Developmental Outlines for a Desktop-based Online Tool for Collaborative Processes

Rupashree Majali

(ABSTRACT)

This research revolves around the rapidly evolving computer-based conferencing technology and the growing need for applications tailored for collaborative environments. It focuses on tools for collaborative processes using an existing software application – CyberQuest as a case study. CyberQuest is a problem solving and innovation support system its main intent being - to help people come up with ideas and with ways to implement them.

The *objective* behind this research is to illuminate the need for desktop based tools for online conferencing, which do more than just audio-video linking and is more than just a simple Internet based "meeting" software. It aims at analyzing the existing CyberQuest software and outline the methodology to be followed to convert it to a Web-based tool for problem solving and innovation support. Chapter 1 describes the existing features of CQ and the reasons as to why this application is suited for adapting to a network environment such as the Internet or an in-company intranet. It elucidates the steps involved in converting CQ to a Web based application and briefly describes the process of how files are processed on the Web. Chapter 3 gives a brief introduction to the Internet and the Web and distinguishes between the two. It also explains what intranets are and the main constituents of a web system. Chapter 4 and 5 explain the conversion issues in detail and list the various scripting languages available and the process of translating the application from Visual Basic into a scripted language. Chapter 6 surveys the results of using an application sharing software to share CQ over 2 or more computers. Lastly, Chapter 7 lists the findings and the future direction of all that is discussed in this study.

When fully developed, WebCQ (Web-enabled CyberQuest) would most certainly prove to be a valuable tool for enabling technology that will support collaborators in different disciplines by connecting them to a vast database of information to help them find solutions and improve performance in almost any field.

The study also includes a set of re-designed user interfaces for the WebCQ.

RESEARCH OVERVIEW

Computer-based Conferencing and Collaboration Systems and the Future of Desktop Conferencing

What is the Aim of this Research?

The purpose of this study is to shed light on the fact that computer-based conferencing systems are going to be the immediate need in the not too distant future when collaborative work environments become a way of life. Using the case of an existing software application called CyberQuest (CQ), it presents a detailed analysis of the process of conversion of this application into an effective Web-based shared application for collaboration. Because of its interdisciplinary nature, CQ could prove to be very useful as a desktop, online tool for idea sharing and group problem solving.

This research aims at providing a developmental outline and analysis towards converting or translating CQ into a Web based decision support system. This will enable multiple users to collaborate on a single session and enhance the co-operative thought process in the achievement of a certain, unified goal. The entire process of CQ revolves around and involves continuous inputs from the user and continuous processing of these inputs as a collection into a database at every stage. Theoretically, in a stand-alone application, which the present CQ is, all the data collected from user inputs is collected and stored in the install directory on the local machine. However, in the proposed -web version this input has to be stored in the server that will host the Web enabled version of CQ (hereafter referred to as WebCQ) and which the client machines will be accessing the WebCQ URL (Universal or Uniform Resource Locator) from. Since more than one user is anticipated to be accessing the site at any given time, the server has to be constantly updating user inputs all given times.

CyberQuest's User Interface (UI) consists of a series of forms the user is asked to fill out in the process of narrowing down the problem scenario. During this solution process, the answers that the user provides at every stage are stored, used and made available for the next step.

For this repeated update to be achieved there has to be a dynamic and continuous action-reaction process between the client machine and the CQWeb server. In other words there has to be a sequential reference to the last form submitted. Every user input fed from the client machine to the server has to have a 'handler' in the server handling that input which eventually decides what the fate of that input is to be and the desired action is taken.

This research seeks to achieve the following:

1. To research the need for and availability of an easily accessible, desktop-based application for idea sharing and group problem solving.
2. To explore the possibility of converting a part of CQ to be a functional web-based application.
3. To investigate the issues regarding the conversion of CQ to a web-based application, available over the Internet. Viz. issues like server-based applications, client-server architecture, etc. and point out the implications of such a conversion.
4. To modify the existing graphical user-interface (GUI) so as to be browser-based. To add new, necessary graphics and modify existing graphics and UI.
5. To explore briefly, the possibility of using an existing conferencing software application as an alternative to re-writing the existing stand-alone one to be web-based.

Background

Predicting the future is, has been and will always be a risky venture. This has never been more evident than it is with the emergence of the "information age" and the World Wide Web (WWW). The Web has been evolving so quickly that some say one Web year is the equivalent of three real years. So now, like the old concept of "dog years" we have this new concept of "Web years" to deal with!

When it comes to Computer-based Conferencing and Collaboration Systems and Web conferencing, though, there is one prediction that is easy to make: it is only going to get better. In particular, user interfaces will improve as JavaScript, dynamic HTML (DHTML), and similar developments will liberate software designers from the fences of HTML that they have been confined within. And performance will only outdo itself as bandwidths increase and enhancements to Hyper Text Transfer Protocol (HTTP) are adopted.

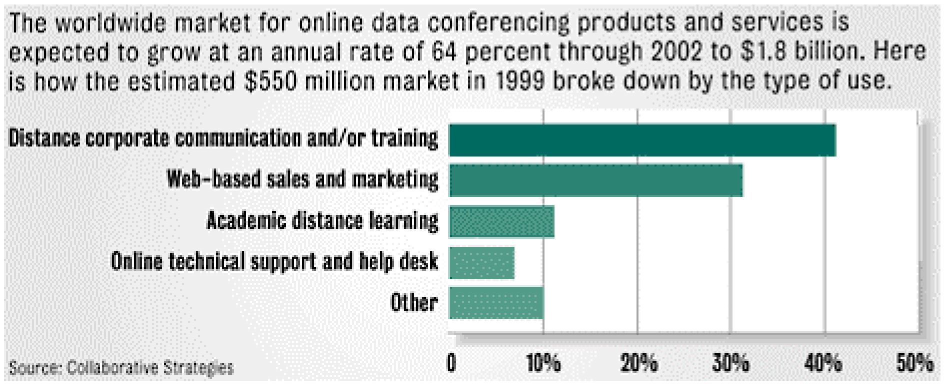
As communication technology rapidly evolves and the Internet increases its influence and grip over every discipline of the work place, most organizations and corporations find themselves constrained by time and space limitations. Limitations that arise from seemingly un-conquerable restrictions placed upon them by factors like geographic location and time zones.

It is almost imminently clear that soon (if it does not already), product development will involve teams of people from multiple organizations working together over networks, supported by computation and information services.   Consequently, there will be a growing necessity to explore the use of computer software and advanced network enabled applications software to enhance and extend group problem solving activity.   As the need to investigate appropriate structural processes for computer-based meetings increases, so will the demand for an online, assistive, problem solving system. As processes begin to demand unconventional access to helpful information, so will there be a significant need for problem solving mechanisms over the network not typically available in a single user environment.

Laboratories like CoLab [1] are already experimenting with new forms of computer-assisted collaboration. Many argue that the current tools for supporting meetings are archaic and unfriendly and many more are suggesting that modern computational technology be used for better support of meetings and cooperative problem solving.

