# Workflow Designer for Exchange: Automating Workflow on Exchange Folders, an End-to-End Developer Walkthrough

Marci Marra, Valerie Whitcomb, and Doug Yoder
Microsoft Corporation

Created April 17, 2000

**Keywords:** team solution, creating; team solution, building; team solution, walkthrough; team solution, sample; team solution, example; Exchange Workflow Designer, end-to-end scenario; Exchange Workflow Designer, walkthrough; Exchange Workflow Designer, creating a team solution; Exchange Workflow Designer, sample process; Exchange Workflow Designer, example; workflow, creating team solutions, scripting

## Summary

This paper guides you through the process of creating a team solution that uses an Exchange public folder, an Outlook form, and workflow to move an expense report through a simplified approval process. The workflow created in this example uses values in an Outlook form to determine how the report is sent through the various states in the process. In addition, it uses script to notify users automatically that the report has been submitted and reviewed.

# Introduction

Does your department need an application for tracking reports or other items in Microsoft® Exchange? Would you like to automate reminders and notifications based on items in an Exchange folder? You can create such a team solution quickly using the Microsoft® Workflow Designer for Microsoft® Exchange 2000 Server.

Using the Workflow Designer, you can add a workflow process to any items in Exchange 2000 folders. A workflow process automates control of items in a folder by moving them through a series of states depending on the actions and conditions you specify. In addition, you can enhance your workflow solution by scripting your own functions that run when the actions are performed. Your workflow process can track any type of item available in Microsoft® Outlook®, such as messages, forms, or tasks, as well as any file, such as documents (.doc), spreadsheets (.xls), or text files (.txt).

The user interface for a workflow process determines how the user submits the item that moves through the process. When designing the user interface for your solution, you have a wide variety of options. You can integrate Microsoft® Visual Basic® applications, Web pages, and any other type of front-end that is able to access the Exchange folder objects.

By completing this walkthrough, you can create a team solution that uses a public folder, an Outlook form, and a workflow to move an expense report through a simplified approval process. The workflow you create in this example uses values in the form to determine how the report is sent through the various states in the process. In addition, it uses script to notify users automatically that the report has been submitted and reviewed.

Because this walkthrough is designed to highlight how to create a simple workflow process, not the user interface for the application, the user interface provided in this example is a simple Outlook form intended only to show how to automate your workflow. The example application created in this walkthrough is not designed to be a real-world solution.

This paper covers the following topics:

- Preparing Exchange for a Team Solution
- Creating the User Interface in Outlook
- Planning the Expense Report Workflow
- Creating a Workflow Process
- Activating the Workflow Process

# Preparing Exchange for a Team Solution

To follow along with this walkthrough, you must have access to an Exchange 2000 server, the Workflow Designer for Exchange, and the appropriate permissions on the server. For more information about preparing Exchange for a workflow process, see the Exchange SDK documentation, available on the Exchange 2000 Web site at http://msdn.microsoft.com/exchange/.

Creating a team solution requires the following components:

- **Server**: Microsoft® Windows NT® 4.0 or Microsoft® Windows® 2000 Server with Exchange Server 2000 as the back-end for the application.
- **Development tool**: Microsoft Workflow Designer for Exchange as the tool for creating the workflow process. This tool can reside on either a client computer with access to the server or the server itself.

Before you can add workflow to your folder, you must have the following:

- **A public folder** in Microsoft Exchange 2000 Server. For example, you could add a folder called Expense Reports to Public Folders.
- **Workflow design permissions** on the Exchange 2000 Server. By creating the folder in Outlook, you automatically have the owner permissions required. For additional information about setting the permissions required for creating workflow processes, see the Exchange SDK documentation, available on the Exchange 2000 Web site at http://msdn.microsoft.com/exchange/.
- **A set of items your team wants to track**. Using this walkthrough, you can create an Outlook form that shows simple controls that make it possible for users to enter and to track Exchange items. In addition to Outlook, you can use .asp pages, a Visual Basic form, or Data Access Pages as the user interface for your folder and workflow. For more information about implementing a user interface for workflow on

a folder, see the Exchange SDK documentation, available on the Exchange 2000 Web site at http://msdn.microsoft.com/exchange/.

- **A plan** for the business process that tracks an item. Using this walkthrough, you can create a simplified workflow process for an expense-reporting form in Outlook.

# Creating the User Interface in Outlook

You can use a workflow process created using the Workflow Designer for Exchange to track any type of item available in Exchange 2000. This walkthrough demonstrates how to create a workflow team solution using a simplified expense report implemented as an Outlook Post form. The solution created in this example is designed only to illustrate how you can integrate a form with a workflow, and it lacks many of the features a real expense report would require. In this example, the user interface is created first because the workflow and its script require the values obtained through controls on the user interface form.

The first step is to create a public folder that will be used to post the expense reports.

