

UNIVERSITÉ ANTONINE
Faculté d'ingénieurs en Informatique,
Multimédia, Réseaux & Télécommunications



Using WorkFlow to create Absence Request

Matière : XML.net et Web Services

Effectué par :

NOM Prénom

INF#

MATTA Elie

Privacy

et al.

applied

Table of Contents

I.	Introduction	3
II.	Technical STUDY.....	4
1.	UML.....	4
1.1	Use Cases Diagram	4
1.2	Activity Diagram	6
1.3	Sequence Diagram.....	8
2.	Technology Used.....	9
3.	DataBase	10
4.	WorkFlow.....	12
5.	SharePoint Services 3.0.....	16
6.	Interaction between WorkFlow and SharePoint	19
III.	Implementation	23
1.	Creating Site using SharePoint.....	23
2.	Creating WF.....	29
2.1	Using MS Visual Studio 2008	29
2.2	Using MS SharePoint designer 2007	38
3.	Creating and Uploading a Document.....	41
4.	Association with SharePoint Site	48
5.	Assigning Mailing Task	50
6.	Designing Forms Using InfoPath 2007	50
7.	Web Solution.....	54
IV.	Backup and restore	58
V.	Conclusion.....	60
VI.	Annex	61
VII.	References	68

I. INTRODUCTION

SharePoint delivers office applications over the Web. An office application can be something as simple as a way to store and manage a library of documents within a small office, or as complex as a project management system used across continents.

SharePoint helps us:

- Find the information you need quickly
- Link to that information to stay current
- Share the information you have with others
- Do all that through standard tools that folks already know

SharePoint does those things by creating a web site for our business that integrates with Microsoft Office applications: mainly, Word, Excel, and Outlook. From a user's perspective, it's just like using the Internet: click on links to go to a new page, search on a phrase to find something, and so on.

What's unique is the way that SharePoint integrates Office documents, task lists, calendars, email alerts, and other features in a way to simplify the flow of work through our business. Instead of routing a document for approval via email, we post the document to SharePoint and collaborate with the reviewers interactively. Because the file is stored in a central location, everyone can see changes as they are made without resending the document each time it is changed; reviewers can discuss changes online, read one another's comments, and assign tasks and deadlines, and all changes are recorded in version history.

We will implement all of these benefits through an Absence Request application workflow where the employees can apply demands to request days off, and their chefs and the manager must approve or reject these demands.

II. TECHNICAL STUDY

As we all know, the phase of conception and modeling is the preparatory phase for any project to be done; and to prepare our project very precisely we used UML (Unified Modeling Language) to create many diagrams that helped us understanding the main functionalities of this program, and creating the units (modules) and the relationships between them.

The Unified Modeling Language (UML) is a language for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling and other non-software systems. The UML represents a collection of the best engineering practices that have proven successful in the modeling of large and complex systems.

1. UML

1.1 Use Cases Diagram

A use-case diagram is typically used to communicate the high-level functions of the system and the system's scope.

→ Actor:

An actor defines a coherent set of roles that users of an entity can play when interacting with the entity. An actor may be considered to play a separate role with regard to each use case with which it communicates.

→ Extend:

An extend relationship defines that instances of a use case may be augmented with some additional behavior defined in an extending use case.

→ Use Case:

Each use case specifies a sequence of actions, including variants that the entity can perform, interacting with actors of the entity.

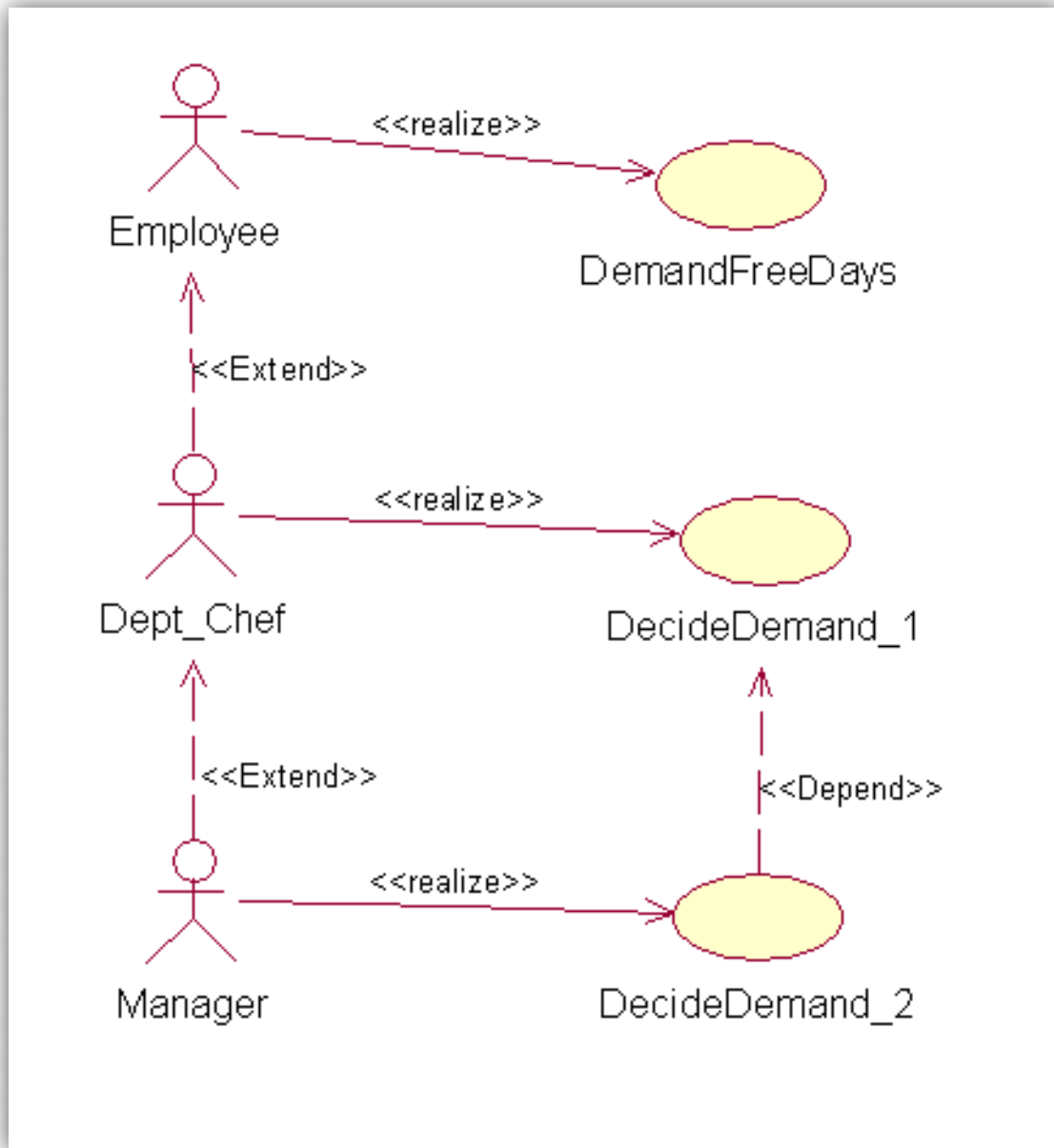


Figure 1 - Use Case Diagram

1.2 Activity Diagram

The purpose of the activity diagram is to model the procedural flow of actions that are part of a larger activity. Because it models procedural flow, the activity diagram focuses on the action sequence of execution and the conditions that trigger or guard those actions.

→ An action:

An action is indicated on the activity diagram by a "capsule" shape with semicircular left and right ends; the text inside it indicates the action.

→ Initial State:

Because activity diagrams show a sequence of actions, they must indicate the starting point of the sequence: *initial state*. The initial state clearly shows the starting point for the action sequence within an activity diagram.

→ Decision point:

Typically, decisions need to be made throughout an activity, depending on the outcome of a specific prior action; that's why we need to model two or more different sequences of actions. The way is to show a single transition line coming out of an action and connecting to a decision point: it is drawn as a diamond on an activity diagram. Since a decision will have at least two different outcomes, the decision symbol will have multiple transition lines connecting to different actions.

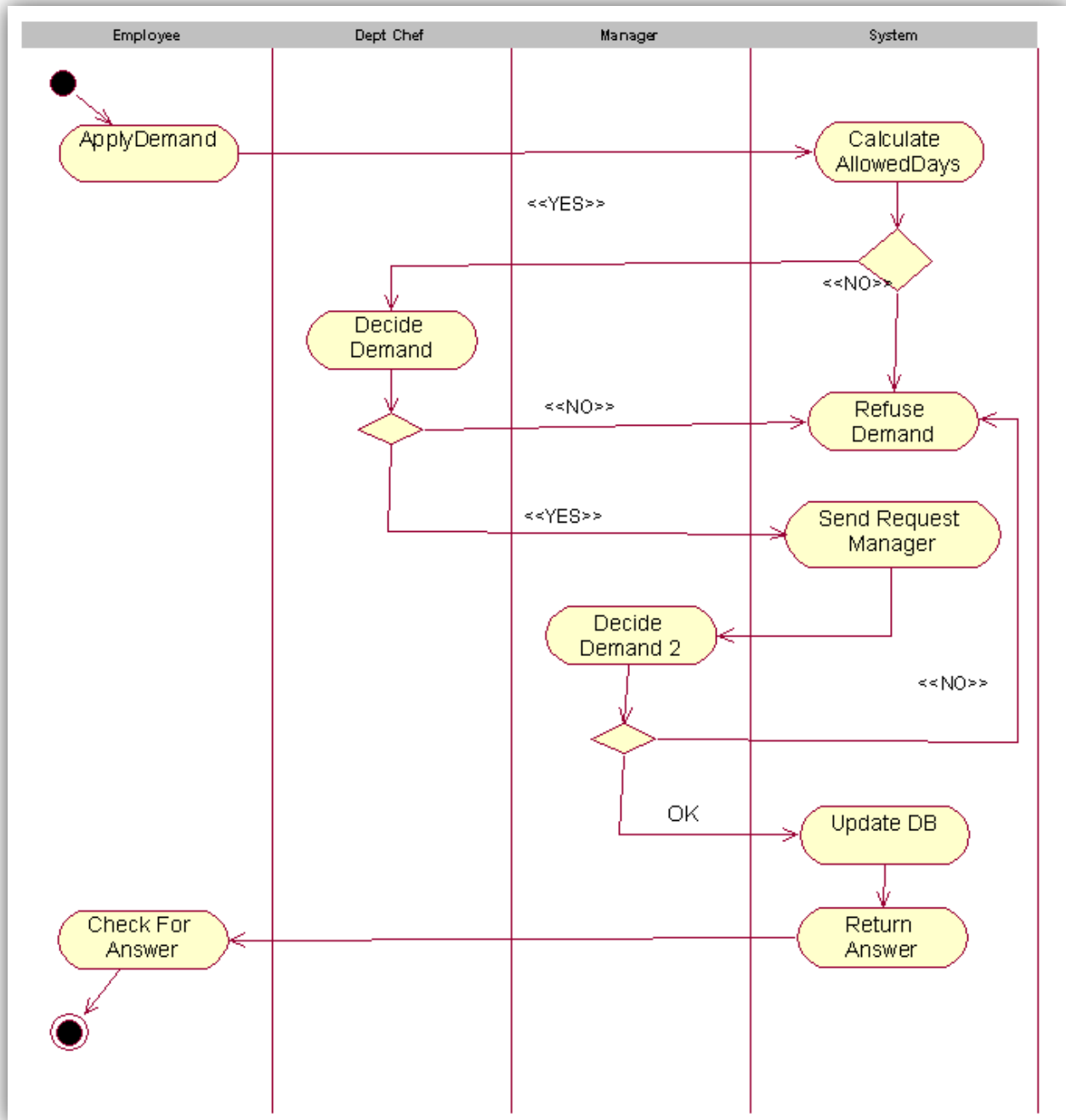


Figure 2 - Activity Diagram

1.3 Sequence Diagram

Sequence diagrams show object interactions arranged in a time sequence. We can use the flow of events to determine what objects and interactions we will need to accomplish the functionality specified by the flow of events. A sequence diagram has two dimensions: The vertical dimension shows the sequence of messages/calls in the time order that they occur; the horizontal dimension shows the object instances to which the messages are sent.

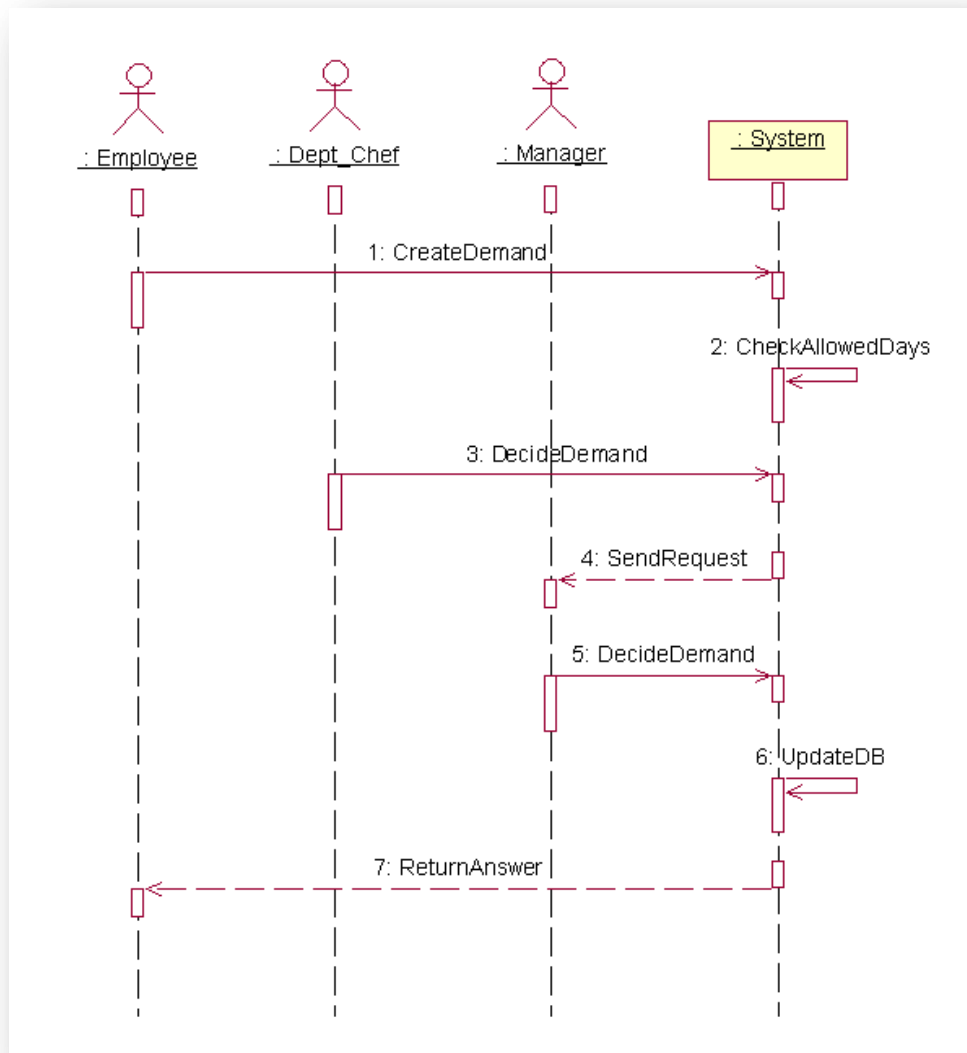


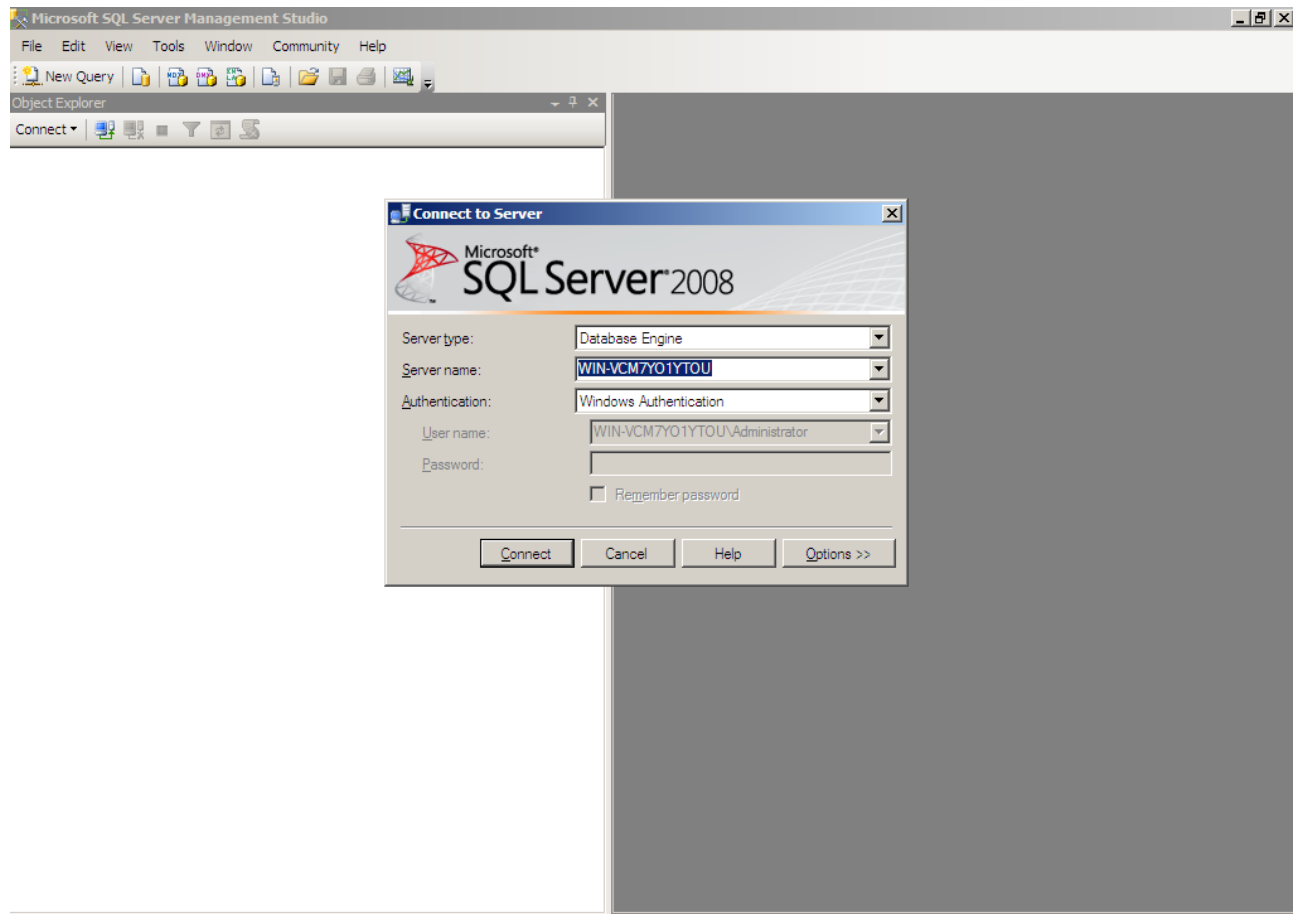
Figure 3 - Sequence Diagram

2. *Technology Used*

- Windows Server 2008
- Visual Studio 2008 + .net Framework 3.5
- Microsoft Office SharePoint Designer 2007
- SharePoint Services 3.0
- SQL Server 2008
- Microsoft Office InfoPath 2007

3. DataBase

Our database was created using MS SQL Server 2008:







Under the name “WSDB”, the database is composed of three tables:










➤ Employee

- 🔑 emp_id (PK, nvarchar(50), not null)
- 📄 emp_name (nvarchar(50), not null)
- 📄 emp_sal (nvarchar(50), not null)
- 📄 emp_dept (nvarchar(50), null)
- 📄 emp_man (nvarchar(50), null)
- 📄 emp_allowed (nvarchar(50), not null)
- 📄 emp_hire_date (datetime, not null)
- 📄 emp_add (nvarchar(50), null)
- 📄 emp_phone (nvarchar(50), null)
- 📄 emp_photo (image, null)
- 📄 emp_usr (nvarchar(50), null)
- 📄 emp_pass (nvarchar(50), null)

➤ Department

-  dept_id (PK, nvarchar(50), not null)
-  dept_name (nvarchar(50), not null)
-  dept_location (nvarchar(50), not null)
-  dept_chef (nvarchar(50), not null)

➤ Demand

-  dd_id (PK, nvarchar(50), not null)
-  dd_emp (nvarchar(50), not null)
-  dd_date (datetime, not null)
-  dd_from (datetime, not null)
-  dd_chef_acc (bit, null)
-  dd_man_acc (bit, null)
-  dd_taken (bit, null)
-  dd_desc (nvarchar(max), null)
-  dd_to (datetime, null)

4. Workflow

Workflows help people to collaborate on documents and to manage project tasks by implementing specific business processes on documents and items in a Microsoft Windows SharePoint Services 3.0 site.

Workflow is sometimes described as a series of tasks that produce an outcome. In the context of Microsoft SharePoint Products and Technologies, workflow is defined more narrowly as the automated movement of documents or items through a specific sequence of actions or tasks that are related to a business process. Workflows can be used to consistently manage common business processes within an organization by enabling organizations to attach business logic to documents or items in a SharePoint list or library. Business logic is basically a set of instructions that specifies and controls actions that happen to a document or item.