Recently, however, an increasing number of organizations and consulting firms are endeavoring to overcome these and other restrictions like these by implementing new strategies, which enable their consultants and employees to keep in touch with one another even while they are constantly on the move. A growing number of these companies have turned to online conferencing technology to help overcome the hurdles of time and place. The technology—which lets participants collaborate in real time over the Internet or another network—will not eliminate all travel for employees. They will continue to work on-site with clients, traveling back to their main offices at the end of the week. But this online conferencing allows the team to stay in touch and collaborate on client documents and presentations, no matter what part of the planet the team member may be.

Fig. A Worldwide Market for Online Conferencing Products



*Based on a survey of 100 users in Fortune 1000 corporations Source: Collaborative Strategies [2]

Table A Features That Appeal to Companies Most in the Products They Buy

Companies using online data conferencing technology rate the following features as the most important in the products they choose
1. Presentation slides
2. Application viewing
3. Web-based learning
4. Instant messaging/buddy list
5. Online help/support for first-time users
6. Document display
7. Integrated e-mail invitation and notification
8. Simultaneous data and telephony
9. Live or text-based questioning and answering
10. Distribution of relevant documents prior to conference
*Based on a survey of 100 users in Fortune 1000 corporations Source: Collaborative Strategies [2]

Considering the speed at which the development of computer-based conferencing systems is being carried out, there will be a definite demand for utilitarian applications that are well suited for use in a conference/collaborative environment. The need for innovative software that facilitates idea sharing, enhances collaborative thought processes and aids in the generation of ideas is the emerging truth out of all the dust that the concept of online conferencing has kicked up.

Contemporary organizations utilize a vast array of computing technology to support their information processing needs. There are many successful computing tools designed as personal computing aids (word processors, spreadsheets, etc.) but fewer tools designed for collaborating groups of people. These latter tools are called *groupware* . Groupware is defined as "systems that support groups engaged in a common task or goal, and that provide an interface to a shared environment" [3]. This study addresses the inherently interdisciplinary nature of such groupware and how useful it would be. Successful implementation of groupware in an organization requires a fundamentally sound technological development in which the social and organizational environment into which the technology is being embedded must be given very careful consideration. Many groupware products associated with the Web have recently been introduced and by many researchers, it is a "hot topic".

And CQ is one such software application, which, if analyzed in detail presents a perfect conceptual basis for a web-based groupware application for problem solving and innovation support.

Why Conference on the Web?

For years we have been shackled to plain ASCII text and people have responded creatively to ASCII's limitations, inventing symbology like emoticons (signs like ;-), :-) etc.).

The Web's multimedia capabilities will greatly enhance conferencing. Many of us have longed to include pictures and sound in our messages, and the Web has finally made this possible. Hypertext links embedded in messages offer a convenient way to convey a message by letting the recipient follow the link and then return to the original message. Links could also be used to post a message in two or more conferences in which it is relevant. There are practical reasons to use the Web for conferencing also. There is an expanding, extensive plenitude of Web client and server software to choose from, supporting a wide variety of hardware. Much of it is free. No proprietary system is likely to match the Web in terms of uniformity and universality of service.

One of the Web's great strengths is that it provides a common user interface for Internet utilities like FTP, Gopher, and WAIS (Wide Area Information Systems). It is natural to extend this to conferencing, as well. People should be able to reach everything the Internet has to offer without leaving the familiar environment of their Web browser.

Finally, a conferencing system on the Web can be designed to scale well. Since the data can be distributed across any number of servers, there are no inherent limits to growth. [4]

Issues and Challenges

There are a number of problems to be solved before a Web-based tool for collaboration Following are two of the most challenging ones:
Compatibility Standards in networked environments: There is general incompatibility between T.120-based conferencing applications and firewall standards

Security: At this time, Web conferencing products and services offer varying levels of security. Almost all provide basic password protection for a given meeting. But, when it comes to encryption of data, the variations are greater. Overall, the move is toward more encryption, especially as corporations move beyond experimenting with Web conferencing and use it for serious business meetings.

CHAPTER 1

A Brief Description of CyberQuest

1.1 What is CyberQuest

CyberQuest is a problem solving and innovation support system its main intent being -
to help people come up with ideas and with ways to implement them.

At present, it has been tried in the following areas

•Technical: Test VLSI Chips Economically

•Management: Schedule Personnel in Health Services

•Social: Plan, Collaborate, and Measure Results for Welfare Reform

•Humanities: Rearrange Bach's Music

•Product Development: Design for Specialty Accent Fabric Pieces.

•Marketing/Sales: Improve Motor Oil Sales Through Gas Stations.

•Education: Learn Mathematics Through Real World Applications.

•Legal: Reduce Court Congestion and Delay

•Business: IBM, Exxon, Hoescht-Celanese, Merck, Sandoz, Union Pacific

•Government: US DOT, Florida DOT, NC Dept. of Human Resources, State Justice Institute, Naval Surface Warfare Center

•Nonprofit: Long Island Association, American Water Works Association, Rephidim Bible Institute

The advantages of using such a program are many:

- Generates numerous ideas in a very short time frame and this presents the pleasant situation of having a variety of solutions to choose from and explore.
- Helps generate better, more efficient solutions
- Clarifies concepts and as a result solutions are presented more lucidly. More often than not this results in better marketing of ideas and increased sales .
- Reduces costs by facilitating efficient, timely solutions also resulting in more time for innovation and development of new products.
- Smoothes out the transition between different stages of the project/problem being tackled and results in more room for intellectual enterprise by generating ideas.

The idea generation process is completed in 6 stages or steps, which are:

1. Problem Description & Analysis
2. Word Selection
3. Idea Generation
4. Idea Screening
5. Idea Packaging & Evaluation
6. Reporting

1.1.1 Stages Involved in the Idea Generation Process

Step 1. Problem Description & Analysis

This step involves:

- A brief description of the situation at hand
- Identification of the:

  --AIM to be achieved.

  --CLIENT to be served.

  --TIME HORIZON for implementing the resulting ideas.
- Analysis of the Current Situation

Step 2. Word Selection

This step involves:

- Choosing key words from a CyberQuest list to describe the aim to be achieved

  --Choose from - Subject Words (Nouns) like *Management, Perfection* etc

  --Choose from - Descriptor Pairs (Opposing Adjectives) like *New<>Old, Flexible<>Rigid*

Step 3. Idea Generation

This step involves:

- Choosing a concept database like *Economics, Art, Biology,* etc
- Having CQ search for analogous concepts

  --Those that have at least one key word the same as ones describing the aim
- Selecting one of the analogous concepts generated by CyberQuest
- Generating an idea based on the selected concept and your experience

Step 4. Idea Screening

This step involves:

- Categorizing each idea

  --Has it been done before?? If so, has it been successful??

