8

Client/Server Technology and Web Services

8.1 INTRODUCTION

The Internet and expanded network connectivity established Client/Server models as the preferred form of distributed computing. When talking about Client/Server models of network communication using web services the broadest components of this paradigm become the web browser (functioning as the client) and web server. So, by introducing web services into the equation, Client/Server models become browser/server models. These models are Server-Centric, which make applications easy to load and install, but reduces rich user interaction. Server-Centric applications are currently available from standard browsers, making them convenient and popular with developers. Therefore, a way of enriching user experience is an essential frontier that must be developed for using browser/server models of distributed computing. One of the revolutions of the personal computer was usability or the ease with which humans could communicate with and configure their computers. This usually occurred through individual configuration and the User Interface (UI). Administratively, this was a nightmare because administrators had to install and maintain applications one machine at a time and manage multiple platforms. Individual installation and maintenance across platforms made web services seem like a good solution. Using HTML tools, developers moved toward giving applications global potential and a uniform protocol for management and deployment. The evolving trend was for developers to create applications that run on the server side, while web browsers became, for all intents and purposes, the standard client interface. Client processing power atrophied as execution of programs took place on central servers and output or responses were transmitted back to the browser through standard IP (Internet Protocols). This improved installation, administration, and maintenance. However, to be intelligible to the wide-array of platforms being targeted, web developers had to write in the lowest common denominator or the most widely accepted standards. This affected user experience negatively, while ensuring that applications could be deployed to the most users.

8.2 WHAT ARE WEB SERVICES?

8.2.1 Web Services History

The World Wide Web was developed by Tim Berners-Lee for his employer CERN or the European Organization for Nuclear Research between 1989-1991. In 1990, he wrote the program for the World Wide Web. This program created the first web browser and HTML editor. It was the first program to use both FTP and HTTP.

FTP (File Transfer Protocol) is used to transfer data over a network. Protocol is the set of standards that defines and controls connection, communication, and the transfer of data between two computer endpoints. It determines the format and defines the terms of transmission. HTTP is the protocol that supports hyper-text documents. Both of these protocols are necessary for communication over the Internet or World Wide Web. The source code for the World Wide Web was made public in 1993, making it available to everyone with a computer. The technology continued to develop and between 1991-1994, extended from communication only between scientific organizations, to universities and, finally, to industry. By 1994, computers could transfer data between each other through a cable linking ports across various operating systems.

The first web server, also written by Berners-Lee, ran on NeXTSTEP, the operating system for NeXT computers. The other technology authored by Berners-Lee that is required for Web communication is URLs (Universal Resource Locators). These are the uniform global identifiers for documents on the Web allowing for easily locating them on the Web. Berners-Lee is also responsible for writing the initial specifications for HTML. The first web server was installed in the United States on December 12, 1991 and at SLAC (Stanford Linear Accelerator Center), which is a U.S. Department of Energy laboratory. In 1994, Berners-Lee created the World Wide Web Consortium (WWC) to regulate and standardize the various technologies required for Web construction and communication. It was created to ensure compatibility between vendors or industry members by having them agree on certain core standards. This ensures the ability for web pages to be intelligible between different operating systems and software packages. After 2000, the web exploded. Till date, there exist more than 110 million web sites on the World Wide Web.

8.2.2 Web Server Technology

At the most basic level, the process for web communication works as follows: a computer runs a web browser that allows it to request, communicate and display HTML documents (web pages). Web browsers are the software applications that allow users to access and view these web pages and they run on individual computers. The most popular web browsers are Internet Explorer, Mozilla, Firefox, and Safari (for Mac). After typing in the URL (or address) and pressing return, the request is sent to a server machine that runs the web server. The web server is the program that delivers the files that make up web pages. Every web site or computer that creates a web site requires a web server. The most popular

web server program is Apache. The server machine then returns the requested web page. See the Fig. 8.1(*a*) and 8.1(*b*), depicting the Web Technology yesterday and today.

