Human–computer interaction

Human–computer interaction (commonly referred to as **HCI**) researches the design and use of computer technology, focused on the interfaces between people (users) and computers. Researchers in the field of HCI both observe the ways in which humans interact with computers and design technologies that let humans interact with computers in novel ways.

As a field of research, human-computer interaction is situated at the intersection of computer science, behavioral sciences, design, media studies, and several other fields of study. The term was popularized by Stuart K. Card, Allen Newell, and Thomas P. Moran in their seminal 1983 book, *The Psychology of Human-Computer Interaction*, although the authors first used the term in 1980 and the first known use was in 1975. The term connotes that, unlike other tools with only limited uses (such as a hammer, useful for driving nails but not much else), a computer has many uses and this takes place as an open-ended dialog between the user and the computer. The notion of dialog likens human-computer interaction to human-to-human interaction, an analogy which is crucial to theoretical considerations in the field

Introduction

Humans interact with computers in many ways; the interface between humans and computers is crucial to facilitating this interaction. Desktop applications, internet browsers, handheld computers, and computer kiosks make use of the prevalent graphical user interfaces (GUI) of today. Voice user interfaces (VUI) are used for speech recognition and synthesizing systems, and the emerging multi-modal and gestalt User Interfaces (GUI) allow humans to engage with embodied character agents in a way that cannot be achieved with other interface paradigms. The growth in human-computer interaction field has been in quality of interaction, and in different branching in its history. Instead of designing regular interfaces, the different research branches have had a different focus on the concepts of multimodality rather than unimodality, intelligent adaptive interfaces rather than command/action based ones, and finally active rather than passive interfaces.

The Association for Computing Machinery (ACM) defines human-computer interaction as "a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them". An important facet of HCI is the securing of user satisfaction (or simply End User Computing Satisfaction). "Because human-computer interaction studies a human and a machine in communication, it draws from supporting knowledge on both the machine and the human side. On the machine side, techniques in computer graphics, operating systems, programming languages, and development environments are relevant. On the human side, communication theory, graphic and industrial design disciplines, linguistics, social sciences, cognitive psychology, social psychology, and human factors such as computer user satisfaction are relevant.

Poorly designed human-machine interfaces can lead to many unexpected problems. A classic example of this is the Three Mile Island accident, a nuclear meltdown accident, where investigations concluded that the design of the human-machine interface was at least partly responsible for the disaster. Similarly, accidents in aviation have resulted from manufacturers' decisions to use non-standard flight instrument or throttle quadrant layouts: even though the new designs were proposed to be superior in basic human-machine interaction, pilots had already ingrained the "standard" layout and thus the conceptually good idea actually had undesirable results.

i.e. ***The human-computer interface*** can be described as the point of communication between the human user and the computer. The flow of information between the human and computer is defined as the *loop of interaction*. The loop of interaction has several aspects to it, including:

- Visual Based :The visual based human computer inter-action is probably the most widespread area in HCI(Human Computer Interaction) research.
- Audio Based : The audio based interaction between a computer and a human is another important area of in HCI systems. This area deals with information acquired by different audio signals.
- *Task environment*: The conditions and goals set upon the user.
- *Machine environment*: The environment that the computer is connected to, e.g. a laptop in a college student's dorm room.
- *Areas of the interface*: Non-overlapping areas involve processes of the human and computer not pertaining to their interaction. Meanwhile, the overlapping areas only concern themselves with the processes pertaining to their interaction.
- *Input flow*: The flow of information that begins in the task environment, when the user has some task that requires using their computer.
- *Output*: The flow of information that originates in the machine environment.
- *Feedback*: Loops through the interface that evaluate, moderate, and confirm processes as they pass from the human through the interface to the computer and back.
- *Fit*: This is the match between the computer design, the user and the task to optimize the human resources needed to accomplish the task.

Goals

Human–computer interaction studies the ways in which humans make, or do not make, use of computational artifacts, systems and infrastructures. In doing so, much of the research in the field seeks to *improve* human-computer interaction by improving the *usability* of computer interfaces.