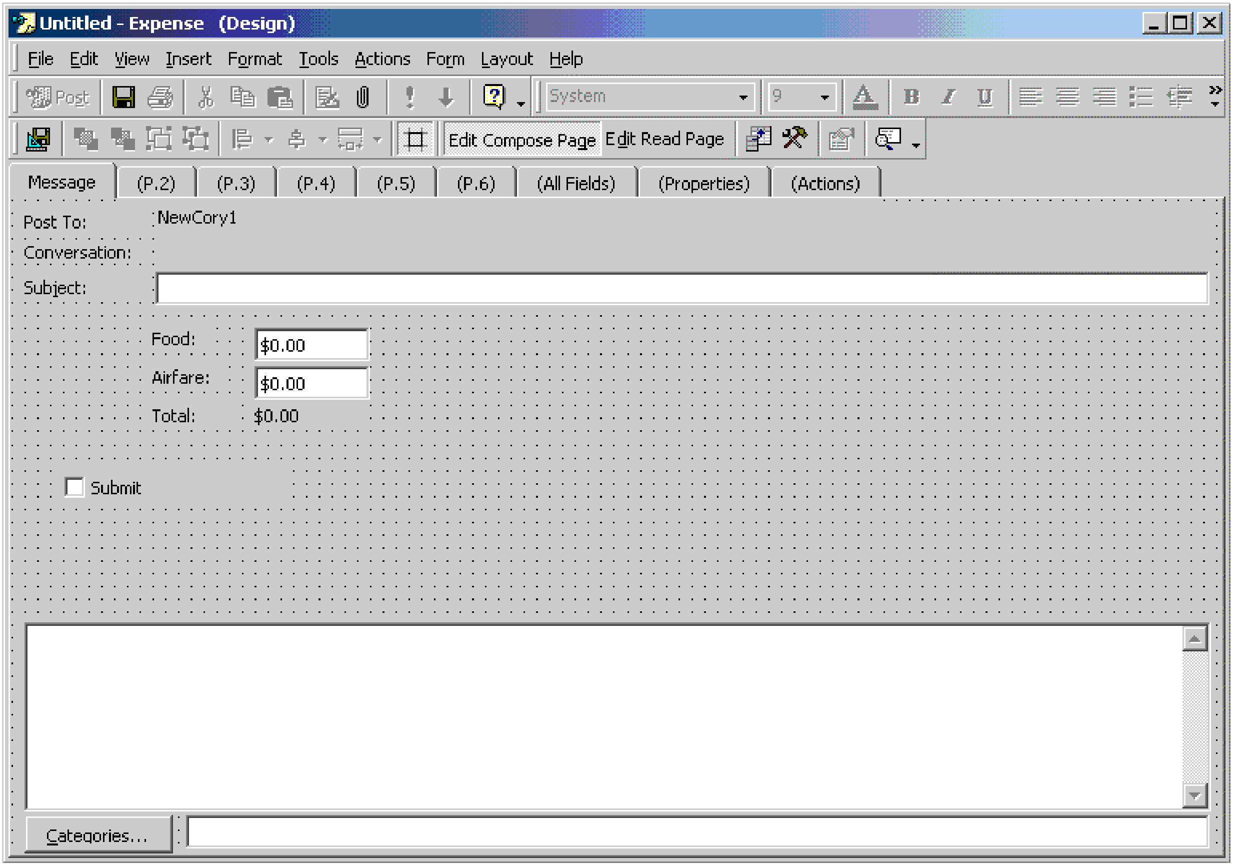
### To add a public folder

1. Start Microsoft Outlook and display the folder list if it's not already shown by selecting **Folder List** from the **View** menu.
2. Expand the **Public Folders** node and then **All Public Folders**.
3. Right-click **All Public Folders** and select **New Folder**.
4. Enter a name for the new folder, such as Expense Reports.
5. Navigate to the new folder by selecting the new folder you just created.

To create the user interface for this walkthrough, you create two different views of the Post form in Outlook:

- **Edit Compose Page view**. This view is used by the person submitting the expense report to input data in the expense fields. It appears when the report is created or composed. You are shown this view automatically when you are designing a form. The text boxes on the form provide values to the script used in the workflow.
- **Edit Read Page view**. This view is used by a manager to approve or reject the expense report. This view of the form contains an approval and rejection combo box in addition to the expense fields. The following figure shows the Edit Read Page view of the form.

**Figure 1. User interface created for this walkthrough.**



Post To: NewCory1
Conversation:
Subject:
Food: $0.00
Airfare: $0.00
Total: $0.00
☐ Submit

By completing this walkthrough, you create and publish an Outlook form and set it as the default form for the folder. The fields in this form are used by the workflow process described in "Planning the Expense Report Workflow" later in this paper.

### To create the Edit Compose Page view of the Outlook form

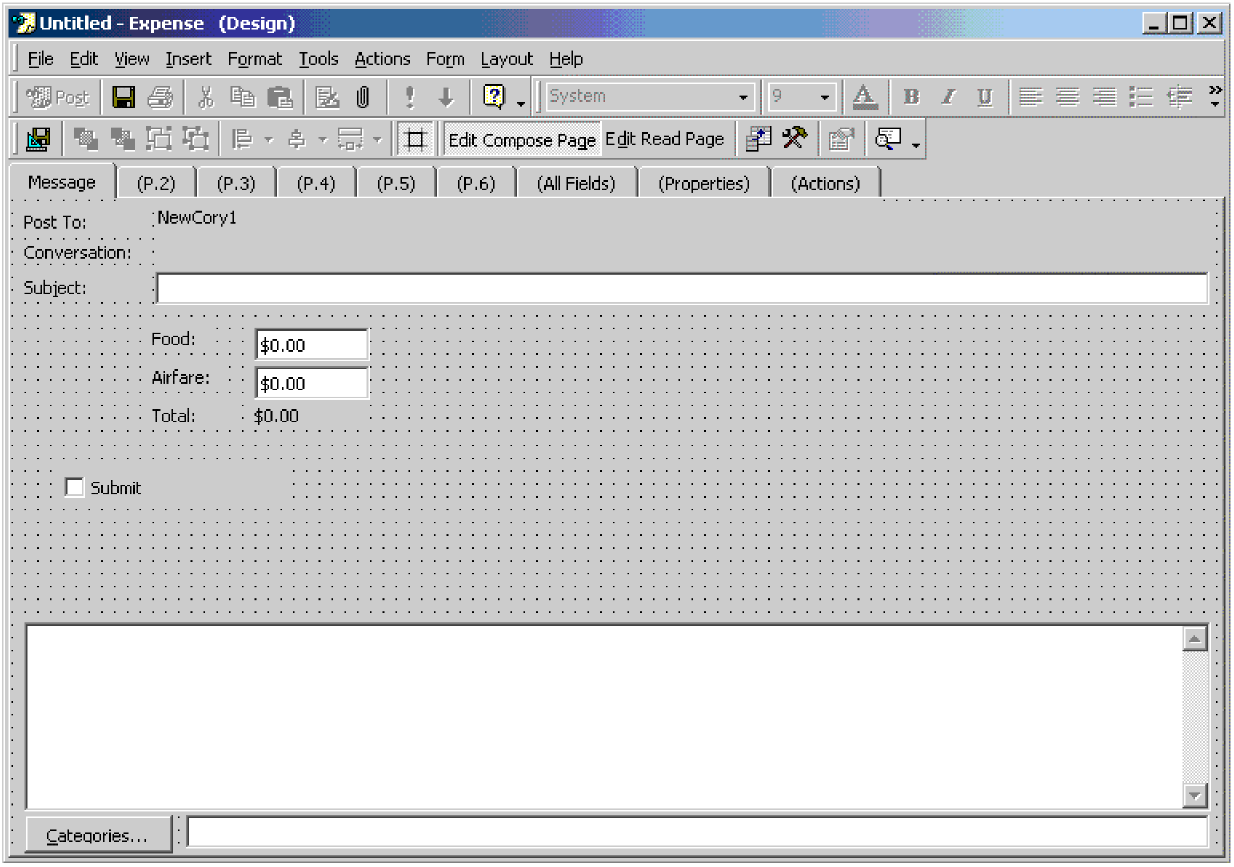
1. In Outlook, open the public folder you want to add the workflow to.
2. From the **Tools** menu, choose **Form**, and then choose **Design a Form**.
3. In the **Design a Form** dialog box, choose **Post**.
4. In the form, resize the message area to create space for the expense report fields.
5. Right-click the form, and choose **Control Toolbox**.
6. Add the following label controls to the form as shown in Figure 1.