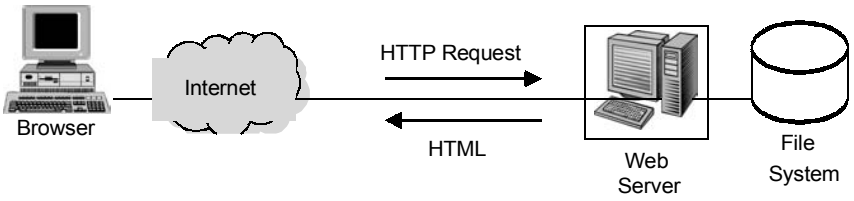

Fig. 8.1(*a*): Web Technology Yesterday

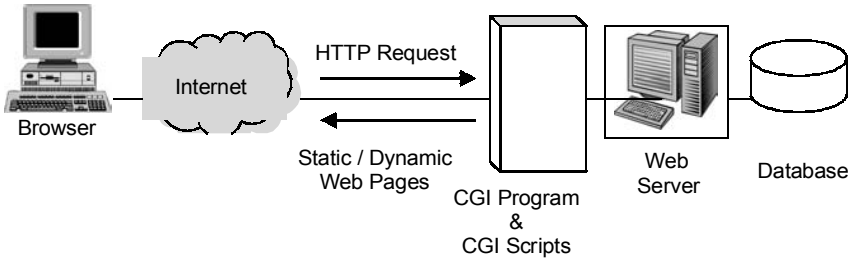

Fig. 8.1(*b*): Web Technology Today

Communication over the Internet can be broken down into two interested parties: clients and servers. The machines providing services are servers. Clients are the machines used to connect to those services. For example, the personal computer requesting web pages according to search parameters (defined by key words) does not provide any services to other computers. This is the client. If the client requests a search from, for example, the search engine Yahoo!. Yahoo! is the server, providing the hardware machinery to service the request. As previously mentioned, each computer requesting information over the Internet requires a web server program like Apache to render the search result intelligible in HTML.

Web servers translate URL path components in local file systems. The URL path is dependent on the server's root directory. The root directory is the top directory of a file system that usually exists hierarchically as an inverted tree. URL paths are similar to UNIX-like operating systems.

The typical client request reads, for example, "http://www.example.com/path/file.html". This client web browser translates this request through an HTTP request and by connecting to "www.example.com", in this case. The web server will then add the requested path to its root directory path. The result is located in the server's local file system or hierarchy of directories. The server reads the file and responds to the browser's request. The response contains the requested documents, in this case, web sites and the constituent pages.

Web Browser (Web Client)

A browser is a software (the most popular web browsers are Internet Explorer, Mozilla Firefox, Safari, Opera, and Netscape) that acts as an interface between the user and the inner workings of the internet, specifically the Word Wide Web Browsers are also referred to as web clients, or Universal Clients, because in the Client/Server model, the browser functions as the client program. The browser acts on behalf of the user. The browser:

- Contacts a web server and sends a request for information.
- Receives the information and then displays it on the user's computer.

A browser can be text-based or graphical and can make the internet easier to use and more intuitive. A graphical browser allows the user to view images on their computer, "point-and-click" with a mouse to select hypertext links, and uses drop-down menus and toolbar buttons to navigate and access recourses on Internet. The WWW incorporates hypertext, photographs, sound, video, etc. that can be fully experienced through a graphical browser. Browser often includes "helper application" which are actually software programs that are needed to display images, hear sounds or run animation sequences. The browser automatically invokes these helper applications when a user selects a link to a resource that requires them.

Accessing Database on the Web Page

Generally, it has been observed that a remote user's web browser cannot get connected directly with database system. But in most of the cases, the browsers are a program running on the web server that is an intermediary to the database. This program can be a common Gateway Interface (CGI) script, a Java servlet, or some code that lives inside an Active Server Page (ASP) or Java Server Page (JSP) document. The program retrieves the information from the page is an ordinary HTML document or the output of some script that Web-based database system. All these activities happen in number of steps explained below and also shown in figure 8.2:

Step1: The user types in a URL or fills out a form or submits a search on a Web page and clicks the Submit button.

Step 2: The browser sends the user's query from the browser to the Web server, which passes it on to a CGI script.

Step 3: The CGI script loads a library that lets it talk to an SQL database server, and it uses that library to send SQL commands to the database server.

Step 4: The database server executes the SQL commands and sends the request information to the CGI script.

Step 5: The CGI script generates an HTML document and writes the HTML document to the Web server.

Step 6: The Web server sends the HTML page back to the remote user.