Workflows can streamline the cost and time required to coordinate common business processes, such as project approval or document review, by managing and tracking the human tasks involved with these processes. For example, by using Windows SharePoint Services 3.0, an organization can create and deploy a basic custom workflow to manage the approval process for drafts of documents in a library. The workflow can route a document to a specified person or a group of people for their review and approval. The workflow can then take specific actions on the document based on the outcome of the workflow. If the document is approved, its status can be updated from **Draft** to **Final**, and the document can be automatically copied to another document library. If a document is rejected, its status can remain as **Draft** and no further actions occur.

When this approval workflow starts, it can create document approval tasks, assign these tasks to the specified workflow participants, and then send e-mail alerts to the participants with task instructions and a link to the document to be approved. While the workflow is in progress, the workflow owner (in this case, the document author) or the workflow participants can check the Workflow Status page to see which participants have completed their workflow tasks. When the workflow participants complete their workflow tasks by approving or rejecting the

document, the workflow ends. The workflow automatically takes the appropriate actions on the document, and it alerts the workflow owner about the outcome of the workflow.

➤ **WF types:**

Windows Workflow Foundation supports two fundamental workflow styles:

- **Sequential workflows** Represents a workflow as a procession of steps that execute in order until the last activity completes. However, sequential workflows are not purely sequential in their execution. Because they can receive external events and include parallel logic flows, the exact order of activity execution can vary.

Sequential workflows can best be represented graphically as a flowchart of actions, with a beginning, an end, and a sequential flow direction from start to finish. Sequential workflows can incorporate flow structures such as repetition, looping, and parallel branches, but ultimately progress from the initial action to the final action.

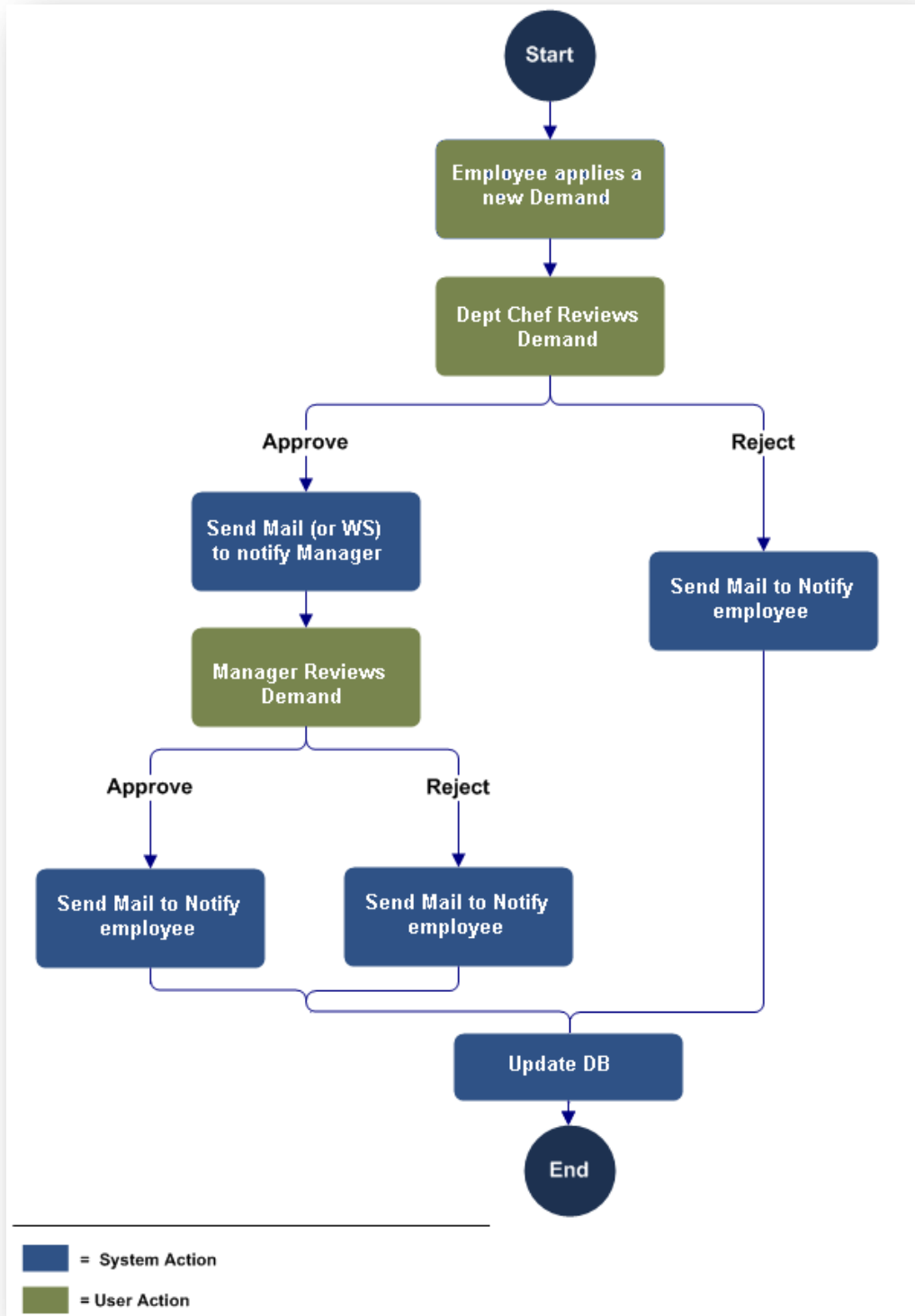


Figure 4 - Conceptual Diagram of our Sequential WF

- **State machine workflows** Represents a set of states, transitions, and actions. One state is denoted as the start state, and then, based on an event, a transition can be made to another state. The state machine can have a final state that determines the end of the workflow.

Unlike sequential workflows, state machine workflows do not have a prescribed execution flow, and need not have an end. Instead, state machine workflows define any number of states which an item may inhabit, and the events that transition the item from one state to another.

The state machine workflow is composed of state activities. Each state activity represents a state for the item. Each state activity can contain optional state initialization, state finalization, and one or more event handlers. Each event handler activity can handle one event. In response to the event handled, some processing can be done, and a transition can be made to another state.

5. *SharePoint Services 3.0*

WSS (Windows SharePoint Services) is an object model for creating web pages and developing web based collaboration, document management, and content publishing.

- At this time WSS is in its third release from Microsoft: WSS 3 is the framework that underlies Microsoft Office SharePoint Server 2007.
- The SharePoint Object Model provided by WSS offers objects to support collaboration and document management functionality, centralized repository for shared documents. WSS also provides support for blogs and wikis. WSS supports browser-based management and administration.
- WSS allows web based document collaboration can be shared for collaborative editing. SharePoint provides access control and revision control for documents in a library.
- Installation of WSS on a server makes available a collection of web parts that can be embedded into web pages to provide certain functionality. SharePoint includes web parts such as workspaces and dashboards, navigation tools, lists, alerts (including e-mail alerts), shared calendar, contact lists and discussion boards.
- SharePoint serves content via IIS Web Sites. These use SQL Server technology to store content.
- SharePoint uses a similar LDAP permissions model to Microsoft Windows, via groups of users. This can be provided via Active Directory. Alternatively, other authentication providers can be added through or even HTML Forms authentication.

The WorkFlow (WF) runtime engine can be hosted in any Windows process. Windows SharePoint Services 3.0 takes advantage of this, acting as a host for this engine. One or more workflow templates, each containing the code that defines a particular workflow, can be installed on a server. Once this is done, an association can be created between a specific template and a document library, list, or content type. This template can then be loaded and

executed by the Windows SharePoint Services-hosted WF runtime engine, creating a workflow instance. The figure below shows how this looks.

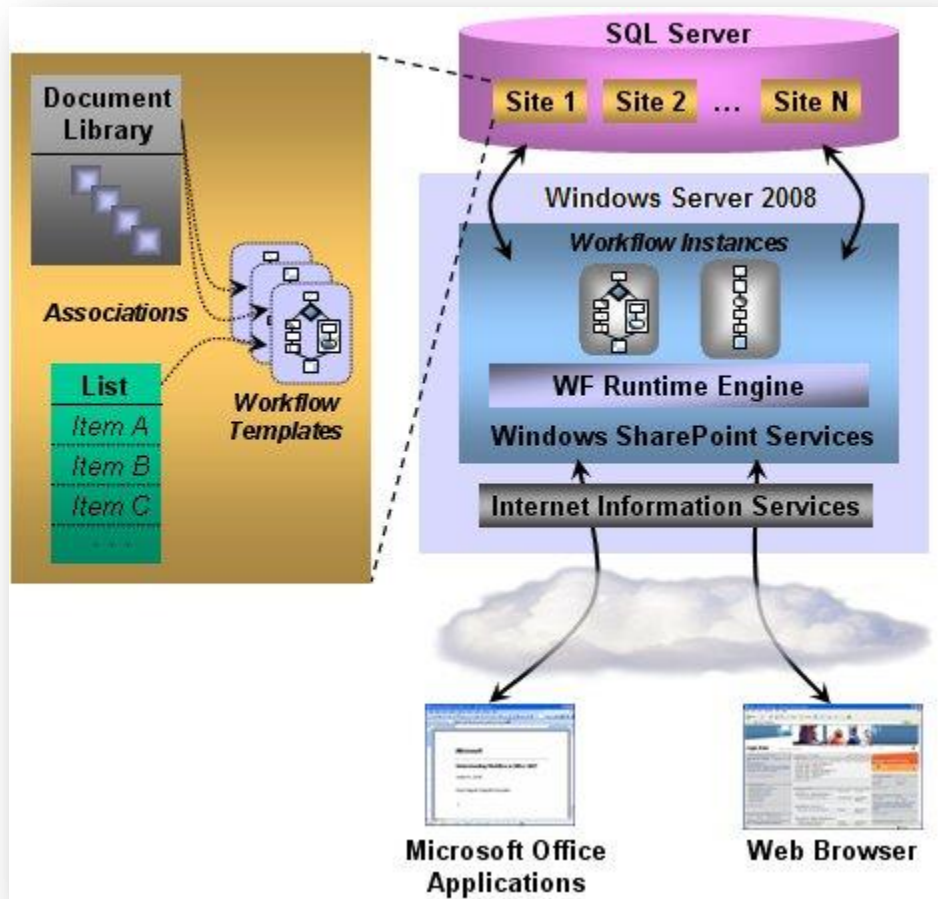


Figure 5 - WF Engine

Like all WF workflows, those based on Windows SharePoint Services 3.0 rely on WF's runtime services. To better support workflows hosted in Windows SharePoint Services, however, Version 3.0 replaces some of those built-in services. WF's standard persistence service has been modified, for example, to allow the state of a persisted workflow to be linked with the document or item with which that workflow is associated.

➤ **Top 10 Benefits of using Windows SharePoint Services:**

- *Improve team productivity with easy-to-use collaborative tools*
- *Easily manage documents and help ensure integrity of content*
- *Get users up to speed quickly*
- *Deploy solutions tailored to our business processes*
- *Build a collaboration environment quickly and easily*
- *Reduce the complexity of securing business information*
- *Provide sophisticated controls for securing company resources*
- *Take file sharing to a new level with robust storage capabilities*
- *Easily scale our collaboration solution to meet business needs*
- *Provide a cost-effective foundation for building Web-based applications*

6. *Interaction between WorkFlow and SharePoint*

The workflow functionality in Windows SharePoint Services 3.0 is built on the Windows Workflow Foundation (WF), a Microsoft Windows platform component that provides a programming infrastructure and tools for development and execution of workflow-based applications. WF simplifies the process of asynchronous programming to create long-running and persistent workflow applications. The WF run-time engine manages workflow execution and allows workflows to remain active for long periods of time and to survive restarting the computer. Run-time services offer functionality such as transactions and persistence to manage errors gracefully and correctly.

The WF run-time engine provides the services that every workflow application needs, such as sequencing, state management, tracking capabilities, and transaction support. The WF run-time engine serves as a state machine responsible for loading and unloading workflows, as well as managing the current state of any workflows that are running. WF allows any application process or service container to run workflows by hosting WF — that is, loading WF within its process.

Windows SharePoint Services hosts the WF run-time engine. In place of the pluggable services that are included with WF, Windows SharePoint Services provides custom implementations of the following services for the engine: transaction, persistence, notifications, roles, tracking, and messaging. Developers can then create workflow solutions that run within Windows SharePoint Services.

This figure shows the workflow architecture in Windows SharePoint Services. Windows SharePoint Services hosts the WF run-time engine within its process, and provides custom implementations of the necessary services. The functionality of the WF run-time engine, as well as the hosting functionality Windows SharePoint Services provides, is exposed through the Windows SharePoint Services object model.

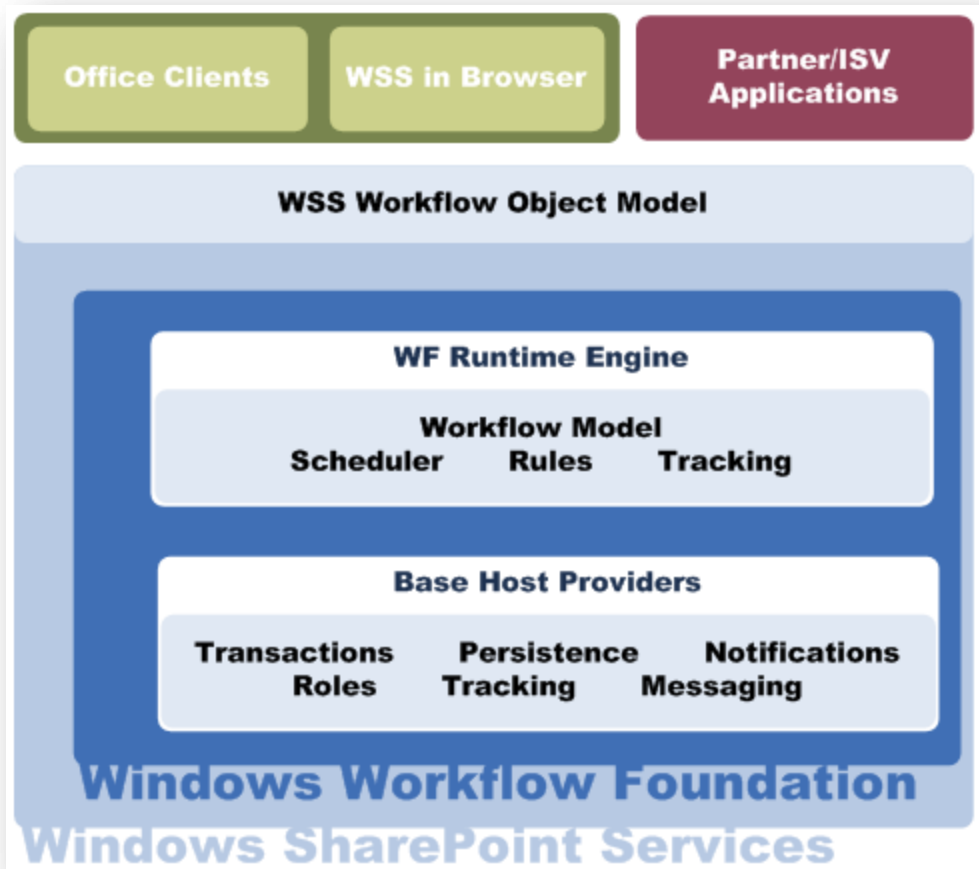


Figure 6 - WF architecture in Windows SharePoint Services 3.0

➤ **Using Forms to Enable User Interaction with Windows SharePoint Services**

Although we can use Windows SharePoint Services workflows to model any number of unique business processes, there are several common stages at which user can interact with the workflow template or instance.

For the user to pass information to the workflow, the developer must provide an interface for that interaction through the use of a form. Adding forms to workflows enables us to make our workflows more dynamic and flexible; gather information from users at predefined times in the life of a workflow, as well as let users interact with the tasks for that workflow.

Technically, we can employ any forms technology with WF workflows, as long as our forms are capable of doing the following:

Préparé par Elie Matta et al.

- Invoking the Windows SharePoint Services object model
- Generating the data necessary to send to the Windows SharePoint Services object model
- Receiving and parsing the required data from the Windows SharePoint Services object model

Whatever forms technology is used, the conceptual operation of each form is the same. Windows SharePoint Services displays the form to the user at the appropriate juncture in the workflow. The user enters information, and submits the form.

As workflow developer, we are responsible for programming what happens when the user submits a form. In most cases, our form calls into the Windows SharePoint Services object model and invokes the appropriate method to pass the form information to the correct workflow instance.

The information string can be in any format, provided that both the form and the workflow activity are programmed to parse the string. In most cases, developers will likely choose to use XML.

Windows SharePoint Services passes the string information to the WF run-time engine, the workflow instance, and ultimately the correct activity within the workflow instance. The activity receiving the information can then respond as programmed. The following figure shows how a form integrates with workflow.