| Control | Properties |
|---|---|
| Food label | Caption: Food |
| Airfare label | Caption: Airfare |
| Total Expense label | Caption: Total Expense |

7. Add the following controls to the form to enter and display the expense values. To modify the associated properties, right-click the control and choose **Properties**. On the **Value** tab, click the **New** button so specify a user-defined field that the workflow script can reference.

| **Control** | **Properties** |
|---|---|
| Food textbox | Name: Food<br>Type: Currency |
| Airfare textbox | Name: Airfare<br>Type: Currency |
| Total Expense label (A label box is used because the user cannot change the value) | Name: TotalExpense<br>Type: Currency<br>Check **Set the initial value of this field to** and enter this formula for the initial value: **VAL([Airfare]) + VAL([Food])** |
| Submit checkbox | Name: Submit<br>Type: Yes/No<br>Format: True/False<br>Check **Set the initial value to this field to** and enter **No** for the initial value. |

**Figure 2. Edit Compose Page view of the Outlook form.**

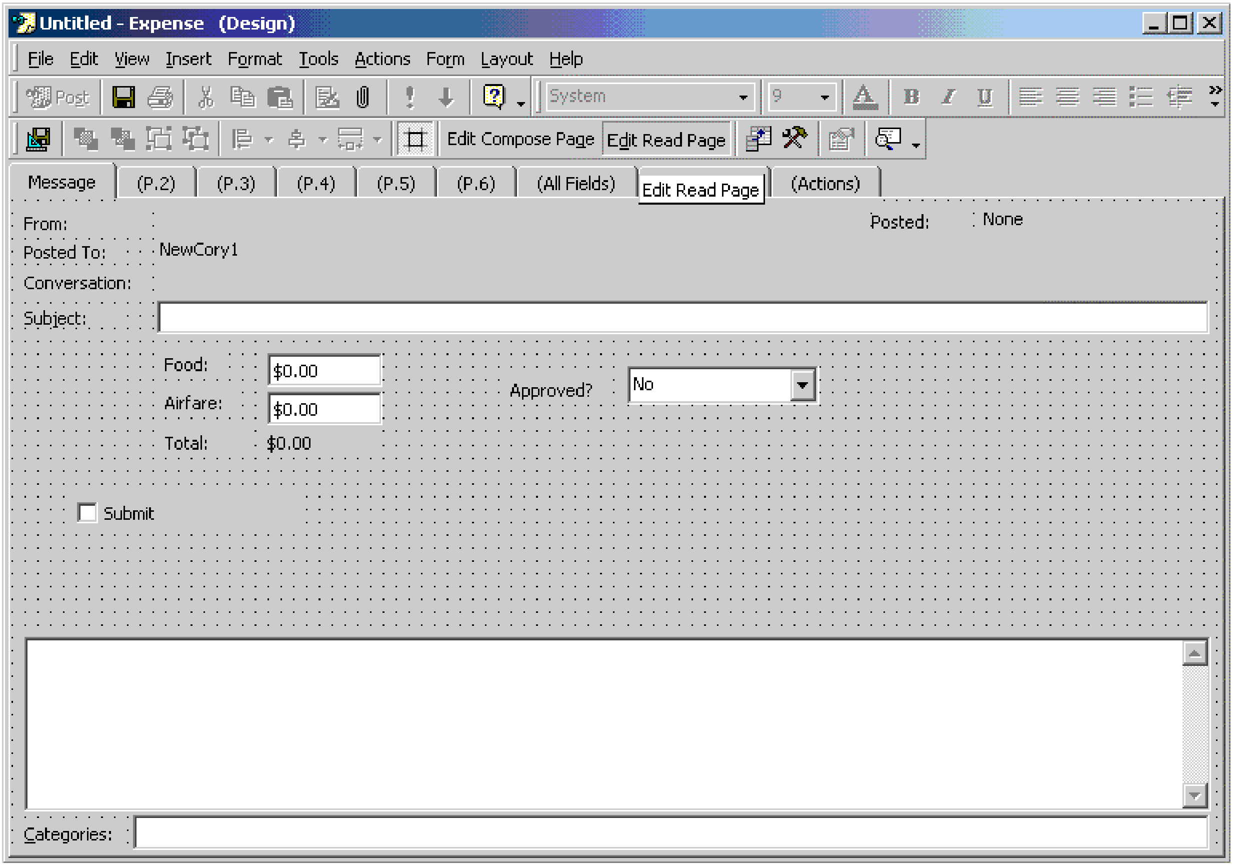


Post To: NewCory1
Conversation:
Subject:
Food: $0.00
Airfare: $0.00
Total: $0.00
☐ Submit

### To create the Edit Read Page view of the Outlook form

1. In the Edit Compose Page view of the form, select and copy all of the controls you added to the form.
2. Above the tabs for the form, click the **Edit Read Page** button.
3. Resize the message area to create space for the controls, and then paste them in place.