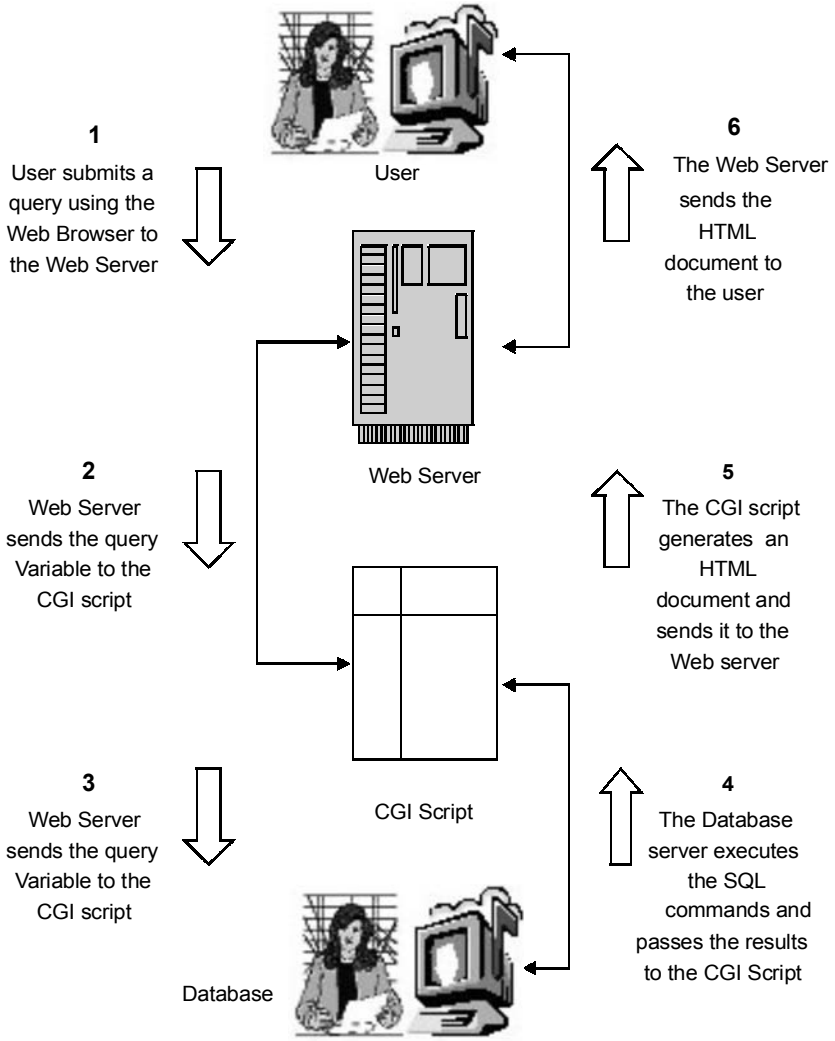

Fig. 8.2: Information Exchange between User and a Web-based Database

If the user has send some information to update a database. Then the CGI Script will generate the appropriate SQL commands and send it to the database server. The database server will execute the SQL commands and then inform the user about the result of execution. A typical example of a database query is search that you perform using a search engine. An example for an insert operation will be filled up a form on your browser to do an online registration for seminar. An example for an update operation will be updating your profile on a portal like naukari.com. In many web sites, even when you type a URL, web pages are generated for you using the information retrieved from a database.

For example, when you browse an online bookshop for a book. There are many details that are dynamic details like price, sales rank, shipping charges, availability, etc. So, it is easy to keep the details about all the books in the shop in a database and generate the web pages as and when requested by the user. The advantage of this is that the changes can be applied to the database and the users always get the up-to-date information.

8.2.3 Web Server

A computer that runs a computer program that is responsible for accepting HTTP requests from clients, which are known as web browsers, and serving them HTTP responses along with optional data contents, which usually are web pages such as HTML documents and linked objects (images, etc.). Although web server programs differ in detail, they all share some basic common features like HTTP and Logging, discussed below.

HTTP: Every web server program operates by accepting HTTP requests from the client, and providing an HTTP response to the client. The HTTP response usually consists of an HTML document, but can also be a raw file, an image, or some other type of document (defined by MIME-types); if some error is found in client request or while trying to serve the request, a web server has to send an error response which may include some custom HTML or text messages to better explain the problem to end users.

Logging: Usually web servers have also the capability of logging some detailed information, about client requests and server responses, to log files; this allows the webmaster to collect statistics by running log analyzers on log files.

In practice many web servers implement the following features also:

- Authentication, optional authorization request (request of user name and password) before allowing access to some or all kind of resources.
- Handling of not only static content (file content recorded in server's filesystem(s)) but of dynamic content too by supporting one or more related interfaces (SSI, CGI, SCGI, FastCGI, JSP, PHP, ASP, ASP.NET, Server API such as NSAPI, ISAPI, etc.).
- HTTPS support (by SSL or TLS) to allow secure (encrypted) connections to the server on the standard port 443 instead of usual port 80.
- Content compression (i.e., by gzip encoding) to reduce the size of the responses (to lower bandwidth usage, etc.).
- Virtual hosting to serve many web sites using one IP address.
- Large file support to be able to serve files whose size is greater than 2 GB on 32 bit OS.
- Bandwidth throttling to limit the speed of responses in order to not saturate the network and to be able to serve more clients.