Much of the research in the field of human-computer interaction takes an interest in:

- Methods for designing novel computer interfaces, thereby optimizing a design for a desired property such as, e.g., learnability or efficiency of use.
- Methods for implementing interfaces, e.g., by means of software libraries.
- Methods for evaluating and comparing interfaces with respect to their usability and other desirable properties.
- Methods for studying human computer use and its sociocultural implications more broadly.
- Models and theories of human computer use as well as conceptual frameworks for the design of computer interfaces, such as, e.g., cognitivist user models, Activity Theory or ethnomethodological accounts of human computer use.

- Perspectives that critically reflect upon the values that underlie computational design, computer use and HCI research practice.

Researchers in HCI are interested in developing new design methodologies, experimenting with new devices, prototyping new software and hardware systems, exploring new interaction paradigms, and developing models and theories of interaction.

Differences with related fields

HCI differs from human factors and ergonomics as HCI focuses more on users working specifically with computers, rather than other kinds of machines or designed artifacts. There is also a focus in HCI on how to implement the computer software and hardware mechanisms to support human–computer interaction. Thus, *human factors* is a broader term; HCI could be described as the human factors of computers – although some experts try to differentiate these areas.

Design

Principles

When evaluating a current user interface, or designing a new user interface, it is important to keep in mind the following experimental design principles:

- Early focus on user(s) and task(s): Establish how many users are needed to perform the task(s) and determine who the appropriate users should be; someone who has never used the interface, and will not use the interface in the future, is most likely not a valid user. In addition, define the task(s) the users will be performing and how often the task(s) need to be performed.
- Empirical measurement: Test the interface early on with real users who come in contact with the interface on a daily basis. Keep in mind that results may vary with the performance level of the user and may not be an accurate depiction of the typical human-computer interaction. Establish quantitative usability specifics such as: the number of users performing the task(s), the time to complete the task(s), and the number of errors made during the task(s).
- Iterative design: After determining the users, tasks, and empirical measurements to include, perform the following iterative design steps:
    1. Design the user interface
    2. Test
    3. Analyze results
    4. Repeat

Repeat the iterative design process until a sensible, user-friendly interface is created.

Methodologies

A number of diverse methodologies outlining techniques for human-computer interaction design have emerged since the rise of the field in the 1980s. Most design methodologies stem from a model for how users, designers, and technical systems interact. Early methodologies, for example, treated users' cognitive processes as predictable and quantifiable and encouraged design practitioners to look to cognitive science results in areas such as memory and attention when designing user interfaces. Modern models tend to focus on a constant feedback and conversation between users, designers, and engineers and push for technical systems to be wrapped around the types of experiences users want to have, rather than wrapping user experience around a completed system.

- Activity theory: used in HCI to define and study the context in which human interactions with computers take place. Activity theory provides a framework to reason about actions in these contexts, analytical tools with the format of checklists of items that researchers should consider, and informs design of interactions from an activity-centric perspective.
- User-centered design: user-centered design (UCD) is a modern, widely practiced design philosophy rooted in the idea that users must take center-stage in the design of any computer system. Users, designers and technical practitioners work together to articulate the wants, needs and limitations of the user and create a system that addresses these elements. Often, user-centered design projects are informed by ethnographic studies of the environments in which users will be interacting with the system. This practice is similar but not identical to participatory design, which emphasizes the possibility for end-users to contribute actively through shared design sessions and workshops.
- Principles of user interface design: these are seven principles of user interface design that may be considered at any time during the design of a user interface in any order: tolerance, simplicity, visibility, affordance, consistency, structure and feedback.
- Value sensitive design: Value Sensitive Design (VSD) is a method for building technology that account for the values of the people who use the technology directly, as well as those who the technology affects, either directly or indirectly. VSD uses an iterative design process that involves three types of investigations: conceptual, empirical and technical. Conceptual investigations aim at understanding and articulating the various stakeholders of the technology, as well as their values and any values conflicts that might arise for these stakeholders through the use of the technology. Empirical investigations are qualitative or quantitative design research studies used to inform the designers' understanding of the users' values, needs, and practices. Technical investigations can involve either analysis of how people use related technologies, or the design of systems to support values identified in the conceptual and empirical investigations.

Display designs

Displays are human-made artifacts designed to support the perception of relevant system variables and to facilitate further processing of that information. Before a display is designed, the task that the display is intended to support must be defined (e.g. navigating, controlling, decision making, learning, entertaining, etc.). A user or operator must be able to process whatever information that a system generates and displays; therefore, the information must be displayed according to principles in a manner that will support perception, situation awareness, and understanding.