  --Is more information needed? Is it a part of a contingency plan?
- Deciding if each idea is a *"Must"* or a *"Want"*

  --Meaning, it is either a necessity or an option
- Discussing and rating each idea in order of its importance in achieving the Aim
- Discussing and rating each idea in order of its ease (or difficulty) of implementation

Step 5. Idea Packaging and Evaluation

This step involves:

- Putting complementary ideas into more powerful packages
- Searching the CQ implementation databases for implementation concepts

  --Organization, Finance, Selling, etc
- Evaluating the resultant ideas package using CQ check lists

  --Rate of return? Type of managers needed? Etc.
- Assigning task responsibilities and scheduling

Step 6. Reporting

This step involves:

- Printing summary reports at any step in the process
- Saving information to files and creating reports as per user preference
- Adding the case to the case library -- so that others with similar problems can use your situational ideas

1.2 Current Limitations of CyberQuest (CQ)

CQ can be described as a brainstorming, ideas-concepts based innovation and decision support application that can be very useful for the generation of ideas, exchanging of these ideas and examining ways to implement these ideas (solutions). These very qualities make it ideally suited for availability over any sort of network. This network could be the Internet or it could even be an organization's administrative intranet [Chapter 3, section 3.1]. Needless to say, any sort of "bringing together" of people or conferencing built into CQ would only increase its practicability and usefulness to a great degree. If CQ were to be available over a network, this would enable users to share and exchange ideas and opinions simultaneously in real-time. This would greatly reduce co-ordination related costs and produce more cost effective solutions for problems. Making CQ available on a network would also eliminate, to a large extent, geographical limitations that a decision-making group or team often encounters. Users and project team members can access such a software remotely, from wherever they are located at that particular time, and electronically participate in a brainstorming session. All this can be done without the usual logistical details which, more often than not, involve time consuming and tedious ground work.

This networking capability, if added to CQ, would make it even more effective and efficient in obtaining results and solutions for current situations and problems. At the present time CQ is a stand-alone application and this limits its functioning as an idea-generation and implementation software. Following is a comparison between an existing session and a probable 'on-line' session in CQ. The following Table 2.11 illustrates the differences between the existing CQ and the proposed WebCQ:

Table 1.1 Comparison between present CQ and proposed WebCQ

Stand-alone version of CQ	**Network-enabled version of CQ**
Input limited to one user at a time	Can handle multiple users at a given time
Users have to print out information for reference and exchange of ideas/opinions	Users can access, exchange and share ideas on-line and store results locally on their hard disks
Requires physical co-ordination of resources and participants	Convenient electronic meeting/session co-ordination
Limited and localized access to application since it resides on an individual machine	Any-time, anyplace access to application since it runs in a browser

CHAPTER 2

Brief Description of Steps Involved in converting CQ to be a Web-Based Application

This chapter discusses the issues and considerations to be kept in mind while converting the existing CQ into a Web-based application. It outlines the basics of how files and pages are processed on the Web, the components that comprise a web system and web server connections. It also explains the interaction of these components during run-time, meaning when a request has been made.

2.1 Process of conversion of CQ to a Web-based application

CyberQuest is presently a standalone application meant to run off a single machine at a time. To convert it to a web-based program would involve adding client-server utility. This would entail:

- Deciding what scripting language will be best suited to convert CQ from its present standalone state to a web-enabled one.
- Researching whether the existing forms in the application can be used, after converting them to ActiveX documents, in the web-based version of CQ or would it have to be completely re-scripted and re-written to accommodate Internet connectivity.
- Dividing CQ's 'tasking' into client-side or server side. Differentiating between what functionalities need to be server-side ones and which ones could be run off the client (browser)
- Researching how to incorporate these converted ActiveX layouts, documents and forms to make a up a web-application.
- Determining the feasibility of the process of conversion. Whether it might be possible to adapt the existing Visual Basic code into a closely related web scripting language or would it be worth completely re-writing it into a totally new scripting language enable internet-connectivity.
- Identifying the database issues. How should ASP or any other web-based scripting language establish a connection to the database and which is the best type of DBMS to use? Presently, CQ uses Microsoft Access databases. Should Access be the database solution on the Web or can there be another solution like ORACLE or SQL(Server Query Language).

- Re-designing the Web-based application's user interface which is usually necessary because it will need to incorporate the browser into its interface.
- Deciding on browser compatibility issues – should the UI of the web-application be designed to be compatible with all browsers or can it focus its solution on a particular type of browser.
- Determining if it would have to be an Internet-based web application or an Intranet one or both?
- Exploring the incorporation of multimedia and associative database idea generators using a search mechanism

2.2 The Basics of Web File Processing

2.2.1 How Web File Processing Works:

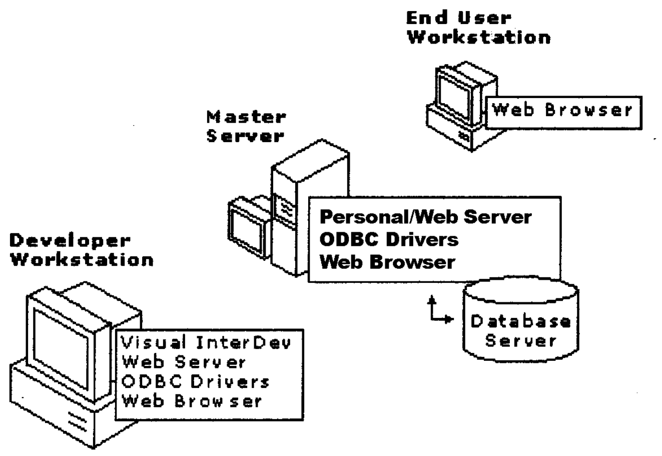
How and when Web files are processed depends on a variety of factors, the most significant one being the needs of the Web application being created. Refer Table 2.1

Table 2.1

Component	**Function**
Developer workstation	Creating a Web project, storing the project information, and connecting to other components
Database server	Storing database definitions and data used by the Web application, if required
Web server	Storing the master Web application, sending your Web application pages to the client, and processing server script and data requests, including connections to the database server
Web browser	Testing and previewing the pages and processing client script

The following fig 2.1 is a diagrammatic representation of the typical set of web system components needed for development of a web-based application.

Figure 2.1 Typical Set of Web System Components



2.2.2 Web Server Connections*:*

Typically the system components interact using HTTP, except for the database components which are likely to use a Local Area Network (LAN) connection or a Wide Area Network (WAN).

Any Internet application software communicates with the master Web server via HTTP. Since CyberQuest currently uses database connectivity, its Web avatar application needs to use a database server, and we need to add an ODBC database connection to the CyberQuest web project.

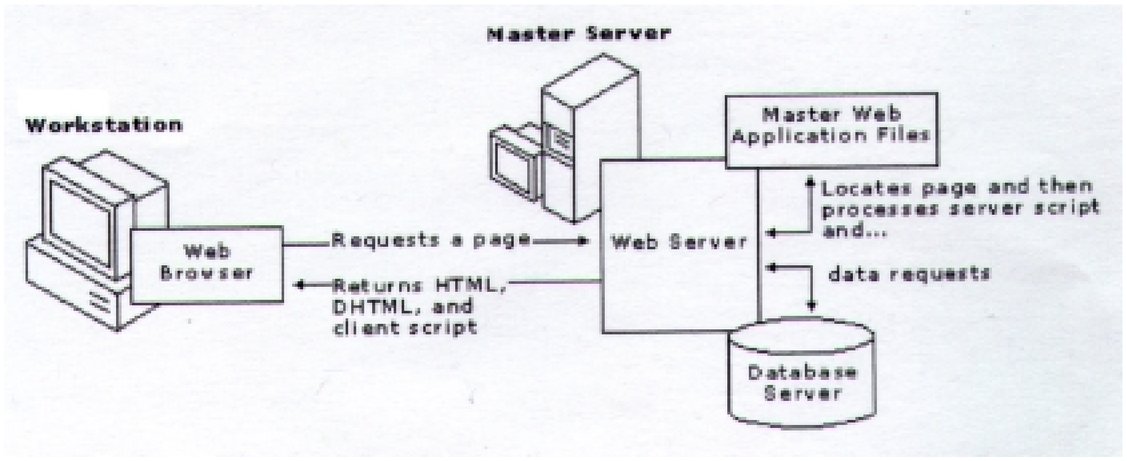
2.2.3 Run-Time Interaction of Components:

The simplest scenario for understanding how the components work together occurs at run time. At run time, the Web server is the central component in the Web application. It receives requests for Web pages from the user's Web browser, sends any requests for data or database commands to the database server, receives data from the database server, processes server script, and sends pages and data back to the user's browser.