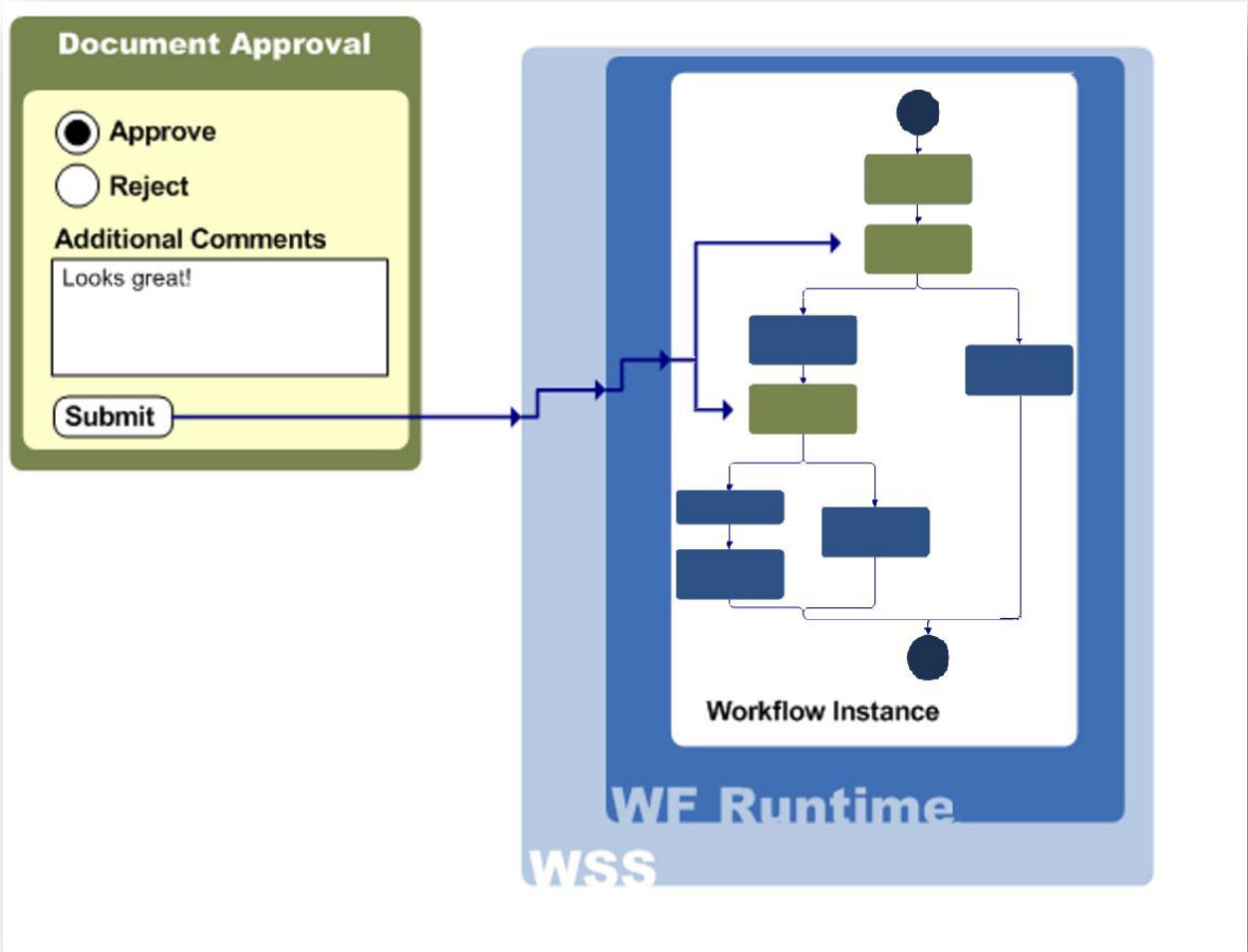


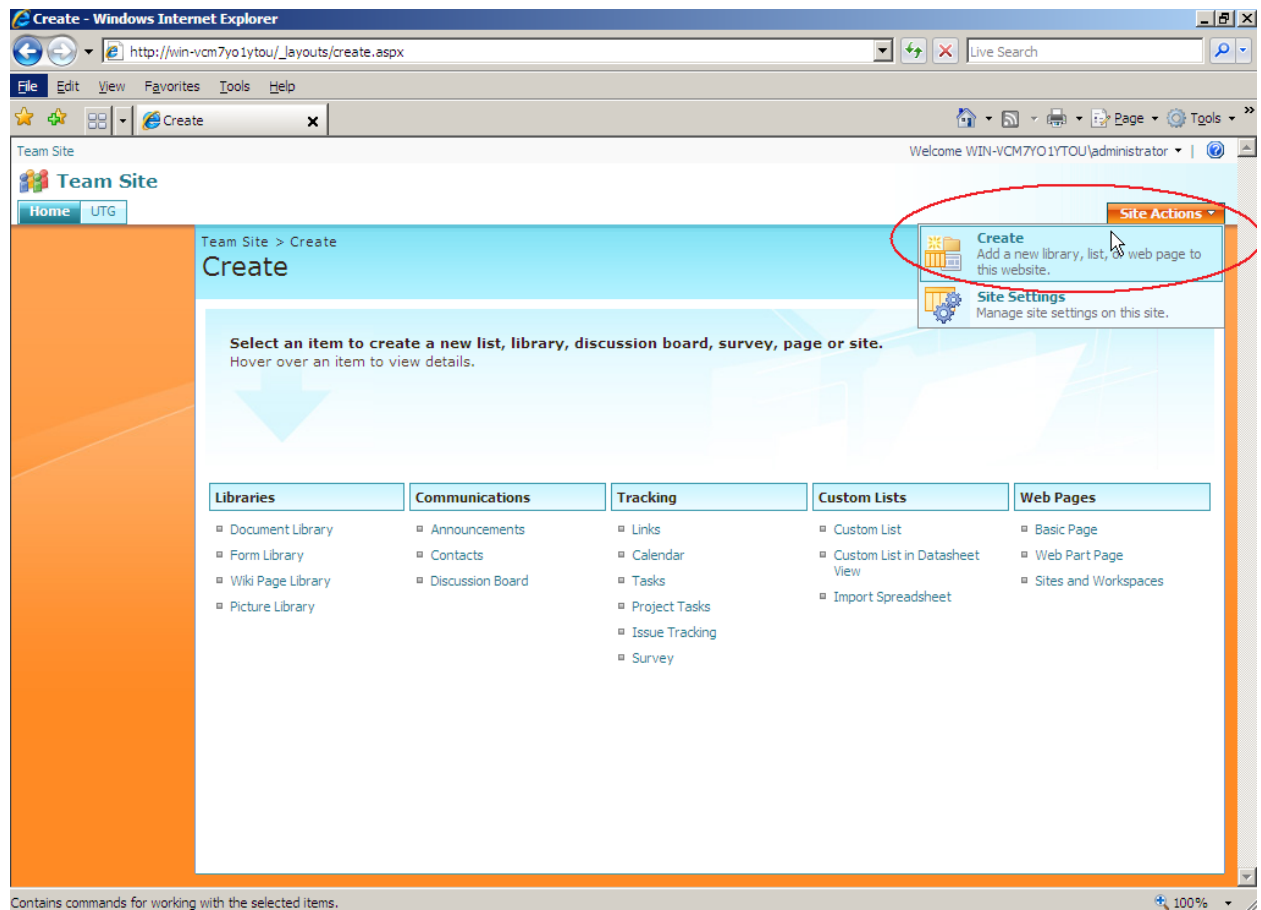
Figure 7 - Form Integration with WF

III. IMPLEMENTATION

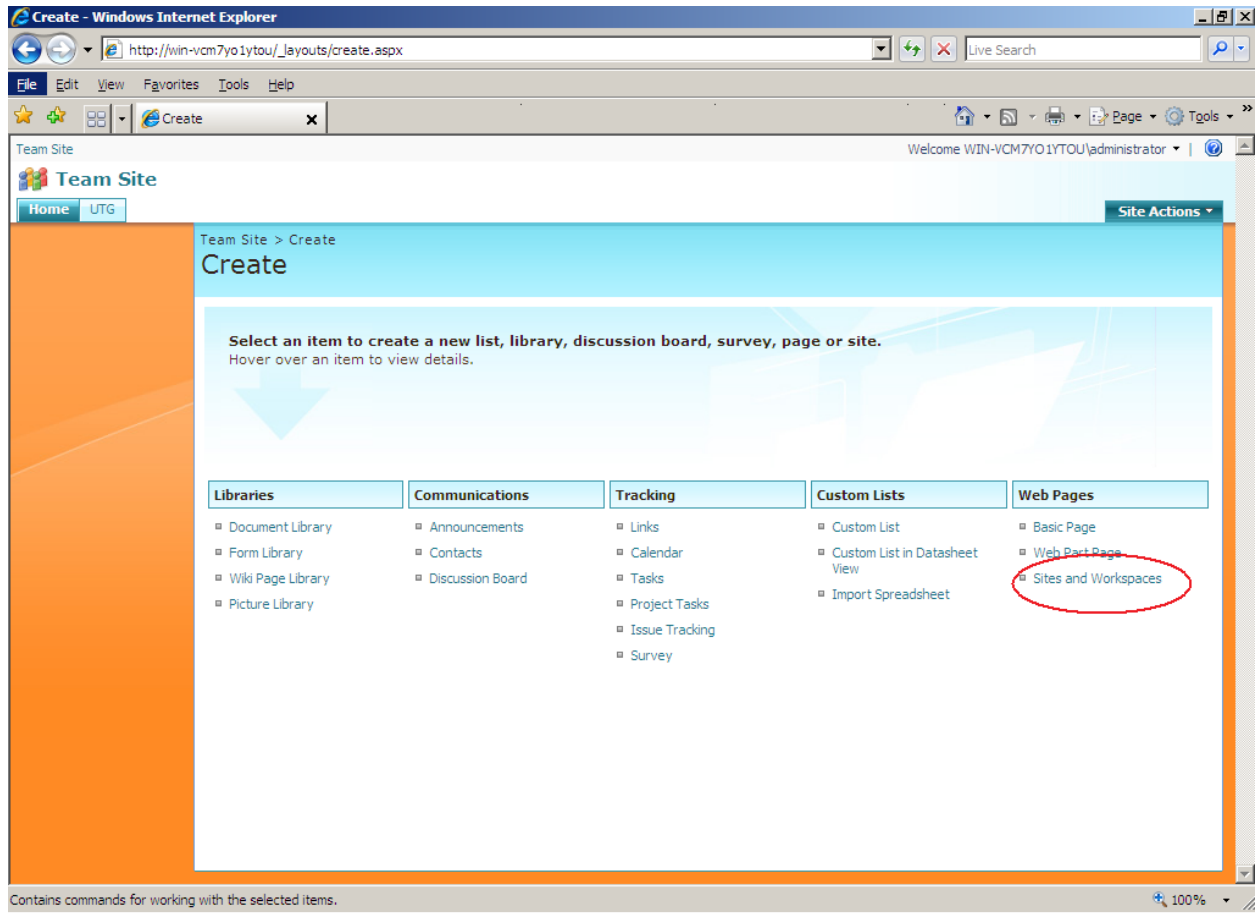
1. *Creating Site using SharePoint*

To create our own site using SharePoint we followed these steps:

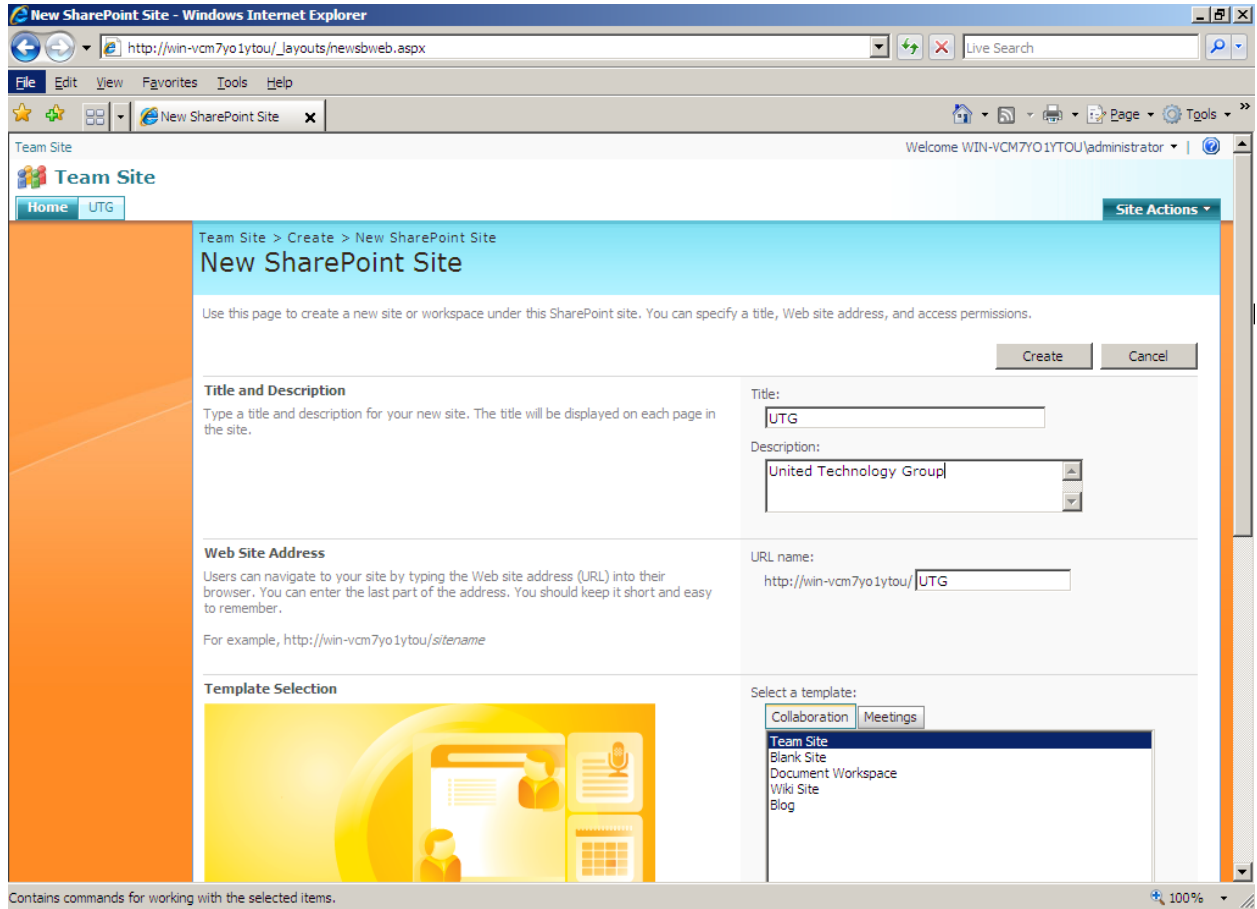
- Click on the Site Actions Menu on the right, and then click create



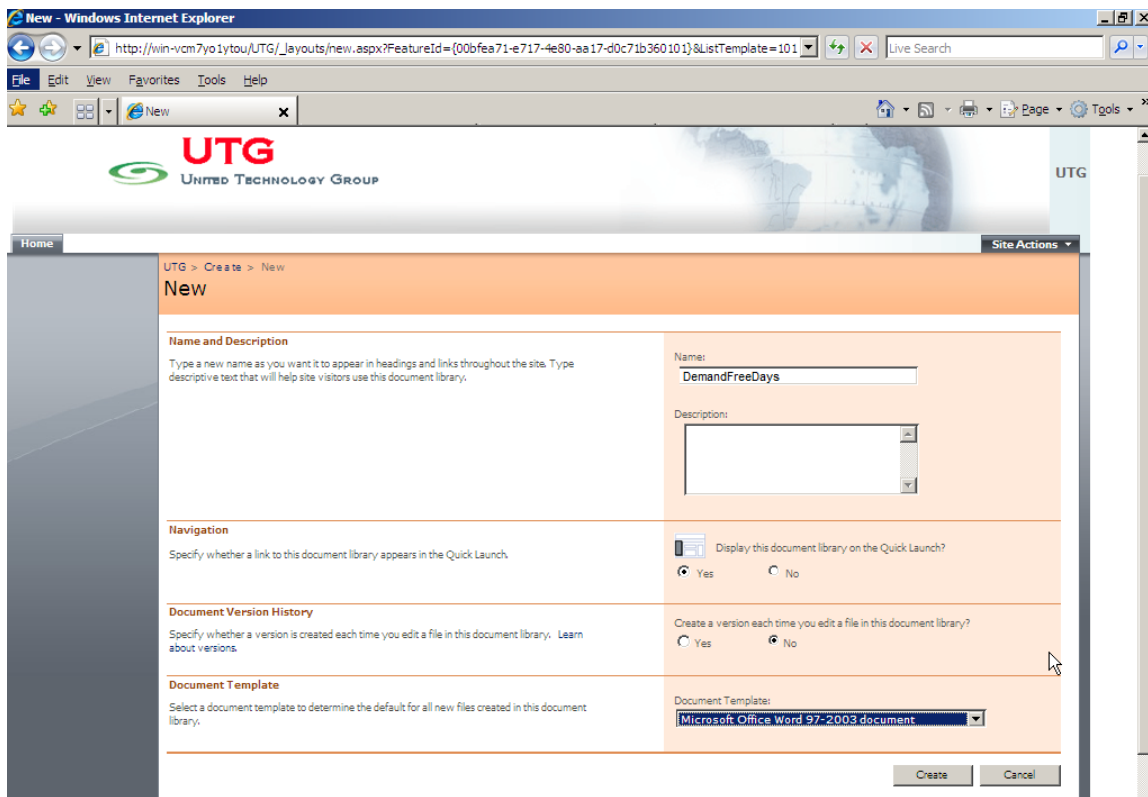
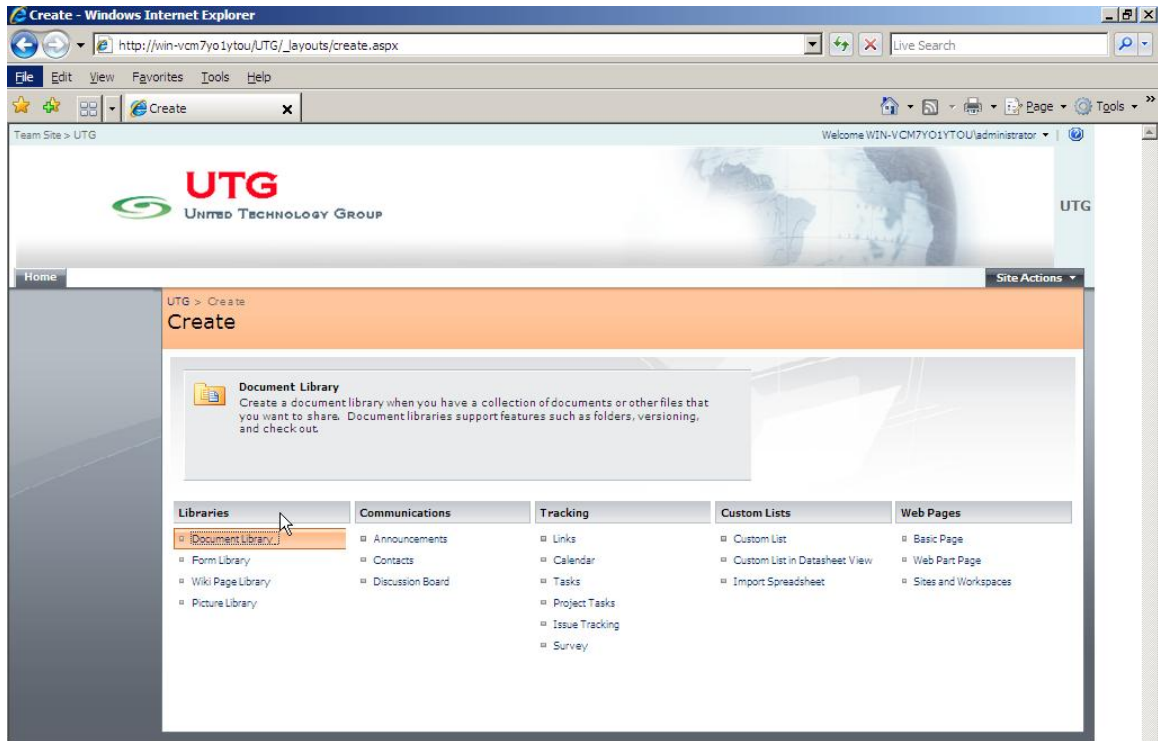
➤ Choose “Sites and Workspaces”



- Fill the site properties and then click “create”

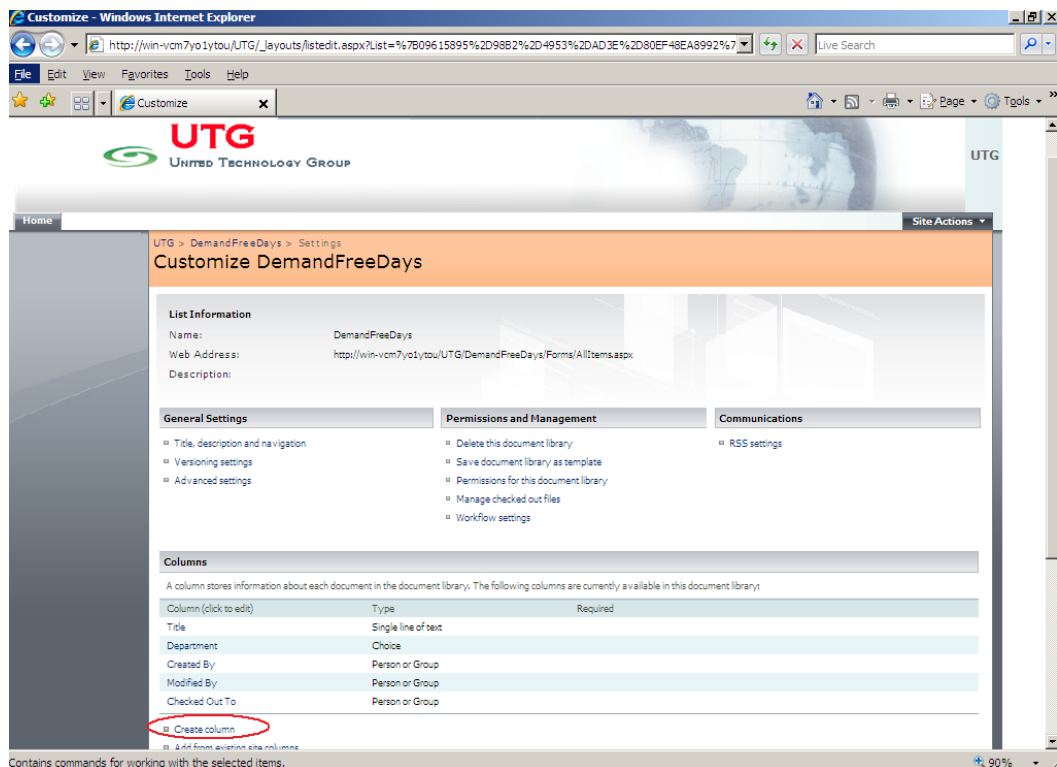
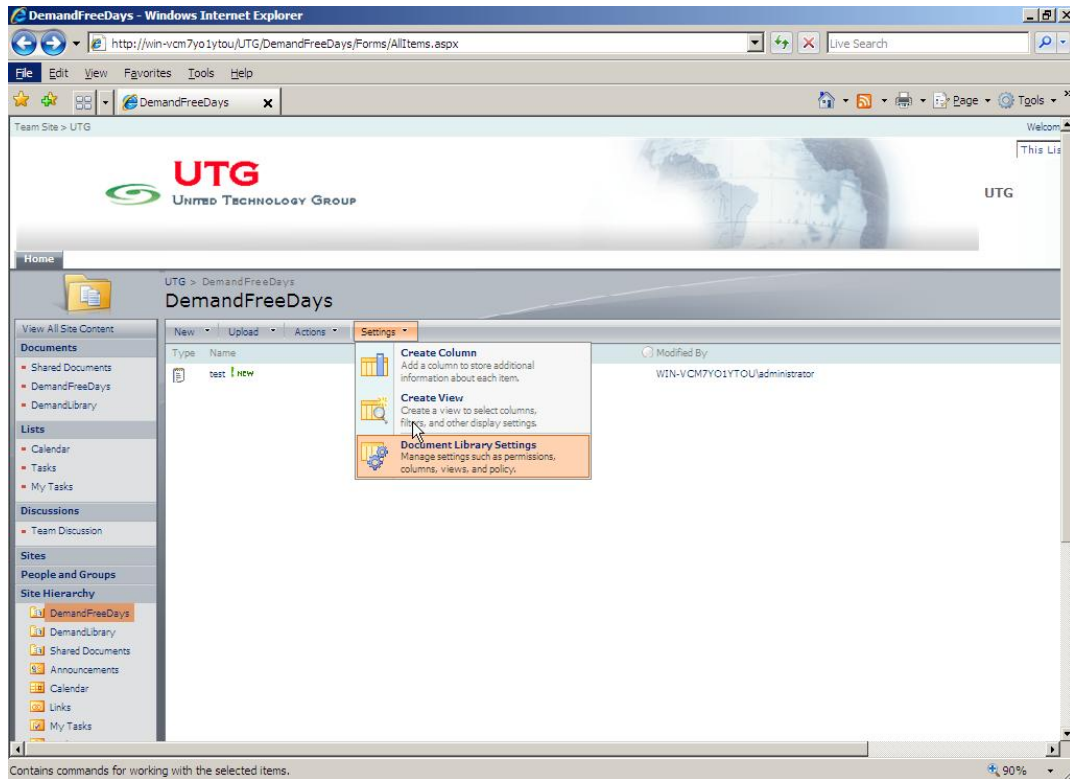


- In our SharePoint we created a Document Library which is easily achieved by choosing “Create” from the “Site Actions” menu (located in the top right corner) and then choosing “Document Library”, we gave it a name and clicked “Create”.

