P-1927

DOCUMENT MANAGEMENT SYSTEM

By

S. Sivakumar
Reg. No. 71204621051

Of

KUMARAGURU COLLEGE OF TECHNOLOGY
COIMBATORE

A PROJECT REPORT
Submitted to the

FACULTY OF INFORMATION AND COMMUNICATION ENGINEERING

In partial fulfillment of the requirements
for the award of the degree
of

MASTER OF COMPUTER APPLICATIONS

July, 2007

KUMARAGURU COLLEGE OF TECHNOLOGY
COIMBATORE – 641006.

DEPARTMENT OF COMPUTER APPLICATIONS

BONAFIDE CERTIFICATE

Certified that this project report titled **Document Management System** is the bonafide work of **Mr. S. Sivakumar (Reg.no 71204621051)** who carried out the research under my supervision. Certified further, that to the best of my knowledge the work reported here in does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

Project Guide **Head of Department**

Submitted for the University Examination held on 03/07/07

Internal Examiner **External Examiner**

May 18th, 2007

To whomsoever it May Concern

This is to inform you that **S. Sivakumar** has successfully completed his project assignment titled **DOCUMENT MANAGEMENT SYSTEM** as a part of **MCA curriculum.**

As a Project Trainee, he started this project on **December 18, 2006** and completed it on **May 18, 2007.**

Please note, as per the company's policies and practices, the company retains ownership of the intellectual property rights concerning work undertaken during projects and disclosure of the source code and any other relevant information or data out of the organization is strictly prohibited.

S. Sivakumar designated, as project trainee will not be delivering the respective source code pertaining to his project.

For Caritor (India) Pvt Ltd,

BHAVANI DEVAIAH
MANAGER – HUMAN RESOURCES

ABSTRACT

In earlier days it was a difficult work for people to maintain and process the documents. Managing and monitoring the whole documents in an enterprise has always been a hectic but an essential process. And if it is manual then it becomes more problematic process. With a small office setup and limited work steps and documents the administration finds it easier to monitor their status without a centralized system, but in a well established firm monitoring and managing the documents without a centralized system is a very tedious task. "**Document Management System**" facilitates us to create and manage the documents electronically through out an organization.

In this **e-Workflow Management** it has various steps for processing a work item, First of all a domain server has to be set for every type of domains in the organization. The user of every domain will be recognized with individual roles and responsibilities. System should be able to get all the work items for processing, and then the work item will be allocated to various work steps in the work flow. Each work step should be able to perform various operations on the work item, such as editing/approving/canceling/ the work item, and adding /deleting notes and pages.

ACKNOWLEDGEMENT

I wish to express my sincere thanks to **Dr. JOSEPH V.THANIKAL,** Principal, Kumaraguru College of Technology, Coimbatore, for permitting me to undertake this project.

My deepest acknowledgement to **Dr. M. GURURAJAN,** Head of the Department, Computer Applications, Kumaraguru College of Technology, Coimbatore, for his timely help and guidance throughout this project.

I also express my thanks to **Mr. A. MUTHUKUMAR,** Project Coordinator, Assistant professor, Department of Computer Applications, Kumaraguru College of Technology, Coimbatore, for encouraging me to do this project.

I am greatly indebted to my guide **Mr. MANIKANDAN,** Senior Lecturer, Department of Computer Applications, Kumaraguru College of Technology, Coimbatore, for her valuable guidance and encouragement at every stage of this project work

I express my sincere thanks to **Mr. SASI ANBUROSE,** Project Manager, CARITOR (INDIA) PRIVATE LIMITED, for his support and assistance at various level of my project work.

Finally, I owe a lot to my beloved parents and family members and to my department staffs without their help and co-operation the project would not have taken a final shape.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CAL	-	Client Automation Layer
COM	-	Component Object Model
OLE	-	Object Linking and Embedding
OMD	-	Omni Desk
RCW	-	Runtime Callable Wrapper
CWM	-	Client Work Manager
CLR	-	Common Language Runtime
CTS	-	Common Type System
CLS	-	Common Language Specification
MSIL	-	Microsoft Intermediate Language
JIT	-	Just In Time

CHAPTER 1

INTRODUCTION

1.1. PROJECT OVERVIEW

The main aim of this project is to build a solution for capturing, indexing, storing and sharing information across the enterprise. Business documents can enter the system via high-speed scan, fax, or the Internet to create electronic documents. It has the following features added with it.

- Full administration from a centralized console,
- Modular components for easy customization,
- Easy and elegant communication with .NET clients,
- XML-based functions to deliver high performance.

The workflow solution has an easy point and click approach to creating workflows without writing code, and routing work items to designated personnel for timely processing.

The **"Document Management System"** will be a core system, which would leverage the process of document handling most effectively and easily based on the User performance for each work item assigned to him. The beneficiaries of the automated system would be primarily the management people of the enterprise. This project helps the enterprise to mange the documents that are sent across various work steps in the organization. This system is developed in .NET Framework version 2.0 by using the existing COM objects. It supports multi user functionality.

1.2. ORGANISATION PROFILE

Caritor India Pvt Ltd incorporated and headquartered in the USA, is a global IT Consulting & Systems Integration firm that delivers high-quality IT services to leading clients around the world. Caritor have been playing the role of a trusted IT partner to our clients since 1993 by helping them translate their IT vision into solid, measurable value. Today it have a presence that spans the USA, UK, France, the Middle-East, India and Singapore with over 3684 Caritorians working across these locations to deliver winning solutions for our clients.

Caritor offers cost-effective and intelligent IT solutions to clients in the Financial Services, Communications, Retails, Manufacturing, High-Technology, Travel & Transportation and Public Sector industries. It also offer IT services in the areas of Application Development, Application Management, Enterprise Business Solutions, Software Testing and System Integration through a global delivery model that ensures security, cost-effectiveness and quality for clients.

From inception it firmly believed in ensuring the highest quality and security for the IT solutions that Caritor deliver to clients. Caritor quality and security processes & certifications are a testament to this commitment – Caritor is one of the very select companies in the world to be certified at SEI - CMM Level 5, PCMM Level 5, CMMI Level 5, ISO:9001 and the BS7799 standards. As a part of continuing quality initiatives it also rolling out Six-Sigma processes internally.

As testament to the growth, capabilities and long-term customer relationships Citigroup Venture Capital International (CVC), a business unit of Citigroup Alternate Investments, came on board as partner for strategic initiatives in December 2004 through a private investment deal.

