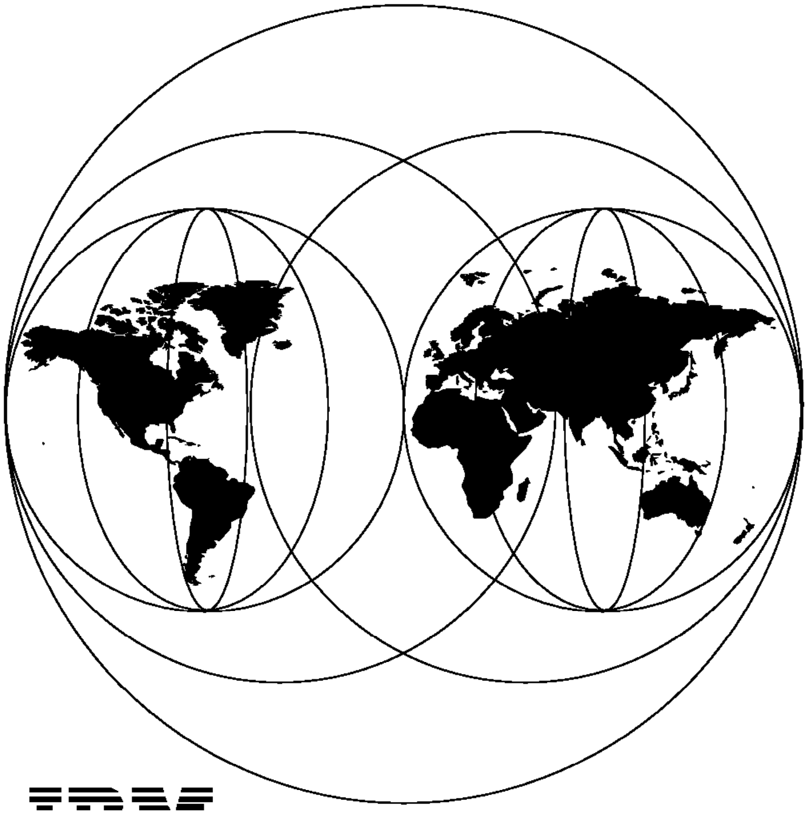
SG24-4805-00

A Guide to the Internet Connection Servers

October 1996



IBM

International Technical Support Organization
Raleigh Center

International Technical Support Organization

SG24-4805-00

A Guide to the Internet Connection Servers

October 1996

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix B, “Special Notices” on page 305.

First Edition (October 1996)

This edition applies to Version 4.1 of IBM Internet Connection Server.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HZ8 Building 678
P.O. Box 12195
Research Triangle Park, NC 27709-2195

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Preface

This redbook provides detailed coverage of the IBM Internet Connection Server family of products. In particular, it focuses on the product′s implementation on multiple platforms.

This redbook was written for Web site administrators and Internet specialists who need to evaluate and implement the Internet Connection Server.

Many practical examples are presented that demonstrate how the Internet Connection Server can be used to establish a presence and engage in electronic commerce on the Internet.

Some knowledge of TCP/IP and the Internet is assumed. However, for the less knowledgeable reader, we have included a chapter that covers basic concepts.

How This Redbook Is Organized

The redbook is organized as follows:

- Chapter 1, "The Internet Connection Family"

  This chapter provides an introduction to the Internet Connection family of products.

- Chapter 2, "Internet Concepts"

  This chapter provides an overview of Internet concepts.

- Chapter 3, "Web Browsers"

  This chapter covers the protocols and functions included in the most widely used Web browsers.

- Chapter 4, "Internet Connection Servers - Installation and Configuration"

  This chapter covers installation and configuration of the Internet Connection server on a selection of the supported platforms.

- Chapter 5, "Establishing a Presence on the Internet"

  This chapter covers the use of the Internet Connection server in establishing a presence on the Internet.

- Chapter 6, "Establishing an Active Presence on the Internet"

  This chapter covers the use of the Internet Connection server in establishing an active presence on the Internet.

- Chapter 7, "Enhancing Your Web Site with Java Applications"

  This chapter provides a practical example of the use of Java in establishing an active presence on the Internet.

- Appendix A, "Two Simple Browsers and One Simple Server"

  This appendix includes two basic browsers and a basic server that were used during the project to investigate the HTTP protocol.

The Team That Wrote This Redbook

This redbook was produced by a team of specialists from around the world working at the Systems Management and Networking ITSO Center, Raleigh.

Eamon Murphy is a Senior International Technical Support Specialist at the Systems Management and Networking ITSO Center, Raleigh. He graduated as a Bachelor of Science in Mathematics and Economics from the University of Sussex. He writes extensively and teaches IBM classes worldwide on all areas of TCP/IP and Internet connectivity. Before joining the ITSO four years ago, he worked in Customer Education, IBM UK as a Senior Networking Instructor.

Jean-Claude Andlauer is a Software Engineer in Availability Services in Luxembourg. He graduated as a Telecommunications Engineer from the Swiss Federal Institute of Technology in Lausanne. He has five years of experience in the fields of TCP/IP networking and Web site administration. He has worked at IBM for two years.

Catherine Ezvan is a Systems Engineer in the AIX National Support and Services Center in France. She graduated as an Electronics Engineer from the Ecole Superieure d′Electronique de l′Ouest in Angers, France. She has four years of experience in the fields of TCP/IP networking, AIX administration and Web site administration. She has worked at IBM for three years.

Manfred Genger is a Workstation Specialist in Technical Marketing in Germany. He has one year of experience in the fields of TCP/IP networking and Web site administration.

Tsuyoshi Kitazawa is a Systems Engineer in the NCC Systems Laboratory in Japan. He graduated as a Master of Engineering from Tokyo Metropolitan University. He has five years of experience in the fields of MVS systems programming, TCP/IP networking and Web site administration. In particular, he was a member of the team who were responsible for the IBM Japan home page.

Thanks to the following people for their invaluable contributions to this project:

David Boone
Systems Management and Networking ITSO Center, Raleigh.

Mike Ensley
Internet Servers Solutions Executive, IBM Raleigh

Ken Parzygnat
ICS Development, IBM Raleigh

Comments Welcome

We want our redbooks to be as helpful as possible. Should you have any comments about this or other redbooks, please send us a note at the following address:

redbook@vnet.ibm.com

Your comments are important to us!

Chapter 1. The Internet Connection Family

As the title indicates, in this redbook we focus on the Internet Connection servers. However, the IC servers are part of a larger family of products that also includes firewalls, gateways, browsers, development tools and miscellaneous others. In addition, there are selected products available from other vendors which may be used to supplement the function of the IC family.

In this section we introduce the Internet Connection family and some supplementary products in a series of tables. The tables show how these products can be used to perform the following tasks:

- Create your own intranet and connect it safely to the Internet
- Establish a presence on the Internet
- Integrate your existing applications with the Internet
- Create new applications for the Internet

Table 1. Create Your Own Intranet and Connect It Safely to the Internet

Operating System	TCP/IP	Firewall	Browser	Secure Browser	Phone
DOS	TCP/IP for DOS	n/a	n/a	n/a	n/a
Windows 3.1x	Internet Connection for Windows	n/a	Netscape Navigator	Netscape Navigator	n/a
Windows 95	Internet Connection for Windows	n/a	Netscape Navigator	Netscape Navigator	Internet Connection Phone
Windows NT	Internet Connection for Windows	n/a	Netscape Navigator	Netscape Navigator	n/a
OS/2 Warp	TCP/IP for OS/2	n/a	Netscape Navigator	Netscape Navigator	n/a
AIX	TCP/IP for AIX	Secured Network Gateway	Netscape Navigator	Netscape Navigator	n/a
OS/390	TCP/IP for MVS	n/a	n/a	n/a	n/a
OS/400	TCP/IP for OS/400	n/a	n/a	n/a	n/a
VM	TCP/IP for VM	n/a	n/a	n/a	n/a

Table 2. Establish a Presence on the Internet

Operating System	Servers	Secure Servers	Commerce Servers
Windows 95	Internet Connection Server for Windows 95	Internet Connection Secure Server for Windows 95	n/a
Windows NT	Internet Connection Server for Windows NT	Internet Connection Secure Server for Windows NT	Net.Commerce for Windows NT
OS/2 Warp	Internet Connection Server for OS/2	Internet Connection Secure Server for OS/2	Net.Commerce for OS/2
AIX	Internet Connection Server for AIX	Internet Connection Secure Server for AIX	Net.Commerce for AIX
HP-UX	Internet Connection Server for HP-UX	Internet Connection Secure Server for HP-UX	Net.Commerce for HP-UX
Solaris	Internet Connection Server for Solaris	Internet Connection Secure Server for Solaris	Net.Commerce for Solaris
OS/390	Internet Connection Server for MVS	Internet Connection Secure Server for MVS	n/a
OS/400	Internet Connection Server for OS/400	n/a	n/a

Table 3. Integrate Your Existing Applications with the Internet

Operating System	DB2 WWW Gateway	CICS WWW Gateway	CICS Client for Java	MQSeries WWW Gateway	IMS WWW Gateway	WWW Bookserver
Windows NT	DB2 WWW Gateway for Windows NT	CICS WWW Gateway for Windows NT	CICS Client for Java for Windows NT	n/a	n/a	n/a
OS/2 Warp	DB2 WWW Gateway for OS/2 Warp	CICS WWW Gateway for OS/2 Warp	CICS Client for Java for OS/2 Warp	MQSeries WWW Gateway for OS/2 Warp	n/a	WWW Bookserver for OS/2
AIX	DB2 WWW Gateway for AIX	CICS WWW Gateway for AIX	CICS Client for Java for AIX	MQSeries WWW Gateway for AIX	n/a	WWW Bookserver for AIX
HP-UX	DB2 WWW Gateway for HP-UX	n/a	n/a	n/a	n/a	n/a
Solaris	DB2 WWW Gateway for Solaris	n/a	n/a	n/a	n/a	n/a
OS/390	DB2 WWW Gateway for MVS	CICS WWW Gateway for MVS	n/a	MQSeries WWW Gateway for MVS	IMS WWW Gateway for MVS	WWW Bookserver for MVS

Table 4. Create New Applications for the Internet

Operating System	Java Development Kit
Windows 3.1	Java Development Kit for Windows 3.1
OS/2 Warp	Java Development Kit for OS/2
AIX	Java Development Kit for AIX
OS/390	Java Development Kit for MVS
OS/400	Java Development Kit for OS/400

Chapter 2. Internet Concepts

In this chapter we introduce some basic Internet concepts. Understanding these concepts will help you to better appreciate the features and functions of the Internet Connection servers.

The concepts introduced herein are:

- TCP/IP
- HTTP
- HTML
- Security
- Firewalls
- Gateways
- Java
- APIs

2.1 TCP/IP

In this section we introduce TCP/IP. For more information, refer to the redbook *TCP/IP Tutorial and Technical Overview*, GG24-3376, published by Prentice Hall as ISBN 0-13-460858-5.

TCP/IP is the acronym for Transmission Control Protocol and Internet Protocol. TCP and IP are the two major protocols in a family of related protocols that is generally referred to as the TCP/IP protocol suite. Because TCP/IP protocols are widely used in the Internet, an alternative name is the Internet protocol suite.

The Internet protocol suite allows Internet-connected hosts to engage in such activities as sharing files with other hosts, logging in to other hosts and many other forms of cooperative processing.

TCP/IP was developed as the result of a project funded by the Defense Advanced Research Projects Agency (DARPA) in the USA. The requirement was for a set of open networking protocols which would foster communications between affiliated defense, research and academic organizations. During the 1970s, the set of protocols, originally known as Network Control Program (NCP), evolved so that by the early 1980s, the TCP/IP-based Internet as we know it today was in existence.

What has changed, particularly in the 1990s, is that the Internet is no longer restricted to the research and defense industries but has been increasingly used for general commercial purposes.

TCP/IP is a communication protocol which, like SNA, is based on layers. It enables heterogeneous computers to communicate by performing network-related processing such as message routing, network control, error detection and correction.

2.1.1 TCP/IP Layers

In this section we describe the four TCP/IP layers.

- **Application Layer**

  The application layer is provided by the program that uses TCP/IP for communication. Examples of applications are Telnet, FTP, e-mail, Gopher and SMTP.

- **Transport Layer**

  The transport layer provides communication between application programs. The applications may be on the same host or on different hosts. Multiple applications can be supported simultaneously. The transport layer is responsible for providing a reliable exchange of information. The main transport layer protocol is TCP. Another is User Datagram Protocol (UDP), which provides a connectionless service in comparison to TCP, which provides a connection-oriented service.

- **Internet Layer**

  The Internet layer provides communication between computers. Part of communicating messages between computers is a routing function which ensures that messages will be correctly delivered to their destination. The Internet Protocol (IP) provides this routing function. Examples of Internet layers are IP, ICMP, IGMP, ARP and RARP.

- **Network Layer**

  The network layer is implemented by the physical network that connects the computers. Examples are X.25, ISDN, Ethernet, token-ring and ATM.

2.1.2 Request for Comments (RFC)

Internet standards are called RFCs or Requests for Comments. A proposed standard is initially issued as a proposal and has an RFC number. When it is finally accepted, it is added to official Internet protocols but is still referred to by its RFC number. We found databases that contain the RFCs at:

```
http://ds.internic.net
http://www.internic.net
```

or

```
http://www.nic.ch/newdom-other.htm
```

which has pointers to European, Asian and American network centers that may also have useful reference data.

2.1.3 TCP/IP Operation

TCP/IP is built on connectionless technology. Information is transferred as a sequence of datagrams. A datagram is a collection of data that is sent as a single message. Each of these datagrams is sent through the network individually. The datagrams are sent in transit; the network does not know that there is any connection between them. It is possible that datagram 10 will arrive before datagram 5. It is also possible that somewhere in the network an error will occur and a datagram will not get through at all. In that case, that datagram has to be sent again.

An IP datagram can cross different physical networks. Physical networks have a maximum frame size, called the Maximum Transmission Unit (MTU). MTU limits the length of a datagram that can be placed in one physical frame. IP requires that each link has an MTU of at least 68 bytes, so if any network provides a lower value than this, fragmentation and reassembly must be implemented in the network interface layer in a way that is transparent to IP.

The terms *datagram* and *packet* are often used interchangeably. To be precise a datagram is the unit of data transfer in a TCP/IP context. A packet is the unit of data transfer in a network layer context.

TCP is responsible for breaking up the message into datagrams, reassembling them at the other end, resending anything that gets lost and putting things back in the right order.

IP is responsible for routing individual datagrams. TCP passes IP a datagram with a destination. IP identifies in its routing table the first host in the path to the destination host and sends the datagram. This process is then repeated by IP in the first host and all subsequent hosts until the datagram reaches its destination.

2.1.4 Applications

In this section we describe some of the most commonly used TCP/IP applications.

2.1.4.1 File Transfer Protocol (FTP)

Files may be transferred between hosts using the file transfer protocol (FTP). FTP is designed in accordance with the client/server model: if both client and server components of FTP are provided on a host then file transfer in both directions is possible. By prompting for a user ID and a password security is provided.

FTP is built on the services of TCP in the transport layer. FTP transfers files as either ASCII characters or binary data. ASCII characters are used to transfer data sets that contain only text characters. FTP provides functions, such as listing remote directories, changing the current remote directory, creating and removing remote directories and transferring one or more files in a single request. For more information about FTP, see RFC 959.

2.1.4.2 Telnet

Telnet enables a user to log in to other computers on the network. By specifying a computer on the network a connection will be established. Once the connection is done, anything you type is sent to the computer. The connection to the remote computer behaves much like a dial-up connection. For example, the remote computer asks for a user ID and password as would be done with direct access to the computer. There are Telnet implementations for 3270 and 5250 emulation available in the IBM TCP/IP products.

Telnet is built on the services of TCP in the transport layer. Telnet provides duplex communication and sends data either as ASCII characters or binary data. For more information about the Telnet protocol, see RFCs 854, 856, 857, 885 and 1091.

2.1.4.3 Network File System (NFS)

The network file system (NFS) allows a system to access files on another computer as if they were on your own computer. NFS uses the Remote Procedure Call (RPC) protocol to communicate between the client and the server. The files to be accessed reside on the server host and are made available to the user on the client host. NFS supports a hierarchical file structure. The directory and subdirectory structure can be different for individual client systems. For more information about NFS, see RFC 1094.

2.1.4.4 Remote Printing (LPR/LPD)

The line printer requester (LPR) allows access to printers on other computers running the line printer daemon (LPD) as though they were on your computer. The clients provided (LPR, LPQ, LPRM or LPRMON) allow the user to send files or redirect printer output to a remote host running a remote print server (LPD). These clients can also be used to query the status of a job, as well as to delegate a job. For more information about remote printing, see RFC 1179.

2.1.4.5 Remote Execution (RSH/REXEC)

Remote shell (RSH) and remote execution (REXEC) are similar protocols that allow you to run programs and commands on different computers. The results are received and displayed on the local host. This can be useful for small computers to harness the power of large computers.

2.1.4.6 Simple Mail Transfer Protocol (SMTP)

The Simple Mail Transfer Protocol is an electronic mail protocol with both client (sender) and server (receive) functions. For more information about SMTP see RFCs 821, 822 and 974.

2.1.4.7 Post Office Protocol (POP)

The post office protocol is an electronic mail protocol with both client (sender/receiver) and server (storage) functions. POP allows mail for multiple users to be stored in a central location until a request for delivery is made by the electronic mail program. For more information about POP, see RFC 1725.

2.1.4.8 Gopher

Gopher is a client/server protocol designed for information location and retrieval. The client function provides a menu-driven interface to access the files stored on a Gopher server. The server function allows descriptive names to be assigned to the files. Thus, making it easier to identify the content of each file. Gopher was designed at the University of Minnesota. For more information about Gopher, see RCF 1436.

2.1.4.9 Hypertext Transfer Protocol (HTTP)

The hypertext transfer protocol is a protocol designed to allow the transfer of hypertext markup language (HTML) documents. Hypertext markup language is a tag language used to create hypertext documents. Hypertext documents include links to other documents that contain additional information about the highlighted term or subject.

2.1.4.10 Domain Name System (DNS)

The domain name system uses a hierachical system for naming hosts. Each host name is composed of domain labels separated by periods. Each label represents an increasingly higher domain level within an internet. The fully qualified domain name of a host connected to a large internet generally has one or more subdomains.

For example:

```
host.subdomain.subdomain.rootdomain
```

or

```
host.subdomain.rootdomain
```

Domain names often reflect the hierarchy level used by network administrators to assign domain names. For example, the domain name eng.mit.edu is a subdomain of edu. Local network administrators have the authority to name local domains within an internet.

You can refer to hosts in your domain by host name only; however, a name server requires a fully qualified domain name. The local resolver combines the host name with the domain name before sending the address resolution request to the domain name server.

Mostly, the default domain appended to a host name is stored in a file in the ETC directory called, in OS/2 for example, RESOLV. For more about DNS, see RFCs 1034 and 1035.

2.1.4.11 Simple Network Management Protocol (SNMP)

The simple network management protocol provides a means for managing an Internet environment. SNMP allows network management by elements, such as gateways, routers and hosts. Network elements act as servers and contain management agents, which perform the management functions requested. Network management stations act as clients; they run the management applications, which monitor and control the network. SNMP provides a means of communicating between these elements and stations so they can send and receive information about network resources. For more information about network management using SNMP, see RFCs 1155, 1157, 1187 and 1213.

2.1.4.12 Talk

Talk allows you to send interactive messages, as opposed to the batch mail capabilities of SMTP. When a local host sends a Talk request to a remote host, the user on the remote host is notified that there is a connection request. The user on the remote host must respond with a Talk message to the local host. Message exchange can thus occur between the local and remote hosts.

2.1.4.13 Finger

The Finger protocol provides an interface for querying the current status of a remote host or a user ID on a remote host. Finger uses TCP as the underlying protocol. For more information about Finger, see RFC 1196.

2.1.4.14 Routing Information Protocol (RIP)

The Routing Information Protocol creates and dynamically maintains network routing tables. RIP arranges to have gateways and routers periodically broadcast their routing tables to neighbors. Using this information, a RouteD server can update a host′s routing tables. For example, RouteD determines if a new route has been created, if a route is temporarily unavailable or if a more efficient route exists. For more information about routing using RIP, see RFC 1058.

2.1.4.15 X Window System

The X Window System protocol supports network-transparent windowing and graphics. For more information about X Window System protocol, see RFC 1013.

2.1.4.16 Sockets Application Programming Interface (API)

The sockets API allows you to write your own applications to supplement those supplied by TCP/IP. Most of these additional applications communicate using either TCP or UDP.

2.1.5 Transport Layer

This section describes the transport protocols in TCP/IP that allow communication between application programs.

2.1.5.1 Transmission Control Protocol (TCP)

The Transmission Control Protocol provides a reliable vehicle for delivering packets between hosts on an internet. TCP takes a stream of data, breaks it into datagrams, sends each one individually using IP and reassembles the datagrams at the destination node. If any datagrams are lost or damaged during transmission, TCP detects this and resends the missing datagrams. The received data stream is a reliable copy of the transmitted data stream. For more information, see RFC 793.

2.1.5.2 User Datagram Protocol (UDP)

The user datagram protocol provides a less reliable mode of communication than does TCP and is an alternative to TCP as a transport protocol.

Like IP, UDP does not guarantee datagram delivery or duplication protection. UDP does provide checksums for both the header and data portions of a datagram. Therefore, applications that require reliable delivery of streams of data should use TCP. For more information about UDP, see RFC 768.

2.1.6 Internet Layer

This section describes the internet protocols in TCP/IP. Protocols in the internet layer provide connection services for TCP/IP. These protocols connect physical networks and transport protocols. RFCs 1118, 1180, 1206 and 1208 provide more information.

2.1.6.1 Internet Protocol (IP)

The Internet Protocol provides the interface from the transport layer (host-to-host) protocols to the physical level protocols. IP is the basic transport mechanism for routing IP packets to the next gateway, router or destination host.

IP provides the means to transmit blocks of data (or packets of bits) from sources to destinations. Sources and destinations are hosts identified by fixed-length addresses. Outgoing packets automatically have an IP header sent to the higher-level protocols. This protocol provides the universal addressing of hosts in an internet network.

IP does not ensure a reliable communication because it does not require acknowledgments from the sending host, the receiving host, or intermediate hosts. IP does not provide error control for data; it provides only a header checksum. IP does not perform retransmissions or flow control. A higher-level protocol that uses IP must implement its own reliability procedures.

For more information about IP, see RFC 791.

2.1.6.2 Internet Control Message Protocol (ICMP)

The Internet Control Message Protocol passes control messages between hosts, gateways and routers. For example, ICMP messages can be sent in any of the following situations:

- When a host checks to see if another host is available (PING)
- When a packet cannot reach its destination
- When a gateway or router can direct a host to send traffic on a shorter route
- When a host requests a netmask or a time stamp
- When a gateway or router does not have the buffering capacity to forward a packet

ICMP provides feedback about problems in the communication environment but it does not make IP reliable. The use of ICMP does not guarantee that an IP packet will be delivered or that an ICMP message will be returned to the source when an IP packet is not delivered or is incorrectly delivered.

For more information about ICMP, see RFC 792.

2.1.6.3 Address Resolution Protocol (ARP)

The Address Resolution Protocol maps Internet addresses to hardware addresses. ARP is not directly available to users or applications. When an application sends an internet packet, IP requests the appropriate address mapping. If the mapping is not in the mapping table, an ARP broadcast packet is sent to all the hosts on the network requesting the physical hardware address for the host.

For more information about ARP, see RFC 826.

2.1.7 Network Layer

This section describes the major protocols that implement the network layer in TCP/IP. Network protocols define how data is transported over a physical network. Typically a single host interface will be defined to use one of these protocols.

2.1.7.1 Network Driver Interface Specification (NDIS)

Network driver interface specification is a medium access control (MAC) interface for local area network (LAN) adapter and protocol drivers. NDIS has become an industry standard, providing a common, open interface that enables network adapters and LAN software from different manufacturers to communicate with each other.

A network adapter driver provides the communication between a network adapter and a protocol, using NDIS as the interface. Network adapter drivers handle the basic transmission and reception of packets on the network.

A protocol driver provides the communication between an application and a network adapter driver using NDIS as the interface. Protocol drivers provide a high level of communication between the data link layer and the application layer.

2.1.7.2 Serial Line Internet Protocol (SLIP)

The Serial Line Internet Protocol (SLIP) allows you to set up a point-to-point connection between two TCP/IP hosts over a serial line, for example, a serial cable or a RS-232 connection into a modem and over a telephone line. You can use SLIP to access a remote TCP/IP network (such as a service provider′s network) from your local host or to route datagrams between two TCP/IP networks.

For more information about SLIP, see RFC 1055.

2.1.7.3 Point-to-Point Protocol (PPP)

The point-to-point protocol allows you to set up a point-to-point connection between two TCP/IP hosts over a serial line, for example, a serial cable or a RS-232 connection into a modem and over a telephone line. You can use PPP to access a remote TCP/IP network (such as a service provider′s network) from your local host to route datagrams between two TCP/IP networks.

For more information about PPP, see RFCs 1717 and 1661.

2.1.7.4 X.25

You can use an X.25 network to establish a TCP/IP connection between two hosts. X.25, recommended as a communication interface standard by the International Telegraph and Telephone Consultative Committee (CCITT), defines the interface between data terminal equipment (DTE) and data circuit-terminating equipment (DCE). A DTE is a computer or workstation connected to a network. A DCE is the equipment at the point of the connection to the network, such as a modem.

For more information about TCP/IP over X.25, see RFC 877.

2.2 HTTP

HTTP is Hypertext Transfer Protocol. In simple terms, HTTP determines how a browser interacts with a server.

The current version of HTTP at the time of writing is 1.0 and it is defined in RFC 1945. However, an Internet draft which describes HTTP 1.1 is also available. Specifications may be obtained at the following site:

```
http://www.w3.org/pub/WWW/Protocols
```

HTTP enables distributed systems to communicate with each other. HTTP is an application-level protocol and has been in use by the WWW global information initiative since 1990.

HTTP is based on request-response activity. A client establishes a connection with a server and sends a request to the server in the form of a request method. The server responds with a status line, including the message′s protocol version and a success or error code, followed by a message containing server information, entity information and possible body content. This communication is simply depicted below.

```
User      sends request to  >.................................> Server
          single connection >.................................<
          response chain    <.................................<
```

Figure 1.   HTTP Client/Server Communication

An HTTP transaction is divided into four steps:

1. The browser opens a connection.
2. The browser sends a request to the server.
3. The server sends a response to the browser.
4. The connection is closed.

On the Internet, HTTP communication generally takes place over TCP/IP connections. The default port is TCP 80, but other ports can be used. This does not preclude HTTP from being implemented on top of any other protocol on the Internet, or on other networks. HTTP only presumes a reliable transport; any protocol that provides such guarantees can be used, and the mapping of the HTTP 1.0 request and response structures onto the transport data units of the protocol in question is outside the scope of this document.