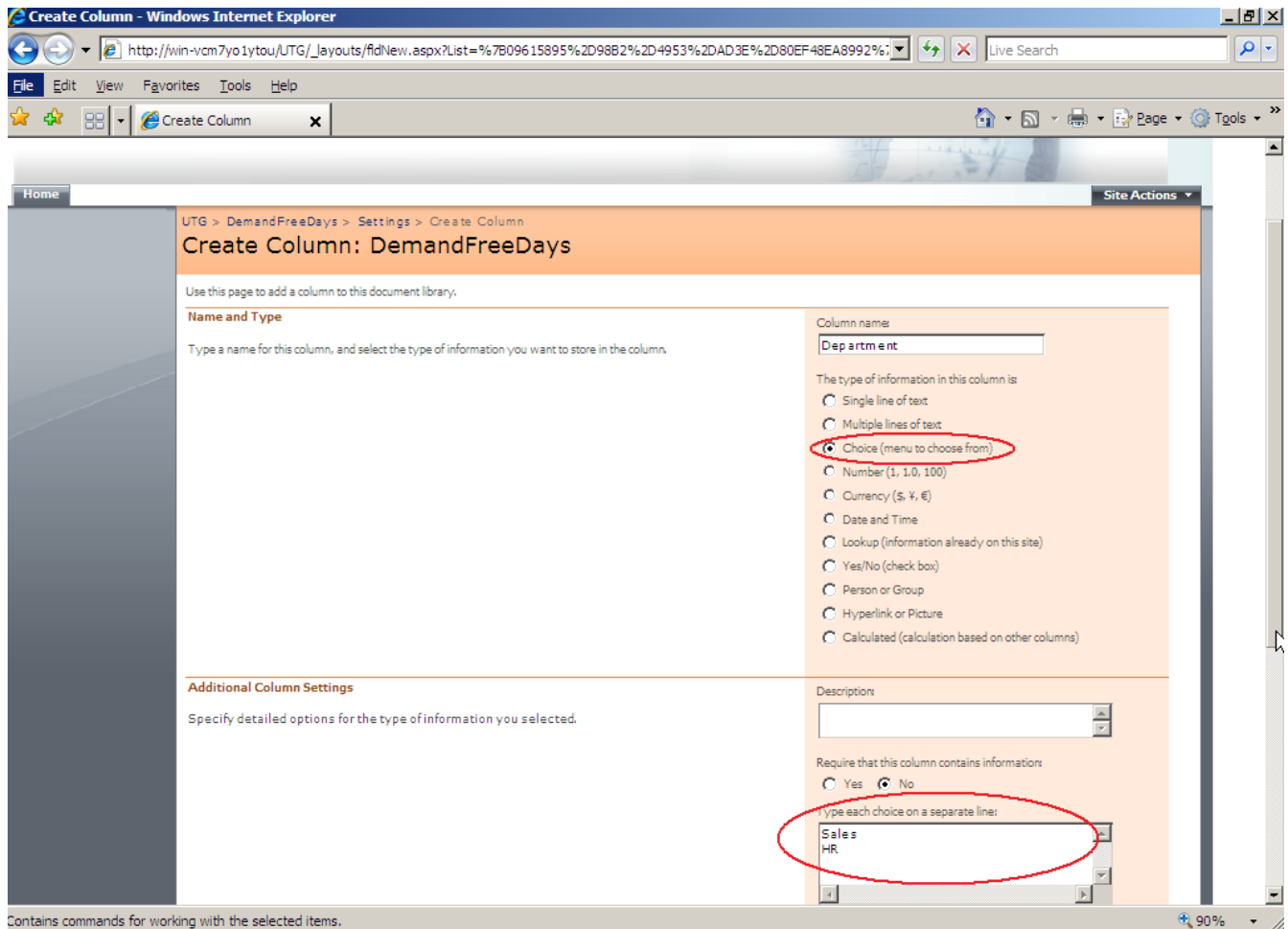


Préparé par Elie Matta et al.

- In our Document Library, we clicked the Settings dropdown and selected Document Library Settings.
- On the Customize Documents page, located the Columns section.



- We created a new column named “Department” which represents the Department of the Employee, we chose the type of the column as “Choice” and we filled manually the names of the departments (“Sales”, “HR”, “Marketing” ...)

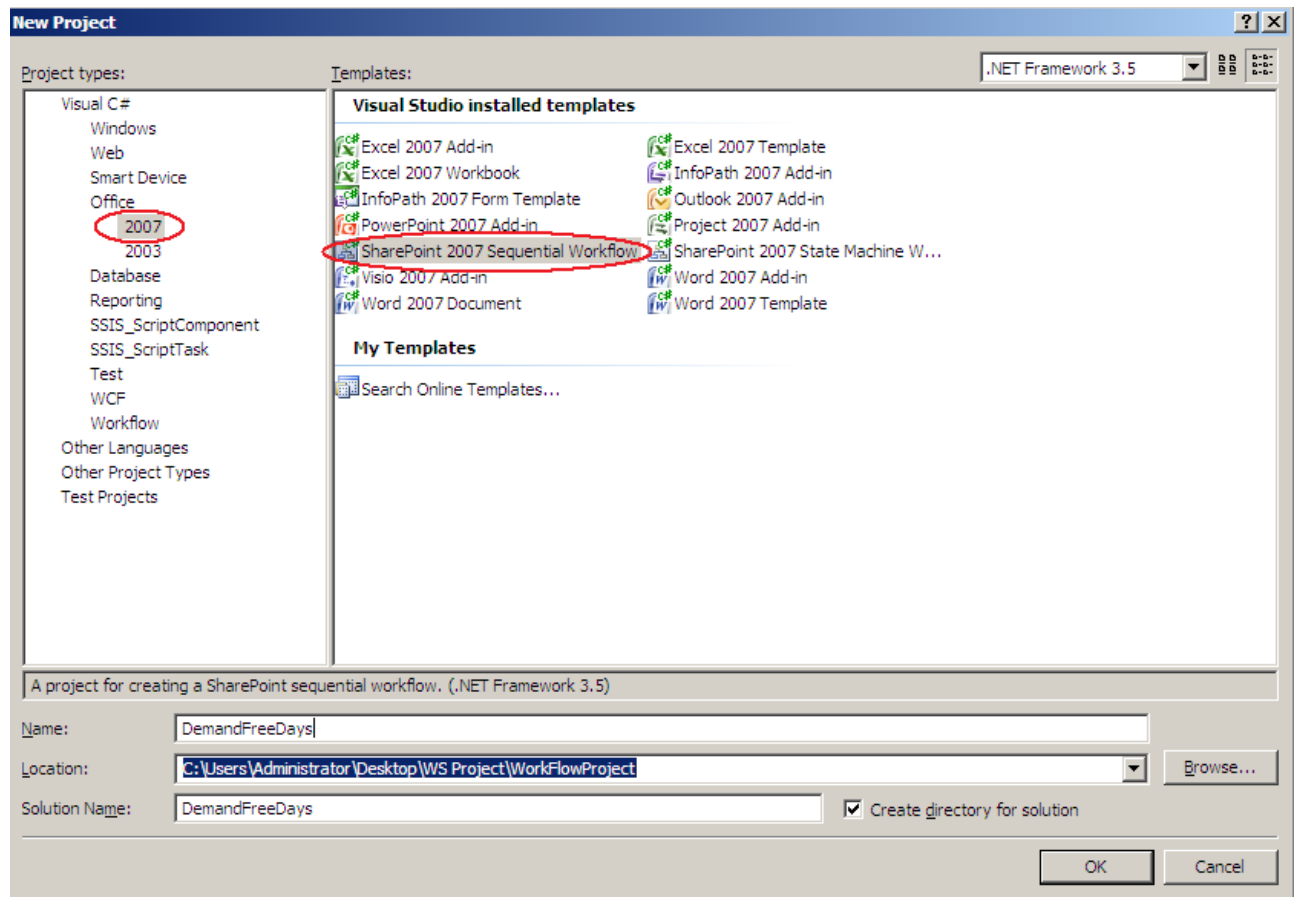


- In the same way as we created the Column Department (concerning a certain employee), we created also the following columns:
 - Employee ID (Single Line of Text)
 - EmployeeName (Single Line of Text)
 - HireDate (Date and Time)
 - AllowedDays (Number [Max=15])

2. Creating WF


2.1 Using MS Visual Studio 2008

- Start Visual Studio.
- In the New Project dialog, in the list of Project Types, by expanding Visual C# we expand Office and then select 2007.
- Select the SharePoint Sequential Workflow project type.
- We named the project “DemandFreeDays”, specified the default location and clicked “OK”.



- The New SharePoint Workflow wizard appears. We used your local site (UTG) for debugging and clicked “Next”.
- A new Dialog Appears, where we chose the list created formerly in the SharePoint as the list to associate with our WF.

New Office SharePoint Workflow [?] [X]

 **Specify the workflow name and site for debugging**

What is the name of the workflow?
Users will identify the workflow by the name you provide.


DemandFreeDays

What local site do you want to use for debugging?
Identify a valid SharePoint Web URL where you want to debug the workflow.

http://win-vcn7yo1ytou/UTG

< Previous Next > Finish Cancel

New Office SharePoint Workflow [?] [X]

 **Select the lists you will use when debugging**

In a debug session you can choose to allow Visual Studio to automatically associate the workflow with a library or list. Visual Studio can also handle the history and task list designations for you.

Automatically associate workflow?
Choose whether you want to handle the association step manually or have Visual Studio do this for you.

Which lists will you use?
This is the library or list to associate with your workflow.

Library or List: DemandFreeDays

The history list displays all of the events that occur while the workflow is running.

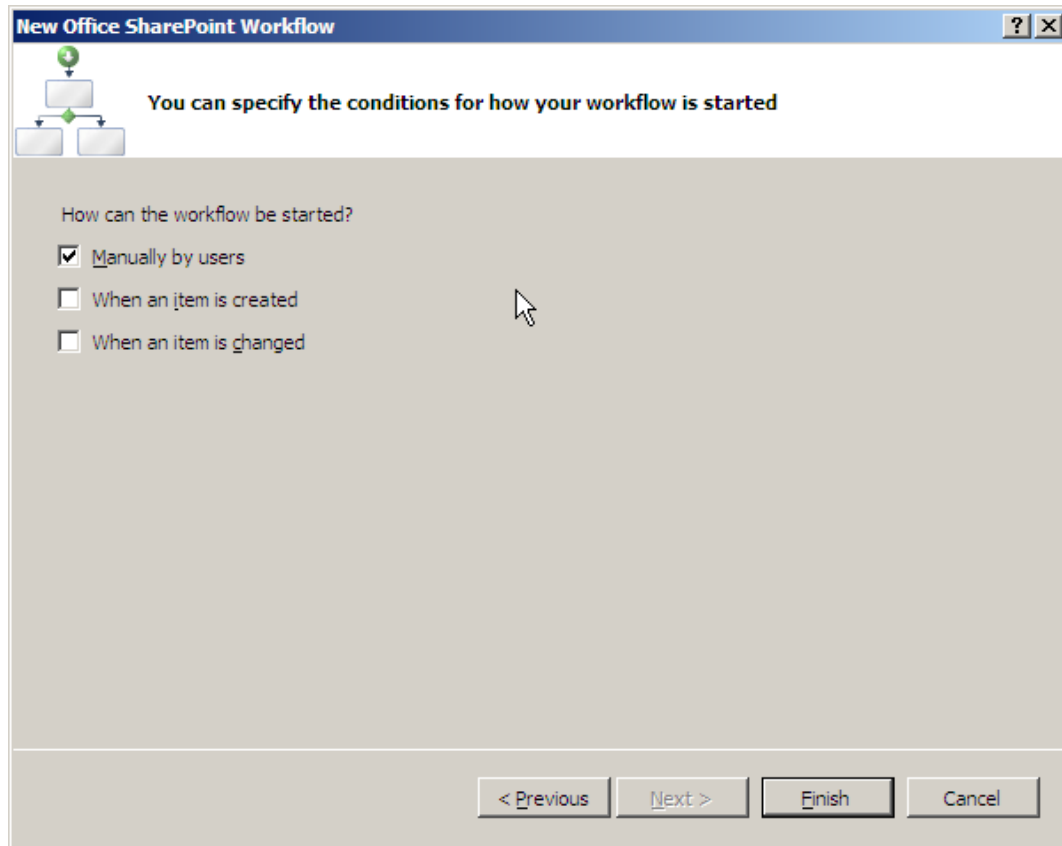
History list: Workflow History

The task list displays the workflow tasks available to each workflow participant.

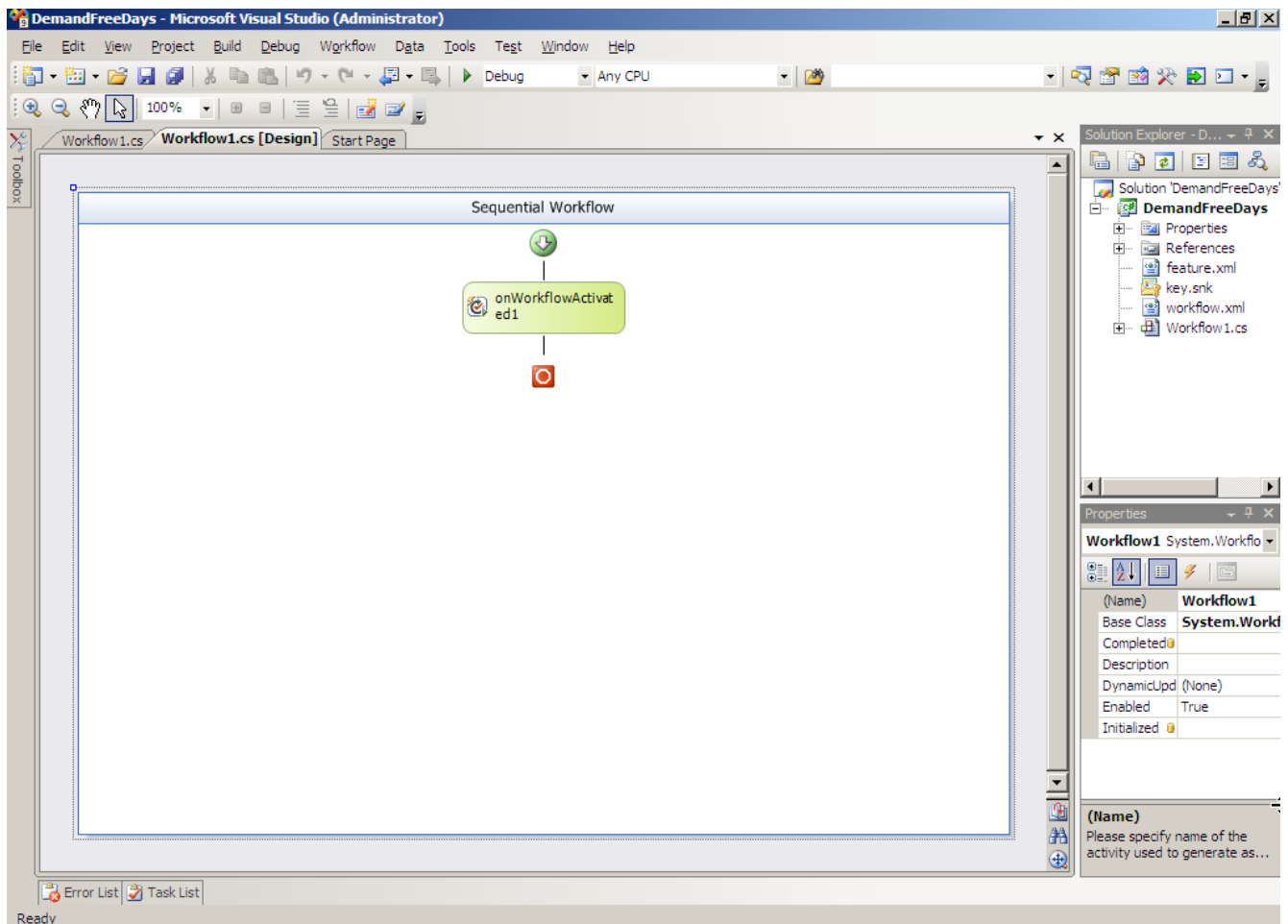
Task list: Tasks

< Previous Next > Finish Cancel

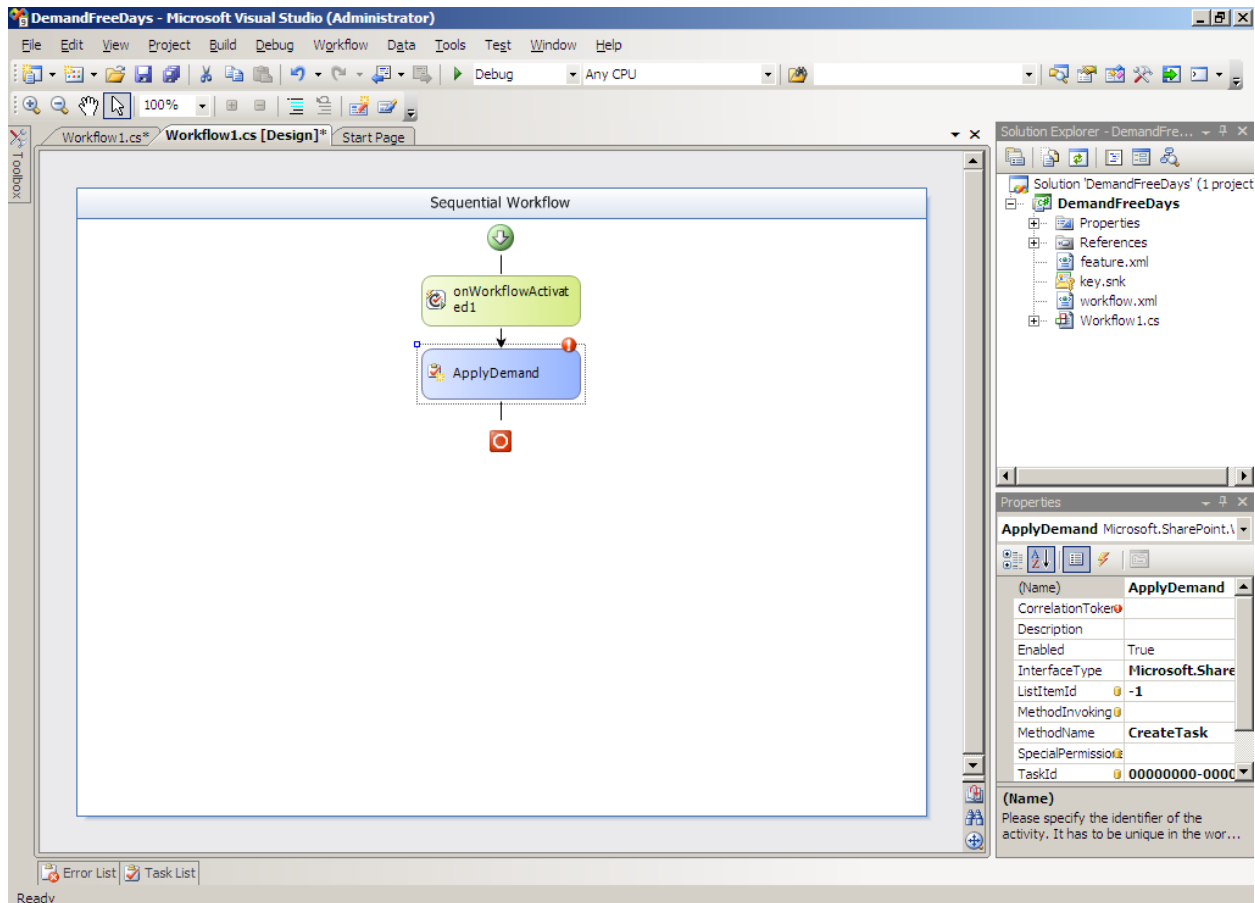
- The last screen describes how the workflow will be started: we chose to start it manually by users so we can see later how.



- After clicking “Finish” the project is created and the new workflow appears in the designer



- From the control toolbox, we drag a CreateTask activity and drop it onto the workflow designer between onWorkflowActivated1 and the end of the workflow (). The new activity is named createTask1 by default; we renamed it to “ApplyDemand”.



We Configured the TaskProperties property of “ApplyDemand”:

- In the Properties window, click the ellipsis for the value of the TaskProperties property. The Bind “TaskProperties” to an activity’s property dialog appears.
- Type the new member name “ApplyDemandProperties”, ensure that the Create property radio button is selected and click OK.
- In the properties window, select the MethodInvoking handler. Type “ApplyDemandCreation” and press the Enter key. The code window appears.

- We added the following code to the “ApplyDemandCreation” event handler and also add the CustomFieldValue function.