1.3. SYSTEM CONFIGURATION

1.3.1. HARDWARE CONFIGURATION :

Server:

Processor	:	Intel Pentium IV @ 3x GHz CPU
Hard disk capacity	:	40 GB
RAM	:	1 GB RAM

Client:

Processor	:	Intel Pentium IV Processor
Hard disk capacity	:	40 GB
RAM	:	256 MB RAM

1.3.2. SOFTWARE CONFIGURATION

Server:

Operating system	:	Windows XP/Vista
IDE	:	Visual Studio 2005
Frame Work	:	DOT NET Framework 2.0/30

Client:

Operating system	:	Windows XP/Vista
IDE	:	Visual Studio 2005
Framework	:	DOT NET Framework 2.0

1.4. PROGRAMMING ENVIRONMENT

ABOUT .NET

NET is the result of a complete makeover of Microsoft's software development products which form a part of the company's new strategy for delivering software as a service.

.NET PLATFORM

The building blocks of .NET platform include the three key entities that make it all possible: the CLR, CTS and CLS. .NET can be easily understood as a new runtime environment by a common base class library. The runtime layer is properly referred to as the Common Language Runtime (CLR). The primary role of CLR is to locate, load and manage .NET types. CLR takes care of number of low-level details such as automatic memory management, language integration and so on.

The primary goal is to acquaint with a number of .NET centric building blocks such as Common Language Runtime (CLR), Common Type System (CTS), Common Language Specification (CLS) and base class libraries.

.NET platform is a common type system which describes all possible data types and programming construct supported by runtime which also specifies how the entities can interact with each other and how they are represented in the .NET Meta data format.

FEATURES OF .NET

1. Fully interoperability with the existing Win32 code
2. **Complete and total language integration:** .NET supports cross-language inheritance, cross-language exception handling and cross language debugging.
3. **Common runtime engines share all .NET aware language:** One aspect of this engine is a well-defined set of types that each .NET aware language "understands".
4. A base class library that provides shelter from the complexities of raw API calls, and offers a consistent object model used by all .NET aware languages.
5. **A truly simplified deployment model:** Under .NET there is no need to register a binary unit into the system registry

SQL SERVER 2005

SQL Server is a 'back-end' database system designed for medium sized enterprises. The main advantages it offers are:

1. It allows a single point of access to the data so better control is possible and special tools and techniques can be applied in a centralized manner.

2. It allows appropriate division of processing between client and server.

3. It can support fairly large numbers of users simultaneously.

The data were made available to applications through views and stored procedures (involving for example queries and updates) and direct access to base tables. Triggers, rules and stored procedures are set up to run on the server in order to control access to and manipulation of data.

FEATURES OF SQL SERVER 2005

INTERNET INTEGRATION

The SQL Server 2005 database engine includes integrated XML support. It also has the scalability, availability and security features required to operate as the data storage component of the largest Web sites.

SCALABILITY AND AVAILABILITY

The database engine can be used across platforms ranging from laptop computers running Microsoft Windows 98 through large multiprocessor servers running Microsoft Windows 2005 Data Center Edition. SQL Server 2005 Enterprise edition supports features such as federated servers, indexed views and large memory support that allow it to scale the performance levels required by the largest Websites.

ENTERPRISE-LEVEL DATABASE FEATURES

SQL Server 2005 includes a set of administrative and development tools that improve upon the process of installing, deploying, managing and using SQL Server across several sites. SQL Server 2005 also supports a standard-based programming model integrated with the windows DNA, powerful and scalable systems. These features allow us to rapidly deliver SQL Server applications that customers can implement with a minimum of installation and administrative overhead.

SECURITY IN SQL SERVER 2005

There are different levels at which security can be enforced with respect to a site which has SQL Server:

1. The world external to site –LAN access, firewalls etc

2. The world internal to the site but external from SQL Server– this is controlled by having logins to SQL Server
3. The world internal to SQL Server – each database has a list of recognized users, which is a subset of the logins. Each user can have different permissions in each database.

CHAPTER 2

THEME OF PROJECT

2.1. SYSTEM ANALYSIS

2.1.1. LITERATURE REVIEW

I got an opportunity to do my academic project in Caritor. They allotted me a project, which I thought as a challenging one and useful for my career.

2.1.2. METHODOLOGY

Methodology followed in this project is Water fall model. Water fall model follows different phases in developing a software project.

Requirements specification

This phase is the process of gathering information. When the requirements are fully completed, one proceeds to design. In our system, we gathered information about the Business rules and made the proposed system with the application of new business rules.

Design

Design should be a plan for implementing the requirements given. In our system all the designing is done through .NET, which is a user-friendly for the end users. All the field level validations are done during the design phase itself. When the design is fully completed, an implementation of that design is made by coders.

Implementation

Implementation is the process of implementing our project by integrating various modules and making it as a workable module. Since the application is a dependent module, it will be implemented after the completion of the other related system.

Verification

Verification is the process of verifying whether the requirement is achieved or not.

Maintenance

Maintenance is the process of maintaining all the details of each domain such as document details.

2.1.3. SYSTEM REVIEW

2.1.3.1. EXISTING SYSTEM

The existing system is able to create the work item but not in an enterprise level. The system will not allow creating more than one domain for an enterprise, and all modules have been done with COM objects. This will give a dependency with technology. They have to make their enhancements in very few languages and it will communicate only with languages like Visual Basic and Visual C++. It uses old optical storage.

These are tedious processes and takes lot of time and inconvenience for customers. If the document collections are more, then the management needs to spend lot of time to get a document processed, and more over only one domain can be set for an organization.

2.1.3.2. PROPOSED SYSTEM

"Document Management System" in effect brings about the concept of electronic document management system across the organization. This can have more than one domain for a single organization. Multiple users can co exists to the domain and perform the entire task that has been assigned to them based on their roles. The roles are to be classified in two types, they are as follows.

- Administrator
- Operator

Each user would be able to perform only the tasks that are listed to user based on the roles assigned by the administrator.

Administrator:

Administrator is the User who would

- Create Users
- Authenticate the Users
- Assign Roles
- Archiving of Work items

Other Users:

Other Users are the one, who would

Create, Edit and modify the work item

Approve or Reject the work item

Sends and retrieves the work item to and from the work item Queue.