Thirteen principles of display design

Christopher Wickens et al. defined 13 principles of display design in their book *An Introduction to Human Factors Engineering*.

These principles of human perception and information processing can be utilized to create an effective display design. A reduction in errors, a reduction in required training time, an increase in efficiency, and an increase in user satisfaction are a few of the many potential benefits that can be achieved through utilization of these principles.

Certain principles may not be applicable to different displays or situations. Some principles may seem to be conflicting, and there is no simple solution to say that one principle is more important than another. The principles may be tailored to a specific design or situation. Striking a functional balance among the principles is critical for an effective design.

Perceptual principles

1. Make displays legible (or audible). A display's legibility is critical and necessary for designing a usable display. If the characters or objects being displayed cannot be discernible, then the operator cannot effectively make use of them.

2. Avoid absolute judgment limits. Do not ask the user to determine the level of a variable on the basis of a single sensory variable (e.g. color, size, loudness). These sensory variables can contain many possible levels.

3. Top-down processing. Signals are likely perceived and interpreted in accordance with what is expected based on a user's experience. If a signal is presented contrary to the user's expectation, more physical evidence of that signal may need to be presented to assure that it is understood correctly.

4. Redundancy gain. If a signal is presented more than once, it is more likely that it will be understood correctly. This can be done by presenting the signal in alternative physical forms (e.g. color and shape, voice and print, etc.), as redundancy does not imply repetition. A traffic light is a good example of redundancy, as color and position are redundant.

5. Similarity causes confusion: Use distinguishable elements. Signals that appear to be similar will likely be confused. The ratio of similar features to different features causes signals to be similar.

Mental model principles

6. Principle of pictorial realism. A display should look like the variable that it represents (e.g. high temperature on a thermometer shown as a higher vertical level). If there are multiple elements, they can be configured in a manner that looks like it would in the represented environment.

7. Principle of the moving part. Moving elements should move in a pattern and direction compatible with the user's mental model of how it actually moves in the system. For example, the moving element on an altimeter should move upward with increasing altitude.

Principles based on attention

8. Minimizing information access cost. When the user's attention is diverted from one location to another to access necessary information, there is an associated cost in time or effort. A display design should minimize this cost by allowing for frequently accessed sources to be located at the nearest possible position. However, adequate legibility should not be sacrificed to reduce this cost.

9. Proximity compatibility principle. Divided attention between two information sources may be necessary for the completion of one task. These sources must be mentally integrated and are defined to have close mental proximity. Information access costs should be low, which can be achieved in many ways (e.g. proximity, linkage by common colors, patterns, shapes, etc.). However, close display proximity can be harmful by causing too much clutter.

10. Principle of multiple resources. A user can more easily process information across different resources. For example, visual and auditory information can be presented simultaneously rather than presenting all visual or all auditory information.

Memory principles

11. Replace memory with visual information: knowledge in the world. A user should not need to retain important information solely in working memory or retrieve it from long-term memory. A menu, checklist, or another display can aid the user by easing the use of their memory. However, the use of memory may sometimes benefit the user by eliminating the need to reference some type of knowledge in the world (e.g., an expert computer operator would rather use direct commands from memory than refer to a manual). The use of knowledge in a user's head and knowledge in the world must be balanced for an effective design.

12. Principle of predictive aiding. Proactive actions are usually more effective than reactive actions. A display should attempt to eliminate resource-demanding cognitive tasks and replace them with simpler perceptual tasks to reduce the use of the user's mental resources. This will allow the user to focus on current conditions, and to consider possible future conditions. An example of a predictive aid is a road sign displaying the distance to a certain destination.

13. Principle of consistency. Old habits from other displays will easily transfer to support processing of new displays if they are designed consistently. A user's long-term memory will trigger actions that are expected to be appropriate. A design must accept this fact and utilize consistency among different displays.