When the Web application is ready for others to see, one can make the Web pages available using the master Web server or you can deploy the Web application to a *production Web server*.

The following Fig 2.2 shows the interaction between the system components at run time.

Fig 2.2 Web System Interaction at Run-time


Master Server
Workstation
Master Web Application Files
Locates page and then processes server script and...
Web Browser
Requests a page
Web Server
Returns HTML, DHTML, and client script
data requests
Database Server

CHAPTER 3

The Basics of the Internet and the World Wide Web

This chapter introduces the Internet and the Web, traces their history and describes the differences between the Internet and an Intranet. It elucidates the ideology behind the establishment of the Web and outlines the components of a web system such as hyperlinks, HTML (Hyper Text Markup Language), HTTP (Hyper Text Transfer Protocol) and browsers. It also explains the conceptual differences between the older static Web and the new dynamic Web along with the older mainframe systems and the more recent client/server architecture over the Web.

3.1 Brief History of the Internet and World Wide Web

Contrary to popular belief, the Internet and the World Wide Web are *not* the same thing. They are closely related undoubtedly, but there is one subtle, yet all too important difference. Briefly, the Internet consists of all the computers connected to it, the connections between them and the methods used to transfer information from one computer to another. It includes chat utilities, email, Use Nets, FTP and even NNTP (News Network Transfer Protocol) for news groups. The web comprises the sum of all the *interlinked information available* on the Internet.

The Internet

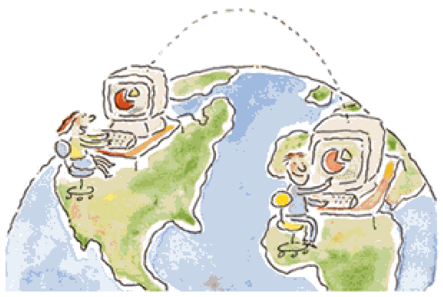
The beginning of the Internet can be traced back to the time of Cold War between the United States of America and the former USSR, when the threat of a nuclear war seemed very real and possible. The Department of Defense (DoD) realized the value of a communications system that could not be easily disrupted in the event of an attack. That fledgling project called ARPANET was started in 1968 and it achieved the unbelievable feat of connecting four computers in four different cities! [5]

It soon became considerably clear that this sort of network had many potential uses beyond the goals of the DoD, including allowing research scientists and engineers to exchange information and resources rapidly and efficiently. Is also became pretty obvious that no single, standalone network, would ever be able to provide this kind of desired functionality. Rather, connecting existing networks was seen as the way to go. In other words, an *internetwork* communication, or the Internet was suggested to be the next step. The early Internet primarily connected universities and research labs.

Major changes were in store for the Internet in 1991. The NSF (National Science Foundation), which had provided most of the financial support for the Internet, withdrew most of this support and opened the Internet to commercial organizations. Change was gradual at first because corporations and individuals were testing the waters and had doubts about the potential and the reach of this then new technology [5]. This was only in the beginning. Today the Internet has soared into importance like it had never known before and revolutionized the way we view, digest and disseminate information.

The World Wide Web

Figure 3.1



The Internet itself existed for quite a while before the Web was born. This early Internet was used primarily for the transfer of data files and for exchanging electronic mail. The birth of the web dates back to the period between 1989-1991 when its initial specifications were developed by Tim Berners-Lee working at the European Laboratory for Particle Physics in Switzerland.

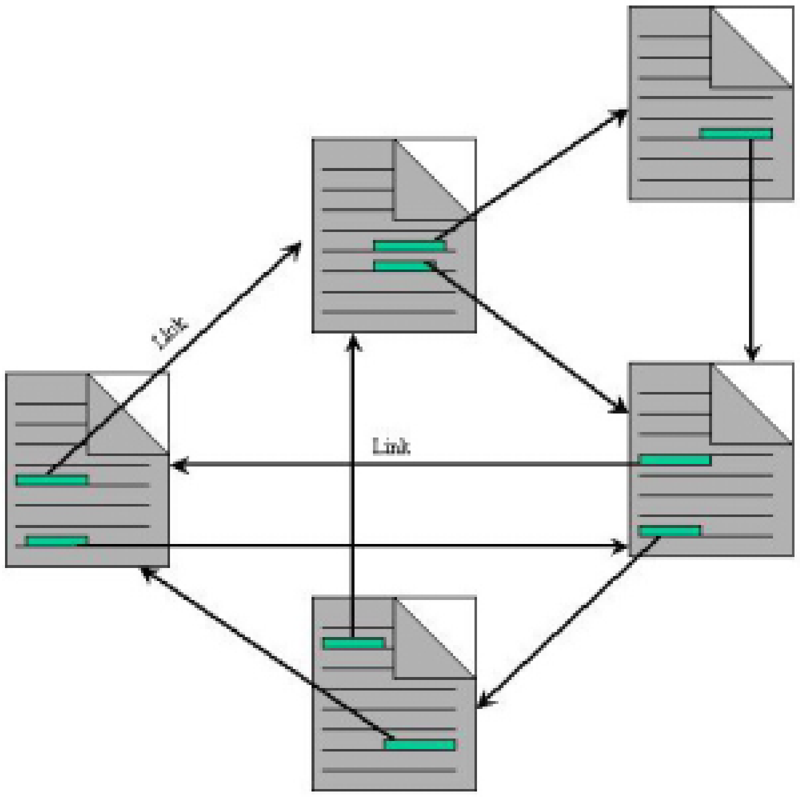
The reasoning behind the Web is based on the following:

- The information that people need is located in documents or files that are stored in computers. Almost all documents make a reference to other documents that contain related or supporting information.
- Those documents in turn reference other documents
- The underlying principle behind the establishment of the Web is to connect these documents not only to other documents they reference but also to provide an easy way to

view (link to) those other documents. Furthermore, an equally important utility would be the fact that it did not matter *where* these referenced documents were located. A document located on a computer in New York could link to a document on a computer in Bangalore as easily as it could link to one down the hallway. The concept of the hyperlink was indeed a powerful one, permitting the creation of a web of related information, hence the term World Wide Web.

Fig 3.2 illustrates the idea of hyperlinks between documents.