Except for experimental applications, current practice requires that the connection be established by the client prior to each request and closed by the server after sending the response. Both clients and servers should be aware that either party may close the connection prematurely, due to user action, automated timeout, or program failure, and should handle such closing in a predictable fashion. In any case, the closing of the connection by either or both parties always terminates the current request, regardless of its status.

What we have just described means that, in simple terms, HTTP is a connectionless protocol. To load a page including two graphics for example, a graphic-enabled browser will open three TCP connections: one for the page, and two for the graphics. Most browsers, however, are able to handle several of these connections simultaneously.

HTTP is stateless, because it keeps no track of the connections. If a request depends on the information exchanged during a previous connection, then this information has to be kept outside the protocol.

For more information about the use of HTTP by Web browsers refer to Chapter 3, “Web Browsers” on page 47.

2.3 HTML

HTML stands for HyperText Markup Language. In this section we discuss:

- HTML characteristics
- Creating an HTML document
- The tagging reference
- How to create a sample document
- Netscape specific extensions
- Microsoft specific extensions
- Creating effective online information
- HTML editing tools

2.3.1 HTML Characteristics

HTML is a scripting language used to create Web pages. HTML is based upon SGML (Standard Generalized Markup Language) and therefore contains many of the same tags. HTML focuses on the content of the document and not on page layout. An HTML document is comprised of ASCII text and as such can be updated using any ASCII editor. The documents created are small and can be sent over the network faster than larger documents.

HTML can be used on any platform which has a browser that understands HTML. Therefore the creator of an HTML document does not need to be concerned with the platform on which a user will be viewing the document. HTML is changing; HTML 3.0 will include things such as aligned text, tables and mathematical equations.

There are also extensions to HTML found in browsers such as Netscape. If you use these extensions, only Netscape can understand them and not other browsers.

2.3.2 Creating an HTML Document

HTML documents are created by using a text editor, word processor or specific tools such as HTML-Wizard, HoTMetaL or HTML Assistant. HTML documents have an extension of HTM or HTML so browsers can find them.

2.3.2.1 What Is an HTML Tag?

For the purposes of discussion we introduce the following example:

```
<HTML> this is the text for this tag </HTML>
```

- Tags in general have a starting point <HTML>.
- Tags in general have an ending point </HTML>.
- HTML tags can be entered in upper, lower or mixed case.
- Text, which will be displayed by a browser, is entered between the starting and ending tags.
- Some tags do not have a starting and ending point.

The following describes how an HTML document is structured:

```
<HTML>
     <HEAD>
<TITLE>
     </HEAD>
<BODY>
 ....
 ....
     </BODY>
     </HTML>
```

Three tags are used to describe a document′s structure:

1. <HTML> starts and ends an HTML document. It indicates that the document contained herein is HTML. The document is created between the <HTML> and the </HTML> tags.
2. <HEAD> describes the prologue to the rest of the document. This section contains a <TITLE> tag to allow the browser to display it.
3. <BODY> delimits the rest of the HTML document. The main portion of the document is bounded by the <BODY> and </BODY> tags.

These structure tags are optional, but recommended. More tools and browsers will be able to understand your document if you use them.

2.3.3 HTML Tags

<HTML> </HTML> The HTML tag identifies the file as an HTML document.

Only one pair of HTML tags is allowed per document. Make them the first and last tags in the document.

<! .. > The ! tag enables you to create comments in your HTML document. Information written between comment tags is not displayed by browsers.

Use comments to include information in a document that you do not want to display, such as the date the document was created. Although browsers do not display the comment contents, the comment is part of the source and is accessible to the reader. Comments should only be one line long, but are stackable.

<HEAD> </HEAD> The HEAD tag identifies the section of the document that contains general document information, such as the title and root URL of the document.

Only one pair of HEAD tags is allowed per HTML document.

<BASE> The BASE tag identifies the root URL of the document, which is used with relative linking. This information is necessary in cases where the document has been copied to another URL. HREF is required to identify the original URL of the document. In cases where the document has been moved from its original URL, this string is appended to any relative links. The string does not consist of a pair of starting and ending tags. When used, it must be included within the HEAD tags.

<TITLE> </TITLE> The TITLE tag must occur within the HEAD tags. It cannot contain any links highlighting or paragraph tags. There can be only one title per document. Titles should be as specific as possible. However, due to the small amount of space that most browsers allow for a title, it is recommended that you not use titles longer than 50 characters. In most browsers, the title is displayed in the window title bar. Also, many browsers display the title in their history list or hot list.

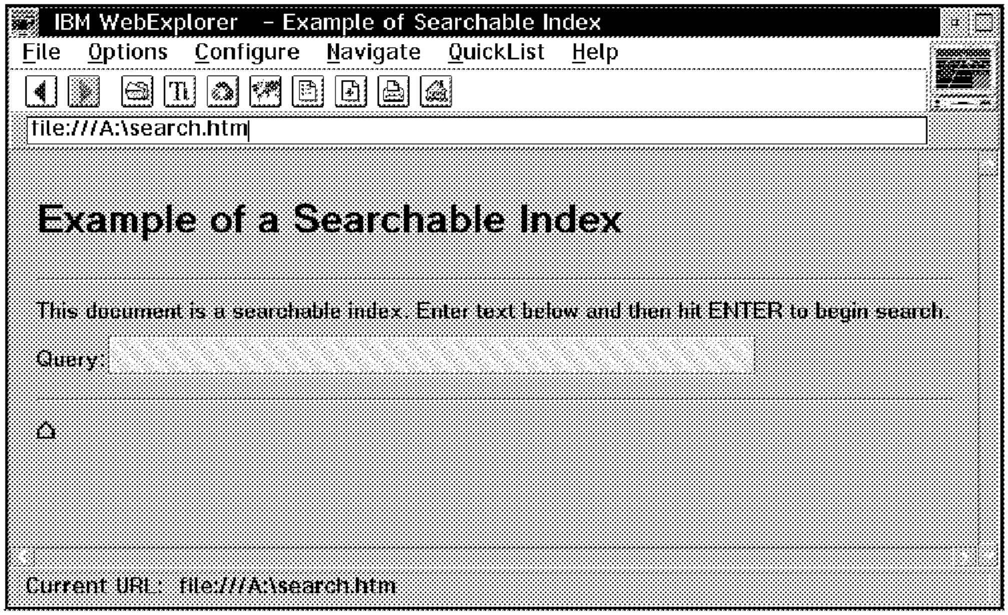
<BODY> </BODY> The BODY tag identifies the section of the document that contains the text and graphics.

<ISINDEX> Use the ISINDEX tag to provide a Web search interface.

The searchable index tag indicates to the Web server that this page provides an interface for keyword searches. The searchable index tag is a single tag; it does not have a paired starting and ending tag. The tag, if used, should occur in the BODY of the HTML document. The searchable index tag automatically generates an entry field and instructions for using it.

An example follows:

```
<HTML>
<HEAD>
<TITLE>Example of Searchable Index</TITLE>
</HEAD>
<BODY>
<H1>Example of a Searchable Index</H1>
<ISINDEX>
</BODY>
</HTML>
```



IBM WebExplorer - Example of Searchable Index
File Options Configure Navigate QuickList Help

file:///A:\search.htm

Example of a Searchable Index

This document is a searchable index. Enter text below and then hit ENTER to begin search.

Query:

Current URL: file:///A:\search.htm

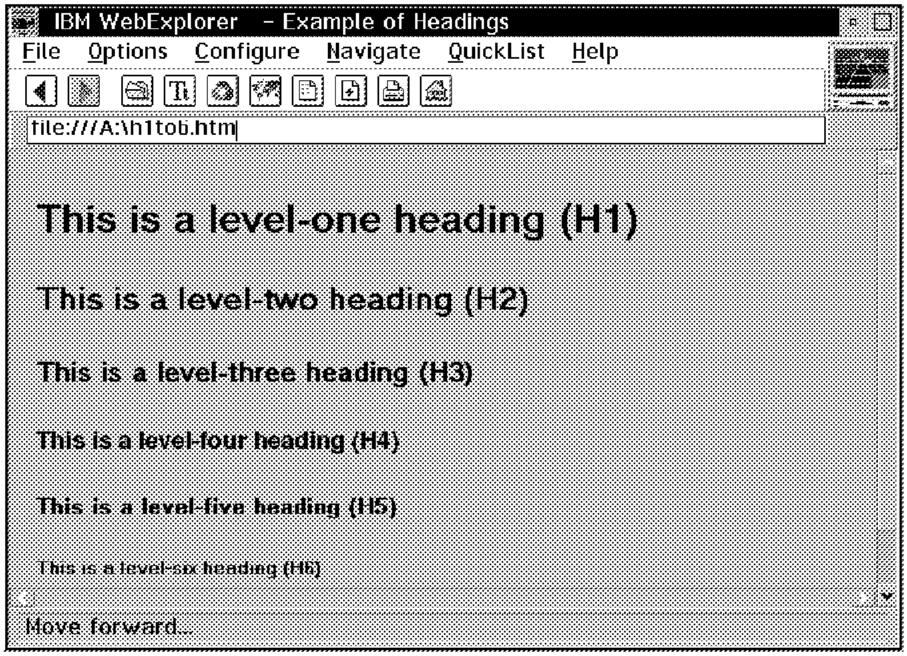
Figure 2. Searchable Index

<H1> </H1> Heading tag levels range from H1 (largest) to H6 (smallest).

You should begin each document with a primary heading <H1>. You should not skip head levels within a document. For example, if the first heading is an H1, subordinate information should begin with an H2, not an H3 tag.

An example follows:

```
<HTML>
<HEAD>
<Title>Example of Headings</TITLE>
</HEAD>
<BODY>
<H1>This is a level-one heading   (H1)</H1>
<H2>This is a level-two heading   (H2)</H2>
<H3>This is a level-three heading (H3)</H3>
<H4>This is a level-four heading  (H4)</H4>
<H5>This is a level-five heading  (H5)</H5>
<H6>This is a level-six heading   (H6)</H6>
</BODY>
</HTML>
```



IBM WebExplorer - Example of Headings
File Options Configure Navigate QuickList Help
file:///A:\h1tob.htm

This is a level-one heading (H1)

This is a level-two heading (H2)

This is a level-three heading (H3)

This is a level-four heading (H4)

This is a level-five heading (H5)

This is a level-six heading (H6)

Figure 3. Different Headings

<P> </P> You can also use the paragraph tag as a single tag <P>. If you use the single tag, place the paragraph tag at the end of a parameter or the beginning of one, but not both. Do not use a paragraph tag in conjunction with tags that also generate a blank line. In general, a paragraph tag generates a blank line or half a blank line of space before the next body of text. With some browsers, a paragraph tag will also cause the first line of the paragraph to be slightly indented.

<A> </A> The A (or anchor) tag is used to assign an ID to text to which you want to link.

Attributes:

- ***NAME***

  Optional. Used to assign an ID to text to which you want to link (a destination).

  For example:

  ```
  <H3><A Name="ingred">Ingredients</A></H3>
  ```

- ***HREF***

Optional. Used to identify the URL or ID of the information to which you are linking. If the destination is outside the current page, the identifier is the URL of the destination surrounded by quotation marks. This URL can be explicit or relative. Explicit URLs specify the protocol, the name of the server and location (path and filename) of the linked file. Relative URLs specify only the location (path and filename) of the linked file and must begin with a slash (/).

For example:

```
It is worth seeing the <A HREF="http://mistral.enst.fr"> site with
links to various museums</A>.
```

If the destination is inside the current page, the identifier is the ID assigned to the destination text (the value of the NAME attribute) preceded by a pound sign (#) and delimited by quotation marks.

For example:

```
<H3><NAME="ingred">Ingredients</A></H3>
 ..
 ..
Before you begin, make sure you have all the necessary
<A HREF="#ingred">ingredients.
```

- ***TITLE***

  Optional. When used with the HREF attribute, it indicates the title of the destination. If the destination document or page has a title, this attribute is informational only. If the destination document or page does not have a title, as with some Gopher servers, many browsers will display the value of this attribute as the title.

2.3.3.1 Highlighting Text

Tags for highlighting text distinguish certain text from the rest of the document by displaying the text in a different manner. There are two types of highlighting:

- Explicit
- Interpreted

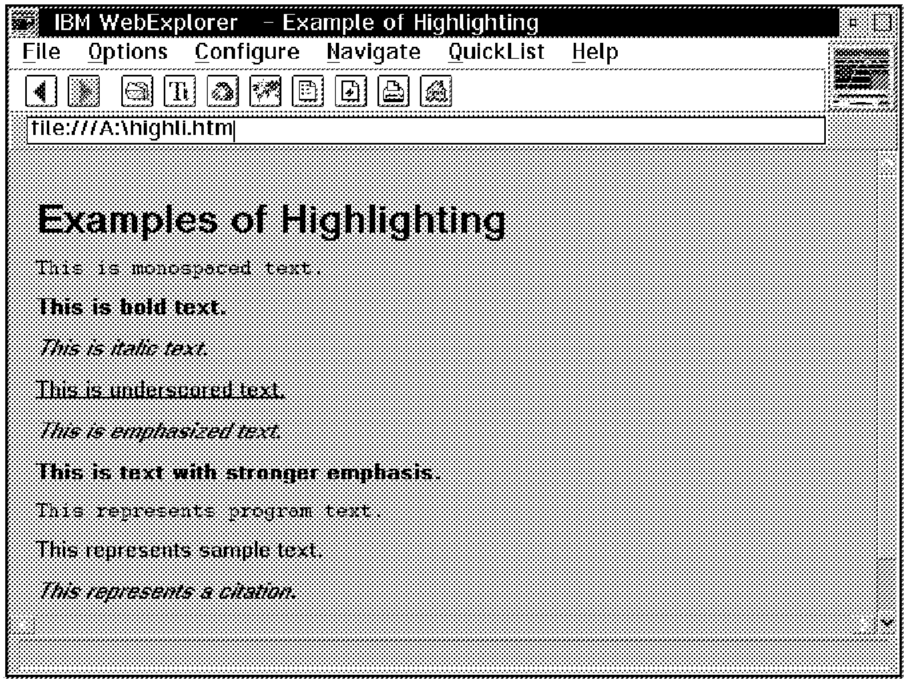
Explicit highlighting instructs the browser to display the text in a specific manner. Interpreted highlighting instructs the browser to display the text in whatever manner it determines is appropriate for the given emphasis.

If a browser does not know how to interpret a highlighting tag, it will not highlight the surrounded text.

An example follows:

```
<HTML>
<HEAD>
<TITLE>Example of Highlighting</TITLE>
</HEAD>
<BODY>
<H1>Examples of Highlighting</H1>
<TT>This is monospaced text.</TT><P>
<B>This is bold text.</B><P>
<I>This is italic text.</I><P>
<U>This is underscored text.</U><P>
<EM>This is emphasized text.</EM><P>
```

```
<STRONG>This is text with stronger emphasis.</STRONG><P>
<CODE>This represents program text.</CODE><P>
<SAMP>This represents sample text. </SAMPLE><P>
<CITE>This represents a citation.</CITE><P>
</BODY>
</HTML>
```



file:///A:\highli.htm

Examples of Highlighting

This is monospaced text.

This is bold text.

This is italic text.

This is underscored text.

This is emphasized text.

This is text with stronger emphasis.

This represents program text.

This represents sample text.

This represents a citation.

Figure 4. Different Highlighting

<TT> </TT> Monospaced text appears in a font resembling typewritten output. Monospaced text is an explicit highlighting that causes the surrounded text to be displayed in a font resembling that of a typewriter where every character (whether it is an i or a w) occupies the same amount of horizontal space.

<B> </B> Bold text is a thicker version of the default font. Bold is an explicit highlighting that causes the surrounded text to be displayed in a bold version of the default font.

<I> </I> Italic text is a slanted version of the default font. Italic is an explicit highlighting that causes the surrounded text to be displayed in an italic (slanted) version of the default font.

<U> </U> Underscored text is the default font with an underline. Underscored is an explicit highlighting that causes the surrounded text to be displayed in the default font with a line drawn beneath.

<EM> </EM> Emphasized text appears in a different font, usually italics.

<STRONG> </STRONG> Strongly emphasized text appears in a different font, usually bold. Strong is an interpreted highlighting that causes the surrounded text to be displayed with stronger emphasis than <EM>. In general, most browsers will display the text in bold font.

<CODE> </CODE> Code is an interpreted highlighting that causes the surrounded text to be displayed as simulated code. In general, most browsers will display the text in monospaced font.

<SAMP> </SAMP> Sample is an interpreted highlighting that causes the surrounded text to be displayed as a simulated sample. In general, most browsers will display the text in a monospaced font.

<KBD> </KBD> Keyboard is a simulated text entry, usually used in a monospaced font.

<VAR> </VAR> Variable is an interpreted highlighting that causes the surrounded text to be displayed with emphasis indicating it is a variable (information that can be changed, such as the value of a parameter). In general, most browsers will display the text in italic.

<DFN> </DFN> Definition is an interpreted highlighting that causes the surrounded text to be displayed with emphasis indicating it is a term or phrase defined in text. In general, most browsers will display the text in bold or in bold italic font.

<CITE> </CITE> Citation is an interpreted highlighting that causes the surrounded text to be displayed with emphasis indicating that it cites a reference, such as a book title. In general, most browsers will display the text in italic.

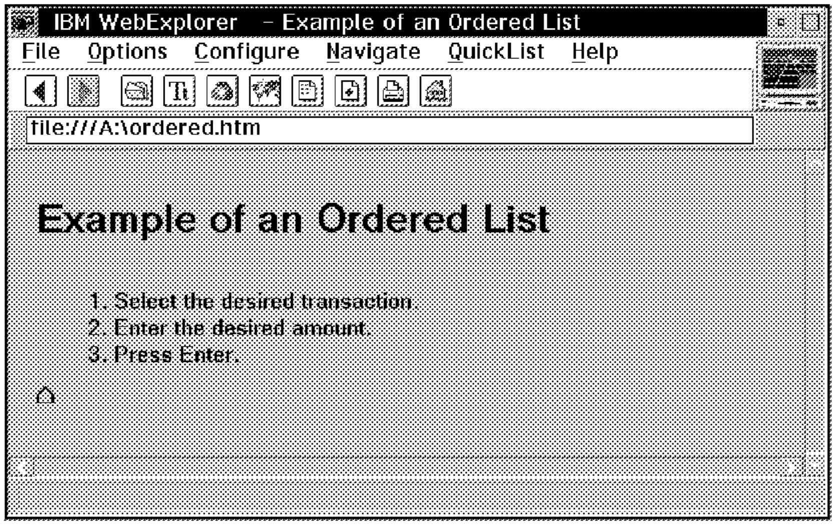
<OL> </OL> The OL or ordered list tags generate a sequential list of individual items. Use an ordered list when the items must be addressed in a specific sequence, such as the steps of a procedure.

Place the beginning and ending tags on separate lines. The first line after the starting tag <OL> must be a list item and begin with <LI>. Each item in the list must be preceded by <LI>.

Each item begins on a separate line and is preceded by a number and a period.

An example follows:

```
<HTML>
<HEAD>
<TITLE>Example of an Ordered List</TITLE>
</HEAD>
<BODY>
<H1>Example of an Ordered List</H1>
<OL>
<LI>Select the desired transaction.
<LI>Enter the desired amount.
<LI> Press Enter.
</OL>
</BODY>
</HTML>
```


IBM WebExplorer - Example of an Ordered List
File Options Configure Navigate QuickList Help
file:///A:\ordered.htm

Example of an Ordered List

1. Select the desired transaction.
2. Enter the desired amount.
3. Press Enter.

Figure 5. Ordered List

<UL> </UL> The UL or unordered list tag is used to generate a bulleted list of individual items. Use an unordered list when the sequence of the items is unimportant, such as a list of benefits.

Place the beginning and ending tags on separate lines. The first line after the starting tag (<UL>) must be a list item and begin with <LI>. Each item in the list must be preceded by <LI>.

<MENU> </MENU> The MENU tag generates a list of individual items without numbers or bullets. Use a menu list when the items are brief (usually no more than one line) and a sequence of the items is unimportant, such as a shopping list.

Place the beginning and ending tags on separate lines. The first line after the starting tag <MENU> must be a list item and begin with <LI>. Each item in the list must be preceded by <LI>.

<DIR> </DIR> The DIR or directory tag generates a tabular list of individual items. Use a directory list when items need to be displayed in columns such as a price list.

Place the beginning and ending tags on separate lines. The first line after the starting tag (<DIR>) must contain the items of the first row. Each item in the list must be preceded by <LI>.

Each row of items begins on a separate line. In most cases, a column is limited to about 25 characters. This is not a widely supported list type.

<DL> </DL> The DL or definition list tag generates a list of paired items. Use a definition list when you want to list and describe items, such as a list of glossary terms and definitions.

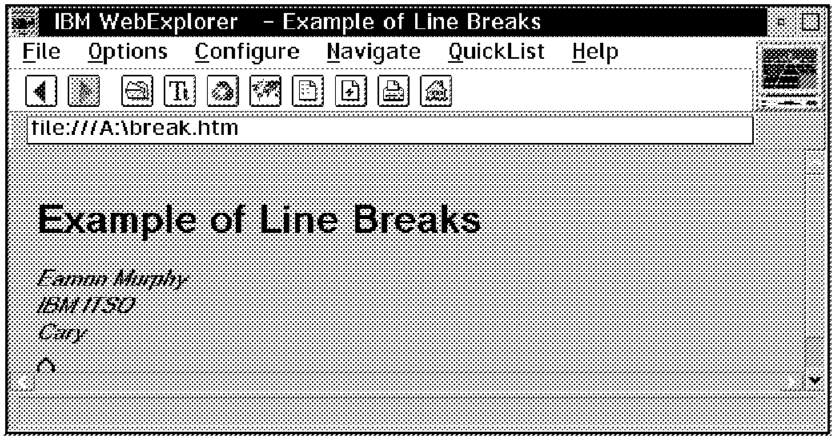
The attribute compact removes the vertical spacing between the term and its description. Place the beginning and ending tags on separate lines. Each term must be preceded by <DT>. Each description must be preceded by <DD>. For each <DT> you must have a corresponding <DD>.

<BR> The BR or line break tag is a single tag; it does not have paired starting and ending tags.

Text following the break is placed on the next line. No additional vertical space is generated.

An example follows:

```
<HTML>
<HEAD>
<Title>Example of Line Breaks</Title>
</HEAD>
<BODY>
<H1>Example of Line Breaks</H1>
<Address>
Eamon Murphy<BR>
IBM ITSO<BR>
Cary<BR>
</ADDRESS>
</BODY>
</HTML>
```


IBM WebExplorer - Example of Line Breaks
File Options Configure Navigate QuickList Help
file:///A:\break.htm

Example of Line Breaks

Eamon Murphy
IBM ITSO
Cary

Figure 6. Line Breaks

<HR> The HR or horizontal rule tag generates a line across the page (no ending tag). Use horizontal rule tags to create a visual break in the page.

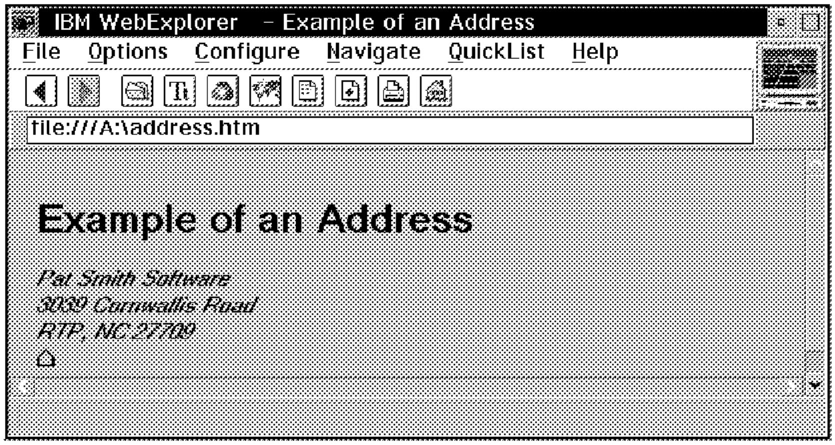
<ADDRESS> </ADDRESS> The ADDRESS tag combines interpreted highlighting and paragraph formatting. Use address tags to set off document identification information and addresses.

Paragraph tags used within address tags act as line break tags. They do not generate additional vertical space. To generate a stacked address format, similar to the format of an address on an envelope, use line break tags inside the address tags.

As with interpreted highlighting tags, such as CITE, the browser controls the emphasis that is placed on the surrounded text. In general, most browsers will display the text in italic. Also, the ending address tag acts as a paragraph tag, causing a line break and generating additional vertical space prior to any following information.

An example follows:

```
<HTML>
<HEAD>
<Title>Example of an Address</TITLE>
</HEAD>
<BODY>
<H1>Example of an Address</H1>
<ADDRESS>
Pat Smith Software<BR>
3039 Cornwallis Road<BR>
RTP, NC 27709
</ADDRESS>
</BODY>
</HTML>
```


IBM WebExplorer - Example of an Address
File Options Configure Navigate QuickList Help

file:///A:\address.htm

Example of an Address

Pat Smith Software
3039 Cornwallis Road
RTP, NC 27709

Figure 7. Address

<BLOCKQUOTE> </BLOCKQUOTE> The BLOCKQUOTE tag highlights text intended as a quote, usually in italic font.

As with interpreted highlighting tags, such as CITE, the browser control the emphasis that is placed on the surrounded text. In general, most browsers will display the text in italic. Some also indent the margins for the surrounded text. Also the ending block quote tag acts as a paragraph tag, causing a line break and generating additional vertical space prior to any following information.

<PR> </PR> The PR or preformatted text tag combines interpreted highlighting and line breaks. Use preformatted text tags to set off information that needs to be displayed as it is in the HTML file, for example, lines from a program or sample file.

HTML tags included within preformatted text tags are interpreted as tags. If you want to include an example that shows HTML tagging, you must use the HTML symbols to create the tag delimiters (<>). Highlighting and linking can be used within preformatted text. However, headings and formatting tags, such as paragraph and address tags, should not be used within preformatted text.

<IMG> The IMG or image tag enables you to include an inline image in your document.

Image tags can take the following attributes:

1. ***SRC***: Required attribute. Specifies the source file of the inline image. The SRC attribute is followed by a character string (identifier) that identifies the image file and its location.

2. ***ALIGN***: Optional attribute. Specifies how the image should be aligned vertically. Possible values are TOP, MIDDLE and BOTTOM.

3. ***ALT***: Optional attribute. Specifies the label for the image that is to be displayed if the browser does not support inline graphical images.

4. ***ISMAP***: Optional attribute. Specifies that the image contains defined areas that, when selected, link to other URLs.

The image tag is a single tag. It is not a paired set of starting and ending tags. Some browsers cannot display inline images, but can display linked images. If an image is crucial to a document, you may want to link to it rather than include it inline.

For browsers that can display graphical images inline, the image is left justified. If a series of images is specified, the images are displayed on the same line, if possible. For browsers that cannot display images inline, the label, if any, is displayed.