Introduction to the World Wide Web

1.0 Introduction

The Internet is rapidly becoming a key resource for locating information relevant to a particular field, engaging in professional discourse, accessing published material, and checking on tomorrow's weather. It has become the foundation for tomorrow's electronic community, providing access to government, media, scientists, and friends and relatives. Access to the Internet is now becoming a requirement of doing business for many enterprises. Commercial use of the Internet is one of its fastest growing uses. Several factors have led to the dramatic increase in the size of Internet including increased bandwidth, relaxation of government restrictions, and less expensive connection options. One major factor which has led to the popularization of the Internet is the World Wide Web (WWW), which provides a hypermedia layer over information and resources available on the Internet.

The current Internet connects over 2 million hosts and nearly 25 million users on every continent in the world. Through the Internet, users can access the latest weather maps of North America, check the New York Stock Exchange quotes for the day, send electronic mail to colleagues on the other side of the world, browse through digital shopping centers, check out the latest electronic magazines, download images from a travelogue on eclectic tourist stops in the Southwest United States, among many other uses.

The Internet impacts us as human-computer interaction (HCI) professionals in two major ways: as users and consumers of the information provided on the Internet, and as designer of information sources and network-based computing systems. This tutorial will focus on exploring the technology and history which has led to the Internet of today, and exploring one of the key emerging technologies for providing information on the Internet, the World Wide Web. The tutorial is hands-on, providing direct experience with many of the tools for accessing the Internet and the information content available. In addition, we will design small information spaces based on hypermedia documents using the HyperText Markup Language (HTML). The tutorial is geared towards HCI professionals which do not have extensive familiarity with the Internet or the World Wide Web.

The tutorial will be presented in five parts. Parts One and Two will focus on the history, technology, and tools which underlie the Internet and provide students with the opportunity to utilize those tools to explore some of the information and resources which are available. Part Three will introduce the basic concepts behind the World Wide Web, and Parts Four and Five will provide information and experience with designing information spaces on the World Wide Web and writing documents to populate those spaces in HTML.

THE INTERNET

This section of the tutorial will present the history of the Internet, starting with its origins as a research project of the United States' Advanced Research Projects Agency (ARPA) and progressing through the interconnection of the ARPAnet and the MILnet to the foundation of the NSFnet and the current backbone architecture. We will then present the key underlying technologies including the Internet Protocol (IP), and how IP addresses are formed, and what they mean; the domain name system (DNS), how domains are registered, and how to find out information about a specific domain; and packet routing and how information actually travels through the Internet. This will be presented at a high level: detailed understanding of networking, or network architectures will not be required. The size of the Internet will then be discussed, as well as its current growth rate, and some discussion of possible future directions for the Internet.

The most commonly used Internet services will then be discussed along with some of the security issues and concerns which arise as a result of the use of these technologies.

Writing documents in html

The HyperText Markup Language (HTML) is the primary mechanism for creating content for information provided on the World Wide Web. HTML is a markup language used to create hypertext documents that are portable from one platform to another. HTML documents are SGML documents with generic semantics that are appropriate for representing information for a wide range of applications.

Students will have the opportunity to gain experience designing HTML documents by creating personal home pages for themselves. They will also learn how to provide links from their home page to other interesting sources of information.

1.1 Simple use of the Web

From your room in Kibabii, you decide to find out what entertainment there is in Bungoma. A company publishes *What's On in Bungoma* online, and you have read that this is available on http://www.bungoma.go.ke as a series of Web `pages' on the Internet. You decide to take a look. You switch on your computer and start it up in the usual way. Just as you might click on an icon to start up your desktop publishing package, so you click on the icon to load your World Wide Web browser. The browser is a piece of software that allows you to display certain kinds of information from the Internet.

Your modem, which connects your computer to the telephone line in the street via an ordinary household telephone socket, makes a series of electronic beeps as it dials up the nearest Internet point of presence and puts you onto the 'Net. Via the interface provided by your Web browser, you can now enter the Internet address, otherwise known as the URL, or Uniform Resource Locator, of the file containing the front page of *What's On in Bungoma*. As with all other information on the Web, this would start with `http://', for example:

http://www.bungoma.go.ke

HTTP (HyperText Transfer Protocol) is the name of the Web's own transmission protocol. Web pages are sent over the Internet to your computer courtesy of HTTP.

The difference between the Web and the Internet

Do not make the rather common mistake of confusing the Web with the Internet itself: the Internet simply provides the medium for the Web to run on, just as a telephone line provides the medium for telephone conversations. What the Web does is provide the technology for publishing, sending and obtaining information over the expanse of the Internet. How the Internet actually works may be a matter of interest, but the Web user does not need to know about it in any detail.