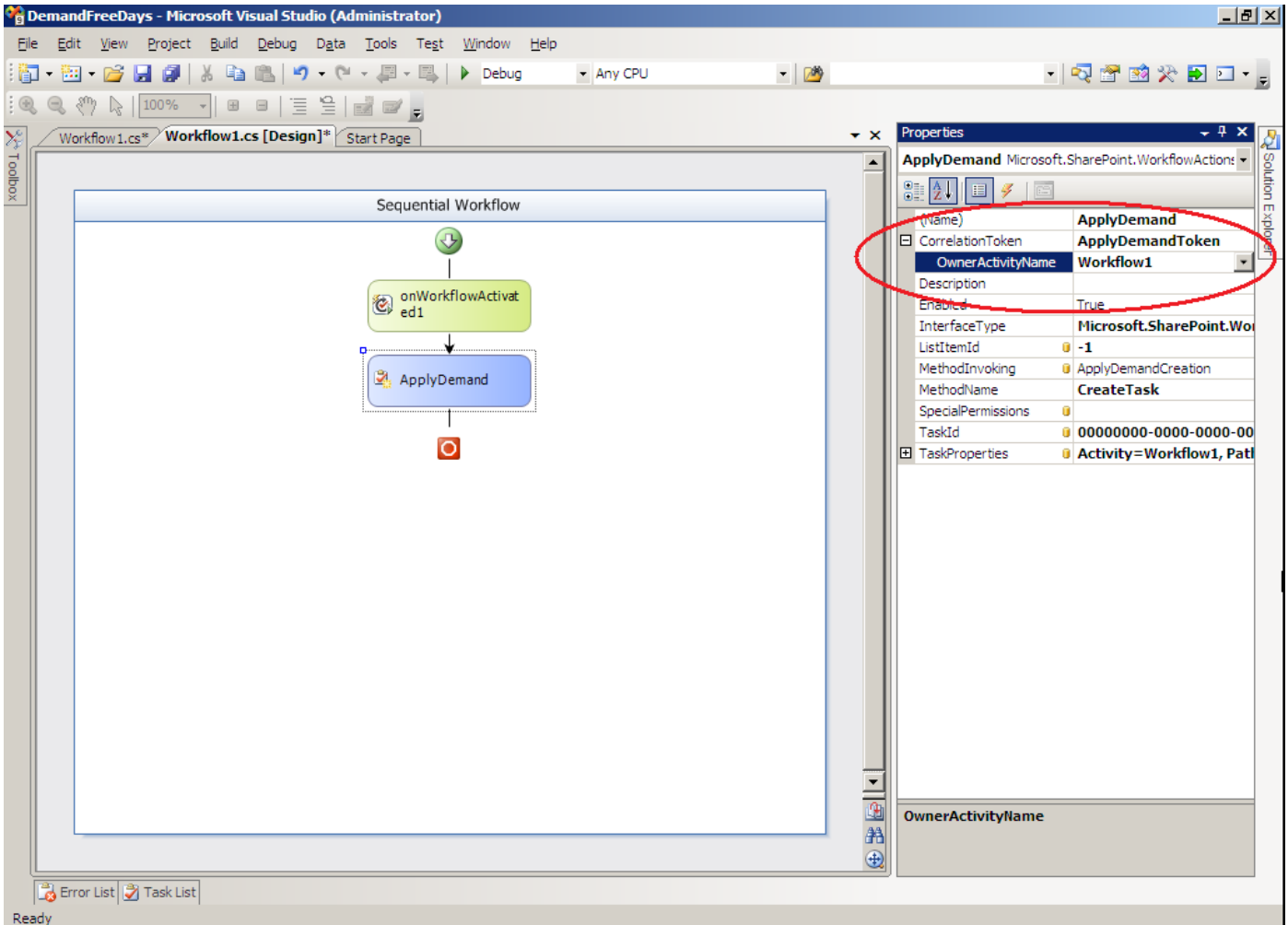
```
private void ApplyDemandCreation(object sender, EventArgs e)
{
    try
    {
        myTaskID = Guid.NewGuid();
        ApplyDemandProperties = new Microsoft.SharePoint.Workflow.SPWorkflowTaskProperties();
        ApplyDemandProperties.PercentComplete = (float)0.0;
        ApplyDemandProperties.AssignedTo = System.Threading.Thread.CurrentPrincipal.Identity.Name;
        ApplyDemandProperties.DueDate = DateTime.Now.AddDays(7);
        ApplyDemandProperties.StartDate = DateTime.Now;
        ApplyDemandProperties.Title = "Apply a Demand for Free Days Workflow Task";
        ApplyDemandProperties.Description = String.Format("The employee asking for free days is from Department",
        CustomFieldValue("Department"));
    }

    catch (Exception ex)
    {
        throw (new Exception("Unable to initialize task.", ex));
    }
}

private string CustomFieldValue(string fieldName)
{
    try
    {
        object item = this.workflowProperties.Item[fieldName];
        string s = this.workflowProperties.Item.Fields[fieldName].GetFieldValueAsText(item);
        return s;
    }

    catch (Exception ex)
    {
        return String.Empty;
    }
}
```

- In the workflow designer, in the Properties window, we selected the text box for the “CorrelationToken” property value; we typed “ApplyDemandToken” and pressed the Enter key.
- Expand the CorrelationToken property to display the OwnerActivityName subproperty.
- For the OwnerActivityName subproperty, we selected the name of the workflow which is “Workflow1” in the dropdown.



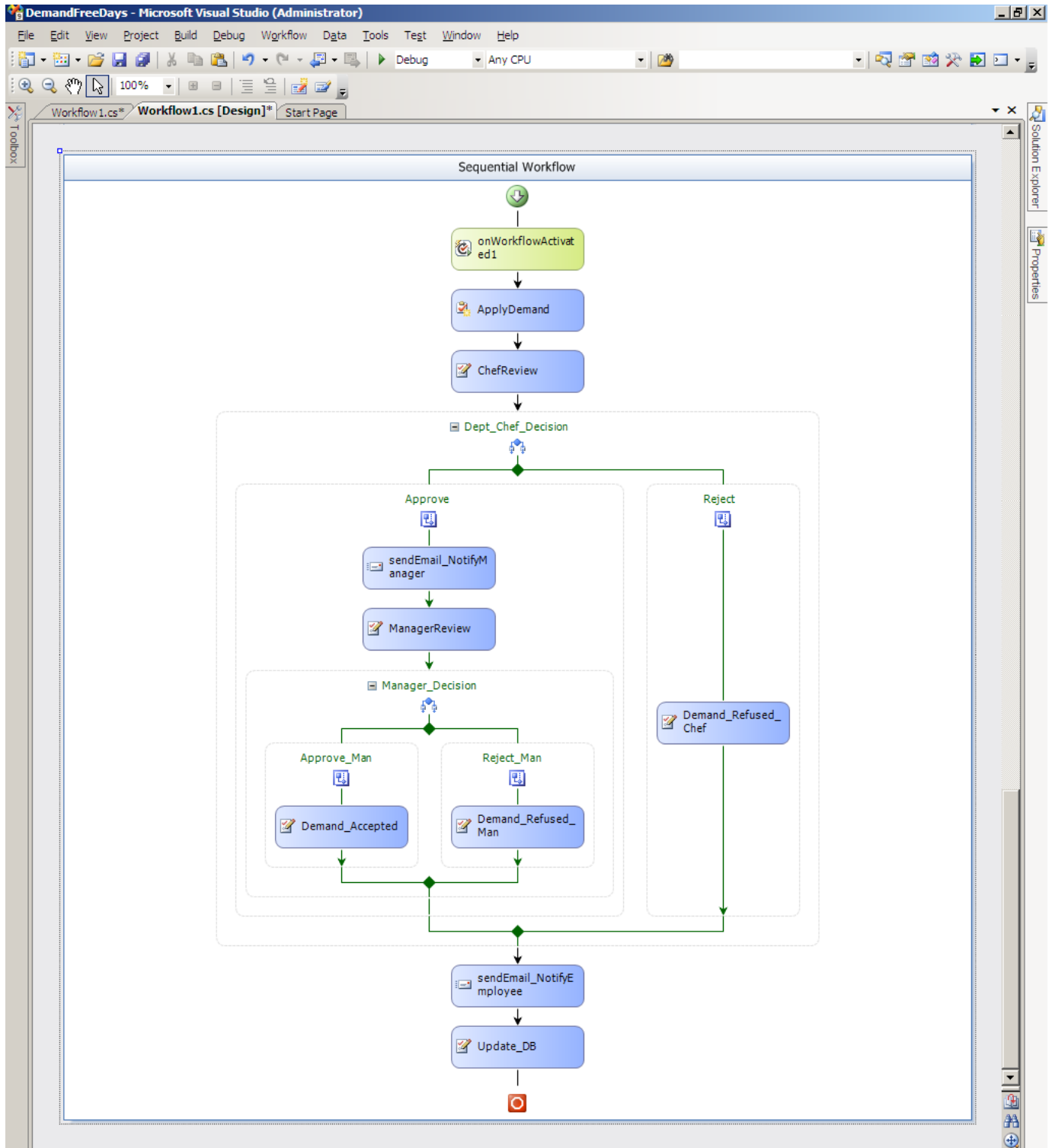
And so on, we created all the tasks that compose our workflow as shown in the Sequential Diagram in paragraph 3.

The screenshot displays the Microsoft Visual Studio (Administrator) interface. The main window shows a workflow design for 'Workflow1.cs [Design]*'. The workflow is a 'Sequential Workflow' containing three activities: 'onWorkflowActivated1', 'ApplyDemand', and 'ChefReview'. The 'ChefReview' activity is currently selected and highlighted with a dashed border. To the right, the 'Properties' window is open for the 'ChefReview' activity, showing various configuration options.

ChefReview Microsoft.SharePoint.WorkflowActions.CompleteTask	
(Name)	ChefReview
CorrelationToken	ChefReviewToken
OwnerActivityName	Workflow1
Description	
Enabled	True
InterfaceType	Microsoft.SharePoint.Workflow
MethodInvoking	
MethodName	CompleteTask
TaskId	00000000-0000-0000-0000-0000-0000-0000-0000-0000-0000-0000-0000-0000-0000-0000-0000-0000
TaskOutcome	Activity=Workflow1, Path=ChefReviewOutcome
Name	Workflow1
Path	ChefReviewOutcome

OwnerActivityName

This is how the final sequential workflow Model looks like:



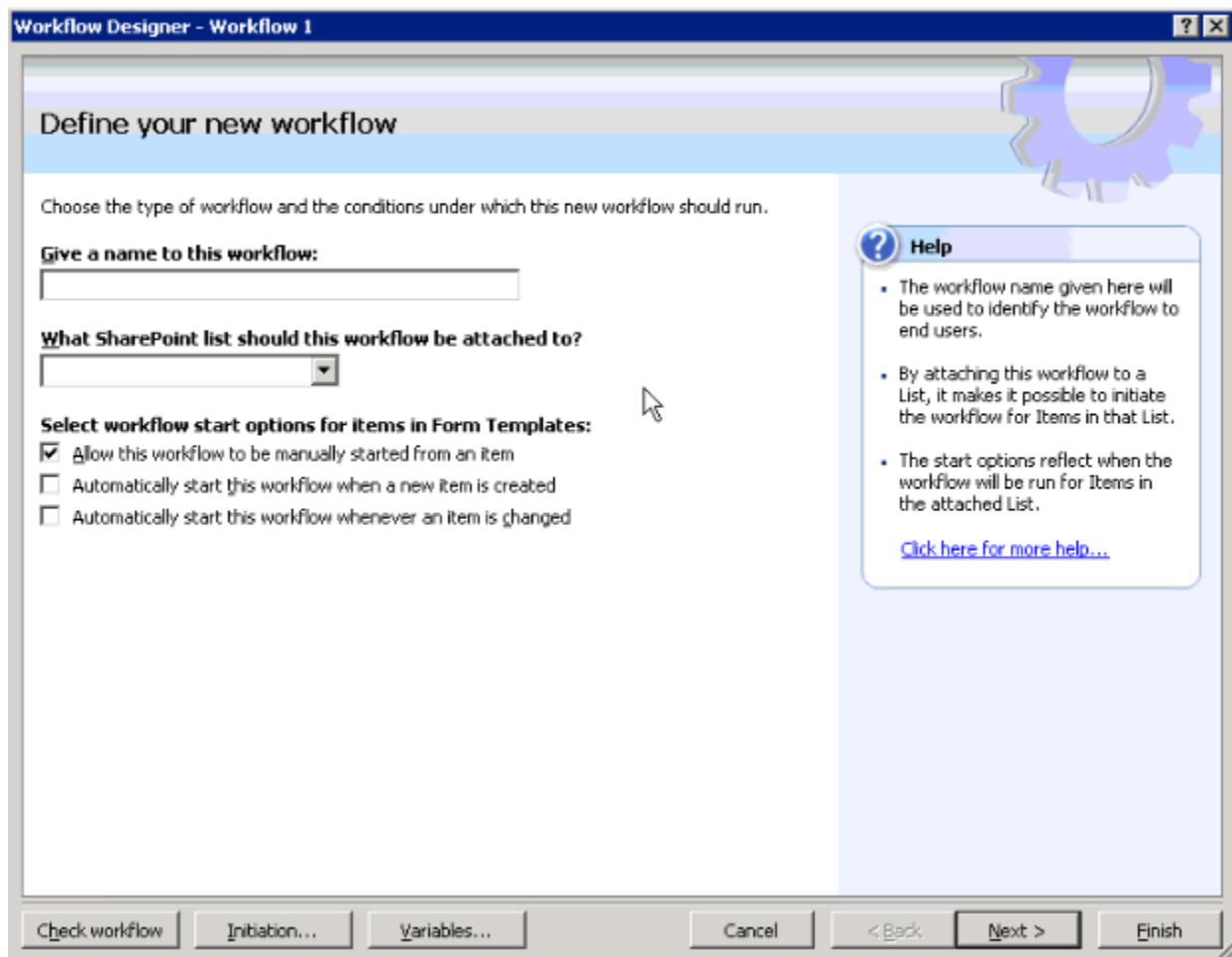
2.2 Using MS SharePoint designer 2007

Step 1- Creating a new Site in the SharePoint Services 3.0

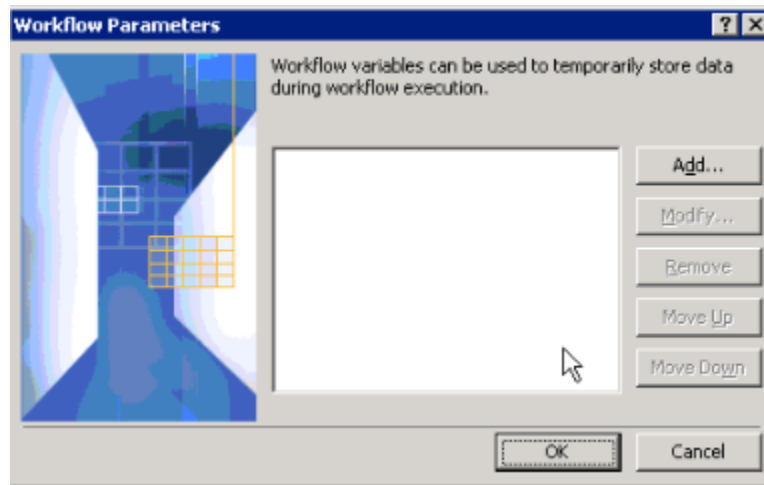
We already have been through this step in earlier stages of our work (see paragraph IV.1)

Step 2- Manipulating WF from the Microsoft Office SharePoint Designer 2007

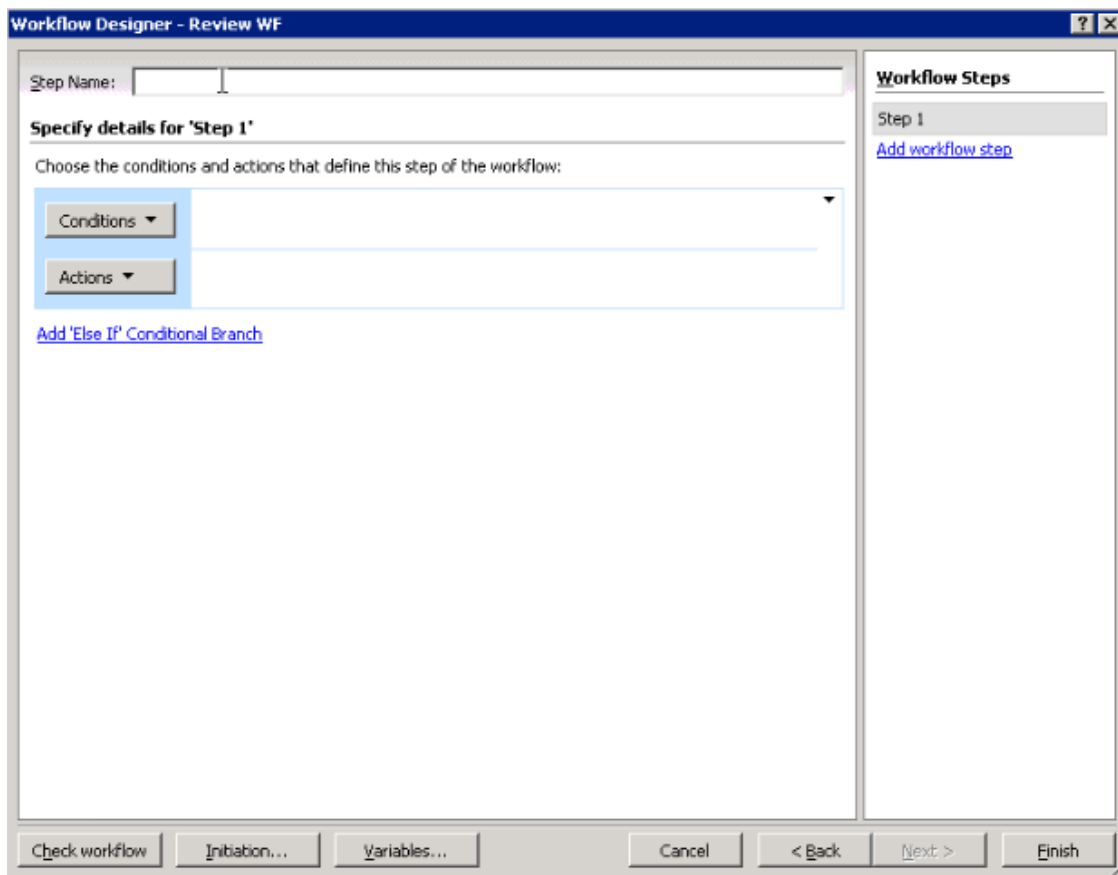
- After launching the MS Office SharePoint Designer 2007, it opens up the default site.
- Now that the site is opened, we should create a custom workflow with the sharePoint designer.
- From the New menu we choose Workflow is the type of items that we can create.
- We give to the workflow and we choose the SharePoint list to be attached to the workflow.
- There are 3 options that we can select for items in Form templates:
 - Allow this workflow to be manually started from an item
 - Automatically start this workflow when a new item is created
 - Automatically start this workflow whenever an item is changed



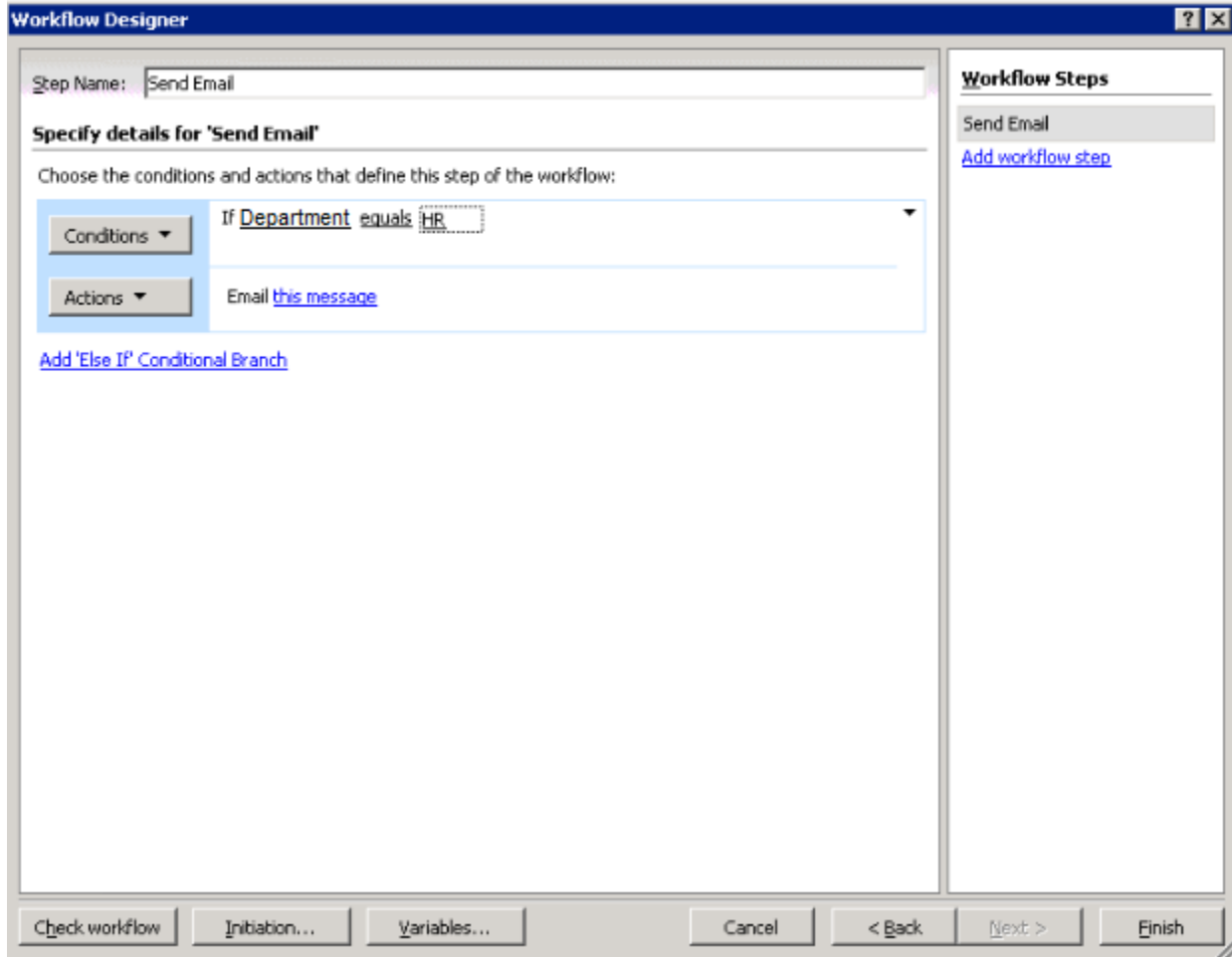
- In our case, we name our workflow: DemandFreeDays and we choose the list created before that is DemandFreeDays. Then we check the option Allow this workflow to be manually started from an item.
- We can also add Initiations and Parameters to the workflow as shown in the figure below:



- Then the next step is to create a step for example for sending an email to Manager:



- For example, to send an email to the HR department we create this:

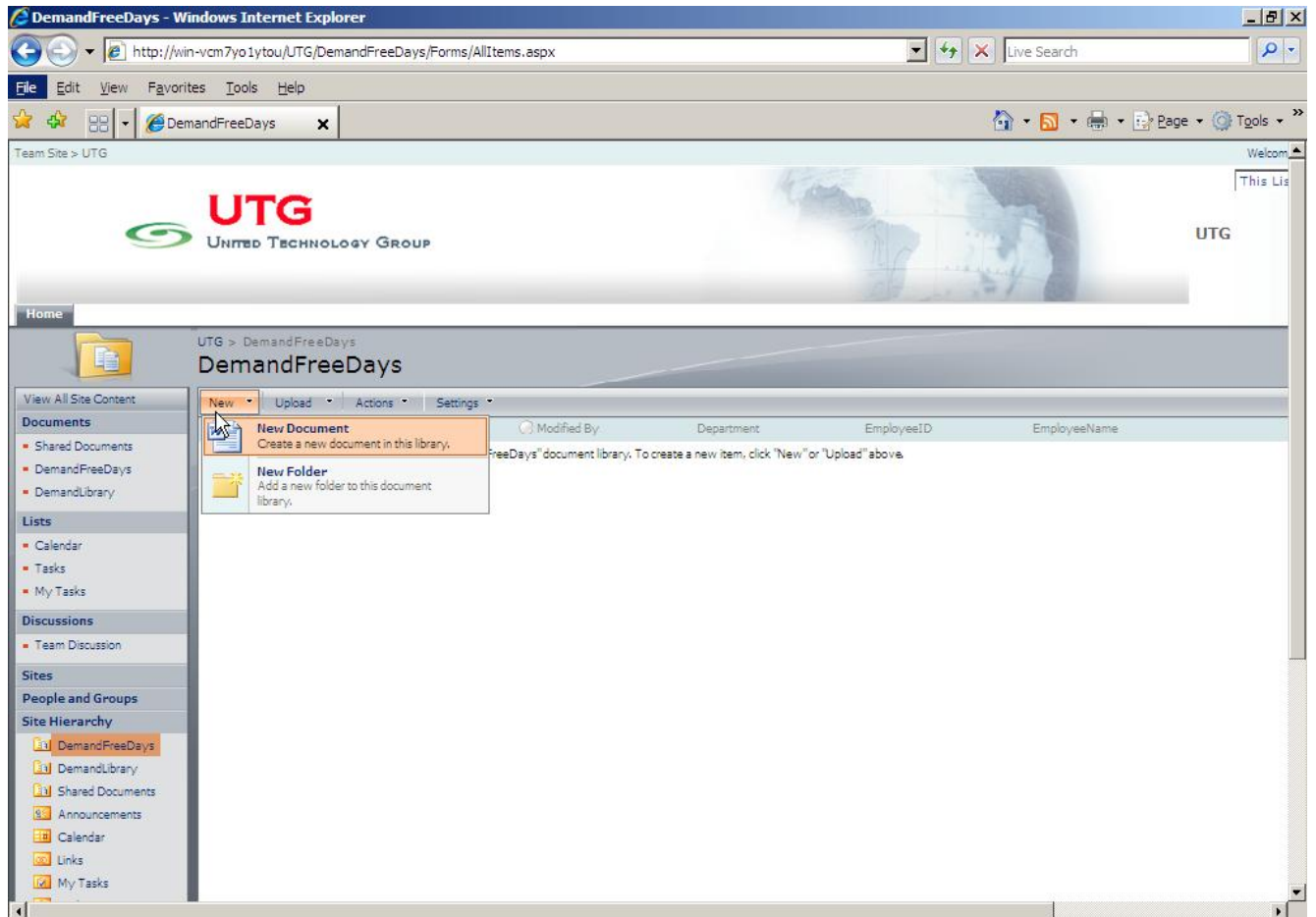


- We can add multiple steps mean multiple conditions and in addition we can create multiple branches.
- Then, we compose the mail to be sent to the members of the HR department that we can integrate parameters created above.
- This will end the creation of the workflow.
- The last step is to launch the workflow by uploading documents associated to different departments.
- The workflow will send the email only for the document related to the HR department.

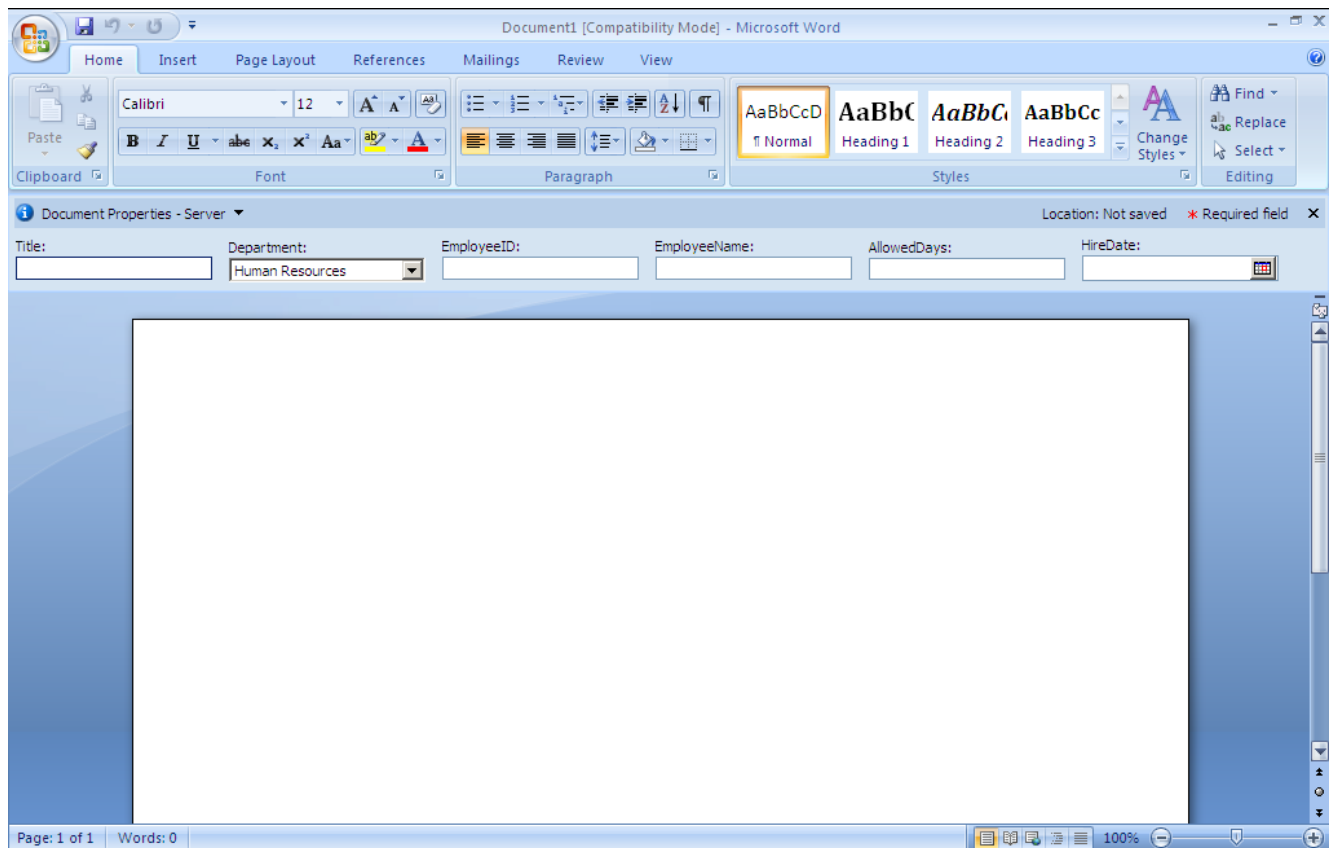
3. *Creating and Uploading a Document*

In our document library home page (DemandFreeDays), and after creating the columns and the WF, we will create and upload a new Office 2007 Word Document using SharePoint:

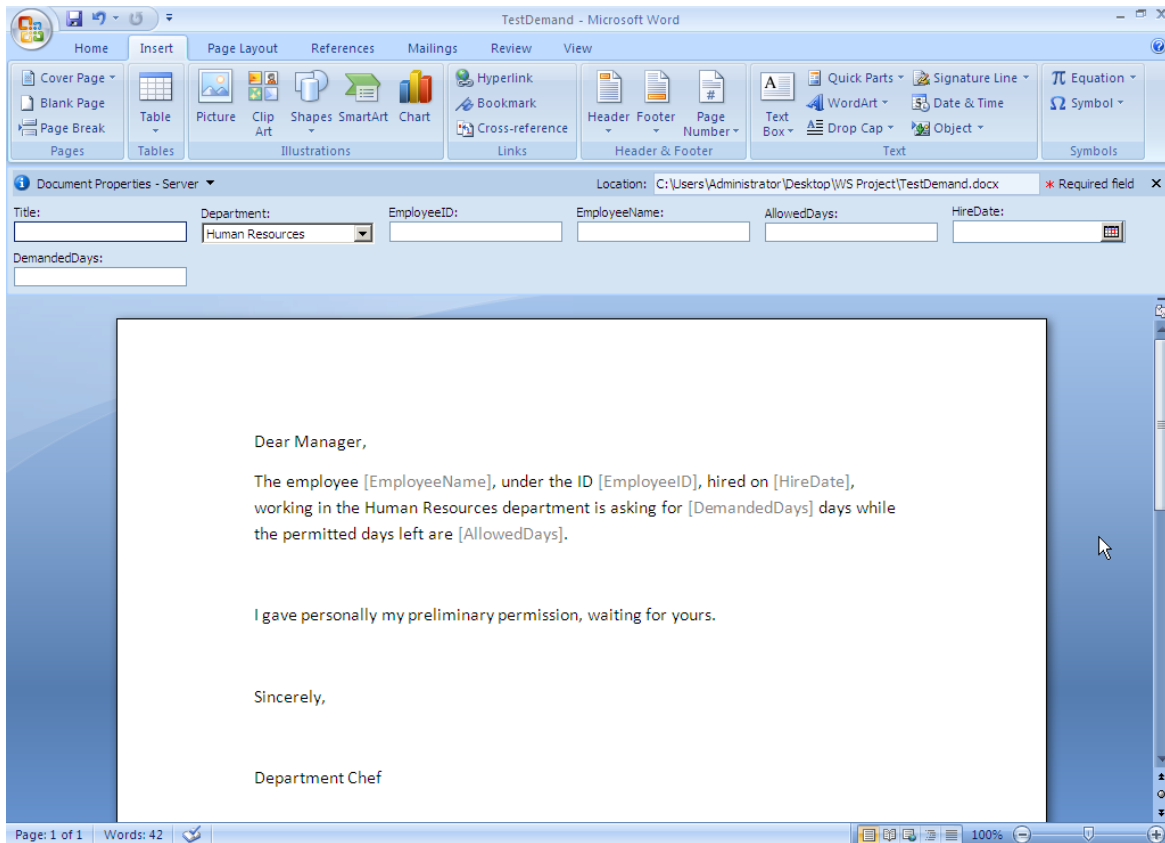
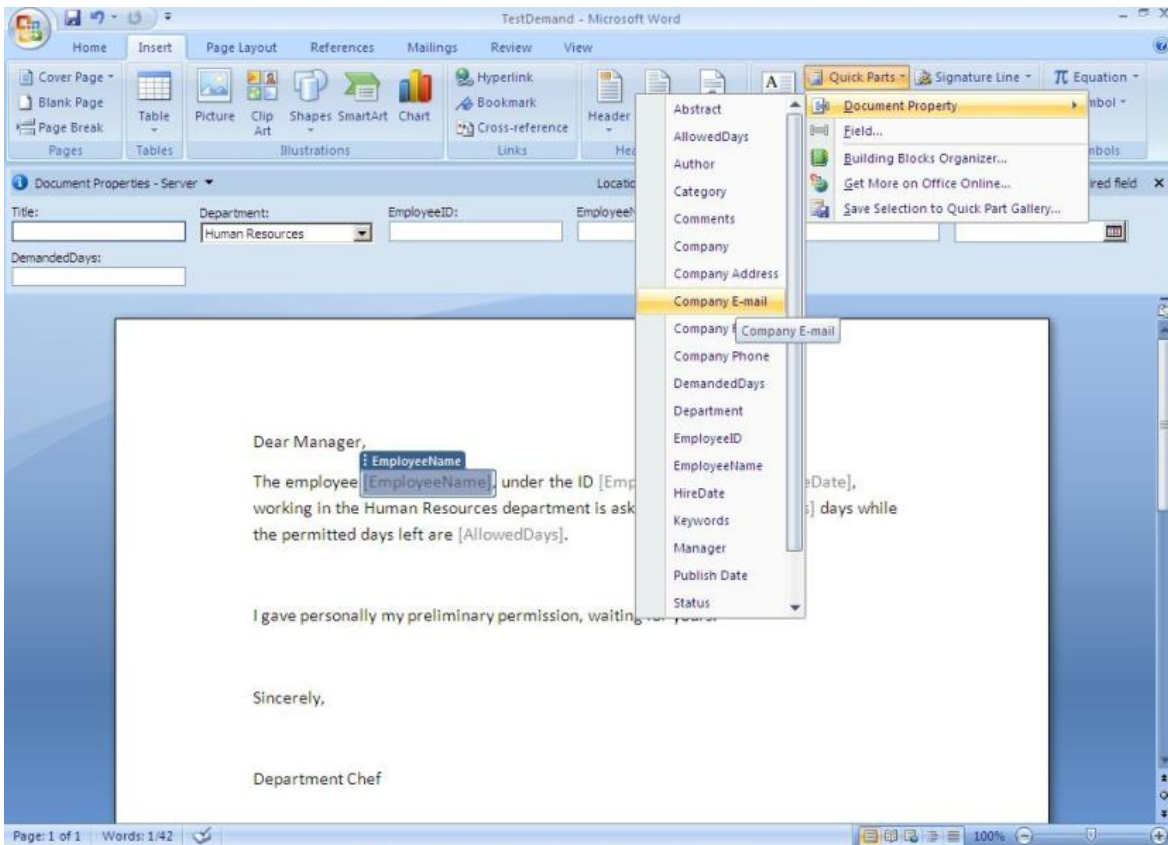
- On the New dropdown in the document library, click Document.



- A new document appears in Word. Observe that there are text boxes for the custom columns from the document library.



- We saved the document in a path of our choice; we were prompted to confirm saving to the new Word 2007 file format, we clicked "Ok".
- We inserted content controls for each of the server properties:
 - Place the selection in the document where you want the content control to be placed.
 - Select the Insert tab on the ribbon.
 - Click Quick Parts, then click Document Property, and then we chose the fields (columns) formerly created in the SharePoint Document Library.
 - We repeated these steps to add all the necessary fields.
 - For now, we left the content controls empty.



- After this we closed the document, we were prompted and clicked “yes”.
- Then return to the document library in Internet Explorer, click Settings and then select Document Library Settings.
- On the “Customize Documents” page, we couldn’t locate the “Content Types” section, so we went to the Advanced Settings and enabled the management of Content Types by switching the radio button to “yes” and clicked “OK” to save it.

UTG > DemandFreeDays > Settings

Customize DemandFreeDays

List Information

Name: DemandFreeDays
Web Address: http://win-vcn7yo1ytou/UTG/DemandFreeDays/Forms/AllItems.aspx
Description:

General Settings | **Permissions and Management** | **Communications**

- File, description and navigation
- Versioning settings
- Advanced settings** (circled in orange)

Permissions and Management

- Delete this document library
- Save document library as template
- Permissions for this document library
- Manage checked out files
- Workflow settings

Communications

- RSS settings

Columns

A column stores information about each document in the document library. The following columns are currently available in this document library:

Column (click to edit)	Type	Required
Title	Single line of text	
Department	Choice	
EmployeeID	Single line of text	
EmployeeName	Single line of text	
AllowedDays	Number	

UTG > DemandFreeDays > Settings > Advanced Settings

Document Library Advanced Settings: DemandFreeDays

Content Types

Specify whether to allow the management of content types on this document library. Each content type will appear on the new button and can have a unique set of columns, workflows and other behaviors.

Allow management of content types?
 Yes No (circled in orange)

Document Template

Type the address of a template to use as the basis for all new files created in this document library. When multiple content types are enabled, this setting is managed on a per content type basis. [Learn how to set up a template for a library.](#)

Template URL:
DemandFreeDays/Forms/template.doc

Browser-enabled Documents

Specify how to display documents that are enabled for opening both in a browser and a client application. If the client application is unavailable, these documents will always be displayed as Web pages in the browser.

Opening browser-enabled documents
 Open in the client application
 Display as a Web page

Custom Send To Destination

Type the name and URL for a custom Send To destination that you want to appear on the context menu for this list. It is recommended that you choose a short name for the destination.

Destination name (For example, Team Library):
URL:

Folders

Specify whether the "New Folder" command appears on the New menu. Changing this setting does not affect existing folders.

Display "New Folder" command on the New menu?
 Yes No

Préparé par Elie Matta et al.

- Now we went back to the previous page and found the Content Type section where we pressed the document link
- In the List Content Type page we clicked “Advanced Settings”

UTG > DemandFreeDays > Settings

Customize DemandFreeDays

List Information

Name: DemandFreeDays
Web Address: <http://win-vcn7yo1ytou/UTG/DemandFreeDays/Forms/AllItems.aspx>
Description:

General Settings | **Permissions and Management** | **Communications**

- [Title, description and navigation](#)
- [Versioning settings](#)
- [Advanced settings](#)

- Delete this document library
- Save document library as template
- Permissions for this document library
- Manage checked out files
- Workflow settings

- RSS settings

Content Types

This document library is configured to allow multiple content types. Use content types to specify the information you want to display about an item, in addition to its policies, workflows, or other behavior. The following content types are currently available in this library:

Content Type	Visible on New Button	Default Content Type
Document	✓	✓

- [Add from existing site content types](#)
- [Change new button order and default content type](#)

UTG > DemandFreeDays > Settings > List Content Type

List Content Type: Document

List Content Type Information

Name: Document
Description: Create a new document.
Parent: Document

Settings

- [Name and description](#)
- [Advanced settings](#)
- [Workflow settings](#)
- [Delete this content type](#)

Columns

Name	Type	Status	Source
Name	File	Required	Document
Title	Single line of text	Optional	Item
Department	Choice	Optional	
EmployeeID	Single line of text	Optional	

- Select the Upload a new document template radio button and then click Browse. We browsed to select the word document we created and then clicked Ok.

UTG > DemandFreeDays > Settings > List Content Type > Advanced Settings

List Content Type Advanced Settings: Document

Use this page to change advanced settings for this content type.

Document Template
Specify the document template for this content type.

Enter the URL of an existing document template:

(Edit Template)

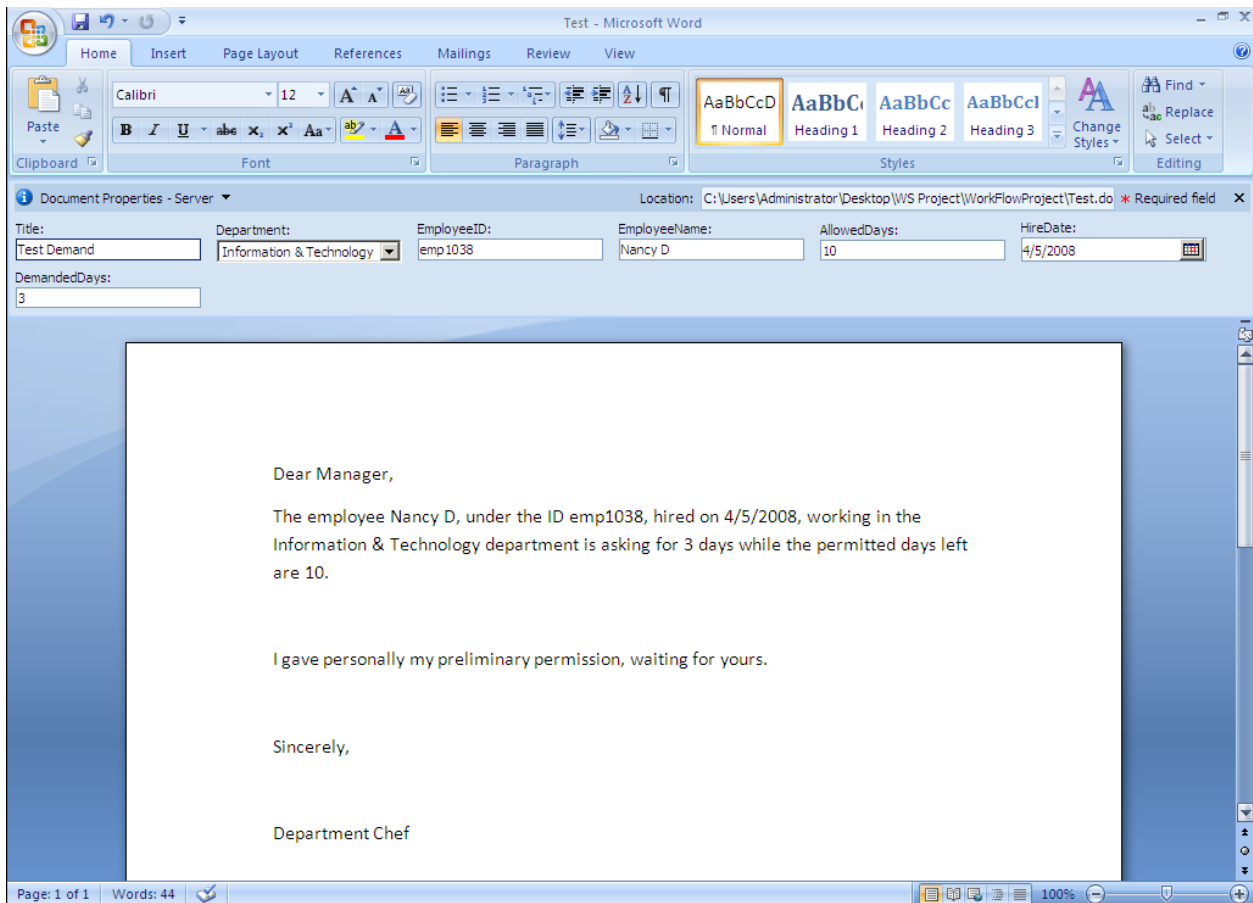
Upload a new document template:

Read Only
Choose whether the content type is modifiable. This setting can be changed later from this page by anyone with permissions to edit this type.