Although web servers differ in specifics there are certain basic characteristics shared by all web servers. These basic characteristics include HTTP and logging. As previously, mentioned HTTP is the standard communications protocol for processing requests between

client browsers and web servers. This protocol provides the standardized rules for representing data, authenticating requests, and detecting errors.

The purpose of protocols is to make data transfer and services user-friendly. In computing, the protocols determine the nature of the connection between two communicating endpoints (wired or wireless) and verify the existence of the other endpoints being communicated with. It also negotiates the various characteristics of the connection. It determines how to begin, end, and format a request. It also signals any errors or corruptions in files and alerts the user as to the appropriate steps to take. HTTP is the request/response protocol used specifically for communicating. HTML documents which is the language hypertext or web pages are written in. However, responses can also return in the form of raw text, images or other types of documents. The other basic web server characteristic is logging. This is a feature that allows the program to automatically record events. This record can then be used as an audit trail to diagnose problems. Web servers log detailed information recording client requests and server responses. This information is stored in log files and can be analyzed to better understand user behavior, such as key word preferences, generate statistics, and run a more efficient web site.

There are many other practical features common to a variety of web sites. Configuration files or external user interfaces help determine how much and to what level of sophistication users can interact with the server. This establishes the configurability of the server. Some servers also provide authentication features that require users to register with the server through a username and password before being allowed access to resources or the execution of requests. Web servers must also be able to manage static and dynamic content. Static content exists as a file in a file system. Dynamic content is content (text, images, form fields) on a web page that changes according to specific contexts or conditions. Dynamic content is produced by some other program or script (a user-friendly programming language that connects existing components to execute a specific task) or API (Application Programming Interface the web server calls upon). It is much slower to load than static content since it often has to be pulled from a remote database. It provides a greater degree of user interactivity and tailor responses to user requests. To handle dynamic content, web servers must support at least any one of interface like JSP (Java Server Pages), PHP (a programming language that creates dynamic web pages), ASP (Active Server Pages) or ASP.NET (it is the successor of ASP).

8.2.4 Web Server Communication

Web servers are one of the end points in communication through the World Wide Web. According to its inventor, Tim Berner-Lee, the World Wide Web is "*the universe of network-accessible information, an embodiment of human knowledge.*" The World Wide Web is the global structure of electronically connected information. It refers to the global connections between computers that allow users to search for documents or web pages by requesting results from a web server. These documents are hyper-text based (written in HTML-Hypertext Markup Language), allowing users to travel to other pages and extend their

research through links. They are delivered in a standardized protocol, HTTP (Hypertext Transfer Protocol, usually written in lower case letters), making HTML documents intelligible across hardware and software variations.

This information travels through web servers and web browsers. The communication initiates from a user request through a web browser. The request is delivered to a web server in 'HTTP' format. The server then processes the request, which can be anything from a general search to a specific task, and returns the results in the same format. The results are written in HTML, which is the language web pages are written in that supports high-speed travel between web pages. HTML is also essential for displaying many of the interactive features on web pages, such as linking web pages to other objects, like images. An important distinction when defining web servers is between hardware and software. A web server is also a computer program (software) that performs the functions outlined above.

8.3 ROLE OF JAVA FOR CLIENT/SERVER ON WEB

Client server models provide the essential mechanisms for working with the Internet. In fact, most of the World Wide Web is built according to this paradigm. In client server models the web browsers run by millions of users are the clients. On the other side of the equation, is the web hosting systems that run at host sites and provide access to processes and data requested by the client. In this case, these hosting systems are the server. This definition is based on software programs, where the client is a program running on a remote machine that communicates with a server, a program running at a single site and providing responses to client requests, such as web pages or data.

Java is a programming language that has been developed specifically for the distributed environment of the Internet. It resembles C++ language, but is easier to use. C++ is a high level programming language that performs low level functions. In computing, low level functions are those that focus on individual components and the interaction between them as opposed to abstracted and systemic features (high level). Java, like C++, is a language that is multi-paradigm and supports object-oriented programming (OOP), procedural programming, and data abstraction. Object-oriented programming is increasingly being used in client server technology. It refers to a programming language model that is organized by objects rather than actions, data rather than logic. OOP identifies objects (sets of data) and defines their relationship to each other. These objects are then generalized into a class. Methods or sequences of logic are applied to classes of objects. Methods provide computational instructions and class object features provide the data that is acted upon. Users communicate with objects and objects with each other through specifically defined interfaces called messages. Java can create applications that are distributed between clients and servers in a network. Java source code (signaled by .java extension) is compiled (source code transformed into object code) into byte code (signaled by .class extension). A Java interpreter then executes this byte code format. Java interpreters and runtime environment run on Java Virtual Machines (JVMs).