Protocols

A protocol consists of salutations exchanged between computers: `good morning', `be with you in a tick', `file coming down the line now' and so on. Each service on the Internet has its own protocol: its own personal way of sending files around the system. The protocol for the Web is HTTP. File Transfer Protocol (FTP) is another common protocol of which you may have heard.

A Web page may also contain forms for conducting commercial transactions across the Internet and include other applications; for example, spreadsheets, video clips, sound clips and so on.

The Web is, therefore, simultaneously a means of online publishing, a way of accessing, storing and retrieving information, as well as a means of sending, acquiring and querying data across the Internet. Most importantly, the Web allows the use of hypertext links that can take you to any computer on the Internet. All these functions work by using the hardware: the wires, the cables, the computers, the satellite links that are used to send information from computer to computer using Internet protocols.

Before the Web, the Internet was largely the domain of computer nerds and others who delighted in the abstract and concise. Interfaces to early Internet applications required almost mathematical precision to operate, and programs such as FTP and Telnet were purely command-driven. Later, when file-retrieval services such as WAIS and Gopher became popular on the Internet, these had the advantage of being menu-driven, but still were rather cumbersome.

What is a URL?

This is a much-simplified explanation for the novice.

Given the fact that there are vast numbers of Web servers, the question remains of how can HTTP possibly locate just the file you want from somewhere on the Internet. The answer is through the use of the URL, the Uniform Resource Locator. This is rather like the telephone number of a computer on the Internet, together with information appended to specify the exact file to be sent to your machine. Taking the URL for our fictitious Web pages in Bungoma, we can see the general pattern that URLs adopt. Look at

http://www.bungoma.go.ke

more closely.

`http' is the name of the Web protocol used to access the data across the Internet.

`www.bungoma.go.ke' is the Internet name or the domain name of the computer on which the information is stored. The `www' indicates that this is a World Wide Web server.'.go' means it's a government organization and `ke' is the country code for Kenya.

To be more specific about the file you want from the server in question, you add to the URL a path and a file name. For example:

http://www.bungoma.go.ke/time_tables/buses.html

which would point to a file called `buses.html' in a directory called `time_tables'. We can imagine that `buses.html' contains bus timetables for the city of Bungoma, which can be conveniently called up on your screen.

URLs can specify files to be accessed using protocols other than `http'. A URL beginning with `ftp://' points to a file to be fetched using FTP - the File Transfer Protocol. Meanwhile, a file beginning with `mailto://' links to an application which allows you to send an email message to a pre-defined address, and `news://' points to a USENET newsgroup and uses the Network Transfer Protocol to transfer data.

There are various conventions when it comes to URLs. Take the letters `.com' for instance. These mean `company' as in: `http://www. microsoft.com'.

Non-profit organizations may use `.org' and educational establishments use `.edu'. Similarly, there are codes for countries. A URL may end in `.us' for the United States, `.fr' for France, `.au' for Australia and so on. Some of the conventions for URLs are listed in Appendices E and F.

How are email addresses different from URLs? Email addresses follow a different format: name@name.name.domain. For example: tiptoes@kibu.ac.ke.

The string of letters following the @ sign identifies the machine to which the mail will be sent, whereas the name of the person who will receive the mail is given directly before the @ sign. There may be more than one person logging on to the machine to read mail. That is why the name of the person becomes important in an email address.

1.2 Basic components of the Web

The basic components of the Web are shown in the following illustration. They are:

- **Web servers**, which are computers that hold information for distribution over the Internet. In the example application in the diagram, one Web server might hold the text and graphics of the online magazine *What's On in Bungoma*, and another server might hold information on which seats are available for a particular concert. The magazine would be formatted using the Web's own publishing language, HTML (HyperText Mark-up Language). The data on available seats and their price would be held in a database with links to specific forms that are published using HTML.