An example follows:

```
<HTML>
<TITLE>Example of Inline Graphics</TITLE>
<BODY>
<H1>Example of Inline Graphics</H1>
This site contains information about the following Acme products:
<P>
<IMG SRC=’ball_pur.gif’><A HREF=’st200.html’>Spring Trap Model 220</A><P>
<IMG SRC=’ball_pur.gif’><A HREF=’cat40.html’>Spring Large Model 40</A><P>
<IMG SRC=’ball_pur.gif’><A HREF=’tun50.html’>Spring Tuned Model 50</A><P>
<IMG SRC=″/av/pix/altavista.gif″ BORDER=0 ALIGN=middle HEIGHT=54 WIDTH=9>
<IMG SRC=line_owl.gif><P>
<IMG SRC=’mailbox.gif’>Feel free to drop a card in our<A HREF=’mail.html’
>suggestion box</A>.
</BODY>
</HTML>
```

Images are mostly .gif files. They are included on pages using the IMG tag. You specify where the image is located using the SRC attribute. The SRC attribute uses the same file conventions as the HREF attribute. Just put the IMG tag where you want the image to appear. Instead of a .gif image file you can also use the following formats:

- jpeg compressed image format (.jpg)
- wave format sound files (.wav)
- mpeg audio (.mp2)
- mpeg video (mpg)
- avi video (.avi)

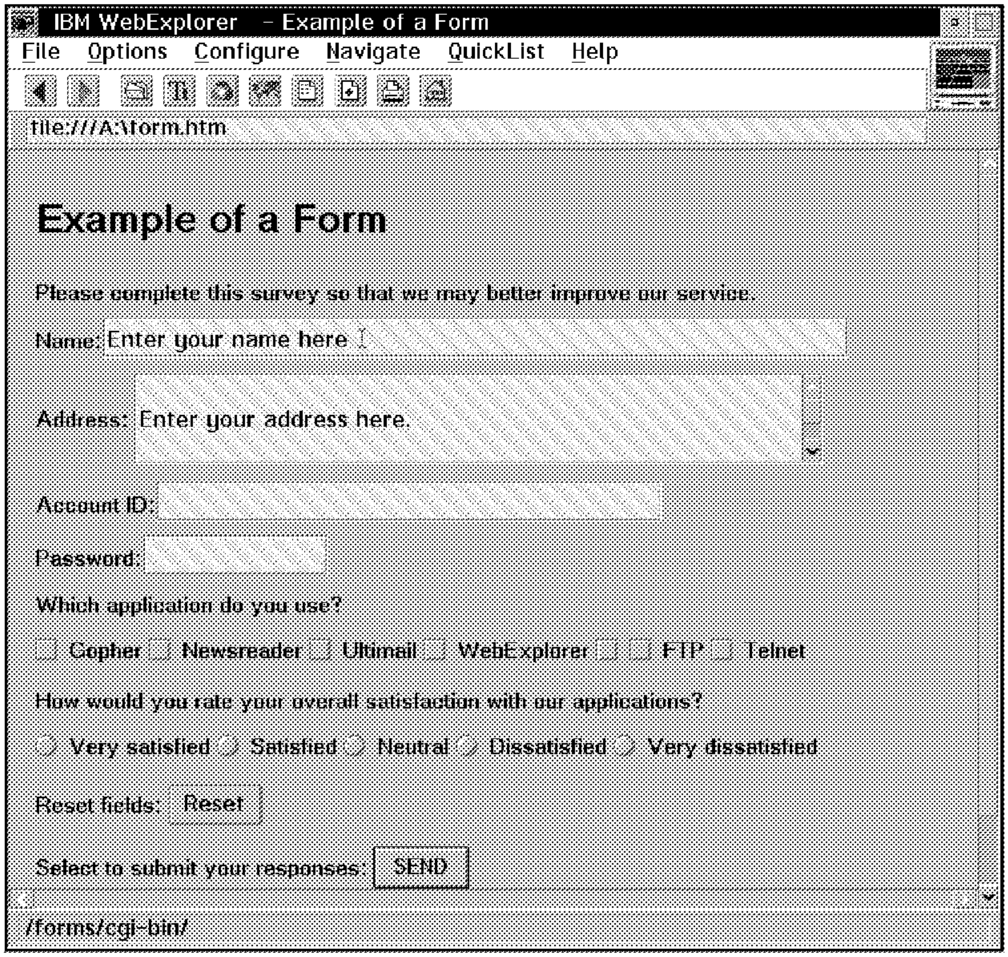
2.3.3.2 Creating Forms

The FORM tag allows you to create input forms.

Data from these forms can be submitted to a Web server for processing and analysis.

Example for creating a form:

```
<HTML>
<HEAD>
<TITLE>Example of a Form</TITLE>
</HEAD>
<BODY>
<H1>Example of a Form</H1>
<FORM METHOD=POST ACTION=″mailto:survey@vnet.ibm.com″>
Please complete this survey so that we may better improve our service.<p>
Name: <INPUT TYPE=″text″ NAME=″name″ SIZE=45 MAXLENGTH=50
VALUE=″Enter your name here″><P>
Address: <TEXTAREA NAME=′address′ COLS=40 ROWS=3>
Enter your address here.
</TEXTAREA><P>
Account ID: <INPUT TYPE=″text″ NAME=″acctID″ SIZE=30><P>
Password: <INPUT TYPE=″password″ NAME=″password″ SIZE=10><P>
Which application do you use?<P>
<INPUT TYPE=″checkbox″ NAME=″apps″ VALUE=″Gopher″> Gopher
<INPUT TYPE=″checkbox″ NAME=″apps″ VALUE=″Newsreader″>
Newsreader
<INPUT TYPE=″checkbox″ NAME=″apps″ VALUE=″ULTIMAIL′>
Ultimail
<INPUT TYPE=″checkbox″ NAME=″apps″ VALUE=″WebExplorer″>
WebExplorer
<INPUT TYPE=″checkbox″ NAME=″apps″ VALUE=″Archie″>
<INPUT TYPE=″checkbox″ NAME=″apps″ VALUE=″FTP″>
FTP
<INPUT TYPE=″checkbox″ NAME=″apps″ VALUE=″Telnet″>
Telnet<P>
How would you rate your overall satisfaction with our applications?<P>
<INPUT TYPE=″radio″ NAME=″rating″ VALUE=″1″> Very satisfied
<INPUT TYPE=″radio″ NAME=″rating″ VALUE=″2″> Satisfied
<INPUT TYPE=″radio″ NAME=″rating″ VALUE=″3″> Neutral
<INPUT TYPE=″radio″ NAME=″rating″ VALUE=″4″> Dissatisfied
<INPUT TYPE=″radio″ NAME=″rating″ VALUE=″5″> Very dissatisfied<P>
Reset fields: <INPUT TYPE=″reset″><P>
Select to submit your responses: <INPUT TYPE=″submit″
Value=″SEND″
</FORM>
</BODY>
</HTML>
```


IBM WebExplorer - Example of a Form
File Options Configure Navigate QuickList Help

file:///A:\form.htm

Example of a Form

Please complete this survey so that we may better improve our service.

Name: Enter your name here

Address: Enter your address here.

Account ID:

Password:

Which application do you use?

□ Gopher □ Newsreader □ Ultimail □ WebExplorer □ □ FTP □ Telnet

How would you rate your overall satisfaction with our applications?

○ Very satisfied ○ Satisfied ○ Neutral ○ Dissatisfied ○ Very dissatisfied

Reset fields: Reset

Select to submit your responses: SEND

/forms/cgi-bin/

Figure 8. An Example of a Form

<FORM> </FORM> The FORM tag indicates that the surrounded information is part of a data entry form.

Form tags can take the following attributes:

- ***METHOD*** - Required option. Indicates the HTTP method used to send the data to the server. Possible values are GET and POST. It is recommended you use POST.
- ***ACTION*** - Optional. Specifies the URL of the processing script. The URL must be enclosed in quotation marks. If the ACTION attribute is not specified, the default is the URL of the document.
- ***ENCTYPE*** - Optional. Specifies how the input data is encrypted.

<INPUT> </INPUT> The INPUT tag can be used to create elements in an HTML document that accept user input. These elements may be in the form of selections or entry fields.

The attributes that an input tag can take depends upon the input type.

- TYPE

  Is required. Specifies the type of fields to be displayed. Possible values for field type are:

  – CHECKBOX

Displays a box that can be selected. Use check boxes for Boolean selections (on, or, off, yes or no). Multiple check boxes can be grouped together. One or more check box in a group may be selected.

Required attributes are NAME and VALUE.

An optional attribute is CHECKED.

- HIDDEN

  Does not accept or display any information to the user. Hidden form fields are used to send status information to the server.

  A required attribute is NAME.

  An optional attribute is VALUE.

- IMAGE

  Displays a graphic, which, when selected, submits the data to the specified URL. Because IMAGE may be obsolete in the future, it is recommended that you use the SUBMIT form type.

  Required attributes are NAME and SRC.

  An optional attribute is ALIGN.

- PASSWORD

  Displays a single line entry field. Password fields are similar to text fields except information entered in this field is not displayed.

  A required attribute is NAME.

  Optional attributes are MAXLENGTH, SIZE and VALUE.

- RADIO

  Displays a radio button (a circle that can be selected). Use radio buttons for multiple choice selections where only one in a series can be selected. All radio buttons in a group should be assigned the same NAME.

  Required attributes are NAME and VALUE.

  An optional attribute is CHECKED.

- RESET

  Displays a push button that when selected returns all the form′s fields to their original values. Use the VALUE attribute to define the label for the push button.

  The default value is RESET.

  An optional attribute is VALUE.

- SUBMIT

  Displays a push button that when selected, submits the data. Use the VALUE attribute to define the label for the push button. The default label is SUBMIT. You can also use the SRC attribute to include an image on the push button.

  Optional attributes are NAME, SRC and VALUE.

  - TEXT

    Displays a single-line entry field. If you require a multiple-line entry field, use the TEXTAREA tag.

    A required attribute is NAME.

    Optional attributes are MAXLENGTH, SIZE and VALUE.

- SRC

  Required with type=′image′.

  Specifies the URL of the graphic to be displayed.

- ALIGN

  Optional.

  Specifies how the graphic should be aligned vertically. Possible values are TOP, MIDDLE and BOTTOM.

- CHECKED

  Optional.

  Indicates the default section.

- MAXLENGTH

  Optional.

  Indicates the maximum number of characters that can be entered into a field. If MAXLENGTH is greater than SIZE, the entry field will allow the field to scroll as information is entered.

- NAME

  Required.

  The identifier assigned to a field. When a form is submitted, the name of a field is paired with its value.

- SIZE

  Optional.

  Specifies the width in characters of a field area.

- VALUE

  Required with TYPE=″checkbox″ and TYPE=″radio″.

  For entry fields, VALUE is used to specify the initial setting. For check boxes and radio buttons. VALUE is used to specify the value assigned to a selection. For SUBMIT and RESET, VALUE is the label to appear on the push button.

<TEXTAREA> </TEXTAREA> The TEXTAREA tag can be used to create multiple-line entry fields.

TEXTAREA tags can take the following attributes:

- ROWS

  Required.

  Indicates the number of rows of input allowed.

- COLS

  Required.

Indicates the number of characters allowed for each row.

- NAME

  Required.

  The identifier assigned to a field. When a form is submitted, the name of a field is paired with its value.

<SELECT> </SELECT> The SELECT tag is used to create a list box of choices, similar to the radio or check box input types.

Select tags can take the following attributes:

- MULTIPLE

  Optional.

  Indicates that multiple selections are allowed.

- SIZE

  Optional.

  Indicates the number of items displayed in the selection list at one time. If SIZE is less than the number of items listed, a scroll bar is displayed to the right of the field, allowing users to scroll through the other items in the list.

- NAME

  Required.

  The identifier assigned to a field. When a form is submitted, the name of a field is paired with its value.

<OPTION> </OPTION> The OPTION tag is used with SELECT tags to create a list of choices, similar to the radio or check box input types.

Option tags can take the following attributes:

- SELECTED

  Optional.

  Indicates that this option is the default.

- VALUE

  Optional.

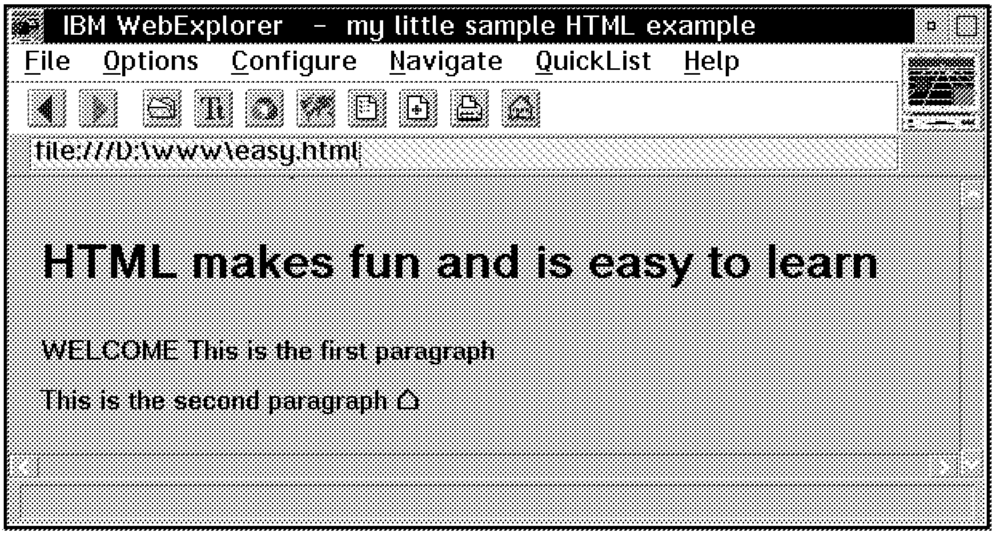
  The value to be paired with the name if this option is chosen. If no VALUE is specified, the text associated with the option is sent as the value.

2.3.4 Sample Document

```
<html>
<head>
<TITLE> my little sample HTML example     </TITLE>
                                          </HEAD>
<BODY>
<H1> HTML makes fun and is easy to learn </H1>

<P>  WELCOME
     This is the first paragraph          </P>

<P>  This is the second paragraph         </P>
```

```
</BODY>
</HTML>
```

The required elements are the <HTML>, <HEAD> and <BODY> tag with their end tags. Because you should include these tags in each file, you want to create a template file with them for further use.



file:///D:\www\easy.html

HTML makes fun and is easy to learn

WELCOME This is the first paragraph

This is the second paragraph

Figure 9. Web Browser View of Sample Document

If you require a complete definition of the HTML commands, you can choose from a wide range of books. In addition, there are numerous WWW links that will assist you in the proper use of HTML. For example:

```
http://www.ncsa.uiuc.edu/General/Internet/WWW/HTMLPrimer.html
http://www.ucc.ie/info/net/htmldoc.html
```

2.3.5 Netscape-Specific Extensions

Here are some extensions to HTML that have been provided by Netscape:

NOBR: No line break

WBR: Word break

BASEFONT: Specifies the default font size for the document

BLINK: Marks the enclosed text as blinking text

APPLET: Used to include an inline applet

PARAM: Defines a parameter for an applet

FRAMESET: Defines the layout of window frames within the browser window

SCRIPT: Used to include program scripts within an HTML document

2.3.6 Microsoft Internet-Specific Extensions

Here are some extensions to HTML that have been provided by Microsoft:

MARQUEE MARQUEE denotes a text string to be scrolled horizontally on the display; the content of the element is the text to be scrolled.

BGSOUND BGSOUND inserts an inline audio snippet. By default, the sound is loaded and played once.

2.3.7 Creating Effective Online Information

Although there are many printable documents available on the Web, the primary purpose of the Web is to provide information in an online format.

Regardless of whether you are producing printed or online information, there are three aspects to consider:

1. Style
2. Content
3. Mechanics

This chapter provides some style guidelines and tips to help you produce Web information. Although you are responsible for the content of your Web information, this chapter provides some style guidelines that you may find helpful.

2.3.7.1 Understanding Form

Information is often displayed in sizable windows. You do not have control over the size of the window as you would with a printed book. Therefore, you should steer away from complex formats, such as multiple columns and large amounts of information in tabular format.

In many cases, the form can reinforce the content. If the information describes a procedure, place the steps in an ordered (numbered) list. If the information contains a set of choices, benefits or considerations, use an unordered (bulleted) list. Lists can enhance the appearance of information as well as aid in user comprehension.

2.3.7.2 Deciding When to Link

In some cases, despite attempts to keep it simple and concise, a topic cannot be covered in a few pages of information. This situation calls for linking. Consistency is one of the most important guidelines of linking. It is important that you adopt a well-defined criteria for deciding when to include information inline and when to separate it out as a separate entity to which you link. Users become accustomed to how information is presented. For example, we are all accustomed to finding the index in the back of the book. Because online information is a relatively young medium, the rules of consistency are still being developed.

In general, you should design the initial page of information for the high-level user; the user who needs the least amount of information. Then, provide links to additional information, such as definitions of terms that a novice user might not understand, technical information that an experienced user might want, or information about related topics. This will help reduce the amount of information on the initial page. In online information, the term page is used to mean discrete blocks of information. This block may be more than a single window′s worth of information. It may also contain links to (and be linked to by) other pages.

2.3.7.3 General Guidelines for Web Information

In this section we give you some general guidelines for providing information in HTML documents on the Web.

Be Brief: The acceptable length of a Web page document varies. First, try to limit the number of times a reader has to scroll through a document. Three times is a good limit. However, keep in mind that the acceptable length of a document should also depend on the topic. On the whole, high-level documents such as overviews, should be kept short. The reader expects this type of information to be brief. In-depth documents are expected to be longer. Remember, you can separate out and link to related information. But do not try to break up a document unnaturally just to make it shorter.

Another element to consider is time. The information a reader is accessing is likely to reside on a computer hundreds or thousands of kilometers away. The amount of time it takes a browser to access and display a document will depend on the speed of the reader′s connection. For example, using a computer attached to a company network that is connected to the Internet via a dedicated line, it might take 12 seconds to display the entire document. Displaying the same document using a computer with a 14,400 baud dial-in connection to the Internet might take 23 seconds. Individually, these access times may seem acceptable. But when you consider that readers may be waiting several seconds for each document, they may not be willing to wait for your document to be displayed if it takes a few seconds longer than expected.

Provide Navigation Aids: It is not unusual for a reader to get lost perusing the thousands of interconnected documents on the Web. In addition to any links that you may have in your document, it will be helpful to your readers if you provide push buttons or icons at the bottom of your document that link back to the parent document or forward to another related topic. If yours is an exceptionally long document, you might consider providing a link at the bottom to the top.

Clearly Identify the Document: When you send out a memo, write a letter or provide a report for others to read, it is important that you include the date and your name. This helps those who receive the document to know where it came from and when. Considering the increased distribution that your document may experience on the Web, this identification is even more important.

Despite the fact that HTML allows you to easily link to documents on other servers, sometimes people prefer to copy a document to their own server. Therefore it is a good idea to always clearly identify your document at the beginning of your document. You should include the date, the status (such as Draft or First Revision) and your name (including your e-mail address). Including your e-mail address will allow others to contact you if they have comments or notify you if they plan to link to your document.

Construct Links Appropriately: The intent of hypertext coding is to allow you to highlight phrases that, when selected, lead the reader to additional information. However, the content of a document should read well without the links. Consider, for example, if someone wanted to print a document. The links would no longer be functional, but the sentences would nonetheless read coherently.

Some examples follow:

- If you want more information about any of these products, contact your **IBM Marketing Representative**.

In this case, the words **IBM Marketing Representative** provide a link to information about how to contact IBM via phone or e-mail. However, if you remove the link, the sentence still makes sense.

The following sentence demonstrate a poor use of linking:

- If you want more information about any of these products, **click here**.

If the link is not operational, the sentence no longer contains useful information.

Avoid Copying Documents: When you find information that you want to include in your document, do not copy it; simply link to it. A good many of the documents on the Web are living documents, which means that they are likely to evolve and change. If you copy a document to your server, the author may update the original document, making your copy obsolete. If you link to the document, your readers will always be linking to the latest information.

Once you begin to maintain your own Web server, you will be surprised at how quickly you can run out of space if you do not plan and manage your information well. In addition to saving time (with change management), linking to a document rather than copying one will also save space.

Using Graphics: Hypertext documents on the World Wide Web represent a significant achievement in documentation. More information is available to more people and the retrievability of the information is relatively easy. With the introduction of graphics and multimedia, these documents can be more effective than their printed predecessors.

Graphics can enhance a document. An illustration of a concept is sometimes more meaningful than words. In most cases, a graphic can convey a message without words. The graphic is therefore language-independent, making your document more useful to those who may not speak your language. Graphics can also add flair to a document, making it more interesting to your reader.

However, using graphics in a Web document is not without its problems. Some browsers do not support inline graphics. Therefore, if your graphic contains important information, such as links to other information, you will need to supply a textual equivalent for the graphic.

Graphics are transferred separately from the text of a Web document; they require additional transfer time. For a reader with a dial-in connection, the additional time can cause the reader to lose interest and cancel the request for the document. This is particularly true for readers who have a browser that does not display Web documents until all associated graphics have been transferred.

Using Multimedia: Multimedia (video and audio clips) can also enhance the appeal and usefulness of a document. Video clips bring a concept to life. For example, without multimedia an overview of heart functions would take paragraphs of text and include numerous medical terms that require further explanation. The chances are, many readers would abandon the document after they hit the second or third paragraph. By including a video clip, the action of the heart comes to life. The video clip captures the attention of the reader while providing the information in a format that can be understood by readers of various knowledge levels. Add to that an audio clip that narrates the action portrayed in the video, and it is like having a biology class in your computer.

It is quite easy to include video and audio clips in your Web document. Once you have created your video or audio file, it is simply a matter of linking to that life. The server and browser take care of the rest.

However, be careful of relying too heavily on multimedia to convey your message. Not all browsers are capable of audio and video playback. Also remember, that for most browsers, the playback of these files is provided through a separate program. Therefore, the entire audio or video file must be transferred to the reader′s computer before it can be played back. This will require additional wait time, additional disk space, and additional memory. This may prevent some of your readers with more modest equipment from viewing or hearing your message.

2.3.8 HTML Editing Tools

An HTML editor assists you in creating Web pages. Depending on the power of the editor, a full set of HTML tags may be supported. There are many editing tools available on the Internet:

- Boxer
- HTML Assistant
- HTML Hyperedit
- HTML Generator
- HTML Wizard
- HotDog
- HoTMetal
- SpHydir
- WebWriter/2
- EPM with HTML extensions

A reference to HTML editors may be found on:

```
http://www.w3.org/pub/WWW/Tools/Overview.html
```

The following are Gif/Jpeg editing programs:

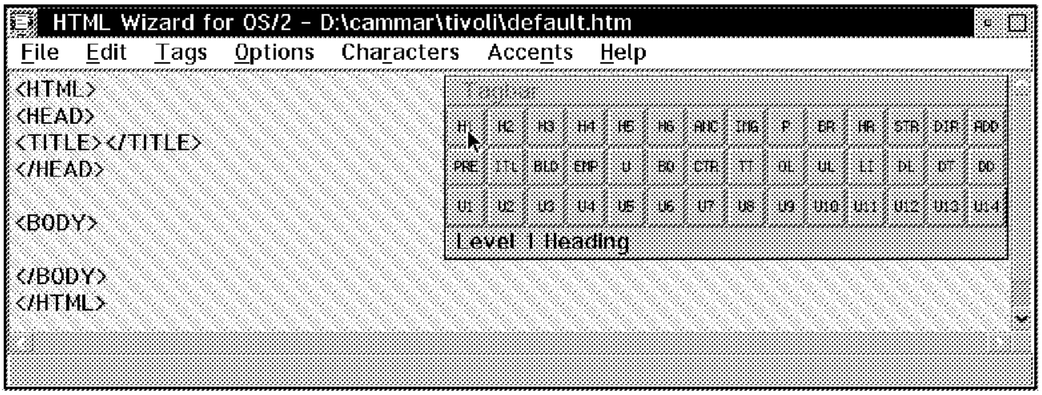
- PMJpeg
- WebGif

2.3.8.1 HTML Validation Service

This service makes use of HTML forms and a CGI script which runs an HTML validation program.

The following are reference Web pages:

```
http://www.arnes.si/books/www-handbook/handbook.long.html
http://www.lionsgate.com/VanOS2/WebCoding/web_tools.html
http://www.webtechs.com/html-val-svc
```


HTML Wizard for OS/2 - D:\cammar\tivoli\default.htm
File
Edit
Tags
Options
Characters
Accents
Help
Toolbar
H1 H2 H3 H4 H5 H6 ANC IMG P BR HR STR DIR ADD
PRE ITL BLD EMP U BQ CTR TT OL UL LI DL DT DD
U1 U2 U3 U4 U5 U6 U7 U8 U9 U10 U11 U12 U13 U14
Level 1 Heading

```
<HTML>
<HEAD>
<TITLE></TITLE>
</HEAD>

<BODY>

</BODY>
</HTML>
```

Figure 10. HTML-Wizard

2.3.9 Icons and Cliparts

To produce HTML documents containing icons and cliparts use this list of resources:

```
ftp://ftp.winsite.com/pub/pc/win3/icons
http://www.cli.di.unipi.it/iconbrowser/icons.html
```

2.3.10 Best of the Web

A link with references to other URLs which show the best pages, is at:

```
http://www.yahoo.com/Computers_and_Internet/Internet/World_Wide_Web/Best_
of_the_Web/index.html
```

PC Magazine Top 100 Web sites are found at:

```
http://www.pcmag.com/special/web100
```

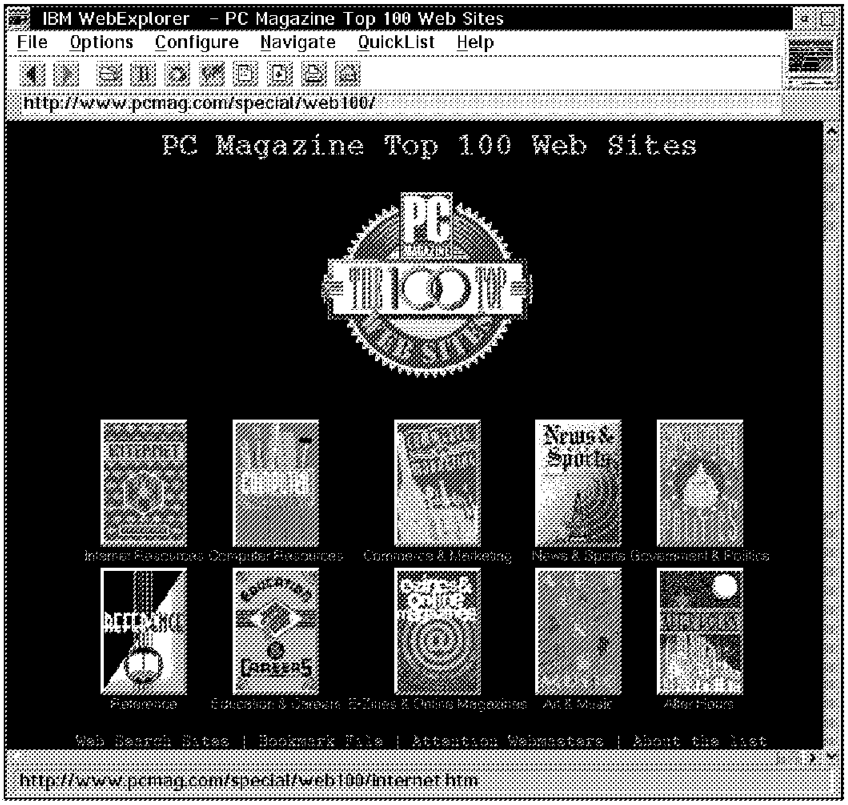

IBM WebExplorer - PC Magazine Top 100 Web Sites
File Options Configure Navigate QuickList Help
http://www.pcmag.com/special/web100/
PC Magazine Top 100 Web Sites
Internet Resources
Computer Resources
Commerce & Marketing
News & Sports
Government & Politics
Reference
Education & Careers
E-Zines & Online Magazines
Art & Music
After Hours
Web Search Sites | Bookmark File | Attention Webmasters | About the list
http://www.pcmag.com/special/web100/internet.htm

Figure 11. PC Magazine Top 100 Web Sites

2.4 Security

One of the major concerns when providing commercial services on the Internet is providing for transaction security and communications security.