The **Client Automation Layer** plays a vital role in this Document Management System, as it works as an intermediate component between the Client side components and Server side components of Document Management System. It was originally written in COM technology. As it have its own drawbacks such as, No garbage collection, Platform dependency and Complexity in accessing database. Current system is being developed in .NET 2.0 frame work; this will have lot of advantages over the existing system.

Those advantages can be proper garbage collection, Side by side execution, Tracing and Logging, Supports Multiple Languages. (i.e.) it will support all managed and unmanaged languages such as Managed C++, Phython, ADA..., and with these things, it also provides Exception handling with the advantage of avoiding DLL hell.

2.1.4. FEASIBILITY ANALYSIS

Feasibility Study is the first stage of System Development Life Cycle of a project. Feasibility Study is a test of a system proposal according to its workability, impact on the organization, ability to meet user needs, and effective use of resources. The solutions are evaluated in the light of their **economic, technical and operational and time schedule implications** in an attempt to establish whether or not it is worthwhile for the organization commit further resources to the project. All these studies have been taken into considerations for developing "Document Management System".

TECHNICAL FEASIBILITY

It is a study of **function, performance, and constraints** that may affect the ability to achieve an acceptable system. Types of **hardware and software** are assessed to determine whether they can support the tasks required.

- **Basic configurations** of "Document Management System" use Microsoft products. The system is developed on these products are very common and very easy to use.
- "Document Management System" is made capable of handling all the transactions **using .NET components** which are developed using C# language.
- As most of the users of "Document Management System" are new to the components, they will be provided with adequate training, which is taken care by the customer support of the organization.

OPERATIONAL FEASIBILITY

Organizational, political and human aspects are considered in order to ensure that the proposed system will be workable when implemented. Impact of "Document Management System" workflow is assessed. Progressive developers can enrich this tool so that the resistance is minimized. It is ensured that the users have been involved in the development of this "Document Management System". The **naive users** are highly **supported** with the help provided by this "Document Management System" since it provides them with the necessary information in the form of informative and formatted screens. "Document Management System" provides **efficiency, accuracy and processing speed**.

ECONOMICAL FEASIBILITY

It is commonly known as **cost/benefit** analysis. It is an evaluation of development cost weighed against the ultimate income or benefit derived from the system or product. Only if the benefits outweigh costs, then only the system will be developed. The "Document Management System" satisfies this constraint successfully.

2.2 SYSTEM DESIGN

System Design defines the Input and Output data or values that are required for the application. Each part of the form and applications require the input values to get on to complete the system objective, which is document management in this case.

The components of the "Document Management System" flow are,

Type of Users:

AdminUsers:

These people will create and authorize users, and their roles.

Users:

These people will work in the workflow; they will approve, reject, edit, and complete the WorkItem.

User will have the following options,

- On right click on CALMaster root node the user has the following options in the menu
  - Create Client
  - Last Error
  - Version
  - OLE Markups Enabled
  - Rendering Thread Property
  - Is Domain Unified Logon
  - About
  - Exit
  - Test Events

Work item:

Documents, Imports, Images and Folders are called work item. These will move around the system. This can be sent to queue, and retrieved from queue.

User can click on the Workitem; he can do the following,

- Info
- Opened Item
- Retain
- Open
- Send
- Send To Default
- Place in Error
- Remove
- Delete Workitem

The System design is classified in to four categories as

- Usecase View
- Class View
- Sequence Diagram
- Input Design and
- Output Design

2.2.1 USECASE VIEW

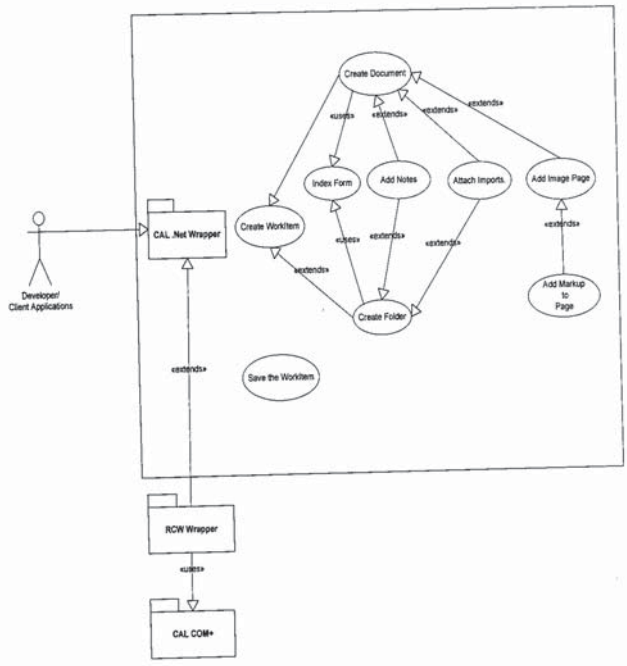

Fig 2.1 UseCase Diagram

1. Developer uses CAL .Net wrapper to create a WorkItem.
2. The CAL.NET custom Wrapper inherits from the RCW classes.
3. The RCW uses the Interop services to marshal calls from the Custom Wrapper to the CAL COM+ layer.
4. WorkItem can be a Document or a Folder.
5. Document and Folders will have the Index form data.
6. Documents and Folders may have Notes, Image pages and Attachments.
7. Image Pages can have markups.

Basic Flow

This use case starts when the actor does an action. An actor always initiates use cases. The use case describes what the actor does and what the system does in response. It is phrased in the form of a dialog between the actor and the system.

The use case describes what happens inside the system, but not how or why. If information is exchanged, be specific about what is passed back and forth. For example, it is not very illuminating to say that the actor enters customer information. It is better to say the actor enters the customer's name and address. A Glossary of Terms is often useful to keep the complexity of the use case manageable—you may want to define things like customer information there to keep the use case from drowning in details.

Simple alternatives may be presented within the text of the use case. If it only takes a few sentences to describe what happens when there is an alternative, do it directly within the Flow of Events section. If the alternative flow is more complex, use a separate section to describe it. For example, an Alternative Flow subsection explains how to describe more complex alternatives.

2.2.2 CLASS VIEW

The CAL.NET component will have its class view; the following are some of them.

- **IIFCALClient, Exposed Interface: ICALClient**

• Properties

Property Name	Description
ClientList	Holds the IIFCALClientList reference
DomainName	Name of the Workflow Domain. Read-only property.
EscalateQueue	Escalate queue for the current user. Read-only property.
IndexFields	Refers to IIFCALIndexFields.
Queues	Refers to a IIFCALQueues
RoleFiles	Refers to a IIFCALRoleFiles
SystemAdministrator	System administrator status of this user. Read-only property.
UserName	Name of the logged-on user for this session. Read-only property.
UserPreferences	Refers to a IIFCALUserPreferences
WorkflowVariables	Gets the list of workflow variables in the system. Read-only property.