- **Servers**, which can be PCs, Macintosh systems or UNIX workstations: it is the server software that makes them special, rather than the computer itself. That said, servers need to be fairly up-market machines. Servers do need to be left on all the time, so that people can access the information on them whenever they want. Another important point about servers: they are relatively difficult to set up. If you are a non-technical person who wants to publish on the Web, the best thing to do is to rent some space on someone else's server.
- **Web clients**, which can be PCs, Macintoshes and other computers that are connected to the Internet and which can retrieve information from Web servers. A Web client is the computer on your desk. PCs, Macintoshes, UNIX workstations and even simple terminals can run client software. Different client software is marketed (or is given away free) for different platforms. Thus, Mosaic has both a Macintosh and a PC implementation.
- **HTTP protocol**, which is used to transmit files between servers and clients. When you click on a hypertext link or fill out a form in a Web document, the results need to be sent across the Internet as quickly as possible, and then to be understood by a server at the other end. Instructions such as `send me this file' or `get me that image' are carried by the Web communications protocol, HTTP. This protocol is the `messenger' that fetches files to and from servers, and then delivers results to your computer every time you click with a request. HTTP has its counterparts in other Internet services: FTP, file transfer protocol, and Gopher are protocols that obtain different sorts of information from across the Internet.
- **Browser** software, which is needed by a Web client for displaying text, images, video clips and so on. This is supplied under the umbrella name `browser', of which Mosaic, Microsoft Corp.'s Internet Explorer and Netscape Communications Corp.'s Navigator and Communicator browsers are probably the best-known examples. Browser software gives you the ability to scan information retrieved from Web servers, as you would browse through a book. It also gives you facilities for saving and printing information obtained on the Web.

A key point that helped make the Web successful is that it is multi-platform. What this means is that it does not matter what kind of computer you are using; you can still view information published on other, usually incompatible, machines. Thus, a PC user can access information published on a Macintosh; a Macintosh can access information published by a PC; UNIX users will find that they are compatible with everyone. The trick is that each computer in its own way can assimilate HTML. What happens when an HTML document gets to its destination is up to the computer on the receiving end. It can display paragraphs in Helvetica 20-point type if it pleases, and headings in Times 14-point bold type, if that is the font available. Whereas the document is transmitted computer-to-computer in a standard format, individual browsers may display it quite differently, depending on the capabilities of the hardware and software in the computer on the receiving end.

1.3 A universally understood publishing language: HTML

To publish information for global distribution on the World Wide Web, you need a universally understood publishing language, a kind of mother tongue, which all computers on the Web can potentially understand. You also need a commonly understood communications protocol for sending published information `down the wire' from computer to computer. This should enable users to download information to their machine at the click of a button, and also to send back information (your address, a credit card number, a query to a database and so on) with little effort.

The publishing language used by the Web is called HTML (HyperText Mark-up Language). Using HTML, you can specify which parts of your text are to be headings, paragraphs, bulleted lists, and which parts are to be rendered in bold-face type, in italicized type and so on. You can use HTML to insert tables into documents, to write equations, to import images and to format fill-out forms for querying databases at a distance. (Some of these features are specific to HTML 3 and are not supported by earlier versions of HTML.) The HTML language itself is very flexible and not difficult to use, although, as with all tasks associated with computing, patience is a necessary virtue for authoring hypertext. Part of the Web's appeal is that almost anyone with a reasonable PC, Macintosh or UNIX computer can publish information without being unduly technical. Judging by the variety of publishers on the Web today, HTML is within the grasp of many.

1.4 The HTTP protocol

The initial protocol for the Web was very, very simple. The client sent a request: `GET this filename' and the other end sent back the file and closed the connection. And that was it. There was no content type to tell you what kind of file was being sent. No status code. Just the file. The client therefore had to guess what it had been given and this developed into a fine art. First, the browser would look at the file extension to see if there were any clues, such as ˙GIF or ˙HTML, and then it would look at the beginning of the file in case the first few bytes gave the game away - all rather precarious.

Then along came MIME, a kind of multimedia extension to email. This was soon adapted to HTTP so that now, when you receive a file, you actually have a status code and a content type. This content type tells you whether or not the file is text/HTML, video/MPEG, image/GIF and so on, which gives the browser a chance to call up the correct viewers to display the file. The burden of finding out what kind of file it was has now moved to the server. On UNIX and DOS, servers still play this game of `guess the file format'; whereas on the Macintosh, this is stored as part of the file itself.