**Figure 3. Edit Read Page view of the Outlook form.**



From:

Posted To: NewCory1

Conversation:

Subject:

Posted: None

Food: $0.00

Airfare: $0.00

Total: $0.00

Approved? No

☐ Submit

Categories:

4. Add a combobox control to the form, right-click the combobox control, and then select **Properties**. Then click the **New** button to specify a user-defined field and set the properties shown in the following table.

| **Property** | **Value** |
|---|---|
| Name | Approved |
| Type | Yes/No |
| Format | Yes/No |
| List type | Dropdown |
| Possible values | Yes;No |
| Initial value | No |

To make the form available to the workflow, you must publish it. In addition, you can specify it as the default form for the folder. You must have editor, publishing editor, or owner permissions to add forms to a public folder.

**Note** Do not enable workflow on a folder before publishing the folder. If you do, you might get a permissions error even though you are the owner of the folder.

### To publish the form and specify it as the default form for the folder

1. From the **Tools** menu, choose **Forms**, and then choose **Publish Form As**.
2. In the **Publish Form As** dialog box, enter a name for the form, and then click **Publish**.

   **Note** The name you provide is the name that the user sees when choosing a form to post in the folder. For the expense report example, you can name it "Expense Report."

3. Close the form, and select **No** when prompted to save your changes (because the form is already published, you do not need to save your changes).
4. In the **Folder** list, right-click the folder you want to add workflow to, and choose **Properties**
5. In the **When posting to this folder, use** box, select the form you created, and then click **OK**.

Your form will appear when a user decides to post to that folder.

# Planning the Expense Report Workflow

To learn about using the Workflow Designer for Exchange, you can use this walkthrough to create a workflow process that includes a series of states through which an expense report item moves and the actions that occur to move the item.

## The Workflow Process

The workflow process in this example is based on the following scenario: An employee creates an expense report form in an Expense Report folder located in the Public Folder. When the report is created, the item enters the Draft state. To move into the Submit to Manager state, the employee must select a Submit check box and save the report. Otherwise, the user can modify and save the item as many times as needed while the Submit check box is not saved. Once the Submit check box is selected, the report is automatically moved to the Submit to Manager state. When the report is moved to the Submit to Manager state, a manager can then either approve or reject the report.

If the manager selects "Yes" in the Approved check box, then the report moves into the Approved state. If the manager selects "No," then the report returns to the Draft state.

Items in the Approved state or Draft state can be deleted. Items in the Submit to Manager state cannot, because a Delete action is not specified for it.

Each workflow state specifies a stage in the workflow process. Details about the specifications for the states, actions, and script used to implement the workflow process in this walkthrough are provided in the procedures in the "Creating a Workflow Process" section later in this paper.

The expense report workflow example includes the following states:

- Draft
- Submit to Manager

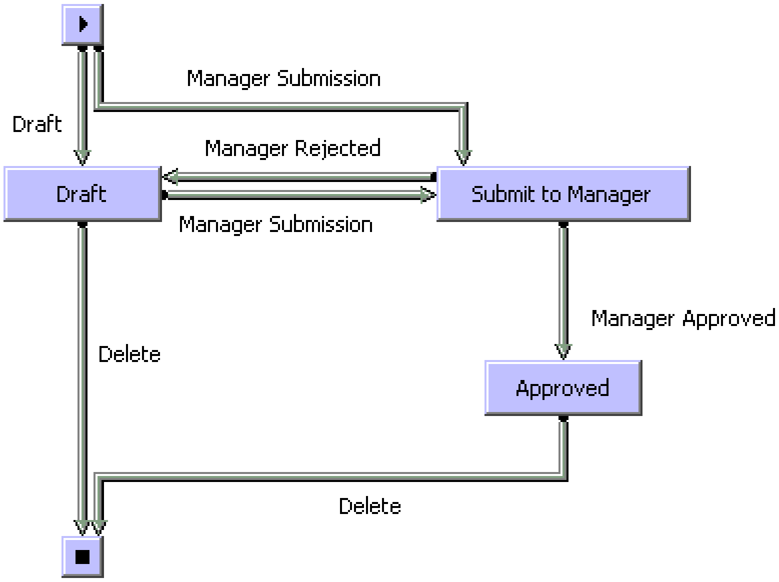
- Approved

Actions are used to transition an item from one state to another. Each state typically has at least two actions. One action moves the item into the state, and one action moves the item out of the state. Each action has a condition expression associated with it that is used in determining which state is transitioned to.

States can have any number of actions. For example, the Submit to Manager state has three transition actions: Manager Approved, Manager Rejected, and Expiry. Transition actions are triggered by a change to the item in the folder. When triggered, the condition on the action is evaluated. If the condition evaluates to True, then the script associated with the action is executed. In this example, the workflow for the expense report has the following actions:

- **Manager Submission and Draft** The conditions for these actions are evaluated after the user posts the report to the folder and are associated with the Create event of the workflow. The conditions for these two actions determine whether the employee selected the Submit check box. Manager Submission is the first action in the list of actions, therefore its condition is evaluated first. If the condition is True, then script for the action is executed. In this walkthrough, when created, the expense report is routed based on whether the employee wanted to submit the report right away or wanted to continue to edit it.
- **Manager Submission** The condition for this action is evaluated after the item in the folder is read and saved. The value of the Submit checkbox control must be True.
- **Manager Approved** The condition for this action is evaluated after the item in the folder is read while in the Submit to Manager state. In this example, the value of the Approved combobox control on the Outlook form must be Yes.
- **Manager Rejected** The condition for this action is evaluated after the item in the folder is read while in the Submit to Manager state. In this example, the value of the Approved combobox control on the Outlook form must be No.
- **Expiry** This action occurs after the report has been in the Submit to Manager state for a user-defined period of time without being opened.
- **Delete1 and Delete2** These actions make it possible for the report to be deleted while in either the Draft or Approved state. Items in the Submit to Manager state cannot be deleted.

**Figure 4. The workflow process for the expense report example.**



## The Workflow Scripts

In your script associated with each action, you can call user-defined functions that you specify in the Shared Script tab of the Workflow Designer. This example uses the following custom functions, along with a set of standard functions that are called from the custom functions. The script used for each function is provided in the "Creating the Shared Script Procedures" section later in this paper.

### Custom Functions

- **SubmitExpenseReport** Called from the Manager Submission action when an expense report is submitted to a manager. Sends an e-mail message to the approval authority containing a link to the posted item awaiting approval.
- **ApproveExpenseReport** Called from the Manager Approved action when the manager either approves or rejects an expense report. Checks the value on the Outlook form for approved/rejected status and sends notification e-mail to the person who submitted the form. If the item is rejected, it also resets the Submit property value and returns it to the Draft state.
- **ExpireExpenseReport** Called from the Expiry action when an expense report remains submitted to a manager for seven days. Sends a reminder message to the approval authority that the expense report needs to be approved.