Figure 3.2 Hyperlinks between documents



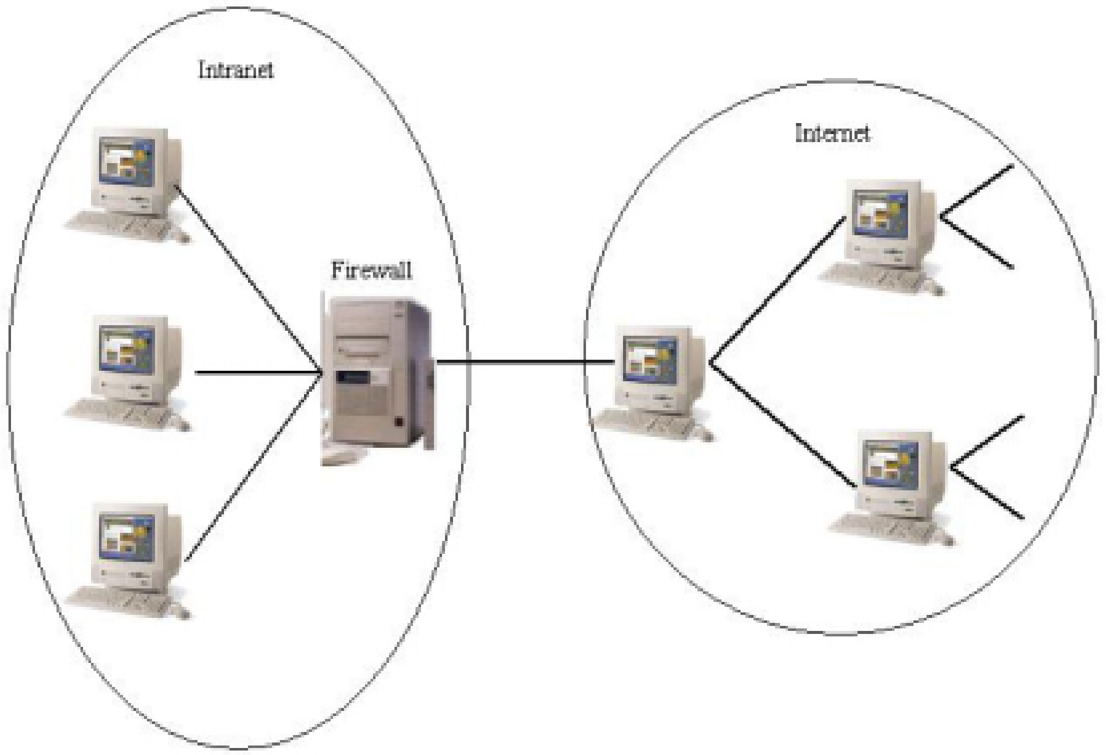
3.2 Internet Versus Intranets

An intranet is in its essential form, identical to the larger Internet in the technology and the protocols used. The difference is that the intranet is partially isolated from the Internet, so access to the computers on the Internet is limited.

A typical use for an intranet would be an organization or corporation that wants its employees to have access to the Internet but does not want the outside world of computing to have access to its computers and network. The network inside the corporation is connected to the Internet through a firewall, which is simply a computer with specialized software installed on it to prevent unauthorized access between the Internet and the intranet. People outside the company can access the intranet only if they have the required access privileges like a password into the appropriate domain. People inside the company can have unrestricted access to the Internet at large, or, in some cases a firewall might also restrict their access to prevent employee misuse and unsolicited Internet access. The relationship between an intranet and the Internet is illustrated in fig 3.3

From the programmer/developer's point of view, writing applications for an intranet is no different than programming for the Internet. There are some consequential minor yet significant differences, and for this reason some Visual Basic technologies are designed specifically for use on an intranet.

Figure 3.3 The Intranet and the Internet



3.3 Hypertext Markup Language

Hyper Text Markup Language (HTML) is the language of the World Wide Web. It is the standard that is used to create all Web documents. HTML documents contain only text and this text can be divided into two categories:

- ***Content*** is the subject matter of the document, i.e., the text that the user sees and reads when the document is displayed.
- ***Tags*** are the special codes that control how the content is formatted, define hyperlinks, display images and perform other tasks depending on what is specified inside of them. For example *<HTML>…….</HTML>*
  are tags which specify that all the subject matter within them are to be read as HTML.

3.4 Browsers

A browser is the one essential piece of software required so as to be able to make use of the Web. One basic job of the browser is to load an HTML file and display its contents with the formatting that is specified by the HTML tags in an HTML document. The other important task is to enable hyperlinks – in other words when a user selects a hyperlink, the browser has to retrieve the HTML document specified in the hyperlink and display it onscreen.
Why do we need to know this? In the context of CQ, it is important to understand the programming utilities created using VB have to be adapted so as to be functional over the Web. This implies that every programming functionality has to be carried out on the server and the resultant document has to be presented over the Web on the client machine in the form of HTML

3.5 What happens to HTML pages?

When a request for a page comes from the browser, the web server performs three steps:

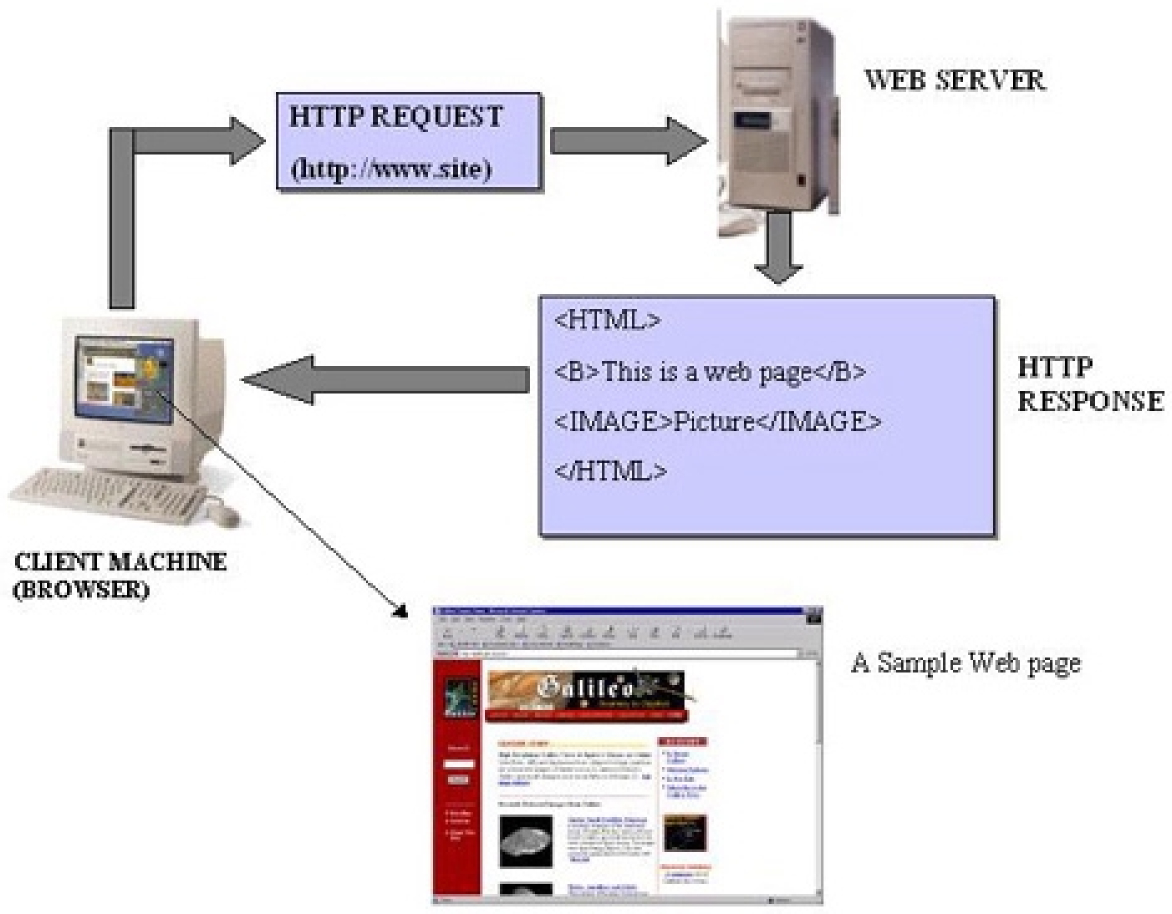
- Reads the request from the browser
- Finds the page in the server
- Sends the page back across the Internet to the browser