The HTTP we use today is the product of collaboration between CERN and a group at the National Center for Supercomputer Applications (NCSA) at the University of Illinois at Champaign-Urbana. Innovations since that time included security features such as Netscape's secure socket layer (SSL), and more recently, the ability to keep open the connection to the server so that the server can make multiple requests. The World Wide Web Consortium is looking into ways of improving this protocol, and into methods of text compression that enable information on the Web to arrive at its destination much quicker.

1.5 More than just text and pictures on the Web

The Web is expanding not only in terms of how much information it holds, but also in terms of the variety of information it holds. The figure below illustrates this general trend. On the left-hand side, you can see the Web as it started, predominantly as a medium for publishing information in textual form, and then it progressed to include photos, diagrams and so on.

Simplified view of components of the Web.
More variety on the Web. From being a simple text-based system, the Web now supports a variety of media.

Toward the right-hand side of our diagram, we show the latest features at the time of writing: the introduction of plug-in modules, the Java programming language that enables all manner of small applications to be sent down the wire and used within Web applications, and so on.

Many of the new features of HTML are associated with this departure away from the simple document and toward a Web which combines text, graphics, video, audio, applications such as spreadsheets, front ends to databases, virtual reality applications and all sorts of other `objects'.

Java

At the time of writing, Java is a very popular buzzword. Java is a programming language that enables programmers to write small applications that are sent over the Web and then executed on the client. It is an object-oriented language related to the C language. What is so special about Java is that programs can be sent to any machine with the right browser to understand them; furthermore, the code will run safely without any risk of adversely affecting the client.

From the user's point of view, Web documents suddenly become much cleverer. It is as though your browser automatically creates features, such as the ability to run a spreadsheet or to play a piece of animation, right in front of your eyes, without you having to load any extra software. The code required to do such tricks is compiled into a special binary format and executed by a Java interpreter in your machine. Java applications are called `applets' (which means `small applications' - it would be another thing entirely to send full-blown application software over the Web) and are small pieces of code that rely on libraries of Java `classes' in the browser. These libraries indicate that the browser has a certain amount of processing knowledge resident in the browser itself: the browser knows how to create a window, respond to events, paint things within a window, draw text within a window and so on. The applets arriving across the Web capitalize on this knowledge and use the library routines to do something useful, such as displaying a simple spreadsheet or a piece of animated graphics.

But, what happens if the applet calls upon a class that is not available at the client end? In such cases, the class has to be fetched over the Internet, a process that indeed may be rather slow. For this reason, it is best to limit the number of these `additional' classes.

Scripts for HTML

Scripts are small programs, which are transparent to the user and which go on `behind the scenes' fine-tuning Web pages in one way or another. The author can write a script in one of several available scripting languages. Scripts themselves may have one of several functions. The classic function is rendering a form to seem `smart' so that it interacts with you as you fill it in. Thus, you might arrange for a script to check each form field in turn to make sure it is properly filled in, and then to advise you if you have mistakenly left out some critical piece of information.

VBScript was invented by Microsoft and is an adapted version of Visual Basic; JavaScript is a Netscape scripting language and Sun has tcl.

In the future, scripts may be used to create various special effects, such as animation. The later chapter in this book on Scripting examines some of the current and upcoming uses for scripting. Scripts can also be used to tailor components of a particular program, such as a spreadsheet program. In this instance, application software developers will supply off-the-shelf components for a program. Let us suppose that the components are for a spreadsheet program; with a script, you can easily integrate off-the-shelf components into the application, a feat that would have previously required many programmer hours to compile. But with the hypothetical script you have written, this almost seamless integration happens in much the same way as it does with Visual Basic; here, the components are written in a kind of systems language and then, the script enables off-the-shelf components to be integrated into the application itself. If scripts are half as good as they promise, they can save many a programmer's headache.

The idea of the Internet media type

If HTML can have all sorts of items embedded in it - pieces of animation to be played by your browser, small programs to run on your computer, and even music - then surely the browser must have some way of knowing what kind of beast is coming down the line. A browser can sense what type a file is and therefore it can know what to do with it by using the *Internet media type*. The Internet media type is a code sent with the file. The browser uses this code to work out what software is needed to interpret and display the data. Some examples include the following: the Internet media type for an HTML document is text/HTML: for a JPEG image, the Internet media type is image/JPEG.