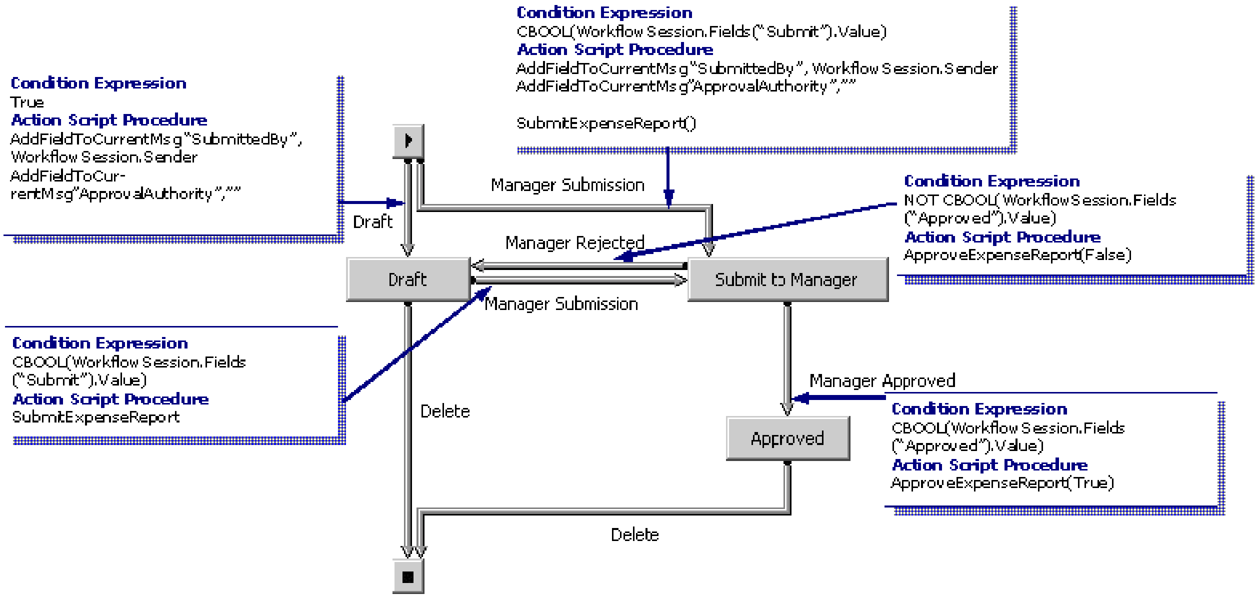
### Standard Functions

- **AddFieldToCurrentMsg** Called from the Draft and Manager Submission actions. Dynamically adds a new property to the item. This is used to add the Approval

Authority property, returned from the GetUserManager script, and the SubmittedBy property created in the WorkflowSessionSender script, to the item.

- **ClearPermissions** Clears all previously set permissions. This is used to change the permissions for the person who submitted the form to read-only after the item has been submitted for approval.
- **GetPathAndSubject** Returns the path to the public folder and the subject of the current item. This makes it possible for the SubmitExpenseReport and ApproveExpenseReport functions to add a link to the specific expense report item in the message body.
- **GetUserManager** Returns application-defined data. In this example, it returns the e-mail alias of the user's manager.
- **SendMessage** Sends an e-mail message in response to workflow actions.
- **SetCurrentMessageField** Sets the value of an existing property of an item. This is used to reset the Submit property value when an expense report is rejected and moved from Approved to Draft states.
- **SetPermissions** Sets permissions making it possible for only the approval authority to make changes to the item and sets read-only permissions to all others.

**Figure 5. Scripting associated with the actions in the expense report example.**



**Condition Expression**
```
CBOOL(Workflow Session.Fields("Submit").Value)
```
**Action Script Procedure**
```
AddFieldToCurrentMsg"SubmittedBy", Workflow Session.Sender
AddFieldToCurrentMsg"ApprovalAuthority",""

SubmitExpenseReport()
```

**Condition Expression**
```
True
```
**Action Script Procedure**
```
AddFieldToCurrentMsg"SubmittedBy",
Workflow Session.Sender
AddFieldToCur-
rentMsg"ApprovalAuthority",""
```

**Condition Expression**
```
NOT CBOOL( WorkflowSession.Fields
("Approved").Value)
```
**Action Script Procedure**
```
ApproveExpenseReport(False)
```

**Condition Expression**
```
CBOOL(Workflow Session.Fields
("Submit").Value)
```
**Action Script Procedure**
```
SubmitExpenseReport
```

**Condition Expression**
```
CBOOL(Workflow Session.Fields
("Approved").Value)
```
**Action Script Procedure**
```
ApproveExpenseReport(True)
```

# Creating a Workflow Process

To create a workflow process, you open the Workflow Designer and specify the states and actions. Creating a workflow includes the following main tasks:

- Adding a workflow to a public folder.

- Adding states that the item will move through.
- Adding actions that determine how the item is handled and specify conditions for special handling.
- Adding script to actions to automate handling of items, sending notifications, or calling other functions you want to use.
- Creating script and functions on the Shared Script tab.
- Activating the workflow on the folder.

**Note** In the sections that follow, the generic procedure describing how to accomplish the designated task is followed by specific instructions for creating the expense report example.

## Adding Workflow to a Folder

After planning your workflow, you can create the workflow on a public folder. The Workflow Designer works directly on the folder, so you must create the folder before opening the Workflow Designer. When you create a new workflow process, the Workflow Designer automatically generates a simple workflow with one state and Create and Delete actions between the starting and ending blocks.

### To add workflow to a public folder

1. From the **Start** menu, choose **Program Files**, then choose **Workflow Designer for Exchange**, and select **Exchange Workflow Designer**.
2. In the **Open Folder** dialog box enter the Exchange Server.
3. Enter **Public/** in the **Folder** box, click the **Browse** button (**...**) to select the folder you created previously, and then click **OK** to start the Workflow Designer.
4. From the **File** menu, select **New Workflow Process**.
5. In the **New Workflow Process** dialog box, enter a name for the workflow process, and then click **OK**.