3.6 HTTP

Hyper Text Transfer Protocol (HTTP) is the protocol of the World Wide Web (Web). Because HTML is the language of the Web, HTTP is the protocol by which HTML documents are transferred over the Internet. It is important to note that TCP/IP is at work behind the scenes and HTTP is just an additional protocol that makes use of TCP/IP specifically for transmitting hypertext documents. An important aspect of HTTP is that it is a stateless protocol. This means that during an HTTP session, no discrete connection is established between the client and the server. HTTP provides for a single request/response interaction as shown in fig 3.4 This is also sometimes referred to as the 'Old Static Web'

In the days when the Web was in its infancy, the older more archaic browsers were only able to handle HTML. Hence it is quite obvious that the more capable the browser (client), the more complicated and 'bigger' it becomes, the slower it operates and the more memory it will devour.

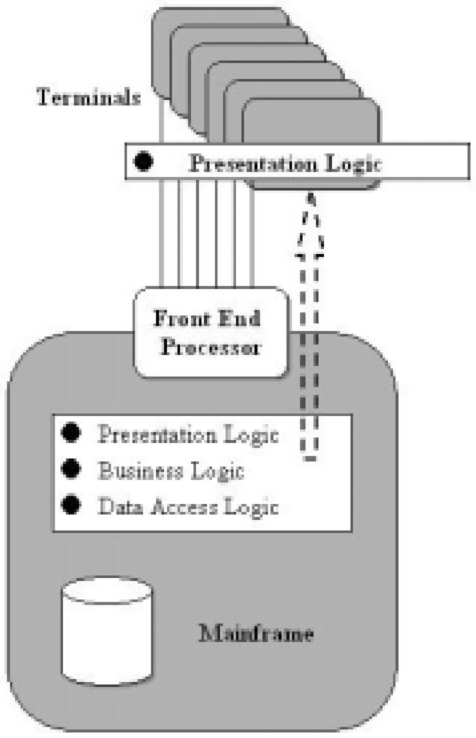
Fig 3.4   How HTTP Works/The Old Static Web



3.7   Client – Server Architecture and the New Dynamic Web

In the early days of networked computers, all of the programs and the data associated with them were stored on one gigantic machine known as the ***mainframe.*** This mainframe acted as the server and the clients took the avatar of mere terminals, which in many trades were scattered all around the building, and had no computing power of their own. They were simply connections to the mainframe and were used merely to execute programs on the server and display the results of these executions. Fig 3.5 illustrates this.

Figure 3.5 Mainframe Architecture



This networking system proved insufficient, as user demands expanded to accommodate more than just simple word processing applications; users demanded more computing power of their own. "Client/Server" architecture grew from an effort to eliminate or then decrease the use of the mainframe. The idea was to separate out the "functions" of the processing of the mainframe to affordable, dependable and maintenance-friendly systems.

3.7.1 Separating the Tasks

The solution to the mainframe problem was, as explained previously, Client/Server architecture. Client/Server architectures separated complex centralized applications into smaller, more manageable tasks or application logic [6]. These tasks and their application logic were divided into three layers:

- **Presentation logic**, which handled user interaction

- **Business logic**, which took care of the business rules
- **Data access logic**, which managed the storage and retrieval of data, and which ensured data integrity

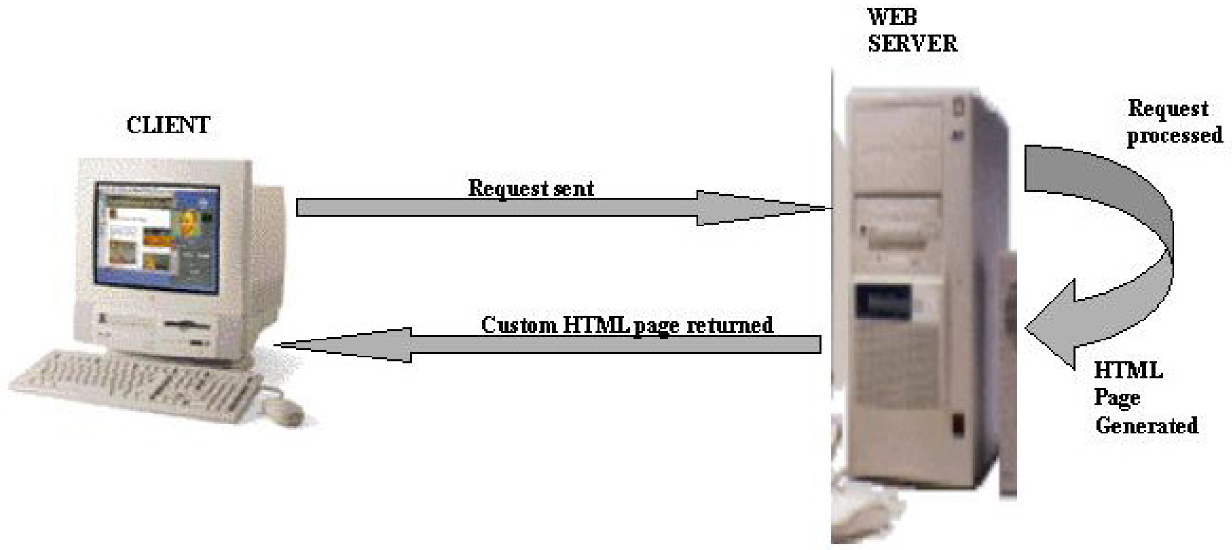
The earliest attempts at client/server applications targeted the replacement of the centralized mainframe by a Relational Database Management System (RDBMS). These RDBMSs provided rapid and highly economical solutions by transferring the presentation and business logic to the client and by enabling multiple clients to safely update the same set of data. This architecture was called “two-tier client/server architecture” because it divided the processing of the data between the workstation and server.

3.7.2 Web Client/Server Architectures

In the beginning, the Web was static. It consisted solely of static content made up of HTML pages or files located on server computers around the Internet. The user could retrieve pages and view them in the browser and that was all. As illustrated in fig 3.4 there was no other interaction between the client and the server and the only way the content of the pages could be updated was for the web site administrator to edit them manually.