Information exchanges are secure if all the following are true:

- Messages are confidential. Your message content is private and not available to others as your messages flow through the Internet. Encryption is used to ensure that the message content is confidential and no one eavesdrops on your communications.
- The information exchange has integrity. With integrity your messages are not altered as they flow from router to router. You can be assured that the message received is the same as the message you sent. For example, financial transactions are unchanged. Encryption and digital signatures ensure integrity.
- Both sender and receiver are accountable. They both agree they took part in the exchange. For example, the receiver knows that the sender signed the contract. Digital signatures assure accountability.
- You can authenticate both parties in the exchange. Not only was the contract signed, but it was signed by the proper person. The sender knows the receiver is authentic and not someone masquerading as the receiver. Authentication ensures that someone is who he or she says he or she is.

The Internet Connection Server has two security implementations that you can choose between:

- Secure Sockets Layer (SSL)
- Secure HyperText Transport Protocol (SHTTP)

2.4.1 Secure Sockets Layer (SSL)

SSL is a security protocol that was developed by Netscape Communications Corporation, along with RSA Data Security, Inc. The primary goal of the SSL protocol is to provide a private channel between communicating applications which ensures privacy of data, authentication of the partners and integrity. SSL provides an alternative to the standard TCP/IP socket API which has security implemented within it. Hence, in theory it is possible to run any TCP/IP application in a secure way without changing it. In practice, SSL is so far only implemented for HTTP connections.

In fact the protocol is composed of two layers:

- At the lowest layer is the SSL Record protocol. It is used for data encapsulation.
- On the top of this layer is the SSL Handshake protocol used for initial authentication and transfer of encryption keys.

The SSL protocol addresses the following security issues:

- Privacy. After the symmetric key is established in the initial handshake, the messages are encrypted using this key.
- Integrity. Messages contain a message authentication code ensuring the message integrity.
- Authentication. During the handshake, the client authenticates the server using an asymmetric or public key.

SSL requires each message to be encrypted and decrypted and therefore has a high performance and resource impact. In addition, since only the server is authenticated, SSL is not suitable for applications, such as electronic banking, which require that the server authenticate their clients.

The great thing about SSL is that it is easy to implement in a WWW server environment. As soon as you have a key pair and a signed certificate (for authentication purpose), you can begin serving SSL-protected documents to SSL browsers. You have to set up SSL security in your WWW server only once. Neither your HTML pages nor your server configuration file require additional statements.

2.4.2 Secure HyperText Transport Protocol (S-HTTP)

S-HTTP is a security-enhanced version of HTTP developed by Enterprise Integration Technologies (EIT). It uses a modified version of HTTP clients and servers to allow negotiation of privacy, authentication and integrity characteristics. Secure HTTP provides transaction security at a document or even a field level. Fields, for example, with an account code or electronic signature, or entire documents may be marked as private and/or signed by the sender.

To use S-HTTP, the document code or server administrator specifies the security options for each document anchor or hypertext link that they want to secure. These cryptographic options, called CRYPOPTS, allow them to specify parameters for communication such as:

- Should the document be encrypted?
- Should the request be signed, encrypted or both?
- What about the reply?
- What cryptographic algorithm should be used?
- What certificate should be used?

CRYPTOPTS can also be specified for directories so that all the files in the directory inherit the parameters for the directory.

When a user requests a secured document, the clients send an HTTP request with its own CRYPTOPT enclosed. The negotiation is initiated. The server knows that it is to be handled by S-HTTP because the URL starts with *shttp:* instead of *http:*. The server responds with its side of the negotiation.

In this negotiation phase, the client and the server exchange messages detailing what CRYPOPT features they accept. If the server is finally satisfied that the client is authorized, it meets the request; otherwise the request fails.
The negotiation is such that integrity, privacy and authentication of both parties are or can be carried out in almost any combination depending on the cryptographic options.

Unlike SSL, S-HTTP requires some HTML changes to invoke it. This implies application involvement. Therefore S-HTTP implementation is not as straightforward as SSL implementation. However, the fact that the granularity of security is at the documentation level increases flexibility and improves performance.

One major advantage of S-HTTP is its ability to perform client authentication. However, the fact that this requires the client to have a public-key certificate limits the degree to which it may be applied.

2.5 Firewalls

When you connect your network to the Internet, you give your internal users the possibility to access this tremendous world but you also offer the opportunity to external users to reach your private network. Potentially, anyone can connect to anyone, that is, any client can communicate with any server. In this way you can expose your network to *attacks*.

This any-to-any connectivity can give you many security problems. You will need to protect your own private data and also protect access to the machines inside your private network against abusive use from outside.

The first step to achieving this is to limit the number of points at which your network is connected to the Internet. The best configuration has a single point of connection as shown in Figure 12 on page 39. If you have a unique path, you gain a lot of control over which traffic to allow into and out of the Internet. We call the gateway that gives you this control a *firewall*.

Firewalls tend to be seen as a protection between the Internet and a private network. But generally speaking a firewall should be considered as a means to divide the world into two or more networks: one or more secure networks and one or more nonsecure networks.

Imagine a company where all the departments are connected to the internal network, including sales, accounts, development and human resources departments. The administrator would like to be able to restrict access from the development department machines to the human resources department machines and from the sales department to the development department.

In the following discussion, for ease of comprehension, we equate a nonsecure network to the Internet or public network and a secure network to a private IP network.

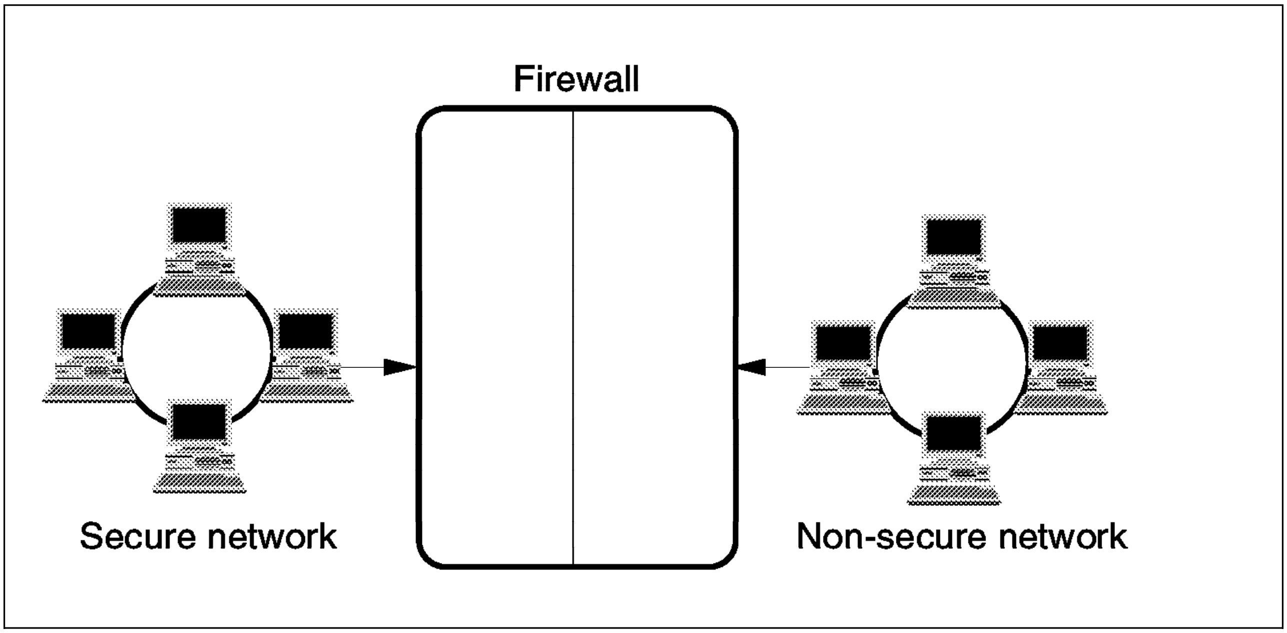


Figure 12. A Single Internet Connection

Firewalls differ in their implementation and in the degree of protection that they offer. The most common types of firewalls are:

- Screening Filter: This uses a router to connect the private network to the Internet. The screening filter looks at each IP packet flowing through it, controlling access to machines and/or ports in the private network and possibly limiting access from the private network to the Internet. Screening filters operate at the Internet Protocol (IP) layer and cannot control access at the application layer.
- Bastion: This is a machine placed between the private network and the Internet which breaks the connection between the two. The bastion relays messages to or from authorized users and denies access to all others. Bastions can control access at the user or application layer, but can be costly if many users are supported.
- Dual-Homed Gateway: This combines a screening filter and a bastion into either a single machine or a series of machines. Combining a bastion and a screening filter in a single machine protects the private network from general access while making some bastion applications accessible.

The IBM Internet Connection Secured Network Gateway (SNG) supports all of the implementations above. It includes also many other technologies to help in implementing security in an Internet environment.

2.5.1 Proxy Servers

Proxy servers are implementations of bastions. They are used to control access to or from the private network relaying only acceptable communications from known users.

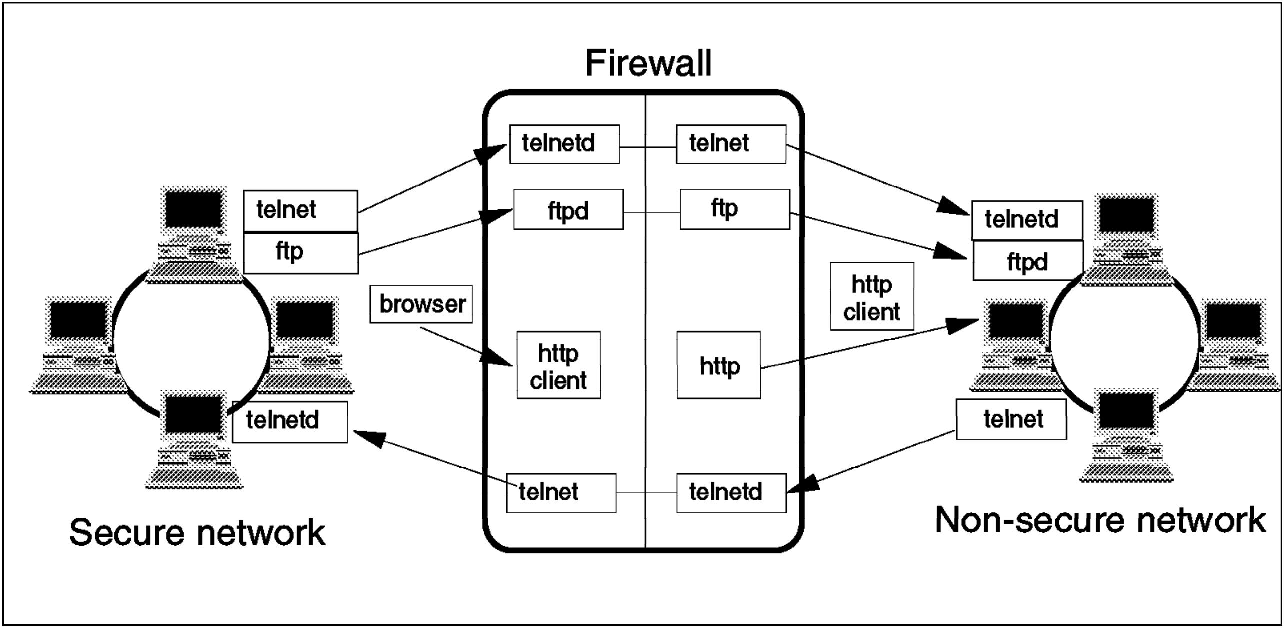


Figure 13. Firewall with Proxy Servers

Users in the private network can access an application, such as FTP, in the proxy server using their usual utilities (clients). Users authenticate themselves to the proxy server and can then access the application on the desired machine in the public network. Proxy servers can also be used from the public network to access applications in the private network, but this exposes login names and passwords to attackers in the public network. The proxy services supplied with SNG include Telnet and FTP.

The Internet Connection servers can act as proxy servers. The internal users ask the server running on the firewall to retrieve documents for them from external servers.

2.5.2 SOCKS Servers

SOCKS servers are like proxy servers without the requirement for double connections. With SOCKS, users can benefit from secure communications without needing to be aware that it is happening.

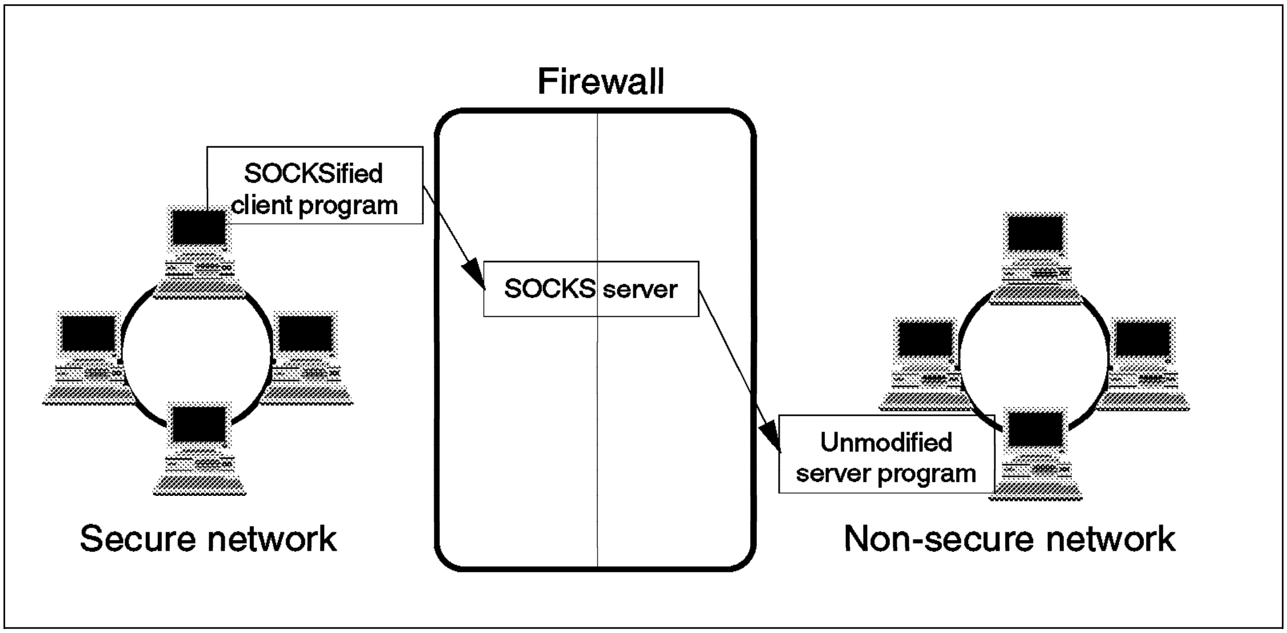


Figure 14. Firewall with SOCKS Server

Users have to use new versions of the client called SOCKSified clients. The SOCKSified client code directs its requests to the SOCKS port on the firewall. Sessions are broken at the firewall, as they are with proxy servers. With SOCKS, however, the connection to the destination application is created automatically once the user is validated.

Both the client and the SOCKS server need to have SOCKS code. The SOCKS server acts as an application-level router between the client and the real application server. SOCKS is for outbound sessions only. It is simpler for the private network user, but does not have secure password delivery so it is not intended for sessions between public network users and private network applications.

SNG includes SOCKSified Telnet and FTP client applications for AIX.

The majority of browsers are SOCKsified and you can get SOCKSified TCP/IP stacks for most platforms. For additional information, refer to:

- `http://www.raleigh.ibm.com/sng/sng-socks.html`
- `http://www.socks.nec.com`

2.6 Gateways

Web browsers can directly access many Internet information services, but some complex data formats and commercial applications often need some help. For instance, application programs using CGI are needed in addition to the basic Web server to read relational databases, dynamically build Web pages containing the database records, and transmit them to the Web browser.

Web gateways allow Web clients to access your database, middleware, or applications through the Web server without your having to make any changes to the existing applications.

IBM has provided a number of Web gateways and in the following section we introduce these.

2.6.1 DB2 WWW Connection

DB2 WWW Connection delivers open Internet access to DB2 data, a simple yet powerful development environment, and the capability of fitting into both a two-tier and three-tier Internet-to-data client/server environment. In a two-tier model (that is a local Web server with one or more Web clients), DB2 WWW Connection lets you build an application that can access DB2 data on your server.

In a three-tier model (that is, a local Web server with one or more Web clients plus access to remote servers), your new application can access DB2 data on the server and data on remote servers connected to the server with Distributed Database Connection Service (DDCS), Client Application Enabler (CAE), or DataJoiner. If the three-tier environment includes DataJoiner, the DB2 WWW Connection application can also access ORACLE, Sybase, Microsoft SQL Server, and other relational and nonrelational data sources.

Web applications for DB2 built with DB2 WWW Connection allow any Web user with an HTML-standard Web browser to access your DB2 data without your having to make any changes to the existing data structure. Using HTML and SQL, you can build applications that query your DB2 data with standard SQL statements.

The key feature of DB2 WWW Connection Version 1 is a CGI run-time engine, which processes the input from HTML forms on the Web and sends SQL commands to a DB2 system specified in a DB2 WWW Connection application. This application consists of a macro file containing HTML input and report form definitions, SQL commands, and variable definitions. DB2 WWW Connection uses the Web page paradigm for application development, which means that the application user on the Web sees only a familiar Web page form that may prompt for user input and then kick off the application transaction, returning the DB2 query results in another familiar looking Web page report.

DB2 WWW Connection applications use native HTML and SQL, exploiting the expressive power of those languages without proprietary limitations. Both HTML input and report forms can support various designs. The run-time engine supports SQL commands for SELECT, INSERT, UPDATE, and DELETE. DB2 WWW Connection applications can use data linking, the capability to use data returned by an SQL query as input to one or more subsequent SQL commands that can drill down further into DB2 data.

Note: For more information about DB2 WWW Connection, see http://www.software.ibm.com/data/db2/db2wgafs.html.

2.6.2 CICS Internet Gateway

The CICS Internet Gateway provides automatic translation between CICS 3270 data streams and HTML. It uses the CICS External Presentation Interface (EPI) and a standard mechanism known as the CGI.

One of the techniques it uses is intercepting the 3270 data stream which is normally sent to end users′ screens. The screen layout is evaluated including variable data fields and static text. By performing efficient translation between

3270 data and HTML, the user interface to your CICS application becomes platform-independent.

In addition, the application can be made globally accessible with a more visually appealing user interface. It means that your CICS applications can have a much improved user interface, using the Web browser as an alternative to the 3270 interface. No CICS-specific software is needed on the end user′s system.

Note: For more information about CICS Internet Gateway, see http://www.hursley.ibm.com/cics/saints/index.html.

2.6.3 MQSeries Internet Gateway

The MQSeries Internet Gateway provides a bridge between the synchronous world of the WWW and the asynchronous world of MQSeries networks. With the gateway, a Web server can provide access to MQSeries applications from a standard Web browser.

The gateway is a conventional CGI application; when the Web server, on which the gateway is available, receives a POST request from an HTML form identifying the gateway, it runs the gateway program. The gateway then collects information from the form and uses it to create an MQSeries message to be passed to a specific queue and queue manager. The gateway then waits for a response from the MQSeries application and when received passes the data content back to the originator of the form.

Note: For more information about MQSeries Internet Gateway, see http://www.hursley.ibm.com/mqseries/aboutmqs.html.

2.6.4 IMS Internet Gateway

An IMS Internet Gateway is planned for the MVS platform. However, currently there is no further information available.

2.7 Java

Java is an important new technology in the world of the Internet. In summary, it is a simple, robust, object-oriented, platform-independent, multithreaded, dynamic general-purpose programming environment for creating applications for the Internet and intranet. Java includes the following components:

- Java

  Java is a programming language developed by Sun Microsystems, which is object-oriented, distributed, interpreted, architecture neutral, and portable. Java can be used to create downloadable program fragments, applets, that augment the functionality of a Java-capable browser such as HotJava or Netscape Navigator.

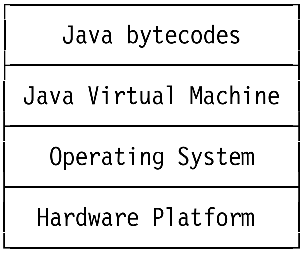
- JavaScript

  JavaScript is an HTML extension and programming language, developed by Netscape, which is a simple object-based language compatible with Java. JavaScript programs are embedded as source directly in an HTML document. They can control the behavior of forms, buttons and text elements. It is used to create dynamic behavior in elements of the Web page. In addition, it can be used to create forms whose fields have built-in error checking routines.

- Java Virtual Machine

  The Java Virtual Machine (JVM) is an abstract computer that runs compiled Java programs. JVM is virtual because it is generally implemented in software on top of a real hardware platform and operating system. In this way, it is architecture neutral and platform independent. All Java programs should be compiled to run in a JVM.

  The following diagram describes in simple terms how Java is implemented:



- HotJava

  HotJava is a Java-enabled Web browser, developed by Sun Microsystems, which lets you view Java applets.

- JavaOS

  JavaOS is a highly compact operating system, developed by JavaSoft, which is designed to run Java applications directly on microprocessors in anything from personal computers to pagers. JavaOS will run equally well on a network computer, a PDA, a printer, a game machine, or countless other devices that require a very compact OS and the ability to run Java applications.

- Java Beans

  An initiative called Java Beans is brewing a similar set of APIs that will make it easy to create Java applications from reusable components. Java Beans will be used in a wide range of applications, from simple widgets to full-scale, mission-critical applications. Many software vendors including IBM have announced plans to support it.

Note: For more information about Java, see http://ncc.hursley.ibm.com/javainfo/ or http://java.sun.com/.

2.8 Application Programming Interfaces

The Web is an interactive medium that potentially allows users to give feedback to a company about its products, use search utilities to locate information on a topic, use conversion programs to convert one value to another, and more. Although ICS does not perform these tasks, they are performed by external programs using information passed to them by the server.

There are three application programming interfaces available today for extending Web servers with external programs:

- Common Gateway Interface (CGI)
- Server-Side Include (SSI)
- Proprietary Application Programming Interface (API)

2.8.1 Common Gateway Interface (CGI)

The CGI allows the server and an external program to communicate. It is a standard interface, supported by almost all Web servers, that defines how information is exchanged between a Web server and an external program (CGI program). CGI programs can be written in any language supported by the operating system on which the server is run. The language can be a programming language, like C++, or it can be a scripting language, such as Perl or REXX. Programs written in programming languages need to be compiled, and typically run faster than uncompiled programs. On the other hand, those written in scripting languages tend to be easier to write, maintain, and debug. CGI programs are stored in the cgi-bin directory.

Note: For more information about CGI, see http://hoohoo.ncsa.uiuc.edu/cgi/.

2.8.2 Server-Side Include (SSI)

SSI allows you to create documents that provide simple information to clients on the fly. Such information can include the current date, and a file′s time stamp and size, with no programming or CGI scripts. In its more advanced usage, it can provide a powerful interface to CGI and external programs.

Note: For more information about SSI, see http://hoohoo.ncsa.uiuc.edu/docs/tutorials/includes.html.

2.8.3 Proprietary Application Programming Interface (API)

An example of a proprietary API is the IBM Internet Connection API (IBM ICAPI). ICAPI allows you to write extensions to do customized processing, such as:

- Publish customized pages based on a client′s code level
- Enhance the basic authentication or replace it with a site-specific process
- Add error handling routines to track problems or alert for serious conditions
- Detect and track information that comes in from the requesting client, such as server referrals and user agent code

The ICAPI has several features that make it a viable choice:

- Efficient

  Designed specifically for the threaded processing used by the ICS.

- Flexible

  Contains rich and compatible functions that can be used to clone other APIs, such as NSAPI, ISAPI and GoServe.

  It is platform independent and language neutral. It runs on all the ICS platforms and applications can be written in any of the programming languages supported by these platforms.

- Simple and easy to use

  Only passes simple data types by reference not value.

  Has a fixed number of parameters for each function.

  Has no return values from functions but does have a return code parameter.

  Includes bindings for the C language and provides an OS/2 GoServe compatibility module.

Does not impact allocated memory. The API does not free the user′s memory and the user does not free the API′s memory.

Note: More information about APIs is available at http://webdev.raleigh.ibm.com/design/api.htm which is in the IBM intranet.

2.8.4 How to Choose a Programming Interface

Choosing a programming interface depends on your priorities. Different programming interfaces have different characteristics. For example:

- Flexibility

  CGI and ICAPI both have great flexibility to develop applications. On the other hand, SSI has the lowest flexibility because it is a substitution mechanism, that is, SSI script is inserted into a document at commented tag locations. You can use many languages that are supported by your operating system, for programming with CGI or ICAPI.

- Development Cost

  SSI has the least development cost, but it does not have much flexibility. CGI is widely known and many tools exist. ICAPI is new, and development requires an expert programmer who knows ICAPI and details of the system. ICAPI development may require the highest costs.

- Operational Risk

  If your CGI/SSI script program abends, the Web server must not be brought down by the bug. ICAPI has safety mechanisms to protect the server against API program failure. However, if you mistakenly write the program to access the server′s private data, it is possible that this will cause a fatal error at the server.

- Running Cost

  CGI and SSI are low performance because the CPU and memory overheads are high. As a result, hardware costs are likely to be higher. If transactions using CGI/SSI increase or CGI/SSI programs become larger, you may have to upgrade your hardware.

Chapter 3. Web Browsers

In this chapter we discuss the characteristics of a typical Web browser.

3.1 The Principles of Browsing

To start with we introduce the basic protocols involved in a browser to server communication.

3.1.1 Hypertext Transfer Protocol

The Hypertext Transfer Protocol (HTTP) determines how a browser interacts with a server.