The portability and usability that characterizes Java stems from the fact that JVMs exist for most operating systems, from UNIX, to Windows, to Mac OS.

Java is one of the most well-suited languages for working on the World Wide Web and the client server model is the primary models for working on distributed networks, of which the World Wide Web is just one. There is natural affinity between the two and this article will discuss some major characteristics of Java and how it can be utilized in building client server systems.

To understand how Java is used in client server systems it becomes essential to understand the major characteristics of Java. Syntactic of Java are similarity to C and C++ languages, Java is simple, simpler, in fact, than the languages it emulates. Java is also a robust programming language, which means it creates software that will identify and correct errors and handle abnormal conditions intelligently. Another major characteristic of Java is that it is object oriented programming, which was described above. OOP is characterized by three properties also present in Java programming: inheritance, encapsulation, and polymorphism. Inheritance is a major component in OOP which defines a general class and then specializes these classes by adding additional details in the already written class. Programmers only have to write the new features since the specialized class inherits all the features of the generalized class.

In OOP, encapsulation is the inclusion of all methods and data required for an object to function. Objects publish their interfaces and other objects communicate with these object interfaces to use them without having to understand how the encapsulated object performs its functions. It is the separation of interface and implementation. Polymorphism in OOP in general and Java specifically, is the ability to assign a different set of behaviors to an object in a subclass from the methods describe in the more general class. Therefore, subclasses can behave differently from the parent class without the parent class having to understand why for change itself. Multi threading is also an important characteristic of Java that increases interactive responsiveness and real time performance. Threading is the way a program splits or forks itself into two or more tasks running simultaneously. This allows for thread based multi tasking. Multi threading creates the effect of multiple threads running in parallel on different machines simultaneously.

Socket-based Client Server Systems in Java

Java builds client server systems with sockets. Sockets are the endpoints of two-way communication between programs running in a network. They are software objects that connect applications to network protocols, so they become intelligible. For example, in UNIX a program opens a socket that enables it to send and receive messages from the socket. This simplifies software development because programmers only have to change or specify the socket, while the operating system is left intact. In client/server models, the server contains sockets bound to specific port numbers. Port numbers identify specific processes that are to be forwarded over a network, like the Internet, in the form of messages to the server. The server only has to monitor the socket and respond when a client requests

a connection. Once connections have been made and bound to port numbers, the two endpoints, client and server, can communicate through the socket. Java sockets are client side sockets, known simply as sockets and server side sockets known as server sockets. Each belong to their own class within a Java package. The client initiates the socket, which contains a host name and port number, by requesting connections.

The server socket class in Java allows for servers to monitor requests for connections. As the socket communicates over the network, Java's server socket class assigns a one port number to each client in the server and creates a socket object or instance. This server socket instance creates a connection and produces a thread or string of processes through which the server can communicate with the client through the socket. Java web servers are built according to this model. TCP (Transmission Control Protocol) works in conjunction with IP (Internet Protocol) to send messages between computers over the Internet. When communicating over the Internet, the client program and the server program each attach a socket to their end of the connection. Then the client and server can read from and write to the socket. Java provides operating system sockets that allow two or more processes to send and receive data, regardless of computer location.

Java's RMI System

The other method for using Java to build client server systems is RMI. RMI stands for Remote Method Invocation. By using Java language and functions, programmers write object oriented programming, so that objects that are distributed over a network can interact with each other. RMI allows for client objects in JVMs to request services through a network from another computer (host or server). Client objects included with the request may call upon methods in the remote computer that change the results. Methods are programmed procedures attributed to a class that are contained in each of its objects (or instances). It is a characteristic of object-oriented programming.

Classes and, therefore, objects can have more than one method and methods can be used for more than one object. Responses then run as if they were local (on the same computer.) This passing back and forth of objects and methods attached to objects is called 'object serializations'. Simply put, RMI requests call upon the method of a remote object. As previously stated, it uses the same syntax it would locally. To make this intelligible to the servers or sites being called upon requires three layers: a client side stub program, a remote reference layer, and a transport connection layer. Each request travels down the layers of the client computer and up the layers of the server. The client side stub program initiates the request. Stub programs are small sections of programs containing routines from larger programs. It is a substitute for programs that may take too long to load or are located remotely on a network. They are provided by the server at the client's request. Stubs accept client requests and communicate requested procedures (through another program) to remote applications. They also return the results to the client or requesting program. These stubs mimic the program being called for service.