It was soon realized that the web could be greatly improved if it could offer dynamic as well as static content. Dynamic content permits the user to send a request for specific information to the server. The Server responds to the request by running a ‘script’ or an application to generate a customized HTML page based on the user’s request. It is this custom page, with up-to-date information, that is returned to the user. Fig 3.6 shows how this works!

Figure 3.6 The New Dynamic Web



3.8 Summary

From this chapter, we now have an idea about how the Internet differs from the Web, what is the difference between the Internet and an intranet. We have seen what constitutes HTML and how files are processed on the Web and also have an idea of client/server architecture on the Web, how the old static Web functioned versus how the "dynamic" technology entered the HTML world.

CHAPTER 4

Detailed Description of Conversion Issues

After the introduction to the Internet and the Web and understanding the differences between the old static Web and the more recent dynamic Web, it becomes important to understand what brings about the "dynamics" in a dynamic Web page. This is where scripting languages enter the HTML picture. This chapter provides an insight to the understanding of the term "scripting" in relation to HTML, the different types of scripting languages in use today and the HTML tags used when scripting. It briefly introduces Java and also introduces Client-Side and Server-side scripts, their advantages over simple HTML and databases on the Web.

4.1 Scripting Languages

In order to add depth to the capability provided by HTML, we can intersperse the HTML code with commands that do not strictly belong to HTML at all. Instead these commands are written in one of a number of **scripting languages** that are available today. We distinguish these almost 'alien' commands, which are embedded within the HTML code, by referring to them as **scripts**. HTML allows us to include scripts at almost any point in the code provided we include them in a legal way.

Scripts are included in HTML by using the <SCRIPT>…</SCRIPT> tags.
Within the beginning <SCRIPT> tag has to be a mention of which scripting language is being used, example

```
<SCRIPT LANGUAGE=VBSCRIPT>

……Vbscript code goes here

</SCRIPT>

or

<SCRIPT LANGUAGE=JAVASCRIPT>

……Javascript code goes here

</SCRIPT>
```

Subsequently, when the page is run by the browser, each script is sent to the script handler (which is nothing but an application that can run a program on a remote computer in another language). On the remote computer (server), the script is then interpreted by a script engine. *Each scripting language needs its own interpreter to translate it.*

There are two main scripting languages, namely VBScript and JavaScript although they are not the *only* two.

JavaScript was the first client-side scripting language. VBScript is Microsoft's scripting edition based on their Visual Basic programming language. [6] Internet Explorer supports both JScript (Microsoft's implementation of JavaScript) and VBScript, however Netscape's Communicator 5.0 supports only JavaScript and VBScript may be added to Navigator with the help of a proprietary plug-in available from http://www.ncompasslabs.com

4.1.1 Where Java Fits in?

Java is an object-oriented programming language (OOPL). It was invented by James Gosling, in 1995. Java's main claim to fame is that it is completely portable, this allows the same Java program to run on any platform without having to be rewritten or recompiled. Java programs are able to run on PCs, Macs, main frames and even in consumer products, such as VCRs, security systems and telephones. Java's portability is achieved in the way that it is compiled and run. Java source code is first compiled into something called *bytecode*. This bytecode contains instructions to the Java Virtual Machine (JVM) that is resident on the platform that the program will be running on.

The JVM interprets the bytecode one instruction at a time and translates it into native machine code. The processor then carries out the specified operation. Any platform which has a virtual machine will run any Java program, using it's own instruction set. Most operating systems, like Windows, UNIX and MacOS have their own virtual machine, as do the web browsers, like Internet Explorer and Netscape Navigator.

Java is interpreted, so it does not run as fast as some other languages. However, Just In Time (JIT) compilers are coming out that take the bytecode, and compile it into native code on the fly, making Java run faster. Two types of Java programs can be written; *applications* and *applets.* Applications are standalone programs that do not require a browser to execute. Applets are Java programs which are embedded in HTML documents and are run by web browsers. When a page containing a Java applet is downloaded, the bytecode for that applet is downloaded along with it. This bytecode is then interpreted by the browser and run on the computer. Java can be used to create dynamic and interactive web pages because of this applet capability.

4.1.2 Client-Side and Server-Side Scripts

A script that is interpreted by the browser is called a **client-side** script. A client-side script is an instruction set that is processed by the client without contacting the server.

Client-side scripting involves the execution of the scripting language by the browser that interprets the web page. The main drawback of client-side scripting is that the method by which the script executes is completely dependent on the browser that executes the script. In other words, client side scripting is browser specific. The advantage of using client-side scripting is that it allows the developer to minimize server calls and executions. Web pages usually load up faster when there are fewer server-side scripts and this also prevents the server from getting overloaded with requests.

- ***Inserting a client –side script***

```
<SCRIPT LANGUAGE=VBSCRIPT>
……Vbscript code goes here
</SCRIPT>
or
<SCRIPT LANGUAGE=JAVASCRIPT>
……Javascript code goes here
</SCRIPT>
```

A script that is executed by the server is called a **server-side** script. A server-side script is very much like an HTML file. It includes HTML tags and text. When the browser comes across a piece of server-side script, it immediately sends it back to the server.

The server then reads the script , interprets it, executes the specified instructions and then returns the whole file as pure HTML. If the server-side scripts contain any ActiveX controls, which in the case of WebCQ almost certainly holds true, then these are interpreted by the browser. These interpreters usually reside on the client machine's hard disk. Most computers these days come with the Java Virtual Machine (JVM) engine pre-installed and the VBScript engine can usually be installed by installing the Windows 95/98/NT Option Pak which installs the PWS (Personal Web Server) extensions on the computer. It can also be obtained from Microsoft's web site.

- ***Inserting server-side scripts***

```
<SCRIPT LANGUAGE=VBSCRIPT RUNAT=SERVER>
……Vbscript code goes here
</SCRIPT>
```

note the `RUNAT=SERVER` instruction.

A point to note, is that developers are predicting that soon, the difference between client and server is going to become fuzzy. "When Java browsers can send as well as receive applets, computers will constantly switch between being a client and a server".[6]

4.1.3 Choosing a Scripting Language

As mentioned earlier, there are two scripting languages one can choose from – VBScript and JavaScript. At this stage, it must be pointed out that JavaScript should not be confused with the programming language Java. As previously Java is a stand-alone cross-platform language for developing applications. It is a simplified version of C++.

Since all of CyberQuest is written in Visual Basic as the programming language, VBScript along with ASP (Active Server Pages) for enhanced interactivity and easy database connectivity seems to be the obvious choice for the main reason being that it can more or less replicate most of

VBA's (Visual Basic for Application) programming functionality over the Web. However, there are some limitations in VBScript's programming capabilities and in those situations, these limitations can be easily overcome by incorporating either JavaScript or even JavaBeans which are simply applets created in Java that are included as classes in the code. Applets are another kind of scripting written in the Java programming language. There are numerous applets available in the market today which can be easily integrated into VBScript and HTML for enhanced dynamism on the Web. Hence, in instances where VBScript might prove to be ill-equipped or limited, Java can be used in the form of applets to compensate for VBScipt's handicap.

Here then, it must be mentioned that simply because CQ is written in VB does not carry the meaning that the developer has to be bound to and by VBScript. Web applications can also be created by combining VBScript, JavaScript and Java (in the form of JavaBeans).