   To complete the expense report example, you can enter the name "Expense Reports."
6. In the **Rename State** dialog box, enter an appropriate name for your initial state, and then click **OK**.

   To complete the expense report example, you can enter the name "Draft."

You can make the diagram easier to read by arranging the states and actions into a more pleasing layout.

### To arrange the diagram

1. From the **Diagram** menu, choose **Arrange Diagram**, and then choose **Vertical**.
2. From the **Diagram** menu, choose **Zoom**, and then choose **Zoom to fit**.

## Adding States to the Workflow Diagram

You can add states to the workflow diagram representing each stage through which an item might pass during processing. In this example, the states used in the expense report workflow diagram are: Draft, Submit to Manager, and Approved.

### To add a workflow state to the diagram

1. Right-click the diagram background, and from the shortcut menu, select **Insert State**.
2. In the **New State** dialog box, specify a name, and click **OK**.

   To complete the expense report example, insert the following states: "Submit to Manager" and "Approved."
3. Right-click the **Delete** action (which is represented by the line between the Draft state and the end point), and choose **Remove Action**.

## Adding Actions to States

You can add actions to states, such as Enter or Edit, and add transition actions, such as Submit or Reject, to establish the relationships between the states in the workflow and to provide a place to add script for automating the workflow. In addition, workflow actions can be used to evaluate conditions and to trigger scripts that automate your workflow process.

> **Note** For more information about the different types of actions, see "Workflow Action Events" in the Workflow Designer for Exchange online documentation.

### To add an action to a state in the Workflow Designer

1. Right-click the state, and from the shortcut menu, select **Insert Action**.
2. Specify an action name (see the next procedure for details about the actions used in this walkthrough) in the **Action Name** text box, and select an **Action Type**.
3. In the **Next state** box, select the state that follows the transition action, and then click **OK** to add the action to the Actions list.
4. In the **Condition Expression** box, enter the condition that must be true for the transition to occur and the script to be executed.

   > **Note** The condition is set to True by default. This means any time the item is read in the folder, the transition is evaluated as True, and the item moves to the next state.
5. In the **Action Script Procedure** box, enter the procedure call to execute a script when the condition value is True.

To complete the expense report example, add the actions, conditions, and script specified in the following procedure. This script calls user-defined functions specified in the "Creating the Shared Script Procedures" section later in this paper.

### To create the workflow expense report example actions

1. To the **Submit to Manager** state, add the following actions with the identified properties.

    - **Action Name: Manager Submission**

    | Property | Value |
    |---|---|
    | Action type | Create |
    | Condition expression | CBOOL(WorkflowSession.Fields("Submit").Value) |
    | Action script procedure | SubmitExpenseReport |

    - **Action Name: Enter**

    | Property | Value |
    |---|---|
    | Action type | Enter |
    | Condition expression | True |

    - **Action Name: Manager Rejected**

    | Property | Value |
    |---|---|
    | Action type | Change |
    | Next state | Draft |
    | Condition expression | NOT CBOOL(WorkflowSession.Fields("Approved").Value) |
    | Action script procedure | ApproveExpenseReport(False) |

    - **Action Name: Manager Approved**

    | Property | Value |
    |---|---|
    | Action type | Change |
    | Next state | Approved |
    | Condition expression | CBOOL(WorkflowSession.Fields("Approved").Value) |
    | Action script procedure | ApproveExpenseReport(True) |

    - **Action Name: Expiry**

    | Property | Value |
    |---|---|
    | Action type | Expiry |
    | Condition expression | True |
    | Action script procedure | ExpireExpenseReport |

2. To the **Draft** state, add the following actions with the identified properties.

- **Action Name: Draft**

| **Property** | **Value** |
|---|---|
| Action type | Create |
| Condition expression | True |
| Action script procedures | AddFieldToCurrentMsg "SubmittedBy",WorkflowSession.Sender<br>AddFieldToCurrentMsg "ApprovalAuthority", "" |

- **Action Name: Manager Submission**

| **Property** | **Value** |
|---|---|
| Action type | Change |
| Next state | Submit to Manager |
| Condition expression | CBOOL(WorkflowSession.Fields("Submit").Value) |
| Action script procedure | SubmitExpenseReport |

- **Action Name: Draft**

| **Property** | **Value** |
|---|---|
| Action type | Change |
| Next state | Draft |
| Condition expression | True |

- **Action Name: Delete**

| **Property** | **Value** |
|---|---|
| Action type | Delete |
| Condition expression | True |

3. To the **Approved** state, add the following action with the identified properties.

- **Action Name: Delete**

| **Property** | **Value** |
|---|---|
| Action type | Delete |
| Condition expression | True |

## Creating the Shared Script Procedures

Shared script procedures are created, edited, and viewed on the Shared Script tab in the Workflow Designer. Think of items on the Shared Script tab as a common library of script routines. When you plan to use a procedure or function in your workflow process, you create it once on the Shared Script tab and then call it from the Condition Expression text box and from the Action Script Procedure text box.

The condition expression is the first script to be evaluated by the workflow engine in response to a given workflow action. By default, the condition expression returns a value of True. You can modify this value by typing "False" in the Conditional Expression scripting box, creating an expression, or calling a function from the Shared Script tab to perform a more complex evaluation.

Action script procedures are used to add automation to your workflow process. For example, if you want to send e-mail, post a message, delete items, or add an audit entry, all of these activities are accomplished using action script procedures. Action script procedures are executed only when the associated condition expression returns a value of True.

> **Note** User-defined fields in the Outlook form are case sensitive. Be sure to reference them appropriately in your script.

### To create a procedure or function on the Shared Script tab

1. In the **Process List**, select the workflow process you are enhancing with script.
2. In the **Workflow Process** pane, select the **Shared Script** tab.
3. Create a procedure, including Sub and End Sub tags, or a function, including Function and End Function tags.
4. Add the procedure call and audit entry to the **Action Script Procedure** text box on the **Design** tab of the **Workflow Process** pane.

> **Note** Success entries must be enabled to use the Audit trail for debugging. For details, see the "To enable script debugging and success entries" procedure later in this paper.