Should this content type be read only?

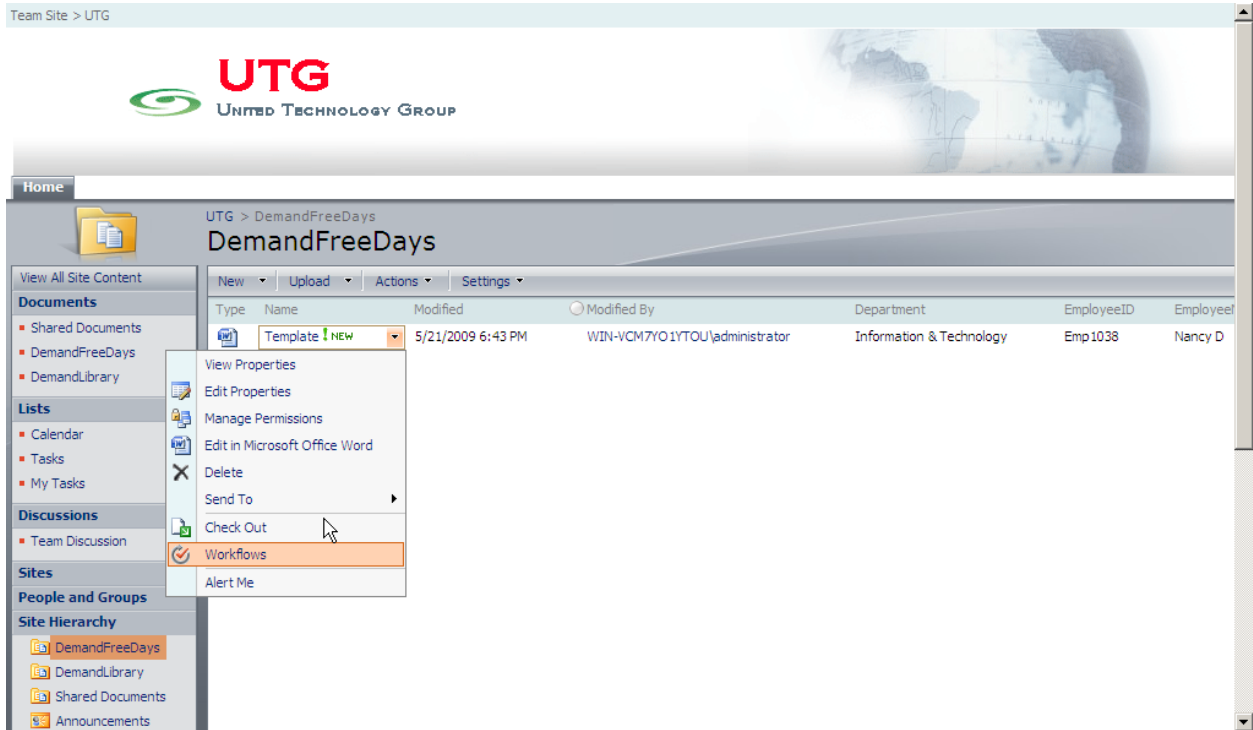
Yes
 No

- In Internet Explorer, return to the document library.
- Upload a new document:
 - In the document library, click the “New” dropdown and then click “Document”. Word starts and loads a new document based on the contract template we created.
 - We filled in the values for Title, Department, EmployeeID, EmployeeName, AllowedDays, HireDate and DemandedDays.
 - Once all required fields have been filled in, on we saved it and exit the word.
 - Our new document appears in the document library. The fields reflect the values you entered in the Word document.



4. Association with SharePoint Site

To associate the WF created with the SharePoint Site, we deploy the project from Visual Studio 2008, and when the UTG site opens, we go to the document library and select the dropdown for the uploaded document and choose Workflows.




- In the list of workflows we select DemandWF (our workflow formerly created) to start the workflow for the document.

UTG > DemandFreeDays > Template > Workflows

Workflows: Template

Use this page to start a new workflow on the current item or to view the status of a running or completed workflow.

Start a New Workflow

 DemandFreeDays
Use this workflow to track items in a list.


Workflows

Select a workflow for more details on the current status or history.

Name	Started	Ended	Status
Running Workflows			
There are no currently running workflows on this item.			
Completed Workflows			
DemandFreeDays	5/21/2009 7:17 PM	5/21/2009 7:17 PM	Completed

with the selected items. 100%

Operation in Progress

 Please wait while your workflow is started...

Waiting for http://win-vcn7yo1ytou/UTG/_layouts/Workflow.aspx?ID=3&List={09615... 3E-80EF48EA8992}&Source=http%3A%2F%2Fwin%2Dvcn7yo1ytou%2FUTG%2FDema... DemandFreeDays%

5. Assigning Mailing Task

We should first navigate to our library list, click Actions → List Settings, and then click “Advanced settings”.

Under the General Settings heading:

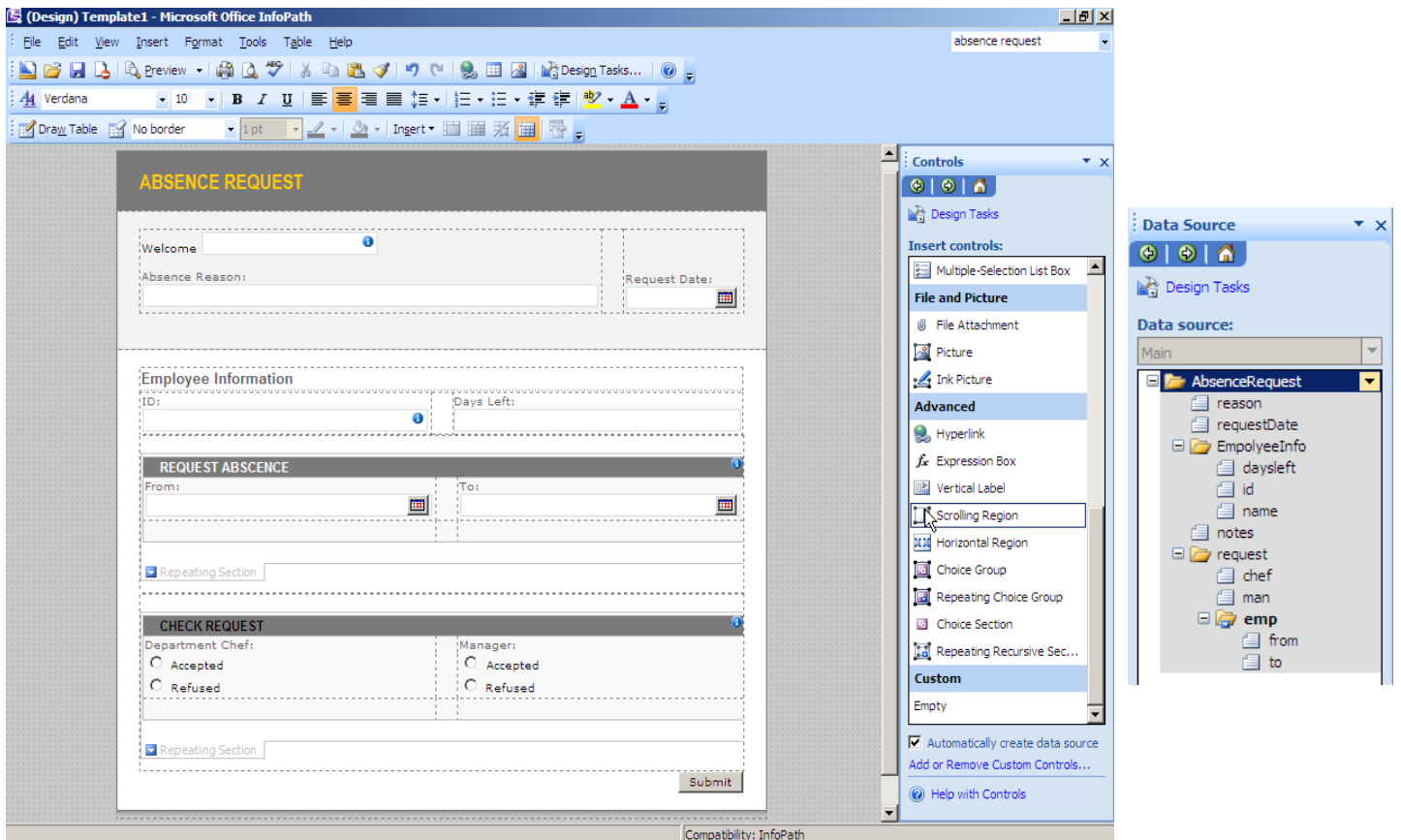
- Select “Yes” in the E-Mail Notification section and click OK.

6. Designing Forms Using InfoPath 2007

There are several steps that allow us to create and associate forms to our WF using MS Office InfoPath 2007.

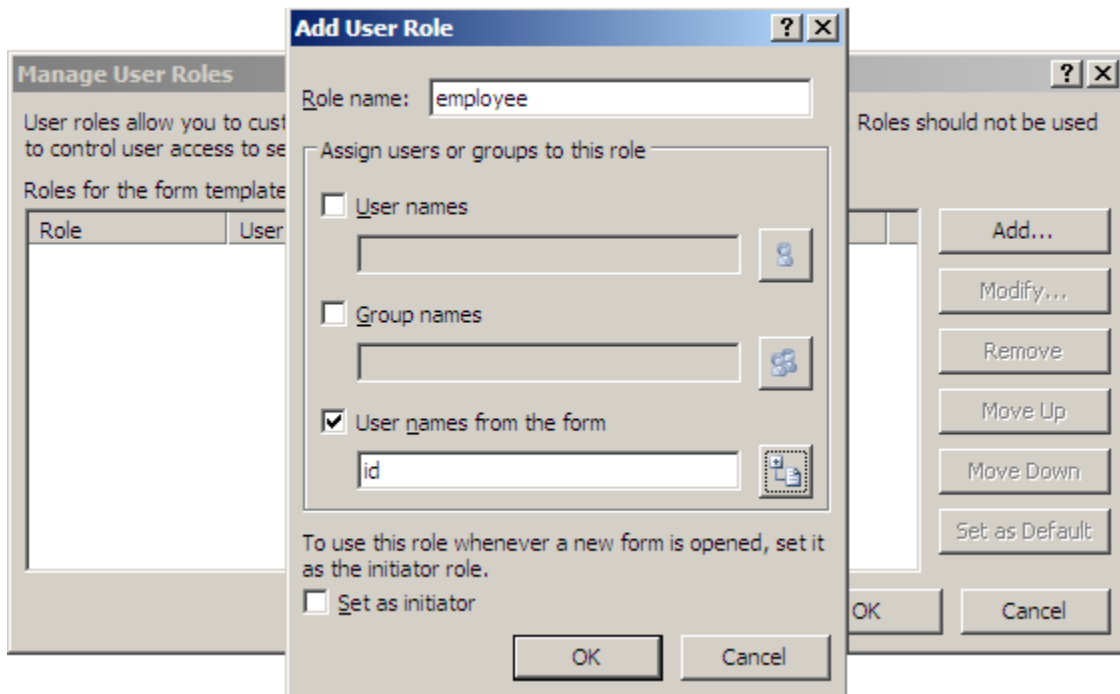
- Create User Roles

After designing our absence request form using InfoPath, we will start the work:



The first step is to create User Roles for this form, we will setup roles based on the User names, Group Names and User Names from the form.

- We will create three roles: the first is for users (regular employees); the second is for department chefs and the third is for the manager. For each one we:
 - Enter Role Name as the User
 - Check the specific checkbox.
 - Click on the button next to the text box and browse to the appropriate field
 - Select the field and click OK

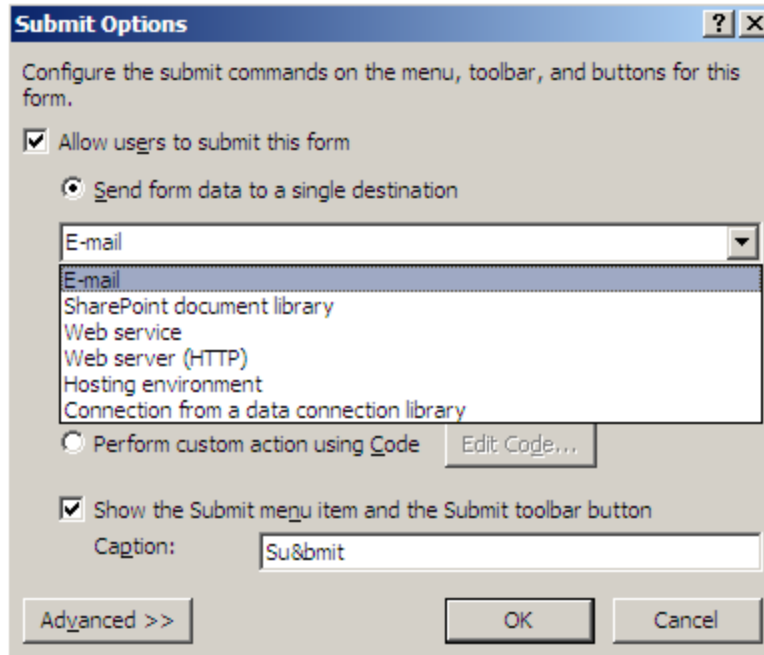


➤ **Setting up Rules for Submitting the form**

Once the user roles are setup, we need to create some rules to setup our workflow process.

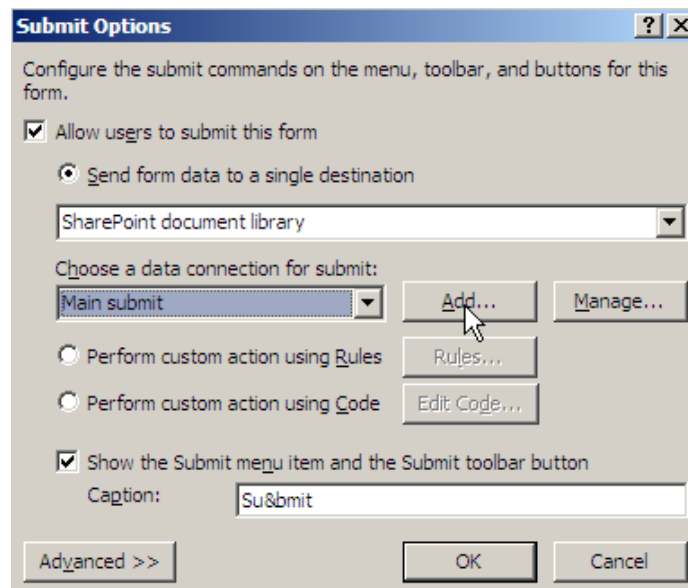
We will consider emailing this form to Manager.

The options provided can be shown by the Submitting Forms window that we can explore by clicking on Tools -> Submitting Forms.

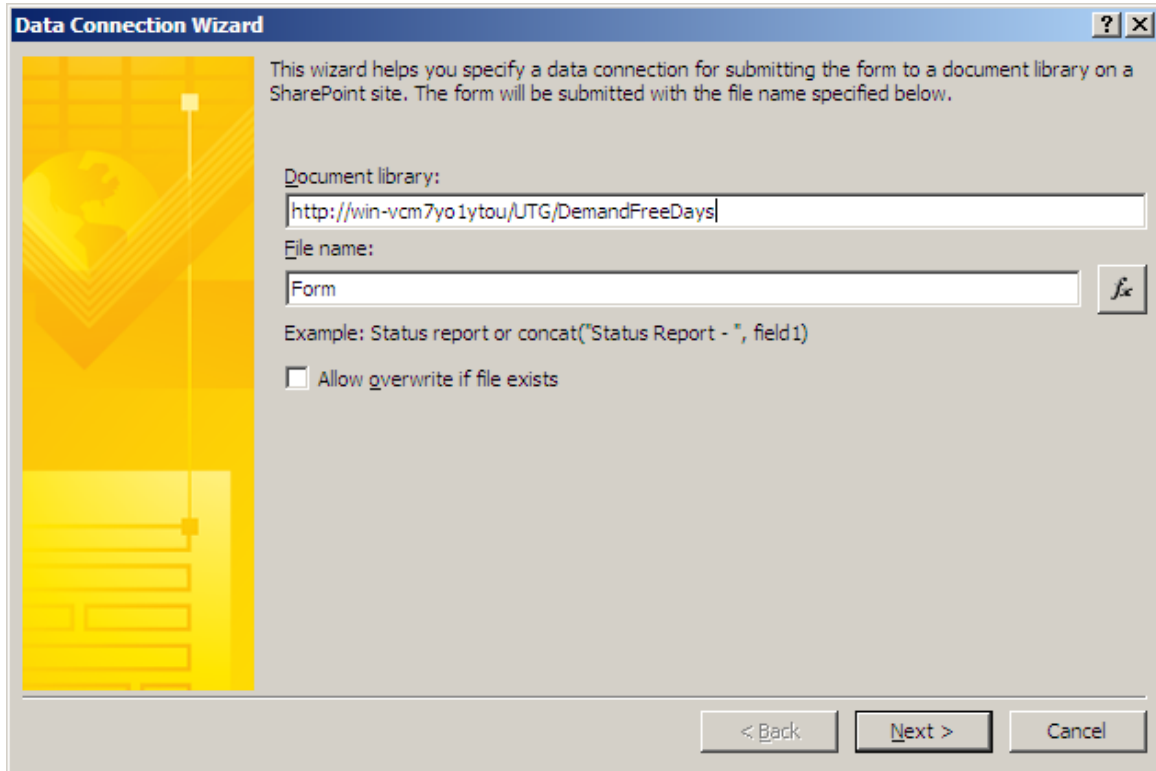


- **Email:** We can email this form to certain user(s).
- **Web Service:** We can use the data to be submitted to some web service for processing.
- **SharePoint document Library:** Basically, the XML data will be submitted to some Forms Library on WSS or SPS site.
- **Web Server:** Submitting it to some web server to store the XML file.
- **Connection from a data connection library:** The database option is only visible if you have created this form using some database connection.

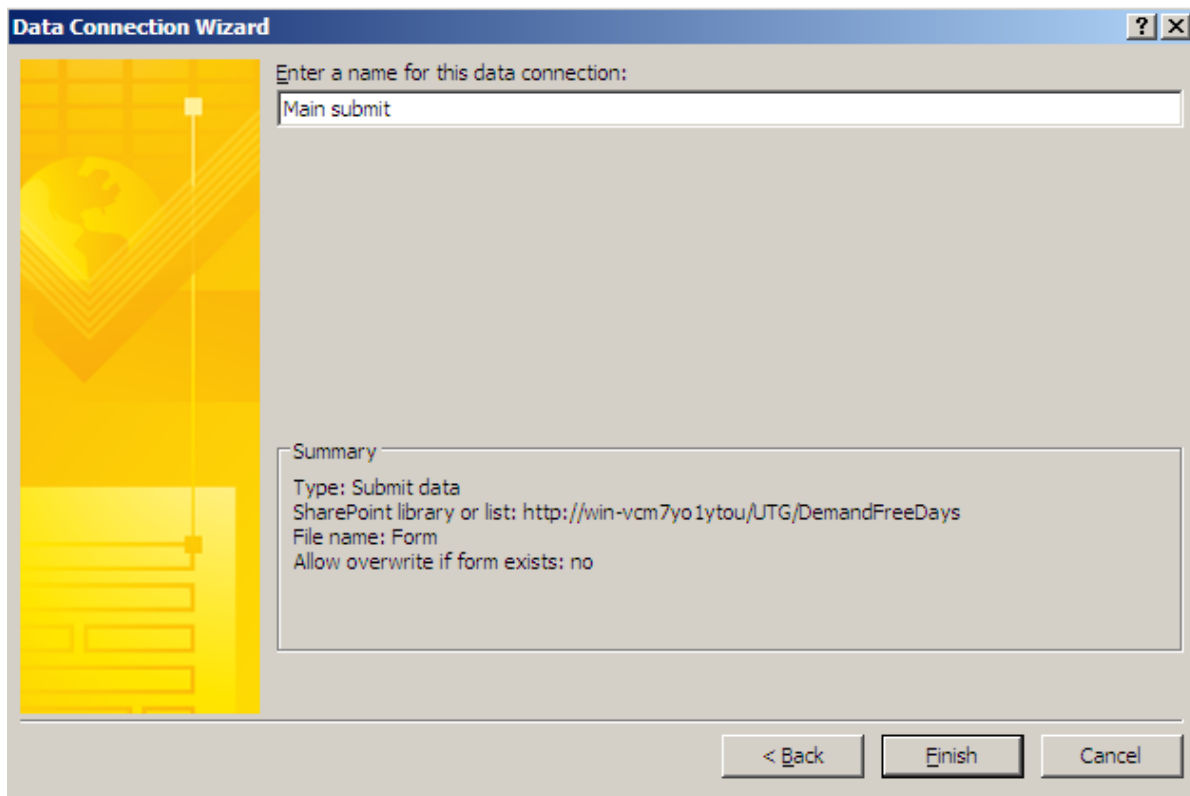
➤ We chose the SharePoint option from the menu and then added a new connection by clicking on the Add Connection button



When clicking “OK” the wizard show and we specify the path of our document library formerly created in SharePoint, and we click next.