WorkitemClasses	Refers to a IIFCALWorkitemClasses

Table 2.1 List of Properties

• Methods

Method Name	Description
CreateDocument	Creates a new document Workitem and returns a reference to its Workitem info.
CreateFolder	Creates a new folder Workitem and returns a reference to its Workitem info.
CreateImport	Creates a new import Workitem and returns a reference to its Workitem info.
Destroy	Destroys the current client session. This logs the user off the system. This call is optional.
FaxDocument	Submits a remote fax request for a document.

Table 2.2 List of Methods

• Events

Event Name	Description
AddItem	Notification sent when a workitem is added to the client list.
CloseItem	Notification sent when a workitem on the client list is closed.

OpenItem	Notification sent when a workitem on the client list is opened.
PageViewerNotification	Notification sent when an interactive close of Client API Viewer is requested.
RemoveItem	Notification sent when a workitem on the client list is removed.
SaveItem	Notification sent when a workitem on the client list is saved.
RemoveAllItems	Notification sent when all workitems on the client list are removed.
ModifyItem	Notification sent when a CAL object is modified as described below.
BeforeLogOut	Notification sent before the client object is destroyed.
AddFolderChild	Notification sent when a child is added to a folder.
RemoveFolderChild	Notification sent when a child is removed from a folder.