Many of the procedures and functions in these examples refer to and rely on each other. When using the example script, be sure to verify that all parameters and dependent functions exist in the workflow script.

### AddFieldToCurrentMsg

To use this code, create the AddFieldToCurrentMsg script on the Shared Script tab of the Workflow Process pane. This script is called from the Action Script Procedure box during the Draft and Manager Submission actions, adding the SubmittedBy and ApprovalAuthority property to the SubmitExpenseReport script.

```
Sub AddFieldToCurrentMsg (strName, Value)

    Dim FieldType

    FieldType = 8 ' BSTR
```

```
      WorkflowSession.Fields.Append CStr(strName), _
         FieldType, , , CStr(Value)
      WorkflowSession.Fields.Update
   End Sub
```

**ApproveExpenseReport**

To use this code, create the ApproveExpenseReport script on the Shared Script tab of the Workflow Process pane. This script checks for a True value in the Approved box on the Outlook form. If the value equals True, then e-mail is sent notifying the person who submitted the form that the expense report has been approved. If the value is not True, the person who submitted the form receives e-mail indicating the expense has been rejected, the expense report item is returned to the Draft state, and the Submit check box is reset.

The ApproveExpenseReport is dependent on the GetUserManager (approval authority), GetPathAndSubject, SendMessage, and SetCurrentMessageField scripts. If the parameter is True, the expense report is approved and the e-mail message indicates that it has been approved. If it is False, the e-mail message indicates that the report has been rejected.

```
   Sub ApproveExpenseReport(bApproved)

      Dim strSMTPAddress
      Dim strBody
      Dim strSubject
      Dim strMessageSubject
      Dim strApproved

      strSMTPAddress = WorkflowSession.Fields("SubmittedBy").Value
      strMessageSubject = WorkflowSession_
         .Fields("urn:schemas:mailheader:subject").Value
      If bApproved = True Then
         strApproved = "APPROVED"
      Else
         strApproved = "REJECTED"
      End If

      strSubject = strApproved & " -- Expense Report " & strMessageSubject
      strSubject = "Your expense report has been " _
         & strApproved & " (" & strMessageSubject & ")"
      strBody = "Your expense report has been " & strApproved _
         & " by " & WorkflowSession.Fields("ApprovalAuthority").Value _
         & vbCRLF & vbCRLF

      strBody = strBody & "View the expense report by clicking _
         here. <outlook://public folders/All Public Folders/Expense/~" _
         & strMessageSubject & ">" & vbCRLF & vbCRLF
      strBody = strBody & "------------------------------------" & vbCRLF
      strBody = strBody & "Expense Starting State: " _
         & WorkflowSession.StateFrom & vbCRLF
      strBody = strBody & "Expense Ending State: " _
         & WorkflowSession.StateTo

      SendMessage strSMTPAddress, strSubject, strBody

      ' Reset submit when rejected
      If bApproved = False Then
         SetCurrentMessageField "Submit", False
      End If

   End Sub
```

### ClearPermissions

To use this code, create the ClearPermissions script on the Shared Script tab of the Workflow Process pane. This script is called when the SubmitExpenseReport is executed. After the expense report is submitted, the permissions for the person who submitted for form change to read-only.

```
Sub ClearPermissions()
   WorkflowSession.ItemAuthors.Clear
End Sub
```

### ExpireExpenseReport

To use this code, create the ExpireExpenseReport script on the Shared Script tab of the Workflow Process pane. This script sends e-mail to the approve authority indicating that an expense report has been awaiting approval and has now expired. The report is returned to the Draft state.

The ExpireExpenseReport is dependent on the GetUserManager (approval authority) and SendMessage scripts.

```
Sub ExpireExpenseReport()

   Dim strSubject
   Dim strBody

   strSubject = "Action Required: Expense Report Approval Request"

   strOutlookURL = GetPathAndSubject
   strBody = "There is an expense report waiting for your approval." _
      & vbCRLF & vbCRLF
   strBody = "View the expense report by clicking here. _
      <outlook://public folders/All public Folders/" _
      & strOutlookURL & ">" & vbCRLF & vbCRLF
   strBody = "----------------------------------------" & vbCRLF
   strBody = "Submitted: " & CTXT(Date) & "  " & CTXT(Time) & vbCRLF
   strBody = "Expense ending state: " & WorkflowSession.StateTo

   strSMTPAddress = WorkflowSession.Fields("ApprovalAuthority").Value

   SendMessage strSMTPAddress, strSubject, strBody

End Sub
```

### GetPathAndSubject

To use this code, create the GetPathAndSubject script on the Shared Script tab of the Workflow Process pane. This script makes it possible the SubmitExpenseReport and ApproveExpenseReport functions to add a link to the expense report in the e-mail message body.

```
Function GetPathAndSubject()
   Dim iPos
   Dim iPathLength
   Dim strPath
```

```
      Dim strSubject

      strPath = WorkflowSession.Fields("DAV:parentname")
      strSubject = WorkflowSession_
         .Fields("http://schemas.microsoft.com/mapi/proptag/0x0037001F")_
         .Value

      iPathLength = len(strPath)

      iPos = Instr(strPath, "backofficestorage") + 18
      iPos = Instr(iPos,strPath, "/") + 1
      iPos = Instr(iPos,strPath, "/")

      iPathLength = iPathLength - iPos
      GetPathAndSubject = Right(strPath, iPathLength) & "/~" & strSubject

   End Function
```

### GetUserManager

To use this code, create the GetUserManager script on the Shared Script tab of the Workflow Process pane. This script is called in the ApproveExpenseReport, ExpireExpenseReport, and SubmitExpenseReport scripts.

```
   Function GetUserManager(strUserAddress)
      Dim mgrDN

      With WorkflowSession
         mgrDN = .GetUserProperty(strUserAddress, "manager",0)
         GetUserManager = .GetUserProperty(mgrDN, "mail", 1)
      End With

   End Function
```

### SendMessage

To use this code, create the SendMessage script on the Shared Script tab of the Workflow Process pane. This script is called in the ApproveExpenseReport, ExpireExpenseReport, and SubmitExpenseReport scripts to send an e-mail message.