4.2 Shortcomings of HTML in the Context of a Web-based CQ

CyberQuest is an *application*, a software program, which requires the user to take some action and provide certain inputs. These inputs, with the help of instructions coded into tools like buttons and menus, are then processed and the information is used to build up a database of case information. The only way to emulate interactivity and achieve the programmed interactive ability of a standalone application over the Web is to use some sort of active scripting which will provide the user with similar input tools like buttons and menus and will also accommodate the underlying programming to achieve the overall capability of the standalone application.

In the case of plain HTML, pages are simply composed of text and static images. They allow you to click on links or other images to get to other pages and these pages in turn display more or less the same kind of information. Innovations such as frames and tables have helped to improve the presentation and usability of these sites, but they have not made the all too critical step towards making these pages truly dynamic or in other words ***interactive***.

From general observation, web sites that contain Scripts written in either VBScript, JavaScript or even ASP, are more interactive and dynamic. They are often tailored to the individual user. The main difference between scripted pages and HTML pages lies in the fact that the web server treats them and deals with them very differently.

4.2.1 What Scripts Can Do that HTML Cannot?

The crucial difference is that pure HTML is merely interpreted by the browser, not executed on the server. By writing code that is to be executed on the web server or on the client server, the developer can create ‘web-applications’ that look, behave and feel just like conventional standalone applications.

One fact must be kept in mind at all times and this is that all scripting languages are designed to be used together with HTML to create dynamic web pages. In fact VBScript, used along with ASP, actually generates HTML code.

4.3 The Web Application

Thus far, from the explanation of client-server task separation on the Web, we have only understood how a scripting language like VBScript can help generate pages in such a way that the developer can specify how, where and when the execution of a particular procedure should be done by the server. It gave us an idea of how the server can dynamically create a page that is returned to the client. But a point to note is that in all those instances of interactions between the client and server, the request and the response to that request existed independent of each other. To convert an existing application like CQ to a web-based application, we need to understand the mechanism that allows the developer to relate two pages or a set of pages, together.

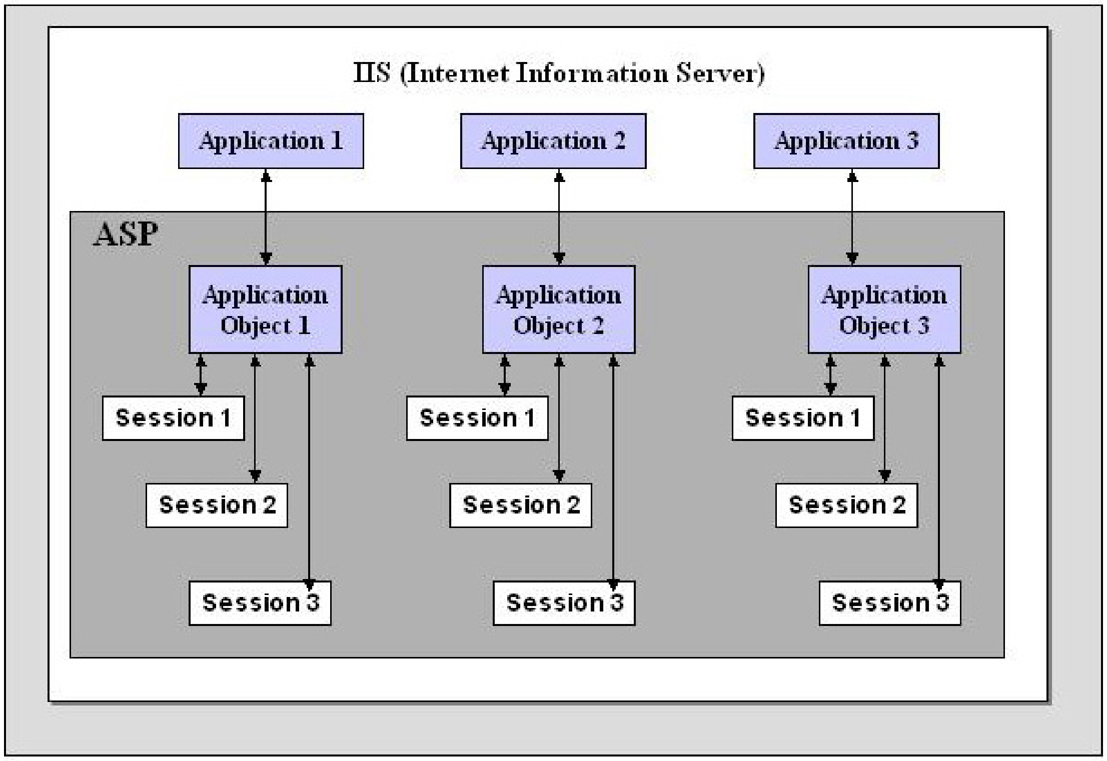
Until recently, the use of existing web technologies meant that the developer had to be satisfied by using cookies [6] and hidden form fields to pass information from one page to another. This is where ASP comes in. ASP, when used along with VBScript, provides a robust and flexible way to do this, by using the **Application** and **Session** objects.

- The **Application object** - is one which allows the tying together of all pages on a single web site into a consistent web application
- The **Session object** - is one which allows us to treat a user's interaction with the web as a continuous action, rather than a series of disconnected page requests.

As the Web moves from just serving up pages to providing access to dynamic information from a whole range of systems, the sites that users might access, are increasingly beginning to look more like a traditional application, such as one written in Visual Basic like CyberQuest. When proposed applications like WebCQ are designed using a combination of VBScript and ASP, each virtual directory that we create on the server, acts like an application. All of the pages in that directory, whether static or dynamically generated are part of the application. Due to this, the developer needs to have some control over the entire application as a whole and for this purpose is used the Application object.

A web server can host any number of Applications at a time, and each application has its own Application Object. This object stores variables and objects for application scope usage. Application-scope means that variables and objects can be accessed by any ASP page that is part of the application. Because any page in the virtual directory is part of the application, this means that any page in the directory has access to an application-scope variable.

Figure 4.3



In the above Fig 4.3, the **Applications** at the top refer to the collections of web-pages and objects defined on the WebCQ server as virtual directories. The **Application Objects** are the ASP objects that control access to WebCQ. The **Sessions** represent the individual client sessions with the application. Any number of sessions can be hosted by one Application

4.4 The Web Server

The web server is simply a computer that provides web services on the Internet, or on a local intranet. A basic web server is designed to locate, address and send out simple HTML pages. The web server makes web pages available to all other users, who can access these pages via their local network or via the Internet.

4.5 Databases and the Web

By far, one of the most powerful aspects of dynamic page content is the ability to connect a Web page to a database. Panoply of information can be and are maintained in a database. Information about products like price, available colors and sizes can be maintained and other information can be extracted from the database and incorporated in a dynamic Web page. Depending upon the requests made by the user, dynamic Web pages can customize a result in real-time and return search results to the user within seconds.

4.5.1 Why is Database Connectivity Important in the Context of WebCQ?

CQ, even in its present standalone version, makes extensive use of databases. At every stage, it either reads from, writes to or appends to the database. Microsoft Access is the database application it uses and with it the Jet Database Engine. The design of the Web-based version of CQ will have to possess efficient and robust database access utilities over the web and in between the client/user and the web-server.