Table 2.3 List of Events

• CLASS DIAGRAMS

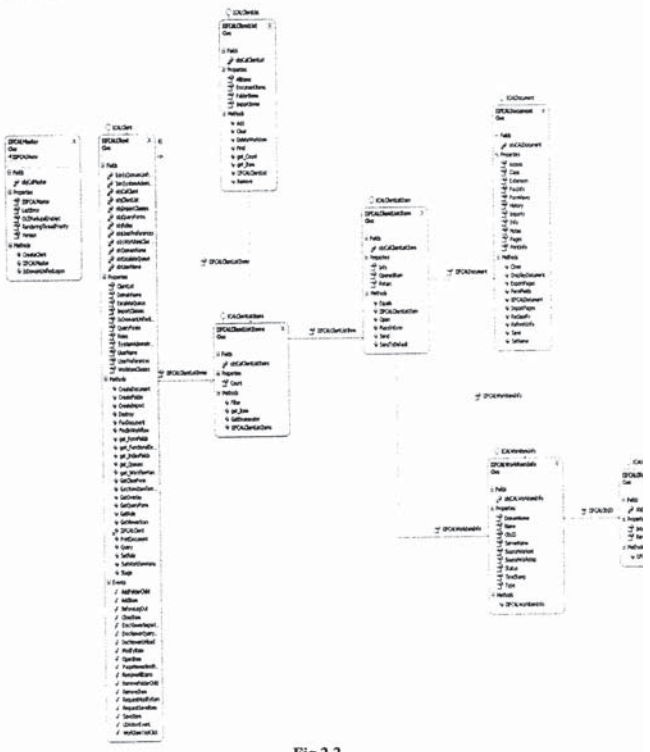

Fig 2.2
Complete Class Diagram for creating and saving workitem

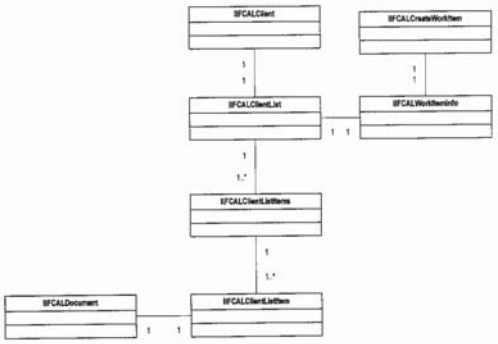

Fig 2.3 Class Diagram of IIFClient

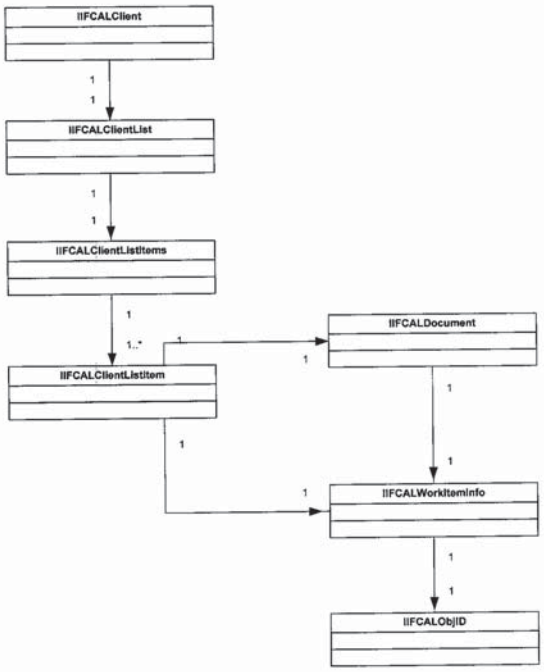

Fig 2.4 Class Diagram of IIFCALDocument

2.2.3 SEQUENCE DIAGRAMS

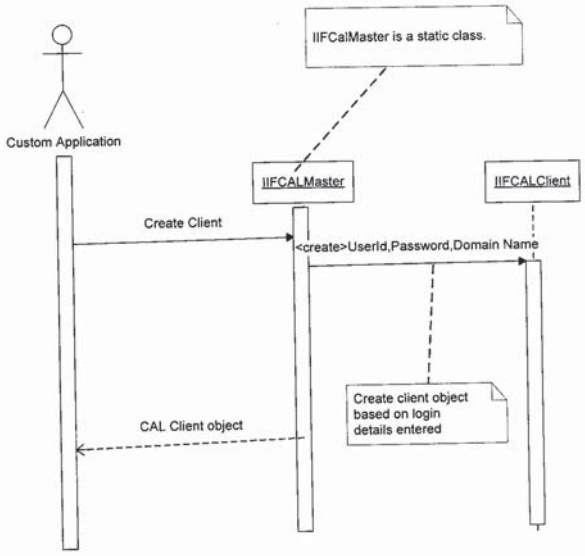

Fig 2.5 Sequence Diagram of Create Client

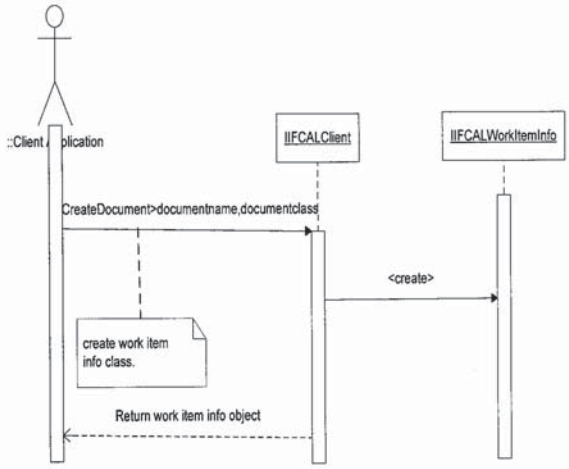

Fig 2.6 Sequence diagram of Create Document

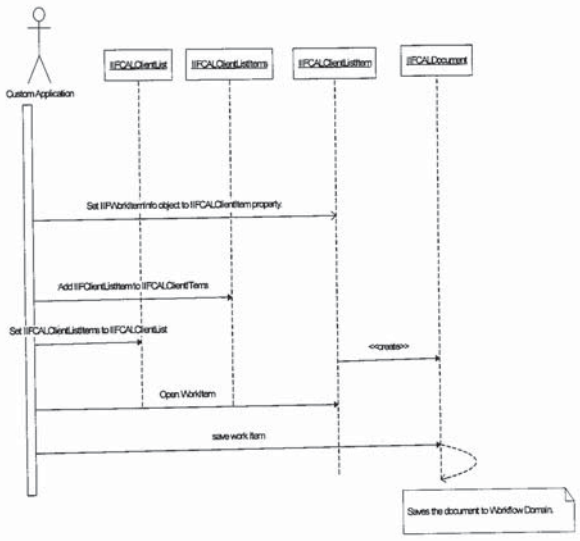

Fig 2.7 Save Document Sequence diagram

2.2.4 INPUT DESIGN

Input format must be designed in such a way to give on easy navigation throughout the component with out the violation of the input validation. Input design is the process of converting user-originated inputs to a computer-based format.

The design of the input depends upon the output of the developed system. To give a clear understanding of Input, a CAL.NET Test application is been developed with clear UI formats, that will give user an easy navigation through out the component. The windows for the **CAL.NET** Test for users based on the **Roles** would be

- Client Configuration
- Work item Management
- Queue Management
- Note and Pages Management

In User management the Administrator creates the Users and their Logins with the desired roles such that the Login Names are not repeated. He would have the rights to create, Modify and Delete the Workitem.

This section provides an alphabetical reference of the OLE automation objects that comprise the CAL interface. The properties and methods of each object are specified in detail. This section contains the following references:

- CALClient
- CALClientList
- CALClientListItem
- CALClientListItems
- CALDocFaxInfo
- CALDocPrintInfo
- CALMaster
- CALNote
- CALNotes Object
- CALObjID
- CALOutQueues
- CALPage

- CALDocument
- CALDocumentImport Object
- CALDocumentImports Object
- CALDocumentViewer
- CALEnumExtraInfo
- CALEnumExtraInfoItem
- CALEnumItem
- CALError
- CALFolder
- CALFolderChild
- CALFolderChildren
- CALFolderChildrenItems Object
- CALFormField
- CALFormFieldInfo
- CALFormFields
- CALFormFieldValueList
- CALImport
- CALImportClass
- CALImportClasses
- CALIndexFields
- CALPageMarkup
- CALPageMarkups
- CALPages
- CALPageViewer
- CALQueryResults
- CALQueue
- CALQueueContentItems
- CALQueueContents
- CALQueues
- CALRect
- CALVersion
- CALWorkflowViariable
- CALWorkflowVariableValue
- CALWorkitemClass
- CALWorkitemClasses
- CALWorkitemHistory
- CALWorkitemHistoryInfo
- CALWorkitemInfo
- CALWorkitemWorkstepList

Table 2.4 List of reference of the OLE automation objects

With these CAL classes we are going to retrieve the .net wrapper objects, this will be the input for the component.

Descriptions of the Inputs:

Users Management

Inputs - User Information's (Users Name, Password, Domain Name)

Workitem Management

Inputs – Documents Name, Obj Id.

Queue Management

Inputs – Document name, Class Name, Obj Id.

2.2.5 OUTPUT DESIGN

Output is the heart of the all software design. The developed system will be said successful only if the output system provides the necessary reports in necessary format. To keep our software in high grade we need to compose an efficient output design. Matching user requirements-the output must produce what the user needs or wants. Though it is a wrapper design, it is clearly evicted to the user with Test application.

2.3. MODULE DEVELOPMENT

The module **CAL.Net (Client Automation API layer)** that is responsible for generating, removing and editing workitems.

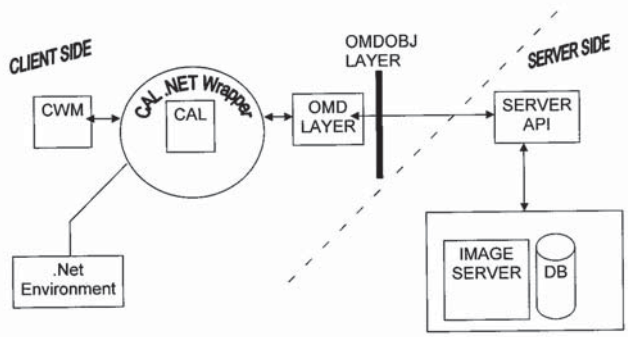

Fig 2.8 Block Diagram of CAL.NET

This module contains the following events,

CALClient Events	Signature
AddFolderChild	AddFolderChild (folderWorkitemInfo , childWorkitemInfo)
AddItem	AddItem (workiteminfo)
BeforeLogOut	BeforeLogOut(Options As DestroyConstants)
CloseItem	CloseItem (workiteminfo)
DocViewerImportActivate	DocViewerImportActivate(workitemInfo , ordinal As Long)
DocViewerQueryUnload	DocViewerQueryUnload(workitemInfo , Cancel Int, UnloadMode Int)
ModifyItem	ModifyItem(workitemInfo , ActionCode As ModifyItemConstants)
OpenItem	OpenItem(Item As Object)
RemoveAllItems	RemoveAllItems()