For more information regarding HTTP, refer to 2.2, "HTTP" on page 12.

An HTTP transaction is divided into four steps:

1. The browser opens a connection.
2. The browser sends a request to the server.
3. The server sends a response to the browser.
4. The connection is closed.

We now describe the four steps in detail. We have developed a basic browser and a basic server using REXX for OS/2 to display the browser′s requests and the server′s responses. Related documentation can be found in Appendix A, "Two Simple Browsers and One Simple Server" on page 299.

3.1.1.1 The Browser Opens a Connection

The browser attempts to connect to the server. If a server is listening, it establishes the connection. If not, the browser times out and displays an error message.

3.1.1.2 The Browser Sends a Request to the Server

Once the connection is established, the browser sends a request to the server, using one of the following request methods:

DELETE The DELETE method is used to delete a resource specified by its URI. Even though this method is supported by most of the recent servers, it is rarely enabled for obvious security reasons.

For example, to delete the /shapes/index.html page the browser would send:

```
DELETE /shapes/index.html HTTP/1.0
```

GET The GET method is used to retrieve a resource specified by its URI. This was the only method supported by the early versions of HTTP (HTTP 0.9), and it is still, by far, the most used one.

For example, to load the /shapes/index.html page the browser would send:

```
GET /shapes/index.html
```

To tell the browser to use HTTP 1.0 rather than a previous version of the protocol, complete the method by HTTP/1.0 and two carriage return line feeds (CRLFs) like so:

```
GET /shapes/index.html HTTP/1.0
```

Note that it is also possible to launch CGI scripts using GET, although this is usually achieved using the POST method. Clicking on a clickable image map would send the following request:

```
GET /cgi-bin/htimage/shapes/shapes.map?29,31
```

HEAD The HEAD method is used to get the header or the meta-information of a resource specified by its URI. This is very useful to determine if a resource is still accessible, or if it has been modified since the last time it was retrieved.

To get information about the /shapes/index.html page, the browser would send:

```
HEAD /shapes/index.html HTTP/1.0
```

LINK The LINK method is hardly ever used. Refer to RFC 1945 for more details.

POST The POST method is used to submit the data of a form to a CGI script.

```
POST /action HTTP/1.0
Content-length: 55

cookie=99567&mode=compact&maxHits=25&searchText=Surfing
```

PUT The PUT method is used to put data on the server. Even though this method is supported by most of the recent servers, it is rarely implemented for obvious security reasons.

UNLINK The UNLINK method is hardly ever used. Refer to RFC 1945 for more details.

3.1.1.3 Real-Life Requests

More elaborate browsers include more information in their requests. The following figures represent the requests sent by Charlotte (an IBM VM browser), the IBM WebExplorer Java Technology Demo, and the Netscape Navigator 2.02, respectively.

```
GET /shapes/index.html HTTP/1.0
Accept: text/*
Accept: text/plain
Accept: text/html
Referer: http://9.24.104.247:8888/shapes/index.html
User-Agent: Charlotte/1.2.1 VM_ESA/1.2.2 CMS/11  via proxy gateway  CERN-HTTPD/3.0 libwww/2.17
```

Figure 15. A GET Request by Charlotte

```
GET /shapes/index.html HTTP/1.0
Accept: */*; q=0.300
Accept: application/octet-stream; q=0.100
Accept: text/plain
Accept: text/html
Accept: application/pdf
Accept: audio/x-pn-realaudio
Accept: application/inf
Accept: audio/x-wav
Accept: audio/x-aiff
Accept: audio/basic
Accept: video/avs-video
Accept: video/x-msvideo
Accept: video/quicktime
Accept: video/mpeg
Accept: image/x-bitmap
Accept: image/bmp
Accept: image/tiff
Accept: image/jpeg
Accept: image/gif
Accept: application/editor
User-Agent: IBM WebExplorer DLL /v960311 Beta
```

Figure 16. A GET Request by the IBM WebExplorer Java Technology Demo

```
GET /shapes/index.html HTTP/1.0
Connection: Keep-Alive
User-Agent: Mozilla/2.02 (Win16; I)
Host: 9.24.104.247:8888
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */*
```

Figure 17. A GET Request by the Netscape Navigator 2.02

3.1.1.4 The Server Responds to the Browser

Whether the request could be honored or not, the server sends a response to the browser.

If the server uses a version of HTTP that is older than 1.0, the response only contains the requested data, or the error message in HTML format, as described in the following two figures.

```
<HTML>
<HEAD>
<TITLE>Shapes - Polygon</TITLE>
</HEAD>
<BODY BGCOLOR=FFFFFF>
<CENTER>
<H1>Shapes<BR>Polygon</H1>
<I>You have selected a polygon</I>
<HR>
<A HREF=”/shapes/index.html”>Back to previous page</A>
<HR>
</CENTER>
</BODY>
</HTML>’
```

Figure 18. The HTTP 0.9 Response to a Successful GET Request

```
<HTML>
<HEAD>
<TITLE>Error</TITLE>
</HEAD>
<BODY>
<H1>Error 404</H1>

Not found - file doesn’t exist or is read protected ·even tried multi‘

<P><HR><ADDRESS><A HREF=”http://win-nt.itso.ral.ibm.com/”>
IBM Internet Connection Secure Export Server</A></ADDRRESS>
</BODY>
</HTML>
```

Figure 19. The HTTP 0.9 Response to an Unsuccessful GET Request

If the server uses HTTP Version 1.0, then the response is composed of a header followed by the requested data, as described in the following two figures.

```
HTTP/1.0 200 Document follows
Server: IBM-Secure-Export-ICS/4.1Beta4
Date: Thursday, 06-Jun-96 21:22:16 GMT
Content-Type: text/html
Content-Length: 457
Last-Modified: Friday, 24-May-96 15:44:07 GMT

<HTML>
<HEAD>
<TITLE>Shapes</TITLE>
</HEAD>
<BODY BGCOLOR=FFFFFF>
<CENTER>
<H1>Shapes<BR>A Sample Clickable Image Map</H1>
<I>Select a shape</I>
<HR>
<A HREF=″/cgi-bin/htimage/shapes/shapes.map″>
<IMG ALT=″Please use the text links″ ISMAP SRC=″/shapes/shapes.gif″></A>
<P>
<A HREF=″/shapes/polygon.html″>Polygon</A> -
<A HREF=″/shapes/circle.html″>Circle</A> -
<A HREF=″/shapes/rectangle.html″>Rectangle</A>
<HR>
</CENTER>
</BODY>
</HTML>′
```

Figure 20. The HTTP 1.0 Response to a Successful GET Request

```
HTTP/1.0 404 Not found - file doesn′t exist or is read protected ·even tried multi‘
Server: IBM-Secure-Export-ICS/4.1Beta4
Date: Thursday, 06-Jun-96 20:48:56 GMT
Content-Type: text/html
Content-Length: 276

<HTML>
<HEAD>
<TITLE>Error</TITLE>
</HEAD>
<BODY>
<H1>Error 404</H1>

Not found - file doesn′t exist or is read protected ·even tried multi‘

<P><HR><ADDRESS><A HREF=″http://win-nt.itso.ral.ibm.com/″>
IBM Internet Connection Secure Export Server</A></ADDRRESS>
</BODY>
</HTML>
```

Figure 21. The HTTP 1.0 Response to an Unsuccessful GET Request

In the successful example, the server returned an HTTP 1.0 response, the status code was 200 (which is OK), and the reason phrase was `Document follows`.

In the unsuccessful example, the server also returned an HTTP 1.0 response, the status code was 404, and the reason phrase was Not found.

In both cases the header contained the Content-Type which indicates the nature of the data transmitted: text/html. This is described in more detail in the next section.

3.1.1.5 The Browser or the Server Closes the Connection

The connection is closed when the server has transmitted its response to the browser.

3.1.2 The Multipurpose Internet Mail Extensions (MIME)

In all the responses stated in the previous section, the Content-Type field of the header was text/html. What would happen if we requested another type of resource, such as a graphic, for example? To find out, we sent the following request to our server:

```
GET /shapes/shapes.gif HTTP/1.0
```

The server's response then was:

```
HTTP/1.0 200 Document follows
Server: IBM-Secure-Export-ICS/4.1Beta4
Date: Friday, 07-Jun-96 15:38:11 GMT
Content-Type: image/gif
Content-Length: 835
Last-Modified: Friday, 24-May-96 15:44:09 GMT
X-httpd-warning: Your browser didn't send the Accept header line for this

GIF89a...
(data flow truncated)
```

Figure 22. A Response to GET Requesting an Image

As this simple example shows, HTTP can transfer much more than the hypertext its name mentions, such as graphics. After all, the World Wide Web has become popular mainly because of its ability to carry sights and sounds (sons et lumieres).

This is made possible by the Multipurpose Internet Mail Extensions which is defined by RFCs 1521 and 1522.

When a browser exchanges data with HTTP 1.0, the request (PUT) or response contains a MIME header, defining the data content and type. In the examples above we encountered the following:

```
Content-Type: type/subtype
Content-Type: text/html
Content-Type: image/gif
```

Upon reception of this header, the receiving end (generally the browser) can decide how to process the data flow (should it display it with an internal or external viewer, launch an application, etc.). This provides a high level of flexibility to browsers.

3.1.3 HTML

The Hypertext Markup Language (HTML) is used to write the pages that appear on the World Wide Web. Please refer to 2.3, “HTML” on page 14 for more details.

3.2 The Features of a Good Web Browser

There are many browsers available in the marketplace. In this section we attempt to pinpoint the characteristics of a good browser.

3.2.1 Protocols

One characteristic of a good browser is that it should interface to most Internet protocols. It should allow you to display pages, download files, send e-mail, read from and append to newsgroups, etc.

3.2.1.1 HTTP

If the software does not support HTTP then it is not a browser.

3.2.1.2 FTP

The File Transfer Protocol (FTP) is certainly the most popular way to exchange files of any format on the Internet. It is very convenient to have an FTP client integrated in the browser. This allows you to get files from an FTP browser simply by linking to them, as if they were WWW pages, rather than having to launch a specific FTP client.

Note that most browsers do not allow you to put files on an FTP server. It is therefore useful to keep an FTP client handy.

3.2.1.3 NNTP

The Internet also carries newsgroups. Newsgroups are a convenient way for a group of people to share their experience or their data. You might consider setting up news servers on your intranet if groups of people in your staff should share the same flow of information.

Some browsers now support the newsgroup scheme name. This allows you to link to the newsgroup simply by specifying news:newsgroup URL. Some browsers use HTML pages that guide you through the news groups at the server. However in most cases HTML pages are not appropriate for the mass of information that some servers carry. We therefore preferred the browsers that dedicate a specific window for reading news, such as Netscape Navigator.

Where possible, opt for a MIME-compliant news client, in order to be able to process documents, images, etc. imbedded in appends to the newsgroups.

3.2.1.4 SMTP

Most browsers now support the mailto scheme name. This means that linking to the mailto:name@host URL allows you to send an e-mail to name at host. To send this e-mail, the browser must communicate with a mail server, and this means the server must support the Simple Mail Transfer Protocol (SMTP).

Furthermore, some browsers allow you to receive and read e-mail from a POP server. This implies that the browser uses the Post Office Protocol (POP).

In any case MIME-compliant mail clients are preferred so that you are able to process documents imbedded in your mail.

3.2.1.5 Other Protocols

Some browsers also implement other protocols, such as Finger or the ICMP-based PING and TRACEROUTE. Although it is convenient to access all possible clients from a single interface, it is impossible to gather all IP-based protocols into one browser; they are too numerous.

3.2.2 HTML

The Hypertext Markup Language (HTML) is used to write the World Wide Web′s hypertext pages. A browser should be able to parse the highest possible version of this language. The current version is 3.0 at the time of writing.

The early versions of HTML were rather basic and included only a few tags. The need to create more sophisticated pages has caused HTML to change quickly from Version 1.0 to 3.0 in recent times. But this has not been fast enough for some browser developers who like to implement their own tags and then, in some cases, they become de facto standards. Frames were introduced in this way. These proprietary tags sometimes cause strange results if the browser does not support them.

Therefore, although we advise you to use the most complete browsers, we also recommend that you develop pages using standard HTML.

3.2.3 Java

Sun describes Java as a simple, object-oriented, distributed, interpreted, robust, secure, architecture neutral, portable, high-performance, multithreaded and dynamic language.

All these attributes are important, but the most important ones are architecture neutral and portable. They mean that a Java applet will run on any browser on any platform if this browser and platform support Java (not only the Netscape Navigator or the Microsoft Internet Explorer). Note that Java has been or will be ported to all major IBM platforms.

3.2.4 JavaScript

JavaScript is a programming language developed by Netscape. Here is a description taken from the Netscape Web site:

> JavaScript (is) a programmable API that allows cross-platform scripting of events, objects, and actions. It allows the page designer to access events such as startups, exits, and user mouse clicks. Based on the Java language, JavaScript extends the programming capabilities of Netscape Navigator to a wide range of authors and is easy enough for anyone who can compose HTML. Use JavaScript to glue HTML, inline plug-ins, and Java applets to each other.

Most browsers (other than the Netscape Navigator) do not support JavaScript.

3.2.5 ActiveX

The latest Microsoft Internet Explorer also provides, with Aptivex, controls and scripting that allow you to create animated or interactive pages.

The major problem with these languages is that they are browser dependent, if not proprietary. A person not using the right hardware, the right operating system and the right browser (that already includes a lot of people) will not be able to properly display your interactive multimedia pages. All your efforts will have been in vain.

Therefore think simple, think standard, love all, serve all.

3.2.6 Security

Security is a very important issue on an intranet or on the Internet. When it comes to browsers the security on the net covers two aspects:

- TCP/IP security, that protects your resources (hardware such as computers, printers, etc. and software such as applications, files, etc.) from misuse or destruction
- HTTP security, that ensures the confidentiality of the data exchanged between browsers and servers.

3.2.7 Network Security

Another important characteristic of a good browser is its network security features.

3.2.7.1 Proxies

One way of implementing TCP/IP security is to set up proxy servers on firewalls. For more information about the firewall and proxy concepts, please refer to 2.5, “Firewalls” on page 38 and 2.5.1, “Proxy Servers” on page 40. In addition there is a chapter dedicated to proxies in the *Internet Connection Server Up and Running* manual.

HTTP supports proxy functions and most browsers can make use of this feature. Furthermore, browsers have become more and more powerful, and they now integrate more protocols than simply HTTP. For instance, it is possible to connect to an FTP server from the browser. Since some of these protocols also support proxying, such as FTP, Gopher and WAIS, a good browser should allow you to configure these proxies.

3.2.7.2 SOCKS Servers

Another way of implementing TCP/IP security is to install SOCKS servers on firewalls. For more information about SOCKS servers, please refer to 2.5.2, “SOCKS Servers” on page 40. A good Web browser should allow you to configure a SOCKS server.

3.2.7.3 S-HTTP

A good browser should support S-HTTP. For more information, refer to 2.4.2, “Secure HyperText Transport Protocol (S-HTTP)” on page 37.

3.2.7.4 SSL

A good browser should support SSL. For more information, refer to 2.4.1, “Secure Sockets Layer (SSL)” on page 37.

3.3 Available Browsers

There are many browsers available for use on the World Wide Web. Here we list the ones that are most widely used in the IBM Internet world.

3.3.1 IBM WebExplorer

IBM WebExplorer is IBM′s browser that is shipped with the OS/2 platform. There are three versions: Base, Secure and Java. Although the Java version would complement the Secure version, no such combination was available at the time of writing.

During our project, no announcement had been made regarding the availability of Netscape Navigator on the OS/2 Warp platform. However, the situation has changed since that time: for more details refer to 3.3.2, “Netscape Navigator.”

3.3.1.1 IBM WebExplorer Base Version

IBM WebExplorer Base version supports HTTP 1.0 and FTP. It also supports the mailto and news schemes.

3.3.1.2 IBM WebExplorer Secure Version

IBM WebExplore Secure Version adds S-HTTP and SSL support to the features of the base product. Unlike the other WebExplorers, this one is chargeable.

3.3.1.3 IBM WebExplorer Java Technology

IBM WebExplorer Java Technology Demo offers Java support along with the base product′s features. It only works in combination with IBM′s Java Development Kit for OS/2.

3.3.2 Netscape Navigator

Netscape Navigator is available on most graphical platforms. The actual market share of this product (about 75 percent) promotes every feature of this browser as a must-have for other browsers, if not as a de facto standard. Please link to the Netscape home page for more details about the Navigator.

IBM and Netscape have recently made available a beta test version of Netscape Navigator that is ported to the OS/2 Warp platform. Although WebExplorer is part of the recently available OS/2 Warp V4 it should be understood that Netscape Navigator is functionally superior in most areas. Netscape Navigator will therefore be the browser of choice in most cases.

3.3.3 Other Browsers and Ways to Surf the Net

As we have demonstrated at the beginning of this chapter, rolling your own browser requires little time and minimal skills. An Internet-oriented language such as Java or REXX which offers a full grown parser can also help in this process.

Thus, you can easily write a browser to load and display a specific resource, a meteorological map for example, or create a robot-browser to automatically send request methods, or a tool to download all the resources pointed to by the HTML anchors of a page.

You can also integrate basic HTTP features into your existing applications, as most software vendors have done. For example, it is already possible to load Web pages directly as a document into some text processors.

Chapter 4. Internet Connection Servers - Installation and Configuration

The purpose of this chapter is to provide you with guidelines for installing an Internet Connection server. This chapter covers:

- An overview of the Internet Connection server features.
- A step-by-step description of how you should install and configure your Internet Connection server.
- A discussion on how you should operate your Internet Connection server. This is comprised of sections on security, backup and logs.

As you will read in 4.1, “Standard Functions,” the Internet Connection server is available for many IBM and non-IBM platforms. Naturally, the installation process varies from one platform to another. Consequently this chapter contains separate sections for each platform providing specific information related to the installation and administration procedures.

Note that, with the exceptions of the OS/390 and OS/400 platforms, the Internet Connection server featured is Version 4.1.

4.1 Standard Functions

In this section we introduce the standard functions of the Internet Connection servers.

- **Home Page Repository**

Acts as a repository for resources (home pages) created with Hypertext Markup Language (HTML).

- **HTTP Support**

Honors requests from Web browsers (clients) using Hypertext Transfer Protocol (HTTP) to transfer the document.

- **Multimedia Serving**

Simplifies explanations or livens up Web pages, HTML documents often including in-line images, or links to audio or video clips.

- **Logging**

For each request, the server logs pertinent information such as the date, time, IP number of the client machine, and the text of the request. There are various types of logging which we discuss in 4.6.8, “Logging Procedures” on page 148.

- **Proxy Support**

Retrieves documents from other servers on behalf of clients. Optionally, these can be stored locally in a cache at the server.

Acts as an agent for clients to access remote servers not directly accessible by the browser due to security access restrictions. In some instances this process involves a referral to a firewall for access to the Internet. For more information about firewalls and security in general, refer to 2.4, “Security” on page 36. The proxy server supports HTTP, FTP, Gopher, WAIS and HTTP-S requests.

- **CGI Support**

Provides access to applications using the Common Gateway Interface (CGI). CGI is an application programming interface between the Internet Connection server and another application such as a database. Sample CGI scripts are provided that will negotiate the movement of data between the Internet Connection server and the outside application.

- **Integration of DB2 Gateway**

Allows you to access DB2 data using standard Web browsers. Using this feature, you can develop Web applications without a proprietary language, data encapsulation or database reformatting. Application programmers can combine dynamic SQL with HTML forms to access DB2 databases. For more information about the DB2 Gateway use your Web browser to go to the following URL:

```
http://www.software.ibm.com/data/db2/db2wgafs.html
```

- **Integration of CICS Gateway**

Allows you to develop and run transaction processing applications on the Web and to access new and many existing CICS applications using standard Web browsers. Typically, no changes are required for CICS applications using this feature, nor is any application code required to use the CICS Gateway. In most cases, you do not need to recompile CICS applications or CICS Basic Mapping Support (BMS) maps. For more information about the CICS Gateway use your Web browser to go to the following URL:

```
http://www.hursley.ibm.com/cics/saints/index.html
```

- **Internet Security**

IBM′s secure servers feature the industry′s two most popular security protocols:

- Secure Sockets Layer (SSL)
- Secure Hypertext Transfer Protocol (S-HTTP)

IBM′s secure servers not only support the industry standards for x.509 certificates used in the SSL and SHTTP protocols, but also provide the facilities to issue certificates for use in a company′s enterprise.

4.2 New Functions

Internet Connection Secure Server V4.1 includes:

- **IBM ICAPI API**

Allows you to extend the server′s base functions. Using this API, you can write extensions to perform site-specific processes, such as publishing customized pages based on a client′s code level, enhancing the basic authentication, and adding error routines to track problems or alert you about various conditions.

- **Netscape API**

Allows you to use the Netscape API on the Internet Connection Server. Using the server′s NSAPI compatibility module, you can port existing NSAPI programs without losing any functions.

- **Server-Side Includes**

Allows you to dynamically insert information into an HTML document that the server sends to a client. The current date, the size of a file and the last change of a file are examples of dynamic information that can be sent to the client.

- **Error Message Customization**

Allows you to customize the messages that your server sends back to the client when error conditions occur. By creating your own HTML message files, you can provide additional information, suggest solutions, and name a contact person, if desired.

- **Multiple IP Address Support**

Allows you to maintain multiple Web sites on a single Internet Connection server. By running the server on a machine with multiple network connections, you can serve different files based on the host name entered by the client making the request. Clients can reach each of your home pages with a simple URL that does not specify a file name or port number.

- **Enhanced Logging and Reporting**

Allows you to control which requests your server logs. The reporting function lets you create and view reports based on the information in your access logs.

4.3 Export Version Restrictions

Two versions of these secure products exist; one for the U.S. and Canada and one for all other countries. The version for U.S. and Canada cannot be announced or made available outside the U.S. or Canada since they contain triple DES cryptographic algorithms for bulk encryption as well as 128-bit RC2 and 128-bit RC4 encryption. These products can not be exported outside the U.S. and Canada to any entities. The version of the products announced outside of the U.S. and Canada use 40-bit RC2 and 40-bit RC4 encryption for data privacy.

A session between an export version and a non-export version will be negotiated between the servers at the lowest level of encryption.

For more information about RC2 and RC4 go to following URL:

```
http://www.rsa.com
```

4.4 Do I Need a Secure Server?

In this section we discuss the need for security in Internet and intranet communications.

At a simple level, it would appear that security is most vital in the case of a corporate intranet with a connection to the Internet. Conversely, we may not need any security within a corporate intranet. In practice, every network we set up will need a thorough security audit prior to *going live*.

Our discussions should help you decide whether you need to install the Internet Connection secure server or whether the nonsecure version will suffice. For an introduction to security concepts please refer to 2.4, “Security” on page 36.

4.4.1 Security Needs

Depending on your involvement with the Internet, you will have different security objectives. If you are an end user that is using an electronic commerce facility on the Internet, you would like to be sure about things such as:

- To whom am I talking?
- Can I trust them?

- What happens with my credit card information?
- Can they use my payment twice?

However, if you are the provider of that electronic commerce facility you will have different security objectives. For example, you would like to know the following:

- Who has access to my systems?
- Can they prove their identity?
- What else on my system can they access?
- How can I bill them?
- Who guarantees thier payment?

Security objectives fall into one or more of the following five categories:

- Access Control

  Assurance that the person or computer at the other end of the session is permitted to do what he or she asks for.

- Authentication

  Assurance that the resource (human or machine) at the other end of the session really is what it claims to be.

- Integrity

  Assurance that the information that arrives is the same as what was sent.

- Accountability

  Assurance that any transaction that takes place can subsequently be proved to have taken place. Both the sender and the receiver agree that the exchange took place (also called non-repudiation).

- Privacy

  Assurance that sensitive information is not visible to an eavesdropper. This is usually achieved using encryption.

4.4.2 What Mechanisms Provide Security?

You can protect your environment by using firewalls. Firewalls can control:

- Which computers on either side of the firewall can be accessed
- In what way these computers can be accessed
- Who can access these computers
- How the end user identifies and authenticates himself

You should consider a firewall when you connect a company intranet to the Internet. It allows you to protect all of the machines on your intranet from any external attacks. You can decide what types of traffic will reach which applications on which machines. You can also implement various authorization schemes for getting through the firewall. Unwanted traffic can be filtered out and you can get audit trails to help in detecting attacks.

In addition, security mechanisms are provided on the server platform where the service is implemented. Platform security can provide, for example, the following security facilities:

- End-user identification and authentication
- Access control for applications and data
- Audit records

The server and the client should agree on the protocols that are used to connect each other and provide enhanced security on the Internet. Among these protocols you will find:

- Secure Sockets Layer (SSL)
- Secure Hypertext Transfer Protocol (S-HTTP)
- Secure Electronic Transactions (SET)

These protocols provide the following for messages that are sent over the Internet:

- Authentication
- Integrity
- Accountability
- Privacy

Internet Connection Secure Server Family supports SSL and S-HTTP. SET at the time of writing is still under development. SET support is planned for the Internet Connection server and associated products like Net.Commerce.

4.4.3 No Confidential Data on Your Web Server

If your Web server is providing company information that should be available for any employee through the company intranet, the server may not need any security functions. You can therefore choose either the standard (nonsecure) version or the secure version of ICS.

4.4.3.1 Confidential Data Included on Your Web Server

There may also be a requirement to restrict some of the information on the Web server to specific users in the organization. Although you might have the impression that you don′t need any strict security practices in place if you are using an internal network, you should realize that most security breaches today are carried out by people inside a company taking a chance to access information they should not access. Therefore, if there is a need to store confidential information on the Web server even within an intranet, some important security aspects must be considered. In this case, you would be wise to consider using the secure version of ICS.

4.5 Installing an Internet Connection Server

In this section we describe how to install your Internet Connection server. We take you from the moment you select the platform on which your server will be running, to the moment you launch it safely for your network.

We have broken down the installation process into the elementary steps that we have identified while installing Internet Connection servers on selected platforms for which this product is available.

These steps are:

- **Prepare Your Platform**

  First of all you should make sure that your platform is ready for the installation of an Internet Connection server. Do you have the appropriate hardware and the proper release of the operating system and TCP/IP stack? Have you applied the required level of maintenance?

- **Prepare Your Network**

  Once your platform is ready you should prepare your IP network.

  While you install and configure your server, the information it advertises may be incomplete or even erroneous. Furthermore, the security settings may not be fully or correctly implemented. You will therefore want to limit the number of hosts that can access your server.

  Your server is a sensible machine; you expect it to deliver the relevant information continuously. You should therefore disable all the unnecessary applications that could stop the server or corrupt its contents, as an unstable release of a browser, or FTP or TELNET servers, respectively.

  You will need to install a browser to read the HTML documents shipped with your Internet Connection server, access the remote configuration and administration forms, and to verify the behavior of your server.