Once the media type has been recognized and the correct software called up to deal with the file, the information in the file can be displayed embedded in the document; that is, it is displayed in an area of the document as though it were simply part of it. Another way of looking at this is that information can be `plugged in' to a document. Indeed, the phrase `plug-in' is commonly used to describe downloading embedded files that are not in HTML, and which require software at the browser end to come to the rescue and display them.

A browser may allow, for example, a PDF (Portable Document Format) file to appear down the wire. The browser sees that this has an Internet media type Application/PDF. It then calls up the Adobe Acrobat reader to display the PDF file. Once the media type is known, the browser loads the correct application software and interprets the file.

1.6 HTML and its relationship to SGML

Early work on representing documents focused on rendering instructions needed to print the documents. Work by IBM on GML (Generalized Mark-up Language) focused on an alternative approach, whereby standard document structures such as headers, paragraphs, lists and so on were marked up by tags inserted into document text. The emphasis on document structure rather than on rendering instructions, made it dramatically easier to move documents from one system to another whether for display on simple terminals, line printers or sophisticated typesetting machinery.

This work led to the Standard Generalized Mark-up Language, which is an international standard ISO 8879:1986. SGML enables you to define a grammar for marked-up documents that defines the ways in which tags can be inserted into documents. For instance, list items only make sense in the context of a list, and table cells only make sense in the context of a table. SGML's formal way of describing the grammar is called the Document Type Definition.

Global hypertext makes worse the problem in moving documents from one system to another; for example, we have Macintosh systems, PCs, a variety of UNIX boxes, simple terminals and even speech I/O devices for the visually impaired. SGML proved ideally suited for this application. Tim Berners-Lee chose SGML to define the HTML document format for the World Wide Web. HTML is formally an application of SGML. The HTML Document Type Definition (DTD) formally defines the set of HTML tags and the ways that they can be inserted into documents.

To the uninitiated, DTDs may seem rather intimidating. This book tries to act as a guiding hand to explain the HTML mark-up language and how to apply it so that it creates documents for Web publishing. HTML is not a static document format, but is evolving rapidly from its simple beginnings as conceived by Tim Berners-Lee.

1.7 The Web for people with disabilities

One of the areas in which the World Wide Web Consortium (W3C) has shown great interest is how to make the Web accessible to the blind. In the case of HTML, this has involved two things in particular, discussed below.

First of all, the IMG tag for inserting images (as explained in our chapter *Graphics on the Web*) is to be superseded by an entirely different tag (OBJECT), which enables the browser to display textual mark-up as an alternative when a visual image appears on the screen.

Suppose you have a photograph as part of your Web pages. Browsers will usually display the photograph as the author intended. A browser used by a blind or visually impaired person will, however, be set up to display a paragraph or two of descriptive text instead of the photograph. This text must be included by the author for the benefit of text-only browsers and may contain hypertext links, itemized lists and all kinds of other mark-up. Seeing such text, the browser will read aloud the words to the user and even use appropriate intonation for bold, emphasized text and so on.

By including textual mark-up as an alternative to images, the blind or visually impaired Web user is in a far better position than previously. The still-popular IMG tag allows only very limited alternative text in lieu of images and thus puts it out of favor with those who rely on speech generation.

Second of all, the W3 Consortium is encouraging authors to separate the *structural* aspects of their documents from those aspects of the documents that are merely to do with *layout*. Once this has been done, the software that enables the text to be synthesized and read aloud to the blind or visually impaired user has got an easier job to do. Think about it: if a page is marked up solely in terms of headings, paragraphs, lists, and other structural items, then it becomes much simpler to understand the HTML and to render the content into speech. But the moment that extraneous information about font size, alignment of text, margin width, color and so on is mixed in with the general mark-up, then translating that mark-up into a spoken equivalent becomes much harder.

An ardent enthusiast when it comes to making the Web available to the blind is Dr T.V. Raman. A clear and original thinker, Raman - himself blind - has contributed widely to discussions and negotiations with browser vendors on the subject of Web access for people with disabilities, and also is an active participant at the Math Working Group. He has also written the Emacspeak Speech Interface, which is a full-fledged, speech-output system that enables the blind and visually impaired to access the Web with a line-mode browser within a UNIX environment.