The parameters used specify to whom the message should be sent, along with the subject and body of the e-mail message.

```
Sub SendMessage (strTo, strSubject, strBody)

   Set oMsg = CreateObject("CDO.Message")

   oMsg.To = strTo
   oMsg.From = WorkflowSession.Sender
   oMsg.Subject = strSubject
   oMsg.TextBody = strBody
   oMsg.Send

   Set oMsg = Nothing

End Sub
```

### SetCurrentMessageField

To use this code, create the SetCurrentMessageField script on the Shared Script tab of the Workflow Process pane. This script is called in the ApproveExpenseReport and SubmitExpenseReport scripts to reset the Submit property value when moving from Approved to Rejected. The parameters specify the name of the property to set and the value to set it to.

```
Sub SetCurrentMessageField(strName, Value)
   WorkflowSession.Fields(CStr(strName)).Value = Value
   WorkflowSession.Fields.Update
End Sub
```

### SetPermissions

To use this code, create the SetPermissions script on the Shared Script tab of the Workflow Process pane. This script is called in the SubmitExpenseReport script; in addition, you must call ClearPermissions prior to setting permissions. The parameter specifies what user (using the user's SMTP e-mail address) should have author access to the item.

```
Sub SetPermissions (strSMTPAddress)
   WorkflowSession.ItemAuthors.Add strSMTPAddress, 0
End Sub
```

### SubmitExpenseReport

To use this code, create the SubmitExpenseReport script on the Shared Script tab of the Workflow Process pane. This script sends the approval authority an e-mail message containing a link to the specific expense report item awaiting approval. The approval authority is determined after the total expense is evaluated. If the total expense is less than $100, the approval authority is the manager of the person who submitted the form. If the total is greater than $100, the approval authority is the manager's manager. In both instances, the GetUserManager script is called and returns an e-mail alias for the appropriate authority.

The SubmitExpenseReport is dependent on the AddFieldToCurrentMsg, GetUserManager (approval authority), GetPathAndSubject, SendMessage, ClearPermissions, SetPermissions, and SetCurrentMessageField scripts.

The ClearPermissions and SetPermissions scripts are required for the manager to be able to approve or reject the expense report.

```
Sub SubmitExpenseReport()
   Dim strSMTPAddress
   Dim strSubject
   Dim strBody
   Dim strOutlookURL

   strSubject = "Action Required: Expense Report Approval Request"

   strOutlookURL = GetPathAndSubject
   strBody = "There is an expense report waiting for your approval." _
      & vbCRLF & vbCRLF
   strBody = strBody & "View the expense report by clicking here. _
      <outlook://public folders/All public Folders/" _
      & strOutlookURL & ">" & vbCRLF & vbCRLF
   strBody = strBody & "----------------------------------------" _
      & vbCRLF
   strBody = strBody & "Submitted: " & CStr(Date) & "  " _
      & CStr(Time) & vbCRLF
   strBody = strBody & "Expense ending state: " _
      & WorkflowSession.StateTo

   strSMTPAddress = GetUserManager(WorkflowSession.Sender)
```

```
        If WorkflowSession.Fields("Total").Value > 100 Then
           strSMTPAddress = GetUserManager(strSMTPAddress)
        End If

        SendMessage strSMTPAddress, strSubject, strBody

        SetCurrentMessageField "ApprovalAuthority", strSMTPAddress
        ClearPermissions
        SetPermissions(strSMTPAddress)

    End Sub
```

# Activating the Workflow Process

After you have completed your workflow, you can implement it on the folder.

> **Note** Finish your work on the forms before activating the workflow. After the workflow is activated, you might find that you no longer have permissions to save your design changes to the folder.

### To activate the workflow on the folder

1. In the **General** tab of the Expense Report workflow process, select **Default Workflow Process for this Folder** and **Run as Privileged?.**
2. Select the folder you want to activate (the first item listed in the **Process List**).
3. In the **General** tab, choose **Activated**.
4. From the **File** menu, select **Save All Changes**.

If you want to use the audit trail to write entries to the Event Viewer for debugging, be sure to enable the success entries.

### To enable script debugging and success entries

1. In the Workflow Designer for Exchange, double-click the folder you want to use.
2. In the **General** tab, select **Enable Script Debugging for this Folder** and **Enable Success Entries for this Folder**.
3. From the **File** menu, select **Save All Changes**.

# Using the Workflow Process

Once you've completed the creation of the Outlook form and the workflow process, you can now test the functionality of your solution.

1. In Microsoft Outlook, select the **Expense Report** folder.
2. Click the **New** button to display the Expense Report form.

3. Enter a name for the Expense Report in the **Subject** field.
4. Enter in values for the Food and Airfare expense values. The total field will calculate the total automatically.
5. Select the **Submit** box to immediately submit your expense report to a manager for approval.

   An e-mail message is sent to your manager indicating that an expense report is waiting approval.

   > **Note** If an error message appears stating that you do you not have permission to save the item, make sure that you have manager information specified in the Exchange address book. If your e-mail alias does not have a manager identified, an error will occur when you try to send e-mail to the manager.

6. Edit the newly added Expense Report by double-clicking it. The Expense Report form now shows the **Approved?** combo box.
7. Select **Yes** from the **Approved?** combo box and select **Save** from the **File** menu.

An e-mail message is then sent to you informing you that your expense report has been approved.

## For More Information

For information about developing and managing team solutions, consult the Workflow Designer for Exchange 2000 Server Developer's Guide in the online documentation and the Microsoft Office Developer Web site at http://msdn.microsoft.com/officedev/.

For the latest information about Microsoft Workflow Designer for Exchange 2000 Server and the Exchange SDK, see the Exchange 2000 Web site at http://msdn.microsoft.com/exchange/.

To access Knowledge Base information, consult the Product Support section of the Microsoft Office Developer Web site.

For additional information about Microsoft Workflow Designer for Exchange 2000 Server, refer to http://msdn.microsoft.com/, and search for "workflow," or refer to Microsoft Office Developer Web Forum at http://search.microsoft.com/us/dev/apps/office/.

For addition information about scripting techniques, see http://msdn.microsoft.com/scripting/. The Microsoft Scripting Web site provides frequently asked questions, sample code, language documentation, and free downloads for people using the Microsoft® Visual Basic Scripting language (VBScript).