RemoveFolderChild	RemoveFolderChild(folderWorkiteminfo,childWorkitem)
RemoveItem	RemoveItem (workiteminfo)
RequestModifyItem	RequestModifyItem(workitemInfo As Object, ordinal, ActionCode As ModifyItemConstants, pCancel As Integer)
RequestSaveItem	RequestSaveItem(workiteminfo, &cancel)
SaveItem	SaveItem (workiteminfo)
WorkItemToolClick	WorkItemToolClick(workitemInfo , sToolID As String)

Table 2.5 List of Events

2.4. TESTING

Testing is a process of executing a program with an intend of finding an error. A good test case is one that has a high probability of finding an as-yet undiscovered error. A successful test is one that uncovers an as-yet undiscovered error. During the development of a project, errors of various types can occur at any stage. At each phase, different techniques are used to detect the errors. However, some error such as those that occur while collecting requirements and some design errors have also to be removed and the system tested for the successful working of any project. Since the code is the only product that can be executed and whose actual behavior observed, testing in the each phase is a process where on detected errors of the previous phase is also detected and corrected. Hence, testing performs a very critical role for quality assurance and for ensuring the reliability of the software. A comprehensive and efficient system is one that has taken into consideration all aspects of the project area and is free from errors or bugs.

Verification and Validation

Verification refers to the set of activities that ensure that system correctly implement a specific function. Validation refers to a different set of activities that

ensure that the system that has been built is traceable to customer requirements. In "Document Management System" verification and validation encompass a wide array of software quality assurance (SQA) activities that include formal reviews, quality and configuration audits, performance monitoring, simulation, feasibility study, documentation review, algorithm analysis, development testing, qualification testing, installation testing.

Unit Testing

Unit testing focuses verification effort on the smallest unit of software design in the module. In "Document Management System", Client Configuration, Work item Management, Queue Management and Creating notes and Pages are to be considered as individual units. Using the procedural design description as a guide, important control paths are tested to uncover errors within the boundary of the module. The relative complexity of tests and uncovered errors is limited by the constrained scope established for unit testing.

Unit testing is normally considered as an adjunct to the coding step. After source-level code has been developed, reviewed, and verified for correct syntax, unit test case design begins. Unit testing is addressed by a module, the number of test cases is reduced and errors can be more easily predicted and uncovered.

Integration Testing

Integration testing is a systematic technique for constructing the program structure while conducting tests to uncover errors associated with interfacing. The objective is to take unit tested modules of "Document Management System" and build a program structure that has been dictated by design. Modules are integrated by moving downward through the control hierarchy, beginning with the main control module. Here it will be done by the Administrator. Modules subordinate to the main control module are incorporated into the structure in either a depth first or breadth first

manner. As integration testing is conducted, the tester should identify critical modules.

System Testing

System testing is series of different tests whose primary purpose is to fully exercise the computer based system. Recovery testing is a test that forces the software to fail in a variety of ways and verifies that recovery is properly performed. Given enough time and resources, good security testing will ultimately penetrate a system. In performance testing, test run-time performance of system within context of an integrated system. Performance testing occurs throughout all steps in the testing process.

Interface Testing

Interface testing is performed to verify the correctness of interfaces between sub-modules such as Client management, Work item management and Queue Management, while performing integration, thus aiding master modules.

In the system the interface testing is performed to check for proper output on combining sub-modules and to ensure that there is no affect on global data.

Black Box Testing

In this approach of testing, test cases are prepared with the knowledge about the function that the product is designed to perform. Tests are done to check if the product is fully operational rather than checking each loops and conditions.
The following testing were done,

- All the possible inputs are given to the user functions on "Document Management System" and validated with corresponding outputs.
- The complete integrity of the data and files were checked.

White Box Testing

This testing is just vice versa of the black box testing. There we do not watch the internal variables during testing. But here we watch them too, along with the output. This will give clear idea what is going on during the execution of our system. The point at which the bug occurs was crystal clear and developer removed it off.

VSTS Testing

VSTS [Visual Studio Team System] is a Unit testing tool which is used in Dot Net Environment. The Unit test cases are written in C# and it is complied and loaded in VSTS. This tool will execute the test cases.

The Attributes of the VSTS are

- TestInitilize
- TestCleanup
- ClassInitialize
- ClassCleanup

TestInitilize and TestCleanup will be called at before execute every Testmethod and after executed a Testmethod respectively.

ClassInitilize and ClassCleanup will be called at before execute every TestClass and after executed a TestClass respectively.

CHAPTER 3

CONCLUSION

3.1. CONCLUSION

The project titled "Document Management System" was developed for CARITOR India Private Limited, Bangalore. All the objectives are met with contentment. All the modules are tested individuality and integrated. This system is compatible and user friendly. The system is tested successfully with the user. Validation checks made the system less error prone. The system was tested and implemented and the performance was found to be satisfactory. All necessary output was generated. It provided quick retrieval of data reducing the accessing time. Thus, the project was completed successfully.

3.2. FUTURE ENHANCEMENTS

Due to time factor and other constraints some features were not incorporated in this system. The system is designed such that it can be upgraded without much modification. The future enhancement of the system is the automatic entry of document through World Wide Web and large number of documents can be stored using data compression and various storage mechanisms. The system has much scope in the future and it can be developed to add more features to help the user.