In Java, stub programs are also referred to as 'proxy'. Remote reference layers manage reference variables for remote objects, using the transport layer's TCP (Transmission

Control Protocol) connection to the server. Reference variables contain class data and therefore include methods. The transport layer protects and maintains end to end communication through a network. The most used transport layer is TCP. After the client requests pass through the transport layer, they pass through another remote reference layer before requesting implementation of the request by the server from a skeleton. These skeletons are written in high level IDL (Interface Definition Language). The server receives the request and sends the response along the same channels, but in the other direction.

8.4 WEB SERVICES AND CLIENT/SEVER/BROWSER—SERVER TECHNOLOGY

Web services and Client/Server technology made it possible for applications to integrate of separate components. These components might exist on separate machines, but they work together through network (Internet) communication. Applications using web services demonstrate the integration of components coming from multiple sources. This makes version management important. One of the benefits of having to focus on version management is to make developers aware of component dependencies and specific areas that require maintenance in each version. This allows developers to customize maintenance for application deployment. These distributed application components use universal formats provided by such programming languages as XML (Extensible Markup Language) and WSDL (Web Standard Description Language).

XML is the W3C standardized language that allows information and services to be written in a structurally and semantically intelligible way that both humans and machine on different platforms can understand. It can be customized with user or industry tags. WSDL uses an XML format to describe network services. These services are described as endpoints that use messages to transmit documents or procedure-oriented information. These operations and messages are abstract, but then they are attached to specific network protocols and message formats to enable communication. Distributed application components also use universal protocols like HTTP (Hypertext Transfer Protocol) and SOAP (Simple Object Access Protocol).

HTTP is the standard protocol for transmitting HTML files or documents. SOAP is a message-based protocol formatted in XML and using HTTP. It is a way for a program running in one operating system to communicate with a program running in another. For the purposes of this discussion, applications that take advantage of web services will be understood as 'balanced computing model' because these applications are designed to take the fullest advantage of both client and server capabilities in the most efficient way. This model of balanced computing improves traditional Client/Server model by not stressing one part of the system and ignoring the capabilities of other parts.

For example, traditional browser-server models were Server-Centric. They could handle user demands but did not take advantage of client-side processing that could predict user behaviour. To predict the behaviour from a diverse customer base requires headroom. Headroom is also known as the attenuation crosstalk ratio. It ensures the network

connections are strong and that signals on the receiving end of a communication are strong enough to overcome any interference. This provides a consistent and customized user experience regardless of unpredictable behaviour in the network.

8.5 CLIENT/SERVER TECHNOLOGY AND WEB APPLICATIONS

There is a gap in user experience between desktop applications and web applications. Desktop applications run on a single computer, while web applications run on the Internet. Since the invention of the Web, developers have been trying to design web applications that demonstrate the speed and interactivity of applications running on the client machine of a LAN (Local Area Network). Despite the explosion of web based applications in the 1990's (and continuing today), many users still prefer desktop applications. Like web sites, desktop applications access up to date information by connecting to the Web through the Internet.

However, desktop applications are designed with a much more refined sensibility and use PC power to customize requests from information stored on the desktop. This user experience is significantly better than when using remote web sites. Many are arguing that desktop applications will be the next wave in the Internet revolution.

So, with desktop applications breathing down their necks, web applications need to keep up the pace. Web applications are accessed by web browsers and run over a network, like the Internet. They are popular because of the power dominance of web browsers serving as thin clients.Thin clients are client applications or devices that simply initiate requests and provide input. They do very little of the processing, letting the server handle the heavy lifting by forwarding requests and contacting different nodes and networks. Web applications are responsible for web based e-mail applications like Hotmail, online shopping sites, online auction sites, wikis, and blogs. In traditional client server computing, every application contained a client program that provided the User Interface (UI) through which users would interact with/make requests from the applications. Each client program had to be installed individually on each user workstation. Web applications are different.