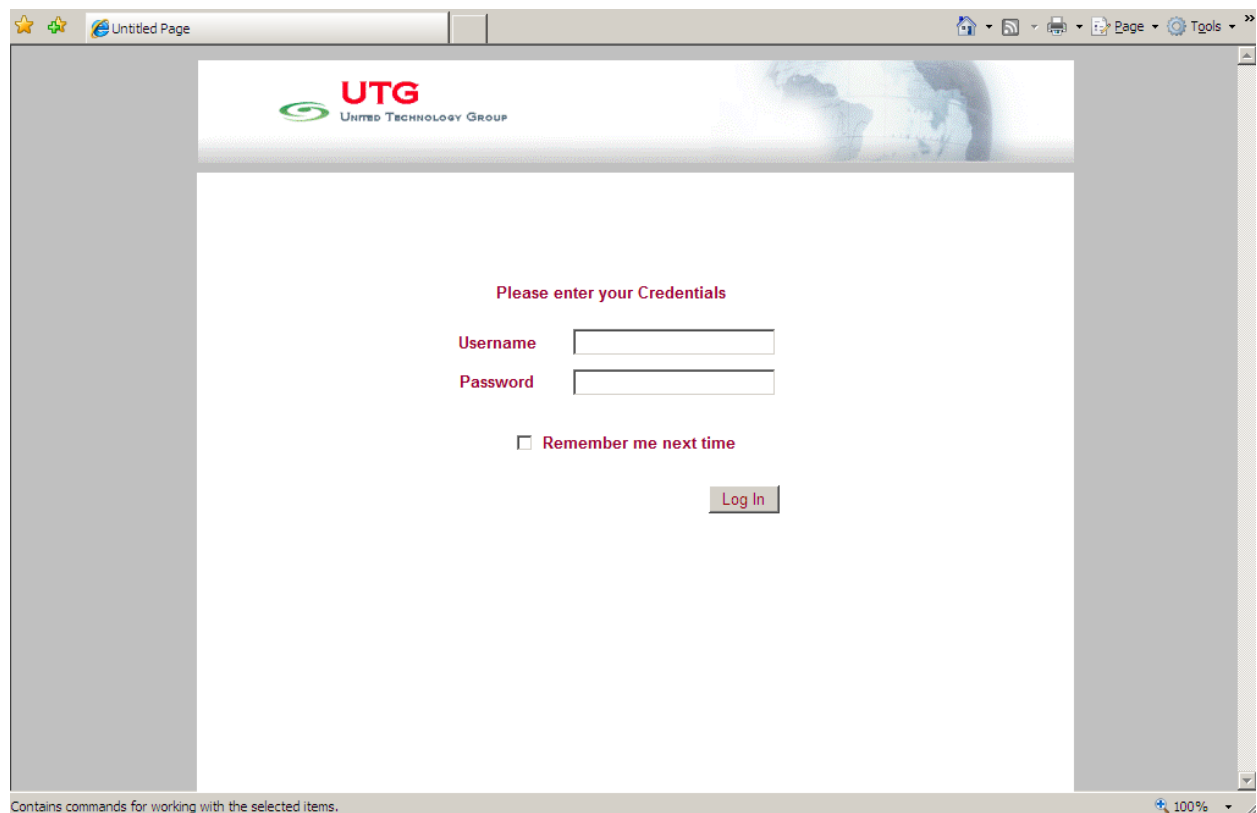


And now we specify a name for the data connection and we click “Finish”



7. Web Solution

Login Page



The user logging to this page can be:

- An employee: he/she will be directed to the Demand page where can find the appliance form to fill and submit his/hers Request.
- A chef of a certain department: he/she will be directed to the Demand page where can find the list of non approved demands (requests) applied by the employees working in his/hers department (only), and a form to approve or reject each demand.
- The Manager: he/she will be directed to the same page where can find for each demand approved by its chef the appliance form, the chef's review and a form to accept or refuse.

Employee

The screenshot shows a web browser window displaying a form for an employee to submit an absence request. The form is titled "Employee" and is part of the UTG (United Technology Group) system. The interface includes a header with the UTG logo and a welcome message: "Welcome Nancy D.". Below the header, there are two main sections: "Employee ID" and "Demand Days Off".

The "Employee ID" section contains the following fields:

- Employee ID: []
- Employee Name: Nancy D
- Allowed Days: 15
- date: 5/25/2009

A "Submit" button is located to the right of the "Allowed Days" field.

The "Demand Days Off" section contains the following fields:

- From: 5/26/2009
- To: 5/29/2009
- Number Of Days: 3
- Description: Sickleave - influenza

The form is displayed in a browser window with a status bar at the bottom indicating "Contains commands for working with the selected items." and a zoom level of 100%.

The employee can fill the details about his absence request like the “from” – “to” dates which defines the number of days off; and the description which is the reason behind the vacation.

By clicking the Submit button, the request is saved as an absence request appliance.

Chef

The screenshot shows a web browser window displaying a page for a chef named Farah F. The page has a header with the UTG logo and a globe. Below the header, there is a section titled "Welcome Farah F." followed by a form. The form includes a "Date" field with the value "5/25/2009 12:00:00 AM", a "Demand ID" dropdown menu with the value "605", an "Employee Name" field with the value "Nancy D", and an "Allowed Days" field with the value "15". A "Submit" button is located to the right of the "Allowed Days" field. Below this section is a "Demand Days Off" section with fields for "From:" (5/27/2009), "To:" (5/29/2009), "Number Of Days:" (3), and a "Description:" dropdown menu with the value "Business - Insurance". At the bottom of the page is a "Chef Review" section with the text "Department Chef: Farah F" and two radio buttons labeled "Accepted" and "Refused". The browser's status bar at the bottom indicates "Contains commands for working with the selected items." and "100%".

The chef finds in the list the demands applied by his employees (in his department) and can observe the details of the request to finally choose his decision (Accept or refuse) by choosing from the radiobuttonlist down the page.

When clicking Submit the decision will be committed.

Manager

UTG
UNITED TECHNOLOGY GROUP

Welcome Josiane H.

Demand ID: Date: 5/25/2009 12:00:00 AM

Employee Name:

Allowed Days:

Demand Days Off

From:

To:

Number Of Days:

Description:

Chef Review

Department Chef: *Farah F*

Accepted
 Refused

Manager Review

Manager: *Josiane H*

Accepted
 Refused

Contains commands for working with the selected items. 100%

In addition to the forms that show in the chef's profile an additional form in the bottom of the page offers to the manager to choose to accept or refuse a certain demand selected from the DropDownList above which represents all the demands applied and approved by their chefs waiting for the manager's decision.

IV. BACKUP AND RESTORE

Microsoft Office SharePoint Server and Windows SharePoint Services are critical components in a network infrastructure. Great care should be taken to back up their components and content. It would be a tremendous task to re-create customized SharePoint sites, not to mention the document data associated with each one. A good deal of attention should be paid to the backup and restore procedures for a SharePoint farm.

There are several different approaches to backing up data in SharePoint. Because there are so many options, SharePoint administrators are often confused over which option is the best for their organization. The following backup/restore solutions and their appropriate use are listed as follows:

- **Recycle Bin:** While not a traditional backup tool, the Recycle Bin included within SharePoint 2007 is the first line of defense in the event that a restore is needed, as all deleted items in a site collection are placed in a dual-stage Recycle Bin, where they can be restored by the individual user after deletion, or by a site collection administrator. Although the Recycle Bin only handles deletions and not such things as item corruption, using the Recycle Bin in a SharePoint environment is key to avoiding using more intrusive tools to restore.
- **Backup and restore options in SharePoint Central Administration 3.0:** Within the SharePoint Central Admin Tool, the option to back up all farm components or individual web applications exists. This type of backup process, although thorough, is not particularly robust and cannot be easily scheduled. It is typically manually invoked before an administrator makes a major change, such as patching the server or updating the service pack levels.
- **STSADM command-line utility backup:** The command line—driven STSADM utility exports entire site collections to flat file formats, allowing for full fidelity backup snapshots of sites at a particular point in time. This type of backup process can be kicked off by a batch file process scheduled with the integrated Windows Scheduled Tasks application or the command-line 'AT' command. STSADM backups also allow individual site collections to be restored to other servers or to different locations on the same server, giving it great flexibility. The only downside to STSADM is that it does not scale well to very large site collections, so it is primarily used by small to medium-sized SharePoint environments.

- **SharePoint Designer backup:** The replacement product for FrontPage 2003 is known as SharePoint Designer 2007. This tool allows individual sites or site elements to be backed up to a .cwp flat file format. This type of backup is typically performed ad hoc, when in the process of designing a SharePoint site or when moving pieces of a site from one location to another. It cannot be easily scheduled.
- **IIS backup script:** A built-in script in Windows 2003 allows the settings of the IIS Virtual Servers on the system to be backed up to an xml-format file, which means that the administrators can keep a copy of the IIS configuration in the event that the SharePoint Server needs to be rebuilt. This type of backup is typically used as part of a scheduled batch file process that is run on a regular basis.
- **SQL backup tools:** Within SQL Server (2000 and above), built-in tools exist that backup SQL databases, including SharePoint databases, and they can be easily set up and scheduled to run full backups of all SharePoint content. This type of backup process is convenient for organizations that already have a SQL database backup plan in place or that want to supplement another backup option, such as STSADM, with a full database-level backup. SQL restore procedures can only restore entire databases, however, and individual site elements cannot be restored using this technique.
- **Third-party backup tools:** Several third-party companies, including “DocAve” and “Veritas”, are currently working on backup tools that will work within SharePoint, allowing for item-level restore of documents and other SharePoint settings. Checking on the features of the preferred backup vendor is recommended for those environments needing a truly automated and enterprise-level backup environment for SharePoint.

V. CONCLUSION

Finally, we have explored three possibilities to work with workflows through this Request Absence application workflow that we discovered in this document.

Human workflow applications can improve the efficiency and accuracy of many business processes. By creating Windows Workflow Foundation and making it a standard part of the operating system, Microsoft has provided the basis for a broad set of workflow applications.

By hosting WF workflows, Windows SharePoint Services 3.0 offers developers and information workers the ability to construct document-oriented human workflow applications. Adding Office SharePoint Server lets these applications interact with their users through InfoPath forms presented in Office 2007 desktop applications. It also provides a group of pre-defined workflows that address common business scenarios.

Windows SharePoint Services and Microsoft Office are popular technologies, and so it's fair to expect that these new workflow features will be quite widely used. Going forward, we look for workflow applications to become more common in the lives of both information workers and developers.

VI. ANNEX

DBServices Class

```
using System;
using System.Data;
using System.Configuration;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
using System.Data.Sql;
using System.Data.SqlClient;
using System.Data.SqlTypes;
using System.Collections;

public class DBServices
{
    #region Global Variables
    private string datasource;
    private string catalog;
    private SqlConnection CnnDB;
    #endregion

    #region Properties
    private static DBServices databaseServices;
    public static DBServices Instance
    {
        get
        {
            if (databaseServices == null)
            {
                databaseServices = new DBServices();
            }
            return databaseServices;
        }
    }
    #endregion

    public DBServices()
    {
        this.datasource = ConfigurationManager.AppSettings["datasource"];
        this.catalog = ConfigurationManager.AppSettings["catalog"];
    }
}
```

```
//Open DB Connection
public bool LoginToDatabase()
{
    try
    {
        string conString = "Data Source=" + this.datasources
            + ";Initial Catalog=" + this.catalog
            + ";Integrated Security=True";

        CnnDB = new SqlConnection(conString);
        CnnDB.Open();
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);
        return false;
    }
    return true;
}

//Close DB Connection
public void CloseConnection()
{
    try
    {
        if (CnnDB != null)
        {
            CnnDB.Close();
            CnnDB = null;
        }
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);
    }
}

//executes sql statement
public bool ModifyData(string sql)
{
    bool res = false;
    try
    {
        SqlCommand cmd = new SqlCommand();
        cmd.Connection = CnnDB;
        cmd.CommandText = sql;
        int i = cmd.ExecuteNonQuery();
        if(i>-1)
            res = true;
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);
    }
    return res;
}
```

```
//returns Dataset
public DataSet getData(string sql)
{
    DataSet ds = new DataSet();
    try
    {
        SqlCommand cmd = new SqlCommand();
        cmd.Connection = CnnDB;
        cmd.CommandText = sql;
        SqlDataAdapter ada = new SqlDataAdapter(cmd);
        ada.Fill(ds);
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);
    }
    return ds;
}

//returns the next ID
public string getNextSequence(string tablename)
{
    string id = "";
    try
    {
        DataSet ds = this.getData("select * from " + tablename + "");
        int n = ds.Tables[0].Rows.Count + 1;
        id = n.ToString();
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);
    }
    return id;
}

}
```

Testing the user logging

```
private void onWorkflowActivated1_Invoked(object sender,
ExternalDataEventArgs e)
{
    loggedid = Session["user"].ToString();
    string sql = "select * from Employee where emp_id='" + loggedid + "'";
    DataSet ds = DBServices.Instance.getData(sql);
    if (this.Label1.Text == "Welcome ")
    {
        this.Label1.Text += ds.Tables[0].Rows[0][1].ToString() + ",";
    }

    //check the type of user
    string s = "select * from Department where dept_chef='" + loggedid + "'";
    DataSet dept = DBServices.Instance.getData(s);

    //if not dept chef nor manager => ordinary employee
    if ((!(dept.Tables[0].Rows.Count > 0)) &&
        (!(ds.Tables[0].Rows[0][4].ToString() == ""))
        {
            currentemp = loggedid;
            this.Panel3.Visible = false;
            this.Panel4.Visible = false;
            this.Panel5.Visible = false;
            this.Panel6.Visible = false;
            this.TextBox1.Text = ds.Tables[0].Rows[0][0].ToString();
            this.TextBox2.Text = ds.Tables[0].Rows[0][5].ToString();
            this.TextBox6.Text = System.DateTime.Now.ToShortDateString();
            this.TextBox7.Text = ds.Tables[0].Rows[0][1].ToString();
        }
    else
    {
        //enable manager and chefs to select applied demands
        TextBox1.Visible = false;
        this.DropDownList1.Visible = true;

        Label2.Text = "Demand ID";
        //if manager
        if (ds.Tables[0].Rows[0][4].ToString() == "")
        {
            manager = loggedid;
            this.Label6.Text = ds.Tables[0].Rows[0][1].ToString();
            string ssql = "select * from Demand where dd_man_acc is NULL
and dd_chef_acc='True'";
            DataSet dss = DBServices.Instance.getData(ssql);
            this.DropDownList1.Items.Clear();
            if (dss.Tables[0].Rows.Count > 0)
            {
                for (int i = 0; i < dss.Tables[0].Rows.Count; i++)

                this.DropDownList1.Items.Add(dss.Tables[0].Rows[i][0].ToString());

                this.DropDownList1.Visible = true;
            }
        }
    }
}
```



```
        this.Panel3.Visible = true;
        this.Panel4.Visible = true;
        this.Panel5.Visible = true;
        this.Panel6.Visible = true;
    }

    //if chef only
    if (dept.Tables[0].Rows.Count > 0)
    {
        chef = loggedid;
        string qe = DBServices.Instance.getValue("select dept_id from
Department where dept_chef='" + chef + "'");
        DataSet empforchef = DBServices.Instance.getData("select * from
Employee where emp_dept='" + qe + "'");
        string ssql = "select * from Demand where dd_man_acc is NULL and
dd_chef_acc is NULL and dd_emp in(";

        for (int l = 0; l < empforchef.Tables[0].Rows.Count; l++)
        {
            if (empforchef.Tables[0].Rows[l][0].ToString() != chef)
            {
                if (l != empforchef.Tables[0].Rows.Count - 1)
                    ssql += "'" +
empforchef.Tables[0].Rows[l][0].ToString() + "',";
                else if (l == empforchef.Tables[0].Rows.Count - 1)
                    ssql += "'" +
empforchef.Tables[0].Rows[l][0].ToString() + "'";
            }
        }

        DataSet dss = DBServices.Instance.getData(ssql);
        if (dss.Tables[0].Rows.Count > 0)
        {
            this.DropDownList1.Items.Clear();
            if (dss.Tables[0].Rows.Count > 0)
            {
                for (int i = 0; i < dss.Tables[0].Rows.Count; i++)

this.DropDownList1.Items.Add(dss.Tables[0].Rows[i][0].ToString());

                this.DropDownList1.Visible = true;
            }
            DropDownList1.SelectedIndex = -1;
        }
        this.Label5.Text = ds.Tables[0].Rows[0][1].ToString();
        this.Panel3.Visible = true;
        this.Panel4.Visible = true;
    }
}
if (DropDownList1.Items.Count == 1)
{
    FillDemandInfos(DropDownList1.Items[0].Value.ToString());
}
}
```

Submitting changes (Update DB)

```
private void Update_DB_MethodInvoking(object sender, EventArgs e)
{
    string choosed = TextBox3.Text;
    if (choosed == "")
    {
        DateTime from = DateTime.Parse(TextBox4.Text);
        DateTime to = DateTime.Parse(TextBox5.Text);
        TextBox3.Text = getDaysBetween(from,to).ToString();
        choosed = TextBox3.Text;
    }
    string allowed = TextBox2.Text;
    if (Convert.ToInt32(choosed) > Convert.ToInt32(allowed))
    {
        Label4.Visible = true;
    }

    loggedid = Session["user"].ToString();
    string sql = "select * from Employee where emp_id='" + loggedid + "'";
    DataSet ds = DBServices.Instance.getData(sql);

    //check the type of user
    string s = "select * from Department where dept_chef='" + loggedid + "'";
    DataSet dept = DBServices.Instance.getData(s);
    //if manager
    if (ds.Tables[0].Rows[0][4].ToString() == "")
    {
        string q = "";
        int index = RadioButtonList2.SelectedIndex;
        string dd = DropDownList1.SelectedItem.Value.ToString();
        if (index == 0) //accepted
        {
            q = "update Demand set dd_man_acc='True' where dd_id='" + dd + "'";
        }
        else if (index == 1) //refused
        {
            q = "update Demand set dd_man_acc='False' where dd_id='" + dd + "'";
        }
        bool res = DBServices.Instance.ModifyData(q);
    }

    //if chef
    Else
    {
        if (dept.Tables[0].Rows.Count > 0)
        {
            string q = "";
            int index = RadioButtonList1.SelectedIndex;
            string dd = DropDownList1.SelectedItem.Value.ToString();
            if (index == 0) //accepted
            {
                q = "update Demand set dd_chef_acc='True' where dd_id='" + dd + "'";
            }
            else if (index == 1) //refused
            {
```

```
q = "update Demand set dd_chef_acc='False' where dd_id='" + dd + "'";
    }
    bool res = DBServices.Instance.ModifyData(q);
}
else
{
    //if employee
    //we should insert the demand
    string id = DBServices.Instance.getNextSequence("Demand");
    string newddid = "dd" + id;
    string emp = TextBox1.Text;
    DateTime dddate = DateTime.Parse(TextBox6.Text);
    DateTime ddfrom = DateTime.Parse(TextBox4.Text);
    DateTime ddto = DateTime.Parse(TextBox5.Text);
    string desc = TextBox8.Text;
    string insert = "insert into Demand
(dd_id,dd_emp,dd_date,dd_from,dd_desc,dd_to) values('" + newddid + "','" +
emp + "','" + dddate + "','" + ddfrom + "','" + desc + "','" + ddto + "')";
    bool res = DBServices.Instance.ModifyData(insert);
}
}

private void sendEmail_NotifyManager_MethodInvoking(object sender, EventArgs
e)
{
    sendEmail_NotifyManager.To="Manager@UTG.com";
    sendEmail_NotifyManager.Subject="New Absence demands";
}
}
```

VII. REFERENCES

- [1] Official Website of Microsoft Office; <http://office.microsoft.com/en-us/sharepointdesigner/>
- [2] Technical Resources of Microsoft; <http://technet.microsoft.com/en-us/library/cc263288.aspx>
- [3] <http://techtalkpt.wordpress.com/2007/06/27/workflows-for-windows-sharepoint-services-30-and-sharepoint-server-2007/>
- [4] Official Website CodePlex; www.codeplex.com/wss3workflow
- [5] MSDN Website; <http://msdn.microsoft.com/en-us/library/aa830816.aspx>
- [6] Official Website of CodeProject; <http://www.codeproject.com>
- [7] Michael Noel, Colin Spence; Microsoft SharePoint 2007 Unleashed; 800 East 96th Street, Indianapolis, Indiana 46240 USA