If the state is TX – Texas, the task is assigned to Joe, else it’s assigned to Pam.

The next scripting example shows how to assign a task to a user based on the department field using a select case statement. We use the "UCase" function to convert the value to upper case so we can type our literal values in upper case to ensure a match.

```
Select Case UCase(ImagingRecord("Department"))  
  
Case "IT"  
    Task.CurrentUser = "Bill"  
  
Case "SALES"  
    Task.CurrentUser = "Joe"  
  
Case "HR"  
    Task.CurrentUser = "Mary"  
  
Case Else  
    Task.CurrentUser = "John"  
  
End Select
```

The next example shows how to update an imaging record with the user name of the person who performed the last task. In this example we have a simple Approve/Deny task for a purchase request. After the request has been approved, we want to fill in our "ApprovedBy" imaging field with the user name of the person who approved the request.

```
`Set the ApprovedBy Field with the user name of the last user to work  
this task.  
ImagingRecord("ApprovedBy") = Task.LastWorkedBy.UserName  
  
`Move to the next task  
Task.SelectedTransitionID = 25
```

The next example shows how to select a role based on the department field and then assign the task to the first user in that role.

```
Dim taRoles
```

```
Dim taRole
```

```
'Create the role collection
```

```
Set taRoles = CreateObject("TACore.RoleCollection")
```

```
'Load all defined roles
```

```
taRoles.DB_LoadAllRoles DB, True
```

```
'Based on the Department field, select a role
```

```
Select Case UCase(ImagingRecord("Department"))
```

```
Case "IT"
```

```
    Set taRole = taRoles("IT Managers")
```

```
Case "HR"
```

```
    Set taRole = taRoles("HR Managers")
```

```
Case "SALES"
```

```
    Set taRole = taRoles("Sales Managers")
```

```
Case Else
```

```
    Set taRole = taRoles("General Managers")
```

```
End Select
```

```
'Assign the task to the first user in that role.
```

```
Task.CurrentUser = taRole.Users(1).Name
```