APPENDIX - 1

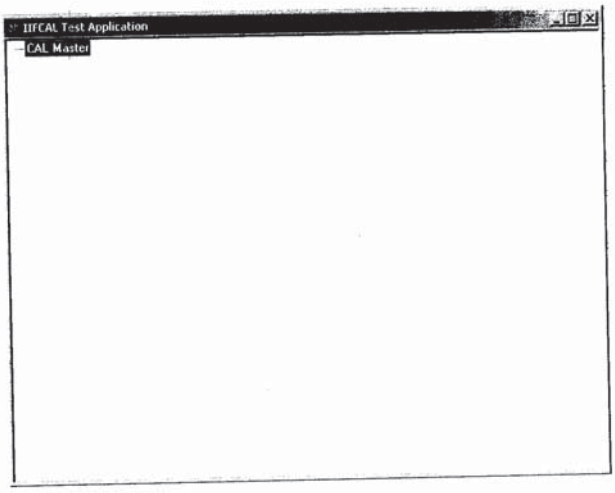

Fig A.1 Main Screen of CAL.NET Test

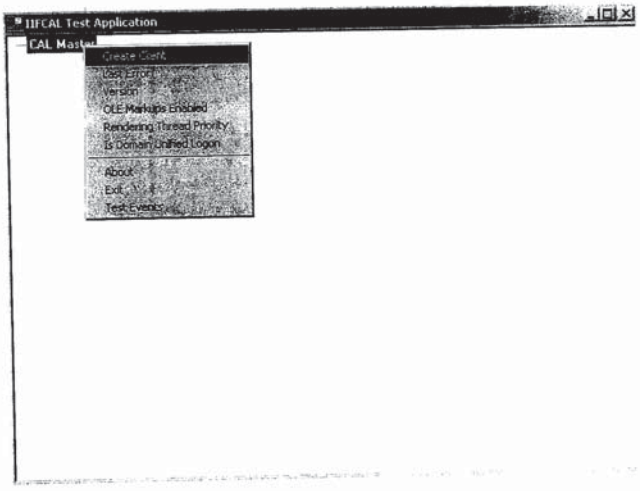

Fig A.2 Configuring Client

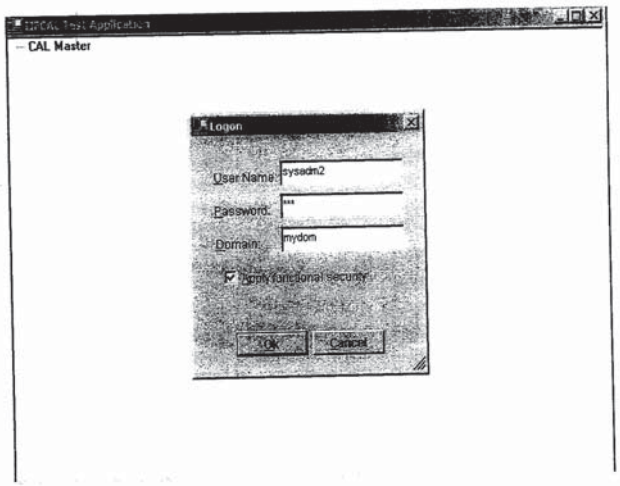

Fig A.3 Logon Screen for CAL.NET

Fig A.4 Client Display in user Node

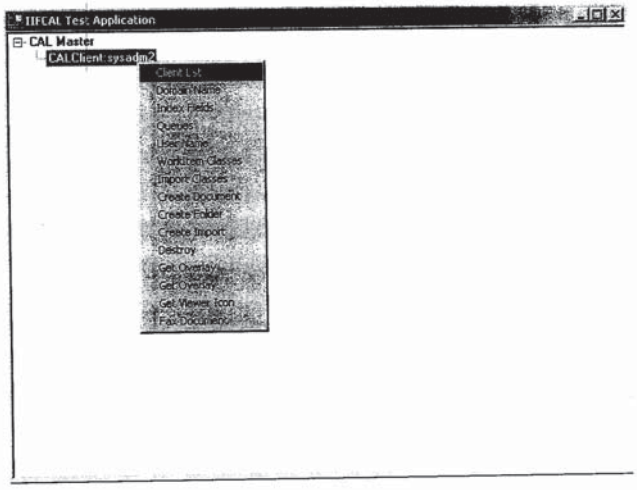

Fig A.5 Client Options

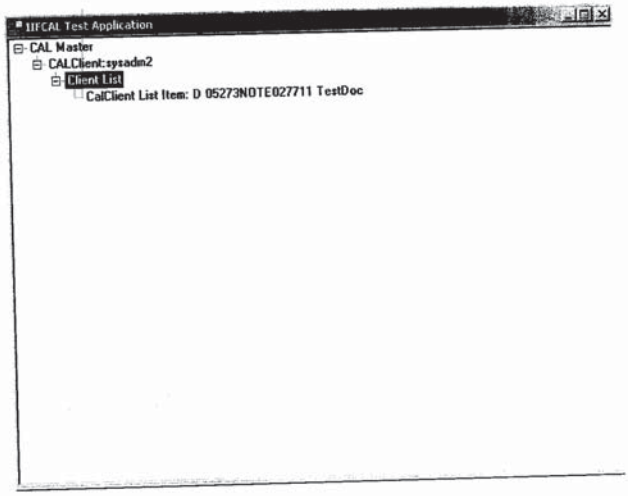

Fig A.6 Client List

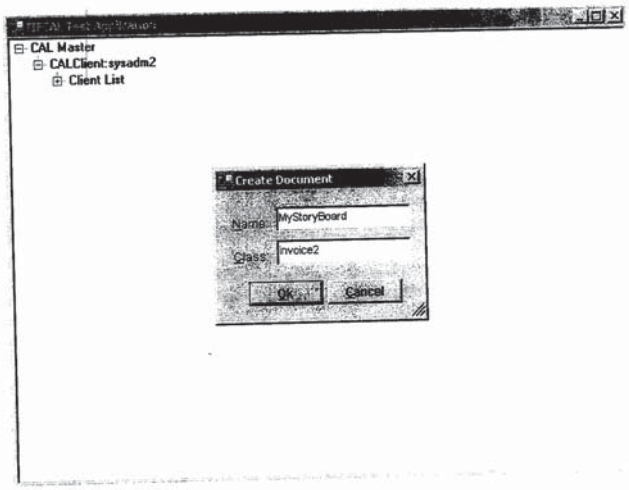

Fig A.7 Create a Document

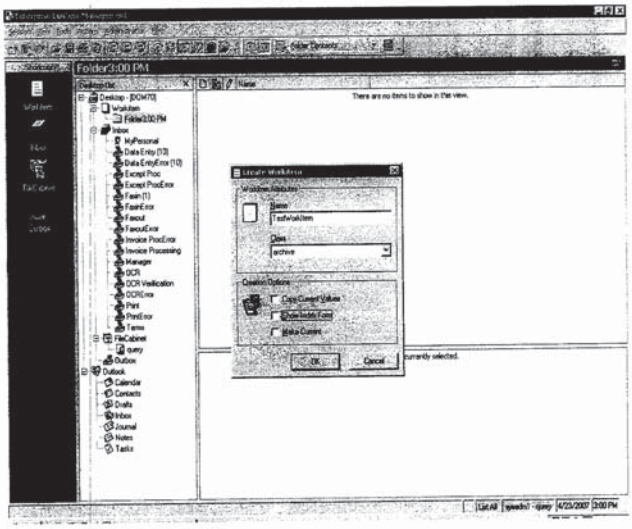

Fig A.8 Create a Workitem

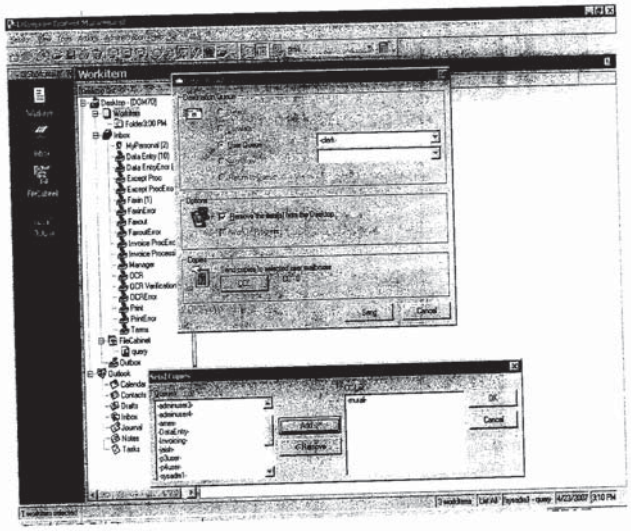

Fig A.9 Send a Workitem

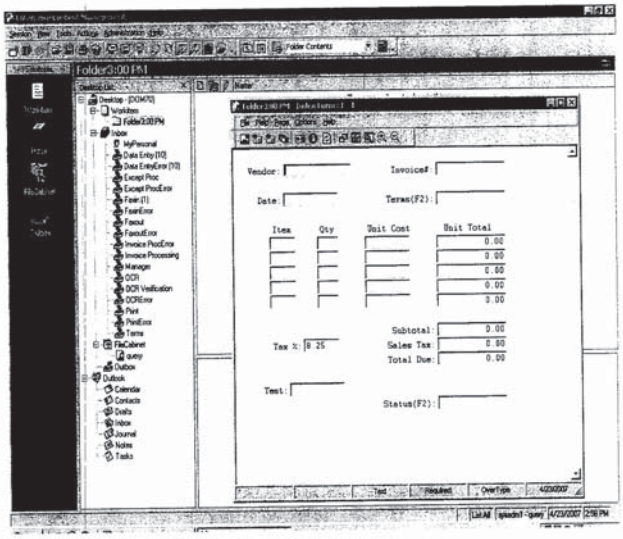

Fig A.10 Open a Workitem

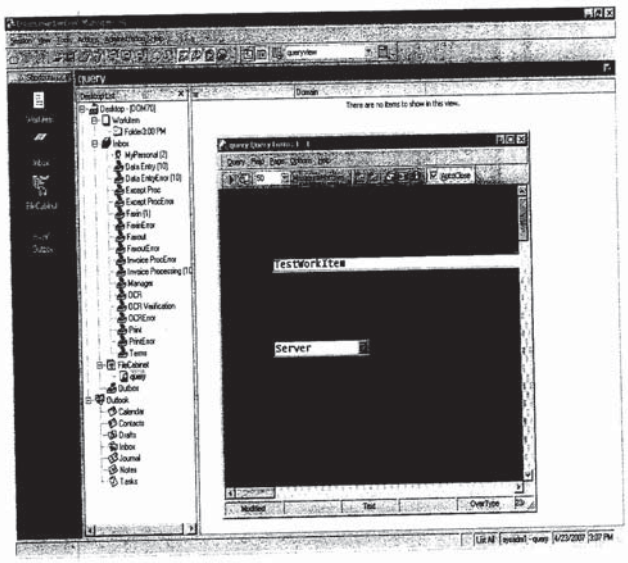