Web applications dynamically generate web documents. These are HTML/XHTML (Hypertext Markup Language/Extensible Hypertext Markup Language) documents transmitted over the Internet through HTTP (Hypertext Transfer Protocol). These documents or pages make up the web site. Using a standard server side scripting language like JavaScript allows for more interactive features in the user interface. Usually, each page is delivered as a static document, but the sequence in which they are presented is interactive. User web forms are embedded in the page markup, customizing responses. The web browser, acting as "universal client", interprets and displays the dynamic web pages. Web application UIs offer a variety of features. For example, the drag and drop feature allows for virtual objects to be moved from location through the mouse. This can create all sorts of actions, from copying to associating two objects to create a new action. By using application specific technologies like Java, JavaScript, DHTML (Dynamic HTML), and Flash, all sorts of graphic

and audio interactive features may be added to a UI. As previously stated, web developers are currently looking for ways to improve user experiences with web applications so they may closely resemble the performance of desktop applications. Remember, the user experience is the most often gauged by the usability of the UI or GUI (Graphic User Interface).

1st Generation Web Applications

The applications that are available now are typified by the technology used/presented in the 1st Generation Web Application; see the Fig. 8.3 shown. This might be characterized as a new form of electronic publishing, but it's richer in some ways than books because it's multimedia publishing. Today most home pages consist of text, photos, and graphics. By early 1997, however, it's likely that animation and 3D applications will be available. This technology is already very useful in information dissemination. Companies are replacing human resources manuals and maintenance manuals with browsers connected over intranets or the Internet to a server which contains the latest information sought.

A primary limitation of first generation applications is that there is no database management system, connected to the web server and the software does not keep the track of who is requesting information or of the last request from that user. It is a stateless connection. The addition of DBMS capabilities to the HTML processes on the server will allow HTML servers to have memory. As the leading DBMS vendors add connections for Web servers, it becomes possible for that server to remember who you are and what you have done from page to page and from visit to visit. The interaction, then, becomes a lot more intelligent and useful.

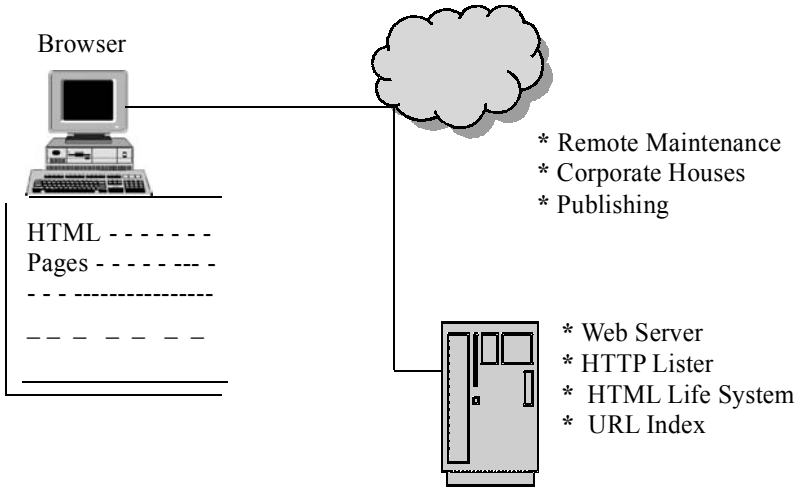

Fig. 8.3: Architecture of Ist Generation Web Application

2nd Generation Web Applications

First Generation Web Applications are quickly going to be joined by newer more capable environments that perhaps we can call the second generation. Several things will define this newer generation that are given as below:

- Support for active clients via downloadable applets (software components),
- Live DBMS links that enable the server to know who you are from page to page, and visit to visit, and
- True interactivity between the client and server (indicated by the two-headed arrow).

2nd generation Web applications have live DBMS connections on the server. Today what is mostly available for such support are SQL DBMS engines from companies like Sybase, Informix, IBM and Oracle. A problem with these engines is that SQL supports traditional business data types such as alphanumeric, date and time. No major SQL product supports extended and complex data types such as voice, maps or video at this time. That is a defect that will be remedied (probably) in the 1997 timeframe. All of the major DBMS vendors are making important strides in this direction.

This software component type of operation will be the first example of widespread use for distributed object computing. Sun's Java technology is the best known example of this approach. Sun describes its Hot-Java browser/Java language technology as "simple, object-oriented, distributed, interpreted, robust, secure, architecture-neutral, portable, high performance, multi-threaded and dynamic." The way this will work is for your browser on the client to have a Java (or ActiveX or C++) interpreter that can activate a component that has been downloaded to your client from the Web server. Your browser, then, becomes event driven and can exhibit various types of behavior.

From a software point of view, we will see both "inside/out" and "outside/in" approaches to write the code to mix applets and normal programming environments. BASIC compilers, for example, will be extended to support embedded HTML code. And, HTML will be extended to handle embedded Java, ActiveX and other component technologies.