Fig A.11 Search a particular Workitem

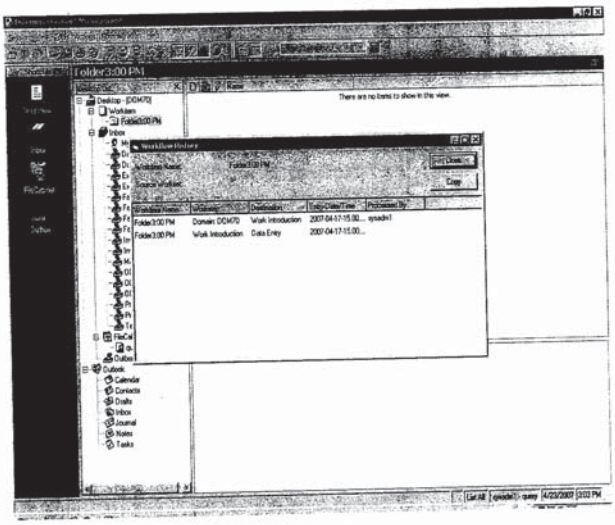

Fig A.12 Workitem History

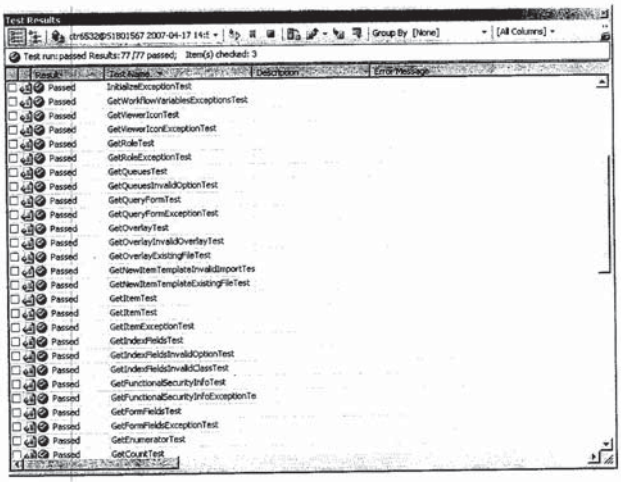

Fig A.13 Test Case Output

REFERENCES

Books:

1. Edward Yourdon (1989), **'Modern Structured Analysis'**, Prentice Hall Inc.,
2. Jesse Liberty (2002), **Programming C#**, 2nd Edition, O'Reilly.
3. Pressman Roger S., **Software Engineering – a Practitioner's Approach,** Tata McGraw-Hill, Fourth Edition.
4. Wagner, **Effective C#,** Oreilly Inc

URLs:

1. www.msdn.microsoft.com/net
2. http://www.gotdotnet.com
3. http://www.srtsolutions.com
4. http://www.oreilly.com/catalog/dotnetfrmess/
5. http://courses.cs.vt.edu/~csonline/SE/Lessons/Waterfall/