In the Java environment pointers are not supported and that makes it impossible for any downloaded applet to address segments of memory outside the virtual Java machine that has been created inside your browser. This enforces security for your client and makes sure that any downloaded applets don't behave in a malicious fashion.

While talking heads and multimedia demos have been used to illustrate the operation of the Java/Hot Java environment, the real benefit to corporate users will come from a new type of application that hasn't been possible before–collaborative computing among strangers. Your server based applications are now available to everyone in the world; your programs can execute on their clients. If you want to do group scheduling, for example, until now everyone in the effort had to have the same client, something like Lotus Notes. With this new environment, it will be easy to accomplish wide area, multi-platform group scheduling via the Internet. The scheduling applet can be downloaded and executed in the component enabled browser.

8.6 BALANCED COMPUTING AND THE SERVER'S CHANGING ROLE

In balanced computing, processing is shared between clients, servers, and any other devices connected via a network. This distribution is inherent in the design since applications are already spread out on different machines and connected through web services. In balancing, the processing required by each new use is often shifted back to the user's system, thereby taking fuller advantage of client-side processing power. This allows for improved scalability, since the processing load is increased insignificantly by the addition of users.

Load balancing can also be achieved by building Service-Oriented Applications (SOAs) where components run on different nodes in multiple locations duplicate services on multiple nodes. This duplicating of services on multiple nodes also improves reliability since there is no single point of failure that will topple the entire application. Through balanced computing, platforms can take maximum advantage of computing and display capabilities on each platform. The virtual application which is balanced across multiple nodes remains transparent (its complex functions hidden) while the user utilizes his or her own collection of devices to run and view the application on the user end.

Balanced computing not only distributes the processing load, but changes the role of the server as well. Instead of computing so heavily, the server primarily directs traffic. Given rich clients and decent Internet connectivity, users directly contact databases instead of requiring server intervention. Rich clients are applicants in user computers that retrieve data through the Internet. As previously discussed, the proliferation of web-based applications replaced the user interface with the web browser. Scripting languages, like JavaScript or VBScript, were embedded into HTML to improve user interfaces (among other things). Java applets were also added. But nothing could compete with the user experience of using an application built from its local environment. Developing technologies like improved web-page scripting languages and AJAX (Asynchronous JavaScript and XML), made web browsers function more like rich client applications.

Another method used to reduce demands on the server uses a connecting device to re-direct previously server-side processing to the client. This depends on the client device capability and Internet connectivity. If these are weak, the server picks up the slack, making application performance consistent on both the client and server ends.

User Experience and Development

Balanced distributed computing models improve user experience and expand development. Developers can focus on user experience by examining devices and customizing features. For example, different user interfaces can be customized for different departments that require different resources to perform their function. Different roles would have their own user interface. Well-defined user roles and profiles that are stored on user machines make more efficient use of server computation. It allows the server to pull customized responses based on the identity/role of the user. To further reduce server demand, clients can communicate directly with databases by requesting information for

the user role profile. This eliminates the web server as middleman for the request and computation on the output side.

Data integration on user platforms offers new opportunities to build applications that draw data from a variety of sources and can add different contexts. In a balanced distributed computing model, web services send information that is usually stored on databases or servers (like financial information) to the user's machine. It accomplishes this by using the client-side's processing power. These responses are formatted in the increasingly popular, universal XML. Desktop applications (on the user's system) can take that information and analyze it in different contexts.

The decentralization of distributed browser-server models also improves security and protects privacy. For example, data repositories are often located in a different location from the server. This makes it more difficult for external attackers to find. It also makes it less accessible to internal attackers. Also, it is safer for user profiles to be stored on individual machines, rather than on a central database.

Distributed computing models address the future of IT architecture and application. Organization must aim to create independent and flexible applications that can respond quickly to a variety of contexts. Connections must be agile. Loosely coupled applications, characteristic of distributed computing models, withstand broken connections and slow Internet performance. This protects core technologies from customer demands and lack of Internet bandwidth.

EXERCISE 8

1. Explain an object web model based on java client and CORBA ORB's on the basis of following points:
   (*i*) Web client
   (*ii*) Protocol used
   (*iii*) Web server
2. Explain end-to-end working of Client/Server web model. (Hint: Use CGI, GET, POST, Web browser and Web server)
3. Explain, with the help of block diagram and example, the Common Object Request Broker Architecture.
4. Discuss the role of traditional and web databases in handling Client/Server based applications.
5. Discuss the role of web browser for providing web service in Client/Server environment.
6. Explain how the Database is being accessed on the Web. Or how the information exchange take place between User and a Web-based Database.
7. Discuss the development of Client/Server Technology based web applications.