- **Get Your Copy of the Internet Connection Server**

  How you get a copy of the Internet Connection server depends greatly on the platform on which you install it. The Internet Connection server is now shipped with the latest releases of some IBM operating systems, such as AIX, OS/390 and OS/400. Furthermore, you may download the Internet Connection server for the other IBM operating systems as well as for non-IBM operating systems from the Internet Connection Family Web site at http://www.ics.raleigh.ibm.com. You may also purchase an Internet Connection server from IBM Direct.

- **Read the Documentation**

  This step is essential! It is possible to download and install an Internet Connection server without reading the installation instructions. This is especially true for OS/2 Warp and Windows NT. But this is definitely not advisable! A Web server exposes the machine and the network on which it is running. You should therefore make sure you know what you are doing. Your first source of information is the Internet Connection Family Web site at http://www.ics.raleigh.ibm.com. Other documents are shipped with the Internet Connection server.

- **Plan for the Installation**

  During this step you will decide what resources you attribute to the server, and which of the installation options you will select.

- **Install Your Server**

  You are now ready to install your Internet Connection server.

As you will discover later in this chapter, Internet Connection servers are very similar across platforms during operation. Installation, however, is very much dependent on the platform. With this in mind, each of the following subsections describes how to install an Internet Connection server on one particular platform.

4.5.1 AIX

In this section we tell you how to prepare and install an Internet Connection Server for AIX.

The following topics are covered:

- Hardware requirements
- Software requirements
- Getting your copy of the Internet Connection Server for AIX
- Securing the AIX server
- Preparing for the installation
- Installing the server

4.5.1.1 Hardware Requirements

- A RS/6000 or IBM Power Series Family with AIX Version 4.1.3 or later.
- Approximately 7 MB of free disk space to install the server. This includes the base file sets, security file sets and message catalog. An additional 4 MB of free disk space is required to install the DB2 and CICS Gateway features.
- A mouse, trackball, trackpoint or pen. Although all functions can be performed with the keyboard, a pointing device is recommended.
- Any communication hardware that is supported by the TCP/IP protocol stack to make network connections.
- 32 MB memory should be sufficient for most servers. However, actual memory requirements will depend on what other applications are running.

4.5.1.2 Software Requirements

- AIX Version 4.1.3 or later

  Note: It is recommended that AIX software be upgraded to Version 4.1.4 or higher to take advantage of the latest enhancements and fixes to the operating system, particularly those that address TCP/IP and security.

- If your server handles a large number of incoming connections, the fix for APAR IX52752 for AIX Version 4.1.3 or later should be applied. This fix increases from 10 to 100 the listen() backlog max limit that is set by AIX.

For capacity and security reasons, it is not recommended to have other applications running on the server. You should install only a minimal AIX, that is the basic file sets that are installed automatically plus the TCP/IP file set (both client and server parts).

4.5.1.3 Getting Your Copy of the Internet Connection Server for AIX

They are several ways to get a copy of the Internet Connection Server for AIX:

- You are a lucky owner of an AIX 4.2 server license. Then you have the product included in the Bonus Pack provided with your license.
- Purchase the product. The software is delivered on a CD/ROM or on diskettes and it comes with the *Up and Running!* documentation. Both secure and nonsecure versions can be purchased. Here are the references for the AIX products:

```
5765-638 Internet Connection Server Version 4.1 for AIX
5697-A74 Internet Connection Secure Server Version 4.1 for AIX
```

- Download the code. You can download the last released version of the Internet Connection Server for AIX, as well as beta versions of the product (both secure and nonsecure versions). You should be aware that the secure beta product is a 60-days demonstration product.

If you already have the *Internet Connection Server* product, you should jump directly to 4.5.1.4, "Securing the AIX Server" on page 67.

Downloading the Code: Prior to download, decide in which directory you want to store the server code. The downloaded file requires approximately 7 MB of temporary storage. The packages that you will extract from this file require approximately 14 MB.

You can get the server code via anonymous FTP on:

```
service2.boulder.ibm.com
```

or at the Internet Connection Family Web site:

```
http://www.ics.raleigh.ibm.com/ics/icfgive.htm
```

The latter is the recommended alternative to download the code since the WWW server helps you in determining which file should be downloaded with regard to criteria such as your National Language Support.

Note: At the time of writing, only the nonsecure servers are available on Boulder FTP server. The downloaded code is a compressed tar file. In order to extract the file sets, you must be in the directory into which you downloaded the server code. At the prompt, enter:

```
zcat <compressed tar filename> | tar -xpvf -
```

The following packages will be extracted :

- cig.rte (CICS Internet Gateway)
- db2www.doc.*xx_xx* (DB2 WWW Product Documentation)
- db2www.www (DB2 World Wide Web Connection)
- internet_server.Bnd (Internet Connection Server Bundle)
- internet_server.base (Internet Connection Server)
- internet_server.security.common (Internet Connection Server Common Secure Files - *secure version only*)
- internet_server.security.export (Internet Connection Server Export Secure Files - *secure version only*)

Printing the Documentation: On the Web page:

```
http://www.ics.raleigh.ibm.com/ics/icftoops.htm
```

you will find references to the Postscript files containing the U.S. English versions of the documentation.

If you do not have a Postscript printer, you won′t be alone. When the server installation is over, you can browse the documentation with your preferred HTML browser; it is online.

4.5.1.4 Securing the AIX Server

Detailing all the mechanisms that you should set up in order to secure your AIX Server is beyond the scope of this redbook. For detailed directives, you can refer to *Safe Surfing: How to Build a Secure World Wide Web Connection* SG24-4564. You should at least apply the following techniques:

Setting Up a User ID for the Web Server: Although the standard recommendation for a Web server is to run it under the *nobody* user and group IDs, we suggest setting up dedicated user and group IDs. The user ID should not be used for login, or for FTP. You can set up the user ID as follows:

```
mkgroup -A www
mkuser pgrp=www groups=www sugroups=system home=/var/nowhere \
gecos=′The Web Server′ login=false rlogin=false www
```

Add the WWW user ID to the file /etc/ftpusers. (Create the file if it does not already exist.)

Removing Unnecessary Services: Many TCP/IP services are enabled by default. You should disable most of them. Some are started via inetd and configured through /etc/inetd.conf; others are started through /etc/rc.tcpip or other commands that are triggered by the init process (see /etc/inittab). In order to disable the undesired services, we suggest you do the following:

- Browse /etc/inetd.conf and disable services such as shell login, exec, fttp, etc. As a minimal configuration, you should keep internal services, telnet and FTP for remote maintenance purposes. If you wish to log FTP activity, which is recommended, the ftpd daemon should be invoked with the -l option as follows:

  ```
  ftp     stream  tcp     nowait  root    /usr/sbin/ftpd         ftpd -l
  ```

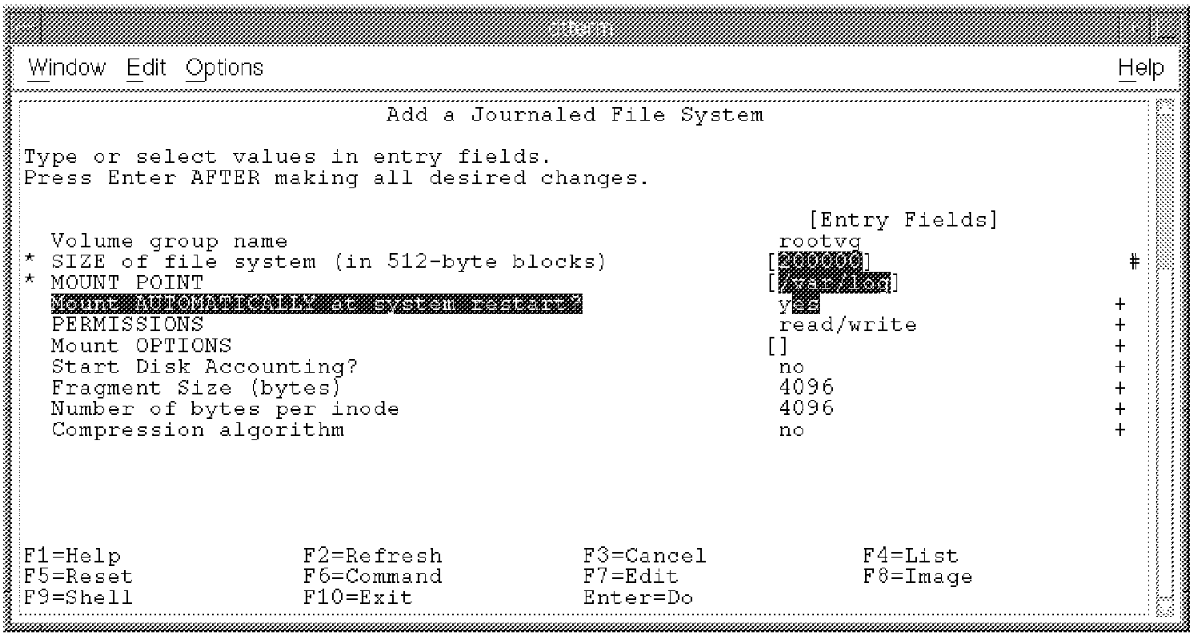
- Browse /etc/rc.tcpip and disable startup commands for subsystems such as routed, gated, etc. As a minimal configuration, you should keep syslogd, sendmail and if required named.
- Browse /etc/inittab and disable the dispatch command for processes such as NFS daemon.

Cleaning Up User IDs and Setting Up Password Rules: You should keep as few user accounts as possible on your Web server. The minimal set of user accounts is: root, daemon, bin, adm, nobody and www (if you chose to add it) plus your regular logins for remote maintenance. You should set up a password policy to encourage users to regularly change their passwords and to choose nontrivial passwords (see default stanza in /etc/security/user file).

Setting Up Logs: You should configure syslogd to log messages related to daemon activities and authentication. As the log files tend to grow larger and could turn the file system they depend on into a full file system, create a separate file system for your log files. Depending on the activity of your server and the frequency with which you will back up your log files, the size of your file system can range from several MB up to 1 GB. First of all, create a new file system by entering:

```
smitty crjfs
```

The following panel appears:



```
                          Add a Journaled File System

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                         [Entry Fields]
  Volume group name                                     rootvg
* SIZE of file system (in 512-byte blocks)             [200000]                 #
* MOUNT POINT                                          [/var/log]
  Mount AUTOMATICALLY at system restart?                yes                     +
  PERMISSIONS                                           read/write              +
  Mount OPTIONS                                        []                       +
  Start Disk Accounting?                                no                      +
  Fragment Size (bytes)                                 4096                    +
  Number of bytes per inode                             4096                    +
  Compression algorithm                                 no                      +




F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure 23. Creating the /var/log File System

The file system is not mounted yet. You should do it manually as follows:

```
mount /var/log
```

Modify the /etc/inetd.conf file to include the following statements:

```
auth.debug /var/log/auth.log
daemon.debug /var/log/daemon.log
```

Then create the log files and refresh the syslogd daemon as follows:

```
touch /var/log/auth.log
touch /var/log/daemon.log
refresh -s syslogd
```

Refer to 4.6.8, “Logging Procedures” on page 148 for information about setting up your Web server to log files.

A new feature in Internet Connection Server 4.1 allows you to set up the syslog daemon to log HTTP daemon error messages.

The channel on which the HTTP daemon sends its messages is user.err. In order to activate this facility, you should add the following statement to /etc/inetd.conf:

```
user.err /var/log/user.log
```

and create the corresponding file:

```
touch /var/log/user.log
```

before refreshing the syslogd daemon.

4.5.1.5 Preparing for the Installation

To perform the following operations and to install the server, you will need root authority.

- Make sure that the hostname of your server machine is correctly set by entering:

  ```
  hostname
  ```

- Check the operation of TCP/IP (ping any known machine, test DNS, etc.)
- Check the consistency of your AIX system by entering:

  ```
  lppchk -v
  ```

  and fix reported problems (if any).

- Although the server installation should not damage your system, think of backing up your system (always a good idea).

4.5.1.6 Installing the Server

To install, you must be logged in as root.

1. Enter:

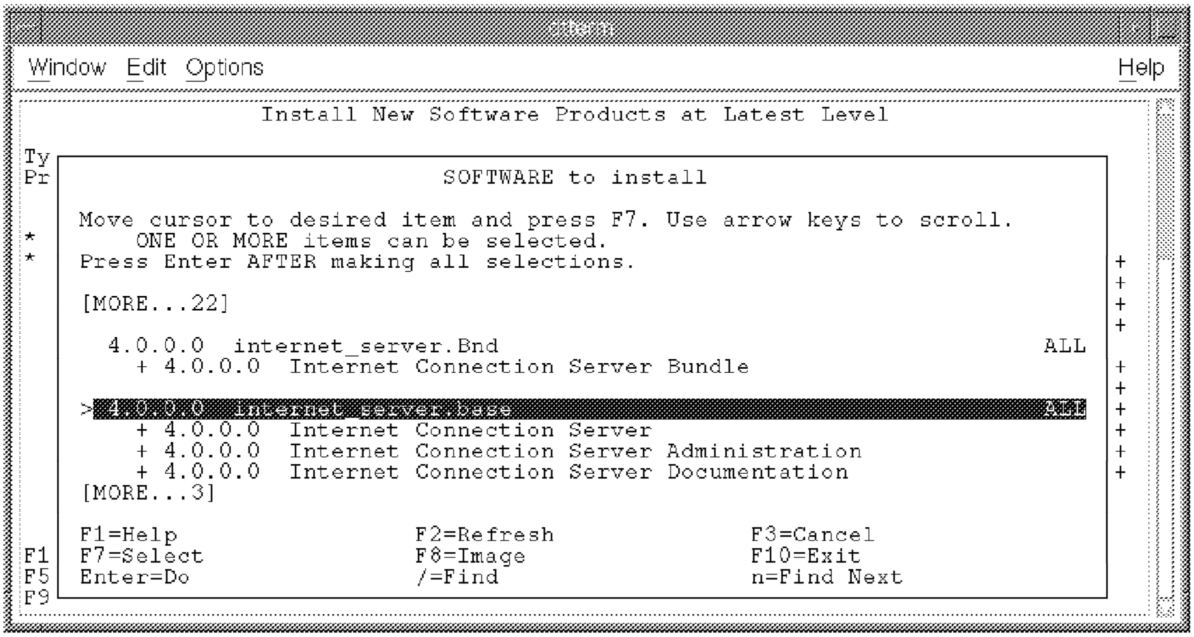
   ```
   smitty install_latest
   ```

2. From the Install Software at Latest Level screen:

   - If you purchased the product and have it on distribution media, press F4 for the INPUT device / directory for Software option and select the appropriate device.
   - If you downloaded the product, enter the name of the directory in which you extracted the packages.

   Then press Enter.

3. From the Install Software Products at Latest Level updated screen, press F4 for the Software to install option. The SOFTWARE to install screen appears as shown below with a list of packages you can selectively install.



```
                     Install New Software Products at Latest Level

Ty
Pr                                  SOFTWARE to install

     Move cursor to desired item and press F7. Use arrow keys to scroll.
*        ONE OR MORE items can be selected.
*    Press Enter AFTER making all selections.                                     +
                                                                                  +
     [MORE...22]                                                                  +
                                                                                  +
       4.0.0.0  internet_server.Bnd                                           ALL
         + 4.0.0.0  Internet Connection Server Bundle                             +
                                                                                  +
     > 4.0.0.0  internet_server.base                                          ALL +
         + 4.0.0.0  Internet Connection Server                                    +
         + 4.0.0.0  Internet Connection Server Administration                     +
         + 4.0.0.0  Internet Connection Server Documentation                      +
     [MORE...3]

     F1=Help                  F2=Refresh                F3=Cancel
F1   F7=Select                F8=Image                  F10=Exit
F5   Enter=Do                 /=Find                    n=Find Next
F9
```

Figure 24. Installing the Internet Server Package

4. Move the cursor to internet_server.base and press F7 to select to install Internet Connection Server, Internet Connection Server Administration and Internet Connection Server Documentation file sets. Move the cursor to the security related packages and press F7 to select (if applicable). Select also the CICS Gateway and DB2 WWW packages if desired. Then press Enter. The SOFTWARE to install option, in the Install Software Product at Latest Level screen is updated to show that you want to install internet_server.base.

5. To install internet_server.base, press Enter. You will have to press Enter again at the ARE YOU SURE screen.

   When the server is successfully installed, you will get `Command: OK` in the COMMAND STATUS screen.

6. Press F10 to exit *smitty*.

Verify that the server is properly installed as follows:

```
lslpp -l ′internet*′
```

You should observe the following:

```
Path: /usr/lib/objrepos
  internet_server.base.admin
                            4.0.0.0  COMMITTED  Internet Connection Server
                                                Administration
  internet_server.base.doc
                            4.0.0.0  COMMITTED  Internet Connection Server
                                                Documentation
  internet_server.base.httpd
                            4.0.0.0  COMMITTED  Internet Connection Server
  internet_server.msg.en_US.httpd
                            4.0.0.0  COMMITTED  Internet Connection Server
                                                Messages - en_US

Path: /etc/objrepos
  internet_server.base.admin
                            4.0.0.0  COMMITTED  Internet Connection Server
                                                Administration
  internet_server.base.httpd
                            4.0.0.0  COMMITTED  Internet Connection Server
```

Your HTTP daemon has been added as the subsystem in your AIX server and automatically started by the installation procedure with the default configuration settings. Verify that it is effectively started:

```
lssrc -s httpd
```

Verify that you can view the Internet Connection Server front page by pointing your browser at your server with the following URL:

```
http://your.server.name/
```

If your AIX server is on the network, anyone can now access your Web server. Since your Internet Connection Server may be configured remotely, you should now stop your server as follows:

```
stopsrc -s httpd
```

and you should restart it only when the Web administrator′s password is changed. Your server is now ready to run.

4.5.2 MVS/ESA

In this section we tell you how to prepare and install Internet Connection Server for MVS.

4.5.2.1 Prerequisites

To start we list the hardware and software requirements for the product.

Hardware Requirements: Internet Connection Server for MVS/ESA (ICS/MVS) operates on all ESA-capable machines supporting MVS/ESA SP Version 5.2.2. Additional requirements include:

- One tape or cartridge drive for installation
- Communication hardware for network attachment supported by the transport provider such as TCP/IP or VTAM
- One or more workstations capable of running a Web browser for configuration and administration

Software Requirements:

- MVS/ESA SP Version 5.2.2 (5655-068 or 5655-069) with APAR OW14736 or
- OS/390 (5645-001)

IBM Internet Connection Server for MVS/ESA(5655-156) exploits and depends upon OpenEdition services and requires installation, configuration and starting of the OpenEdition shell in its specified operating environment. The PTFs associated with the following MVS/ESA OE APARS should be installed:

  - OW15675
  - OW15773
  - OW16636
  - OW13795

In addition, the following are required:

- DFSMS PTFs associated with APAR OW16222
- SMP/E Release 8.1 or later (5668-949)
- IBM C/C++ for MVS/ESA Language Support feature or Language Environment for MVS and VM 5.1 (5688-198) or higher
- RACF 2.1 with APAR OW09834 or 2.2 (5695-039) or higher and PTFs associated with APARs PN76907, PN75884

  Note: An external security system supporting SAF at an equivalent level may be used.

One of the following is required as a transport provider:

- TCP/IP Version 3 (5655-HAL) and PTFs associated with APARs:
  - PN72972, PN70461, PN62474, PN65950, PN71946, PN75274, PN77544, PN71946, PN77696, PN73070
- ACF/VTAM Version 4 Release 3 (5695-117) with AnyNet/MVS feature

Note: For further information, see http://www.raleigh.ibm.com/icf/icfprod.html.

4.5.2.2 Getting the Software

ICS/MVS can be ordered as a product for installation under MVS 5.2.2. In addition, OS/390 Internet BonusPak which includes ICS/MVS, with supplementary tools and samples, is available for MVS 5.2.2 and OS/390.

Unlike some of the other Internet Connection servers, there are no FTP servers from where you can download the ICS/MVS code.

4.5.2.3 Planning for the Installation

ICS/MVS is installed using the SMP/E.

OpenEdition and HFS: Before you install ICS/MVS, you must have OpenEdition MVS installed on your system. Also, you should prepare HFS disk spaces and consider authorization environments.

TCP/IP for MVS: Make sure that the configuration file is updated for OpenEdition MVS. ICS exploits Virtual IP Address (VIPA) through the -h run-time parameter. VIPA allows multiple copies of ICS/MVS to run on the same MVS system.

RACF: ICS/MVS has extensive access control support including RACF validation of user IDs and passwords and access control through SURROGAT user ID support. These control Web access to resources on MVS. Users are identified by an OMVS user ID (UID), which is kept in the RACF user profile, and an OMVS group ID (GID), which is kept in the RACF group profile.

LE/370: To install the ICS, LE/370 must be installed and customized for the MVS OE system.

Web Browser: You can use the Configuration and Administration Forms from a remote client that has access to the server. All you need is IBM WebExplorer (or any other Web browser) and your ICS must be up and running.

4.5.2.4 Installation

The installation steps for ICS/MVS R1 are as follows:

1. Unload the ICS/MVS installation samples.
2. Using sample IMWJALLO, allocate your target and distribution libraries.
3. Create mount point /usr/lpp/internet and mount the target hierarchical file system (HFS) using ISHELL.
4. Using sample IMWJDDDF, provide SMP/E access to your data sets.
5. Using sample IMWMKDIR, create the HFS structure.
6. Using sample IMWJR100, SMP RECEIVE ICS/MVS′s functions.
7. Using sample IMWJEXAP, SMP APPLY ICS/MVS′s functions.
8. Using sample IMWJEXAC, SMP ACCEPT ICS/MVS′s functions.
9. Set up the security environment.
10. Set up the TCP/IP configuration.
11. Copy and customize the Server Procedure.
12. Set up the ownership of the installed files.
13. Manage the configuration file (httpd.conf).

Considerations

- The hardware configuration and the network topology need to be considered when setting up the TCP/IP and httpd configuration files.
- Security and protection
  - Packet filtering and a firewall
  - User name and password
  - Address template
- Character Codes
  - Code page / locale
  - ASCII and EBCDIC issues

    Before creating your own home page, there are some ASCII/EBCDIC considerations you should understand. MVS OE stores HTML files in EBCDIC or ASCII. The ICS/MVS uses different file suffixes to distinguish between EBCDIC, ASCII, and binary. EBCDIC is the default and allows the file to be viewable to MVS OE editors such as ISPF.

    EBCDIC data must be in OpenSystems Latin-1 code page (IBM1047). ASCII data can be stored in OpenSystems Latin-1 code page ISO8809-1 or in any MVS OE code page. Binary data can be delivered without translation by using the AddEncoding and AddType directive.

4.5.2.5 Setup and Customization

- TCP/IP configuration

  Put the TCP/IP data configuration in a location such as SYS1.TCPPARMS(TCPDATA) so that OpenEdition sockets can find the name server.

  In the TCP/IP profile, reserve the port that the Web server uses (default is the port ′80 TCP OMVS′) to access OMVS.

- Start Procedures Table

  Copy the IMWEBSRV sample to a site procedure library, for example, SYS1.PROCLIB or USER.PROCLIB, and modify the parameters in the procedure.

- Security environment

  Create the IMWEB group ID, the WEBADM/WEBSRV user IDs, and surrogate user IDs for the Web server.

- Customize the httpd.conf file

  Copy the sample httpd.conf file to /etc/httpd.conf and modify it for your environment.

- Getting Content on MVS

  One of the biggest issues you will face is building content and getting it on MVS. Generally, HTML content files can be edited directly using OEDIT in the OpenEdition shell. If you edit the files on another platform such as AIX, you have to move them to MVS, using FTP or copy them after mounting the HFS using NFS.

4.5.2.6 Testing

We recommend that you run the following tests:

- Starting and stopping the Web server. Although IMWEBSRV is usually started as a started task, it can be controlled by using OpenEdition commands. You can use the kill or wwwcmd commands to stop or restart the process in OMVS. You can also use the MVS commands PURGE and CANCEL to stop the job.
- Viewing the ICS/MVS Front Page from a Web browser.
- Checking whether authorized/not-authorized clients can access the server.

4.5.2.7 Tuning

Since ICS/MVS uses OpenEdition services, such as threads and HFS, TCP/IP or AnyNet for network services, it is important that these components are tuned correctly for optimal throughput.

System Resources Management (SRM) is one of the MVS components that determines which address spaces should be given access to system resources, and the rate at which the address spaces are allowed to consume those resources. The access to resources is controlled by assigning address spaces to performance groups. The following are areas to consider:

- Updating IEALPAxx to place LE/370 modules in shared storage
- Adding libraries to Virtual Lookaside Facility (VLF)
- Updating SYS1.PARMLIB to set up the MIN/MAX for APPC initiators
- Caching and placement of HPF data sets
- Monitoring TCP/IP performance
- Monitoring NFS performance
- Modifying Max/MinActiveThreads in httpd.conf

4.5.2.8 Backing Up

The following essential files must be backed up:

- httpd.conf configuration file
- Password files
- Group files
- ACL files
- Content files
- Access/error log files

The four main options when it comes to backing up these files are as follows:

1. Back up the HFS data set using IEBCOPY or any other tool
2. Back up the files to another media using standard shell commands such as cp/xcopy, tar and cpio
3. Back up the files individually using ADSM
4. Back up the files to a remote machine using FTP or NFS

There are several backup tools such as DDR, IEBCOPY, and DFDSS for an MVS system. Thus, you can select and combine your favorite methods to back up those files.

4.5.2.9 Inspecting Logs

There are several log tools available that can help you in inspecting or debugging any problems. The tools are as follows:

- wwwlogs (AccessLogs and ErrorLogs)

  AccessLog and ErrorLog files are in the wwwlogs directory. You can use them directly to specify where the server should log all requests made by the client and any internal errors.

  The format of the common log record is:

  remotehost remoteuser authuser date request status bytes

  The parameters are defined as follows:

  remotehost The hostname or IP address if the hostname cannot be resolved

  remoteuser The remote log name of the user as defined by RFC931

  authuser The name with which the user has been authenticated

  date The date and time of the request

  request The request sent from the client

  status The HTTP/1.0 status code returned to the client

  bytes The number of bytes of data transferred to the client

- Master console log (or using SDSF)

  RACF sends access violation messages to the MVS console. These messages contain, for example, information about the following:

  - The resource that was accessed
  - The user ID and group ID that was used in the access check
  - The requested access authority and the access authority this user ID has
  - The function that was used when the violation occurred
  - Date and time stamps

- Operator commands to get a current status

  For example, ′D OMVS,A=ALL′

- RACF SMF reports

  RACF writes records to the system management facility (SMF) for detected unauthorized attempts to enter the system. Optionally RACF writes records to SMF for authorized attempts and detected unauthorized attempts in the following categories:

  - Access to RACF-protected resources
  - Issuing of RACF commands
  - Attempts to modify profiles on the RACF database.

  RACF writes these records to an MVS data set. To list SMF records, you should use the RACF SMF data unload utility. With the RACF SMF data unload utility, you can browse or upload to a database, query, or reporting package, such as DB2.

4.5.3 AS/400

In this section we introduce the Internet Connection for AS/400 System. For information on preparing and configuring the product refer to 4.6.9, "AS/400 HTTP Server Configuration and Operation" on page 165.

4.5.3.1 TCP/IP

In this section we provide some introductory information regarding the TCP/IP capabilities of the AS/400 system. We have not done this for all the platforms in this chapter but it seems appropriate due to the special communications characteristics of the AS/400 system.

To connect the AS/400 system to the Internet, you need the TCP/IP protocol. The AS/400 system has implemented TCP/IP since V1R2. With V3R0M5, it was *free* (no charge), and with V3R1 it is integrated into OS/400.

The TCP/IP protocol has been available on the AS/400 system since V1R2 as a separately-priced licensed program product (LPP) called the TCP/IP Connectivity Utilities/400. But, it was not until V3R1 that it was really integrated into the operating system (OS/400). The product, TCP/IP Connectivity Utilities/400, still exists and includes all of the applications such as Telnet, FTP, and so on, but is a free LPP that comes when you purchase OS/400.

Also on V3R1, we have integrated the C-Sockets interface into OS/400, so client/server applications can be written to work through the Internet or even in a local TCP/IP network. Most of the TCP/IP applications have been rewritten using a C-Sockets Interface on V3R1.

V3 Customers - Excluding V3R05: All of the integrated functions of TCP/IP are on V3R1 and the following list contains some of the highlights:

1. *Free* (no charge).
2. Many of the applications were rewritten from Pascal to ILE C/400.
3. Performance is comparatively better. It is up to eight times faster.

V2 and V3R05 Customers - Excluding V2R1: For those customer that have not upgraded to V3R1 and intend to install TCP/IP on their machines, it is probably cheaper if they go to V3R1 instead of purchasing the TCP/IP product for V2.

Table 5. Cross-Reference: OS/400 Licensed Program x Function

OS/400 Licensed Program	Function
OS/400 (5763-SS1)	TCP/IP protocol stack
	Configuration support for TCP/IP protocol stack
	Configuration support for SLIP 1
	NETSTAT command
	PING command
	SNMP agent
	Configuration support for SNMP agent
	APPC over TCP/IP
	TCP/IP over AnyNet using sockets API
	C sockets API
TCP/IP Connectivity Utilities (5763-TC1)	Pascal TCP and UDP API
	Telnet
	SMTP
	FTP
	LPR/LPD
	POP3 1
	HTTP server 1
	WSG 1
	DB2WWW 1
TCP/IP File Server Support/400 (5798-TAA and 5798-TAZ)	NFS
AS-IS Service Offering	Gopher

Note:

1 New in V3R2

4.5.3.2 Hardware Connectivity

The hardware required to connect an AS/400 to the Internet is dependent on the following:

- What type of connection are you going to use (SLIP, PPP, X.25, and so on)?
- How does your ISP provide that type of connection?

Table 6 (Page 1 of 2). Cross-Reference: Hardware x Connection Type

Type of Connection	Hardware Needed	External Hardware Needed
X.25	Communications port	None
Frame Relay	Communications port	None
SDDI	SDDI IOP	None
FDDI	FDDI IOP	None
Wireless	Wireless IOP	None
SLIP	Communications port	None

Table 6 (Page 2 of 2). Cross-Reference: Hardware x Connection Type

Type of Connection	**Hardware Needed**	**External Hardware Needed**
PPP	LAN Adapter (token-ring, Ethernet, wireless)	IP router

4.5.3.3 Internet Connection for AS/400

> **Important note**
>
> Install PTFs SF32078 (for 5763-TC1) and SF31077 (for 5763-SS1) before using Internet Connection for AS/400.

With the announcement of Internet Connection for AS/400, IBM offers AS/400 customers the capability to take an active part in the ever-growing and exciting World Wide Web. In doing so, IBM followed its commitment to network computing on the AS/400 system which means that everyone can access and distribute information, applications, and services provided by the network.

4.5.3.4 HTTP Server Configuration and Operation

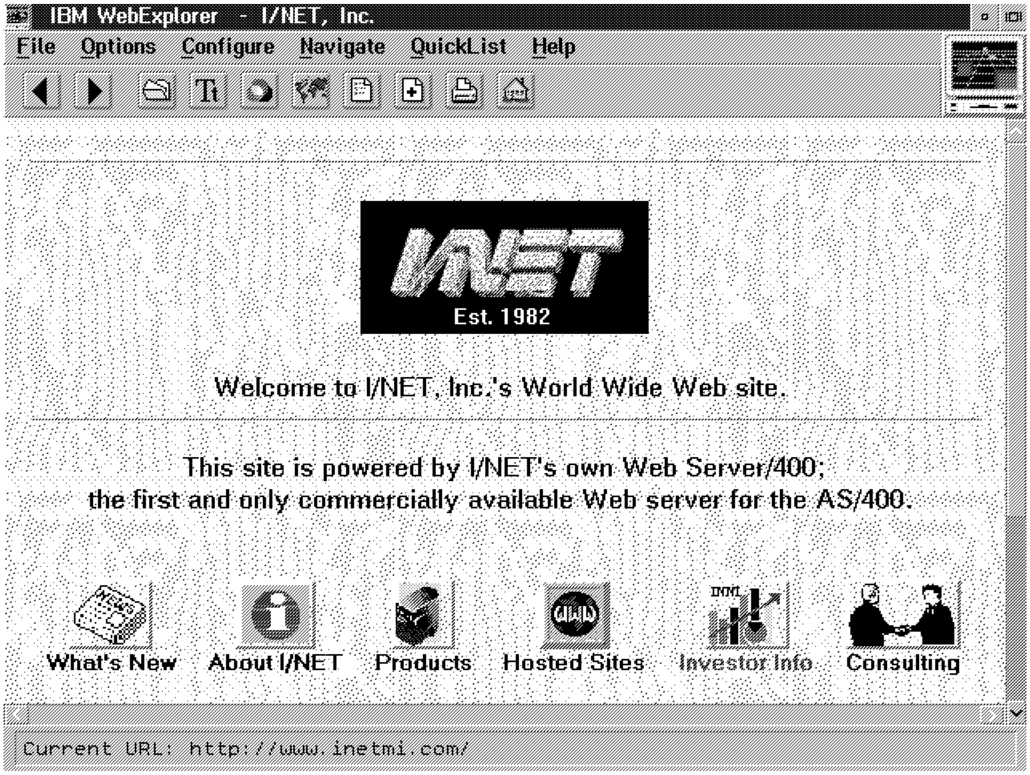
Many of the features of Internet Connection for AS/400 are reliant upon the HTTP server.

4.5.3.5 The HTTP Server Is a Read-Only Server

The HTTP specification defines seven ″methods″ that represent seven types of requests that a remote HTTP client application can send to an HTTP server. The HTTP server implements only the GET, HEAD, and POST requests. The HTTP server does *not* implement two other methods, PUT and DELETE. This makes it impossible for a request from an external client to cause an AS/400 object to be overwritten.

4.5.3.6 I/NET′s Web Server/400

I/NET′s Web Server/400 is another Web server available, at a cost, for the AS/400 system.



Welcome to I/NET, Inc.'s World Wide Web site.

This site is powered by I/NET's own Web Server/400;
the first and only commercially available Web server for the AS/400.

Current URL: http://www.inetmi.com/

Figure 25. I/NET's Web Server/400

This server is fully HTTP/1.0 compliant and as this book is being written, offers the following features:

- Integrated File System support
- Scripts (including REXX, CL, ILE-C, RPG, and COBOL)
- Database, shared folders, and spooled files
- Ultimedia System Facilities support
- Logging
- National language support

For the latest details of their product, take a look at URL: `http://www.inetmi.com`

You can order I/NET's Web server code from:

```
I/NET, Inc.
     643 West Crosstown Parkway
     Kalamazoo, MI  49008
     USA
     ATTN: Web Server/400 Customer Service
Phone: (616) 344-3017 ext. 158
FAX    (616) 344-0186
E-Mail: websales@inetmi.com
```

4.5.4 OS/2 Warp

In this section we tell you how to prepare and install Internet Connection Server for OS/2 Warp.

4.5.4.1 Prerequisites

To start we list the hardware and software requirements for the product.

- Software Requirements:
  - OS/2 Warp V3.0 or later
  - One of the following:
    - TCP/IP V3.0, which is part of Warp Connect
    - TCP/IP V3.1, which is part of Warp Server
    - Internet Connection for OS/2 which is included in all versions of Warp
    - TCP/IP V2.0 with latest CSD
    - A partition formatted using the High Performance File System (HPFS)
  - For the DB2 Gateway feature:
    - DB2/2 Version 1.2 or later
    - 2 MB of free disk space
  - For the CICS Gateway feature:
    - Access to a CICS for OS/2 server
    - CICS client for OS/2 Version 1.0 installed, including updates from Corrective Service Disk (CSD) 1
    - 4.5 MB of free disk space

Please refer to the READ.ME file for further instructions.

This server *must* be installed on an HPFS-formatted drive; it cannot be installed on a FAT-formatted drive. Only the server needs to be on the HPFS partition, not the entire operating system.

- Hardware Requirements:
  - A computer that can support OS/2 Warp V3.0 or later
  - A communications adapter that is supported by the TCP/IP protocol stack
  - 4 MB of free disk space (11 MB with both the DB2 Gateway and CICS Gateway features installed)
  - A minimum of 12 MB of memory (16 MB of memory is recommended.)
  - A 1.44 MB diskette drive or a CD-ROM drive for installation purposes
  - Available documentation which is sent with the product; including readme file(s).

Use the VIEW utility to view any required information files. Note that information files can be identified by their .INF extension.

Refer to the Up and Running in folder IBM Internet Connection Secure Server after the product is installed.

In the folder IBM DB2 World Wide Web Connection for OS/2 you will find the following documentation:

- Application Developer′s Guide in INF and HTML format
- License agreement
- Installation information for DB2 Gateway program

4.5.4.2 Get Your Copy of the Internet Connection Server for OS/2 Warp

You have two possibilities to get a copy of the Internet Connection server:

- Download it from the Internet Connection Family Web site at: http://www.ics.raleigh.ibm.com.
- Purchase it

4.5.4.3 Types of Installation

The different types of installation are as follows:

- First time installation

  Follow the steps under installing for the first time in the *Up and Running* manual.

- Migration

  If Internet Connection Server (without security function) or a different version of Internet Connection Secure Server is installed, you can migrate to this version of Internet Connection Secure Server. When you migrate, you can build upon the configuration you had for your old server.

  Refer to the *Up and Running* manual for more information regarding migration.

- Reinstallation:

  Refer to the *Up and Running* manual for more information regarding reinstallation.

If you are installing for the first time, there are two options to consider:

- Diskettes/CD-ROM

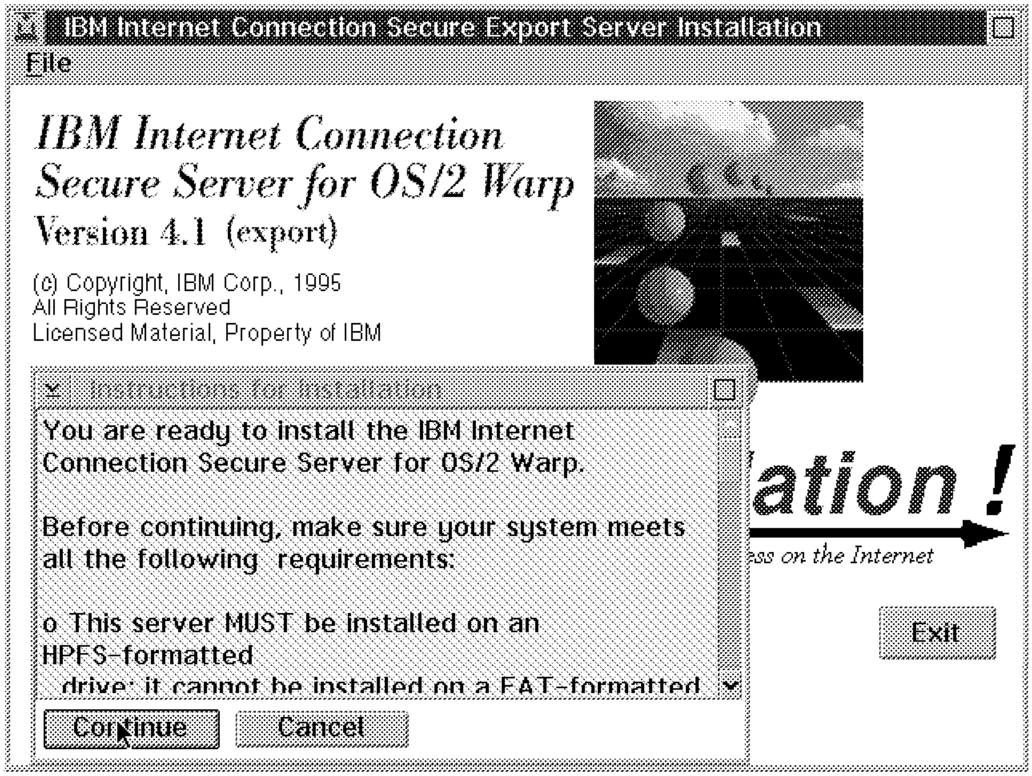
  Enter `X:install`.

  X is the drive where the Internet Connection Secure Server diskette or CD-ROM can be found.

- Remote code server

  Enter `Z:\path\install`.

  Z is the drive where the Internet Connection Secure Server files reside on the code server.


IBM Internet Connection Secure Export Server Installation
File
Instructions for Installation
ation !
ss on the Internet
Exit
Continue
Cancel

IBM Internet Connection Secure Server for OS/2 Warp
Version 4.1 (export)

(c) Copyright, IBM Corp., 1995
All Rights Reserved
Licensed Material, Property of IBM

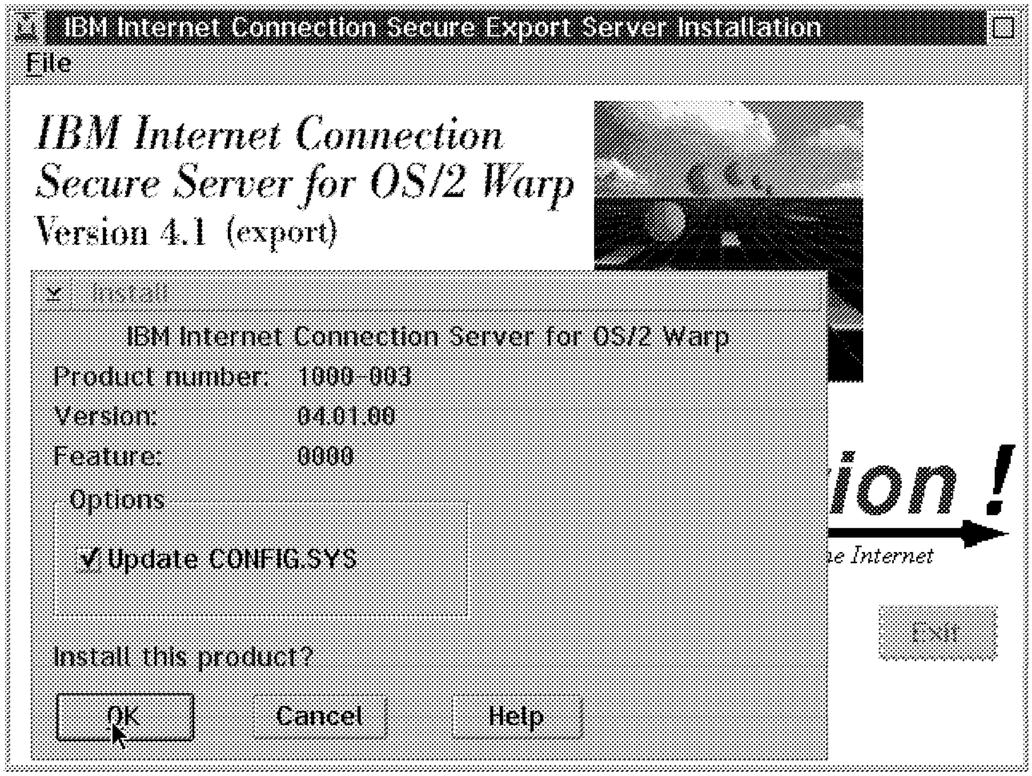
You are ready to install the IBM Internet Connection Secure Server for OS/2 Warp.

Before continuing, make sure your system meets all the following requirements:

o This server MUST be installed on an HPFS-formatted
drive; it cannot be installed on a FAT-formatted

Figure 26. Installation - Check System Requirements

From the Instructions for Installation window shown in Figure 26 click on **Continue** after you have read the information and checked that your system meets the specifications.



IBM Internet Connection Server for OS/2 Warp

Product number: 1000-003

Version: 04.01.00

Feature: 0000

☑ Update CONFIG.SYS

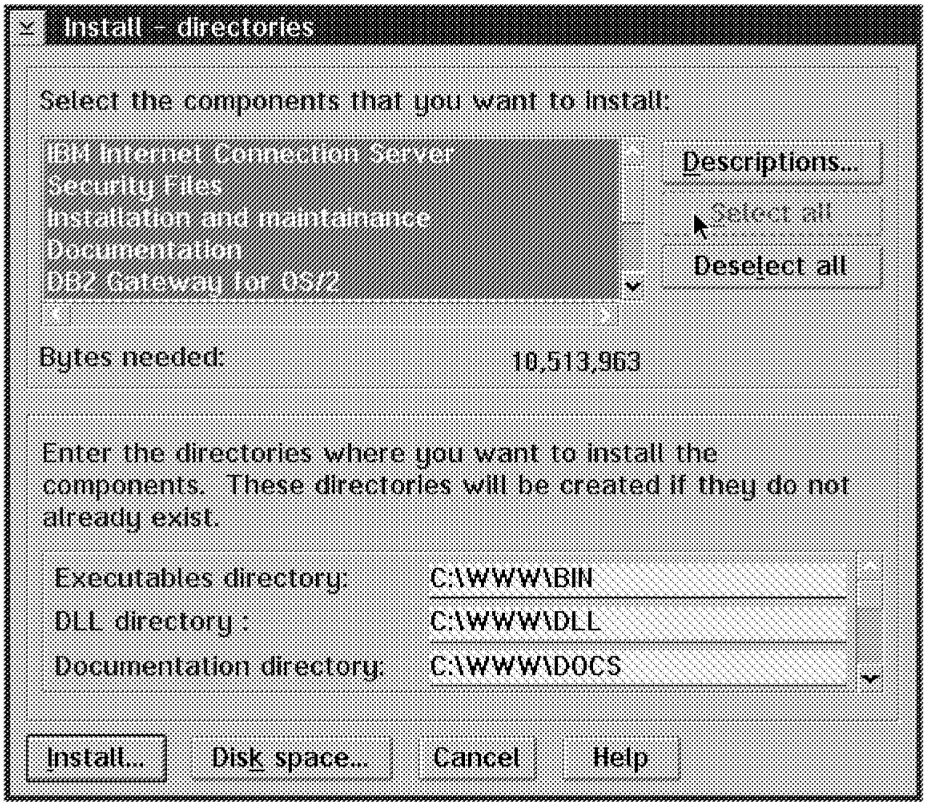
Install this product?

Figure 27. Installation - Update CONFIG.SYS and Install Product

From the Install window shown in Figure 27 you have the following options:

- Click on **Update CONFIG.SYS**.
- Click on **OK** to start installation.

Update CONFIG.SYS. You should leave this box checked so that the installation procedure will automatically update your CONFIG.SYS file to include the directories where you install the Internet Connection Secure Server.



Select the components that you want to install:

Bytes needed: 10,513,963

Enter the directories where you want to install the components. These directories will be created if they do not already exist.

Executables directory: C:\WWW\BIN
DLL directory : C:\WWW\DLL
Documentation directory: C:\WWW\DOCS

Figure 28. Installation - Select Components and Directories

From the Install - directories window shown in Figure 28 click on **Select all** or on the components that you wish to install. Available components are as follows:

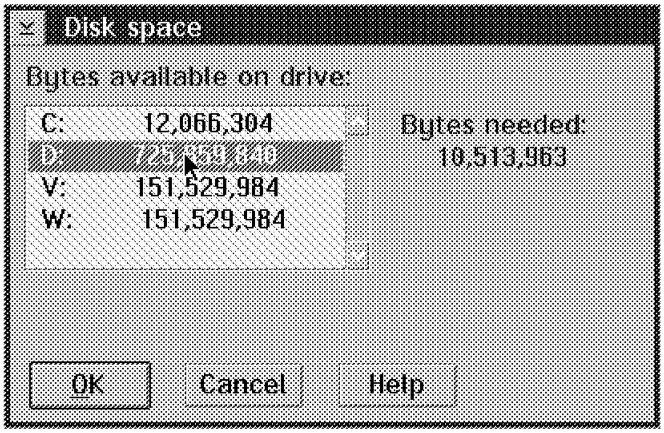
- IBM Internet Connection Secure Server
- Documentation
- DB2 Gateway for OS/2
- DB2 NLS package
- DB2 Sample application
- DB2 Documentation
- CICS Gateway for OS/2

Click on **Disk space** if you will not install on the C drive.

Click on **Install** for continuing the installation.

Select the components you want to install by highlighting them or click the **Select all** button to install all products.

Select **Disk space** to get a list of all available drives to install.



Bytes available on drive:

C:	12,066,304
D:	725,859,840
V:	151,529,984
W:	151,529,984

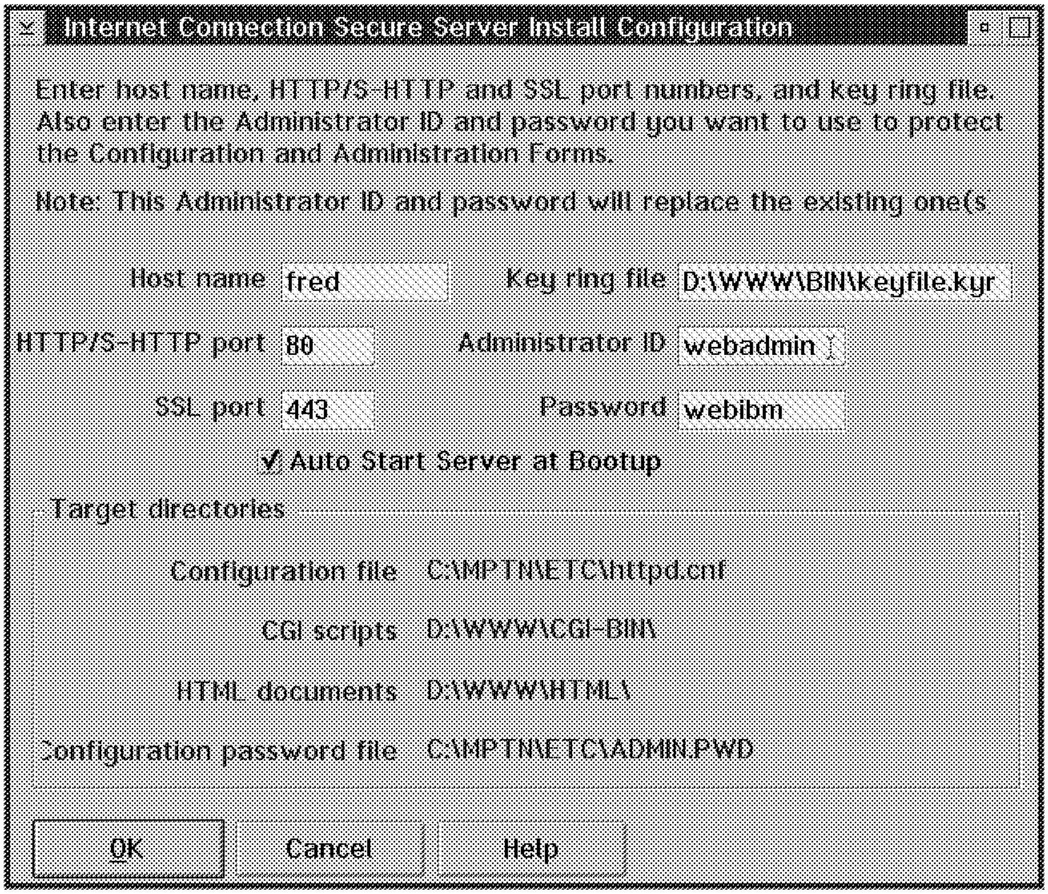
Bytes needed:
10,513,963

Figure 29. Installation - Disk space

From the Disk space window shown in Figure 29, select the drive where the product should be installed and click on **OK**. This action will return you to Figure 28 on page 84.

If all is OK, click on the **Install** button to begin the installation.

4.5.4.4 Initial Configuration



Enter host name, HTTP/S-HTTP and SSL port numbers, and key ring file. Also enter the Administrator ID and password you want to use to protect the Configuration and Administration Forms.

Note: This Administrator ID and password will replace the existing one(s

Host name: fred

Key ring file: D:\WWW\BIN\keyfile.kyr

HTTP/S-HTTP port: 80

Administrator ID: webadmin

SSL port: 443

Password: webibm

☑ Auto Start Server at Bootup

Target directories

Configuration file: C:\MPTN\ETC\httpd.cnf

CGI scripts: D:\WWW\CGI-BIN\

HTML documents: D:\WWW\HTML\

Configuration password file: C:\MPTN\ETC\ADMIN.PWD

Figure 30. Configuration

Next you will see the Internet Connection Secure Server Install Configuration window as shown in Figure 30.

We recommend that you immediately change the default administrator ID webadmin and password webibm. If you do not do this, you may allow an unauthorized user to access your system.

Explanation of the fields:

Host name The name by which your server will be known must be unique to your LAN. The default value is the host name defined in your CONFIG.SYS file.

HTTP/S-HTTP Port The default value of 80 is the well-known port number for Hypertext Transfer Protocol (HTTP) and Secure Hypertext Transfer Protocol (S-HTTP).

SSL Port The port you want your server to listen on for requests for documents protected by the Secure Sockets Layer (SSL) protocol. The default is 443.

Key ring file The name of the file where you want to store public-private key pairs that the server can use for secure communications.

Administrator ID The ID of your server administrator. Anyone attempting to use the Internet Connection Secure Server Configuration and Administration Forms will be prompted to enter this ID. The default value is webadmin. As mentioned before it is strongly recommended that you change the default during installation.

Password The password you want to use to protect access to the Configuration and Administration forms. Anyone attempting to use the Internet Connection Secure Server Configuration and Administration forms will be prompted to enter this password. The default value is webibm. It is strongly recommended that you change the default during installation. Note that the password is case-sensitive.

Use the Auto Start Server at Bootup check box to indicate if you want the Internet Connection Secure Server to start automatically when you start your host machine. If you check the box, the server will be added to your OS/2 startup folder. You can start Internet Connection Secure Server from any OS/2 command prompt by entering HTTPD. You can stop the server the same way that you stop other OS/2 programs. We list below the methods of stopping the server:

- Double-click on the icon in the top left corner of the Internet Connection Secure Server window.
- Select **Exit** from the server pull-down menu on the menu bar of the Internet Connection Secure Server.
- Press F3 from the Internet Connection Secure Server window.
- Press Ctrl+Esc. Click with the right mouse button on the window list entry for Internet Connection Secure Server. Click on **Close**.

Target directories are shown here for information and cannot be changed. You may want to make a note of the information in this window for future reference.

The directory structure is as follows:

executables directory d:/www/bin

dll directory	d:/www/dll
documentation	d:/www/docs
CGI binscripts	d:/www/cgi-bin
HTML	d:/www/html
remote admin directory	d:/www/admin
icons+graphics	d:/www/icons
logs	d:/www/logs
DB2 WWW base directory	d:/db2www
DB2 WWW macro directory	d:/db2www/macro
CICS Client base	d:/cicscli

You can view the directory structure online by clicking on the Installation Utility in the Internet Connection Secure Server folder. Once it is loaded, select **Details** and then **Product status** from the pull-down menu.

Once the server is running you should back up the following files:

- Configuration file located at drive/mptn/etc/httpd.cnf
- Password files located at drive/mptn/etc/admin.pwd
- Key ring files located at drive/www/bin/keyfile.kyr

4.5.4.5 Installed Folders

After your installation has successfully completed you have a new folder on your desktop as shown in Figure 31.

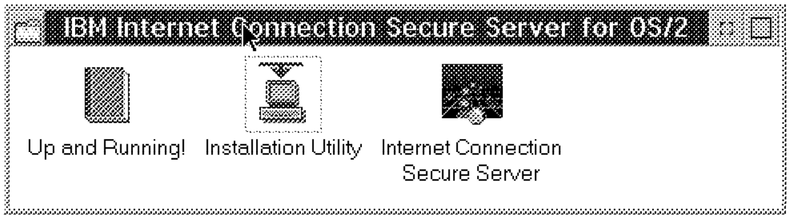

IBM Internet Connection Secure Server for OS/2
Up and Running!
Installation Utility
Internet Connection Secure Server

Figure 31. IBM Internet Connection Secure Server for OS/2 Warp

The options that are available from this folder are as follows:

- Up and Running!

  For viewing all of the information that is provided with Internet Connection Secure Server for OS/2

- Installation Utility

  For installation and maintenance for IBM Internet Connection Secure Server for OS/2 (view installed products, view directories where the products are installed, update/delete products, etc.)

- Internet Connection Secure Server

  For starting the Internet Connection Secure Server and DB2 WWW Connection (if you selected to install it)

If you also installed DB2 WWW Connection then you will have the additional new folder shown in Figure 32 on page 88.

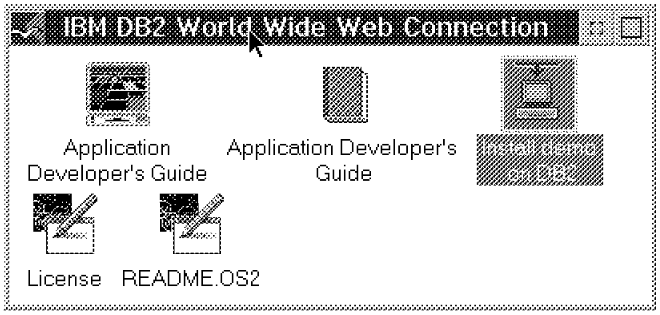

IBM DB2 World Wide Web Connection
Application Developer's Guide
Application Developer's Guide
Install demo on DB2
License
README.OS2

Figure 32. DB2 WWW Connection

The selectable options here are:

- Application Developer′s Guide in HTML and INF format
- Install demo on DB2
- License agreement
- A README file providing installation and configuration information

4.5.4.6 Features of DB2 and CICS

Web applications for DB2 built with DB2 WWW Connection V1 allow any Internet user with an HTML standard Web browser to access your DB2 data. Changes to the existing data structure are not necessary. Using standard hypertext markup language (HTML) and structured query language (SQL), you can build applications that query your DB2 data with standard SQL statements.

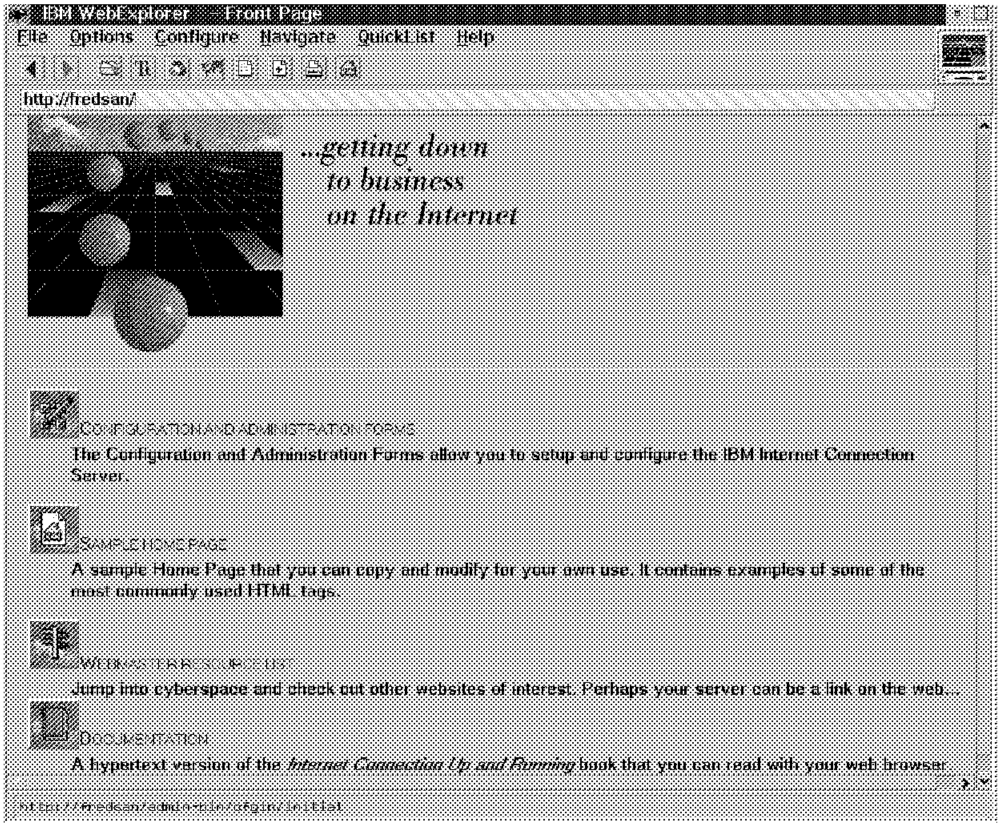
The CICS Internet Gateway provides automatic translation between CICS 3270 data streams and HTML. It uses the CICS External Presentation Interface. CICS dialogs can now be conducted via the Internet.

4.5.4.7 Viewing the Front Page

After you install the product and have the server running, you can use the WebExplorer or any other Web browser to look at the Internet Connection Secure Server Front Page. Point your browser at your server with the following URL:

```
http://your.server.name/
```

Where your.server.name is the fully qualified name of your host. You should see something similar to Figure 33 on page 89.



CONFIGURATION AND ADMINISTRATION FORMS

The Configuration and Administration Forms allow you to setup and configure the IBM Internet Connection Server.

SAMPLE HOME PAGE

A sample Home Page that you can copy and modify for your own use. It contains examples of some of the most commonly used HTML tags.

WEBMASTER RESOURCE LIST

Jump into cyberspace and check out other websites of interest. Perhaps your server can be a link on the web...

DOCUMENTATION

A hypertext version of the *Internet Connection Up and Running* book that you can read with your web browser.

http://fredsan/admin-bin/cfgin/initial

Figure 33. Internet Connection Secure Server Front Page

The server front page contains the following links:

- Configuration and Administration Forms
- Sample Home Page
- Webmaster Resource List
- Documentation

4.5.4.8 Changing the Default Home Page

The home page is the document that your server returns when a client sends a request that does not point to a specific directory or file. The server tries to serve the request from the document root directory, which is the directory you specified as your HTML directory during installation. The installation default is C:/WWW/HTML.

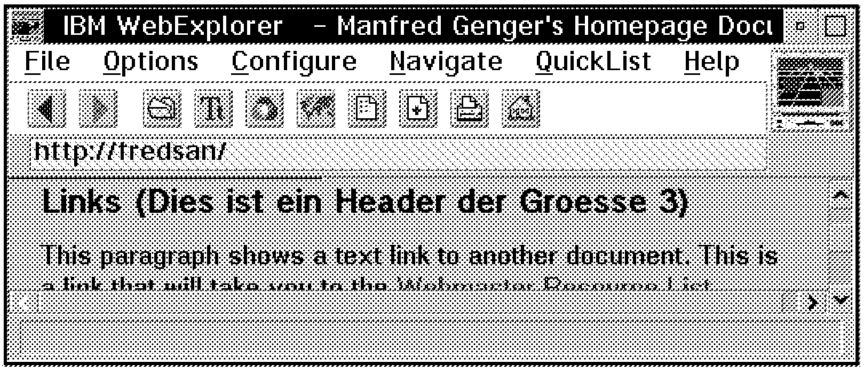
The default configuration file defines a list of four welcome pages. The order of the welcome pages is important because the server goes down the list from top to bottom when looking for the file it should return.

1. Welcome.html
2. welcome.html
3. index.html
4. Frntpage.html

We now take you through the steps that are involved in changing the default home page:

1. Make sure the server is running.
2. Go to the Internet Connection Secure Server Front Page.
3. Click on **Sample Home Page**.
4. Open the file with your browser editor.
5. Make some changes; for example, change ″This is a size 2 header″ into your local language.
6. Save the file with a name of Welcome.html and copy it to drive /www/html.
7. Make sure that the cache of your browser is disabled.
8. Repeat steps 1 to 3 and you should be able to see the changes. The results of our efforts are shown in Figure 34.

Be careful in changing the file. Use an editor which supports a file length greater than 256 characters a line or a specific HTML browser like HTML Wizard.



http://fredsan/

Links (Dies ist ein Header der Groesse 3)

This paragraph shows a text link to another document. This is a link that will take you to the Webmaster Resource List

Figure 34. New Default Home Page

4.5.5 Windows NT

In this section we tell you how to prepare and install the Internet Connection Server for Windows NT.

4.5.5.1 Check the Prerequisites

First of all, you should make sure you have everything you need to install Internet Connection Server for Windows NT.

You will find these requirements on the Internet Connection Family Web site at http://www.ics.raleigh.ibm.com, in the installation package′s READ.ME file, and in *Internet Connection Server for Windows NT Up and Running!*. We explain how to read and print these documents later in this subsection. The two first references contain late breaking information, so please read them carefully.

The hardware and software prerequisites are:

- A computer running Windows NT with TCP/IP configured
- 4 MB of free disk space on an HPFS or NTFS formatted drive
- A CD-ROM drive (only if your Internet Connection Server for Windows NT comes on a CD-ROM).

4.5.5.2 Plan for the Installation

- **Windows NT Server vs Workstation**

  You may install an Internet Connection Server for Windows NT on both a Windows NT server and a Windows NT workstation. We advise you to use a workstation for the following reasons:

  – A Windows NT workstation is much cheaper than a Windows NT server. Installing an Internet Connection Server for Windows NT on a Windows NT server that is not used as a server would thus be unnecessarily expensive.

  – A Windows NT server may be a sensible piece of your network when it comes to availability and security issues. However, a Web server exposes the machine and the network on which it is running to user mistakes or even sabotage. Installing an Internet Connection Server for Windows NT on a Windows NT server that is used as a server would thus be quite risky.

  Installing an Internet Connection Server for Windows NT on a Windows NT workstation is therefore the cheapest and the safest solution.

- **HPFS vs NTFS**

  You must install the Internet Connection Server for Windows NT on an HPFS or NTFS formatted drive. We advise you to use an NTFS formatted drive, because it will allow you to use the Windows NT security features.

- **Normal Application vs Windows NT Service**

  You may run the Internet Connection Server for Windows NT as a normal application or as a Windows NT service.

  To start the Internet Connection Server for Windows NT as a normal application, you have to log on as a user, start the server, and remain logged on as long as the server is to be up. As a consequence, the server is running with the authority of the user that started it. Furthermore, no other user may log on to the machine while the server is running.

  You start the Internet Connection Server for Windows NT as a Windows NT service through the Services Control Panel. As with all Windows NT services, you will be able to launch the server upon the startup of your system. The server will run in the background with its own authority, and users will be able to log on normally, without affecting the server′s operations. Only one instance of the Internet Connection Server for Windows NT may be run as a service, all other instances have to run as normal applications.

4.5.5.3 Read the Documentation

Since the Internet Connection Server for Windows NT is a rapidly evolving product, we advise you to read all the available information carefully.

Documentation covering the Internet Connection Server for Windows NT may be found in the following places:

- **The Internet Connection Family Web site**

  The Internet Connection Family Web site provides you with the prerequisites, updated installatio procedures and information about the latest code. The Internet Connection Family Web site can be found at http://www.ics.raleigh.ibm.com.

- **The installation package′s READ.ME file**

  You will find the latest information about your copy of the Internet Connection Server for Windows NT in the installation package′s READ.ME file.

- *Internet Connection Server for Windows NT Up and Running!*

  This is the Internet Connection Server for Windows NT manual. Read it! It contains all you ever wanted to know about your server.

  If you purchased your copy of the Internet Connection Server for Windows NT, then you received a hardcopy of *Internet Connection Server for Windows NT Up and Running!* with the software package.

  If you downloaded the Internet Connection Server for Windows NT from the Internet, then you will find a softcopy of this manual on the Internet Connection Family Web site at http://www.ics.raleigh.ibm.com, and within the server′s installation package. Unfortunately, it is not possible to read this last version of the manual prior to installation. We therefore advise you to print it from the Internet, whenever possible.

4.5.5.4 Get Your Copy of the Internet Connection Server for Windows NT

You have two possibilities to get a copy of the Internet Connection Server for Windows NT:

- Download it from the Internet
- Purchase it

4.5.5.5 Prepare Your IP Network

Connect the server′s host to your intranet or to the Internet. Enable the IP-based servers that you may need to maintain and operate your Internet Connection server remotely; disable all the others. Although FTP and Telnet are very useful, we do not advise you to enable them, because they could expose your server even more than it already is.

4.5.5.6 Installing your Internet Connection Server for Windows NT

- Visit the Internet Connection Family Web site at http://www.ics.raleigh.ibm.com. Read the latest news about the Internet Connection Server for Windows NT, check the prerequisites, and print the updated installation procedure.

- Get your copy of the Internet Connection Server for Windows NT: download or purchase it.

- If you downloaded your Internet Connection Server for Windows NT as a zip file from the Internet, unzip the installation files.

  Suppose you have downloaded the SERVER.ZIP file into the U:\DOWNLOAD directory, that your unzip program is V:\ZIP\UNZIP.EXE, and that you would like to unzip the installation files into the W:\INSTALL directory. Then do the following:

  – Open an MS-DOS window.

  – At the command prompt enter `w:`.

  – Enter `cd \install`.

  – Enter `v:\zip\unzip u:\download\server.zip`.

  - Enter `exit`.

- Read the READ.ME file.

  From now on, the installation files are located in a well-known directory. It is the W:\INSTALL directory described in the previous step, or, if the Internet Connection Server for Windows NT was shipped on diskettes or on a CD-ROM, it is W:\, where W is the name of the diskette or CD-ROM drive. We refer to it as W:\INSTALL for the remainder of this subsection.

  We used the Notepad application to edit the READ.ME file.

  - Within the File Manager select the **File** menu.
  - Select **Run**.
  - Enter `NOTEPAD W:\INSTALL\READ.ME` at the command line.
  - Select **Ok**.

- Install the Internet Connection Server for Windows NT.

  - Run the setup program:
    - Within the File Manager select the **File** menu.
    - Select **Run**.
    - Enter `W:\INSTALL\SETUP`.
    - Select **Ok**.
  - Read the Welcome window and click **Next**.
  - From the Select Components window:
    - Select the **IBM Internet Connection Server** check box if you want to run your Internet Connection Server for Windows NT as a normal application.
    - Select both the **IBM Internet Connection Server** and the **NT Service** check boxes to launch your Internet Connection Server for Windows NT as a Windows NT service.
  - From the Choose Target Directory window, do the following:
    - The default destination directory for the server is C:\WWW\. Click **Browse** if you want to change the drive or directory destination.
    - Select **Next**.
  - From the Select Component Directories window, do the following:
    - Review the destination subdirectories for the server components. These directories define where you want to install the Internet Connection Server components and where you want to store the resources you will be making available through the Internet Connection Server.
      - **Remote Administration directory** - The files used by the Internet Connection Server Configuration and Administration Forms are installed in this directory.
      - **Executables directory** - The Internet Connection Server executable program files and other related files are installed in this directory.
      - **CGI Bin scripts directory** - The directory where you want to put your script programs that use the Common Gateway Interface

(CGI); the Internet Connection Server htimage program is installed in this directory

- **Documentation directory** - The Internet Connection Server documentation files are installed in this directory.
- **HTML directory** - The directory where you want to put your HTML documents; the Internet Connection server sample HTML pages and the Internet Connection server front page are installed in this directory.
- **Icons and graphics directory** - The default directory list icons are installed in this directory. You may also choose to put your own icon and graphics files in this directory
- **Logs directory** - The directory where you want the Internet Connection server to put log files.

- If you have no changes to make, do not select any of the check boxes.
- If you want to change any of the default installation directories, select the corresponding check box.
- Click **Next**.
- If you opted to change any of the component directories, you will be prompted on additional windows for those directories.

– From the Select Program Folder window, do the following:

  - Optionally, change the default program folder name of Internet Connection Server by selecting one of the existing folders listed.
  - Click **Next**.

– From the Configuration Parameters window, do the following:

  - Review the default values for hostname, port, administrator ID and password.
    - **Host Name** - The default value is the host name defined as part of your TCP/IP configuration. If you want to use an alias, you can change this field to a fully qualified host name that is defined in your domain name server.
    - **Port** - The default value of 80 is the well-known port number for Hypertext Transfer Protocol (HTTP). Other port numbers less than 1024 are reserved for other TCP/IP applications. Port numbers 8080 and 8008 are commonly used for testing servers.
    - **Administrator ID** - The ID of your server administrator. Anyone attempting to use the Internet Connection Server Configuration and Administration Forms will be prompted to enter this ID. The default value is webadmin.
    - **Password** - This is the password you want to use to protect access to the Configuration and Administration forms. Anyone attempting to use the Internet Connection Server Configuration and Administration forms will be prompted to enter this password. The default value is webibm.
  - If you have no changes to make, do not select any of the check boxes.

        - If you want to change any of the default values, select the corresponding check box. We recommend that you change the default administrator ID and password now.
        - Click **Next**.
    - The server files will be installed and the server icons added.
- Read and print *Internet Connection Server for Windows NT Up and Running!*. To do so, we advise you to install a Web browser. Start browsing from the FRNTPAGE.HTML file, located in your server′s documentation directory. If you have kept all the defaults during the installation, this file is C:\WWW\DOCS\FRNTPAGE.HTML. From there, link to the online documentation.
- Configure your server. Please do not start your server before you have configured it. We explain how to configure your server later in this chapter.
- Start your server.
    - To start your server as a normal application:
        - Double-click on the **Internet Connection Server** icon in the Internet Connection group. You may also enter `whttpg` at the command prompt of a DOS window.
    - To start your server as a Windows NT Service:
        - Double-click the **Control Panel** icon from the Main program group.
        - Select the **Services** icon from the Control Panel program group.
        - From the Services panel, highlight the Internet Connection Server.
        - Click **Start** to start the service.

          Click **Stop** to stop the service.

          Click **Startup** to configure the service. To display the server′s graphical user interface the next time the server service is started, check the **Allow the Service to Interact with Desktop** check box. Please read the Windows NT manual for more information about Windows NT services.
    - To start another instance of the Internet Connection Server for Windows NT:
        - Open an MS-DOS window.
        - If you installed your Internet Connection Server for Windows NT as a normal application, enter `whttpd`.
        - If you installed your Internet Connection Server for Windows NT as a Windows NT service, enter `whttpd -noservice`.
- Uninstalling your server

  To remove the Internet Connection Server icon and product from your machine, select the **Uninstall Internet Connection Server** icon in the Internet Connection Server group.

4.6 Configuration

There are two ways to configure an IBM Internet Connection server:

- By editing the HTTPD configuration file
- By using the Configuration and Administration forms

4.6.1 Editing the HTTPD Configuration File

The behavior of every IBM Internet Connection server is controlled by its configuration file, whose name is HTTPD. (The extension of this file depends on the operating system.) It is possible to edit this file using standard editors on all the platforms for which the Internet Connection server is available. Editing the HTTPD file is probably the fastest and safest way to configure your server. It is fast because you can modify all the directives at once, rather than having to browse through numerous forms. It is safe because your server does not need to be online or even up while you configure it.

This method has two drawbacks however: it requires some understanding of the directives in the configuration file, and it is prone to syntax errors. (There is no spell or syntax checking.)

4.6.1.1 A Configuration Example

In this section we describe how we configured one of our IBM Internet Connection servers by editing the HTTPD file.

We have divided the configuration process into the following steps:

1. Identify and document the topology of your IP network.

   This step is important as it will help you to identify the resources located on your intranet. These resources include the hosts (clients and servers) and their peripherals that can be accessed from the IP network. In addition, your analysis will need to include your demilitarized zone(s), the Internet as well as your intranet.

   This information will allow you to place your server optimally in terms of accessibility, security and throughput.

2. Categorize the information you are serving and group the people accessing it.

   You should always be able to associate a security level and a group of people to the information you serve. This will allow you to serve everybody differently.

In our case we have identified the following categories of pages and groups of people.

Table 7. Categories of Pages and Groups of People

	Administrators	Project Members	Intranet	Internet
Public Pages	X	X	X	X
Local Pages	X	X	X	X
Test Pages	X	X		
Configuration and Administration Forms	X			

Categories of pages:

- Public pages contain the information we want to be available on the whole of the Internet. In our example the public pages are located in the C:\Internet\ directory.
- Private Pages contain the information we want to be available on our intranet only. In our example the private pages are located in the C:\Intranet\ directory.
- Project Pages are pages developed by the project members; only the project members should be able to read them from their host for security and confidentiality reasons. Project members also have access to their own pages. In our example the projects pages are located in the C:\Project\ directory, and the member directories are:

  ```
  C:\Andlauer\,
  C:\Ezvan\,
  C:\Genger\,
  C:\Kitazawa\, and
  C:\Murphy\.
  ```

- Configuration and Administration forms allow you to remotely configure and administer the server; only the administrators should be able to access them from a very limited number of hosts. In our example we have moved the Configuration and Administration forms from the default C:\WWW\Admin\ to the C:\Nothing\ directory.

Let us now tell the server where to find the pages that the browsers will request. Here is how we modified the HTTPD configuration file:

```
Exec /admin-bin/*           C:\Nothing\*
Exec /cgi-bin/*             C:\WWW\CGI-Bin\*
Pass /Project/*             C:\Project\*
Pass /Andlauer/*            C:\Andlauer\*
Pass /Ezvan/*               C:\Ezvan\*
Pass /Genger/*              C:\Genger\*
Pass /Kitazawa              C:\Kitazawa\*
Pass /Murphy/*              C:\Murphy\*
Pass /httpd-internal-icons/* C:\WWW\Icons\*
Pass /icons/*               C:\WWW\Icons\*
Pass /Internal/*            C:\Intranet\*
Pass /*                     C:\Internet\*
```

Figure 35. Exec and Pass Directives

The groups of people are:

- The Administrator - Responsible for the server operation and contents; he or she should therefore be able to access all the data hosted on the server.
- The Project Members - Develop pages and applications on the server; they have developed the pages hosted on the server, and are still busily working; they should have access to all their works to finish and correct them.
- The Intranet - The people located on the intranet should be able to read information hosted on the server that is internal to the service, as well as information that the service wants to advertise on the Internet.
- The Internet - The people located on the Internet should only be able to access the information hosted on the server that the service wants to advertise on the Internet (and nothing more).
- Apply the corresponding security

  We used the htadm tool to attribute passwords to the administrator and the project members, as below:

```
htadm -create  C:\Anything\Administrators.p
htadm -adduser C:\Anything\Administrators.p Laurel       Sun     Jean-Charles
htadm -adduser C:\Anything\Administrators.p Hardy        Mercury Manfred

htadm -create  C:\Anything\ProjectMembers.p
htadm -adduser C:\Anything\ProjectMembers.p Catherine    Pwd1    Catherine
htadm -adduser C:\Anything\ProjectMembers.p Eamon        Pwd2    Eamon
htadm -adduser C:\Anything\ProjectMembers.p Jean-Charles Pwd3    Jean-Charles
htadm -adduser C:\Anything\ProjectMembers.p Tsuyoshi     Pwd4    Tsuyoshi
htadm -adduser C:\Anything\ProjectMembers.p Manfred      Pwd5    Manfred
```

  Then we created the associated group files, as listed below:

```
Sheriff:  Laurel@(9.24.104.181,9.24.104.247)
Sidekick: Hardy@9.24.104.181
```

Figure 36. The Administrators.g Group File

```
UMurphy:    Eamon@9.24.104.18
UEzvan:     Catherine@9.24.104.27
UKitazawa:  Tsuyoshi@9.24.104.118
UAndlauer:  Jean-Charles@9.24.104.244
UGenger:    Manfred@9.24.104.245
GResidents: UMurphy,UEzvan,UKitazawa,UAndlauer,UGenger
GProject:   UMurphy,GResidents
```

Figure 37. The ProjectMembers.g Group File

Note that we could have placed the passwords and the groups in one single password file and one single group file. But the method we use compartments the users and the group names, so that it is impossible to grant an administrator privilege to a project member unless the wrong password and group files are used.

Furthermore, notice that the team members may access their pages from their computer only.

Let us now restrict access to the files to the authorized users. We added the following lines to the HTTPD configuration file:

- For the administrators:

```
Protection  Administration {
      GroupFile    C:\Anything\Administrators.g
      PasswdFile   C:\Anything\Administrators.p
      Mask         Users
      PostMask     Users
      PutMask      Users
      GetMask      Users
      AuthType     Basic
      ServerID     Server_Administration
}
Protect  /admin-bin/*  Administration
```

- For the project members:

```
Protection  Project-Prot {
      GroupFile    C:\Anything\ProjectMembers.g
      PasswdFile   C:\Anything\ProjectMembers.p
      Mask         GProject
      PostMask     GProject
      PutMask      GProject
      GetMask      GProject
      AuthType     GProject
      ServerID     Project_Member
}
Protect  /Project/*  Project-Prot
```

- For one resident (here Catherine Ezvan):

```
Protection  Catherine-Prot {
      GroupFile    C:\Anything\ProjectMembers.g
      PasswdFile   C:\Anything\ProjectMembers.p
      Mask         UEzvan,UMurphy,UAndlauer
      PostMask     UEzvan,UMurphy,UAndlauer
      PutMask      UEzvan,UMurphy,UAndlauer
      GetMask      UEzvan,UMurphy,UAndlauer
      AuthType     UEzvan,UMurphy,UAndlauer
      ServerID     Jardin_Secret
}
Protect  /Ezvan/*  Catherine-Prot
```

- For a user on the intranet:

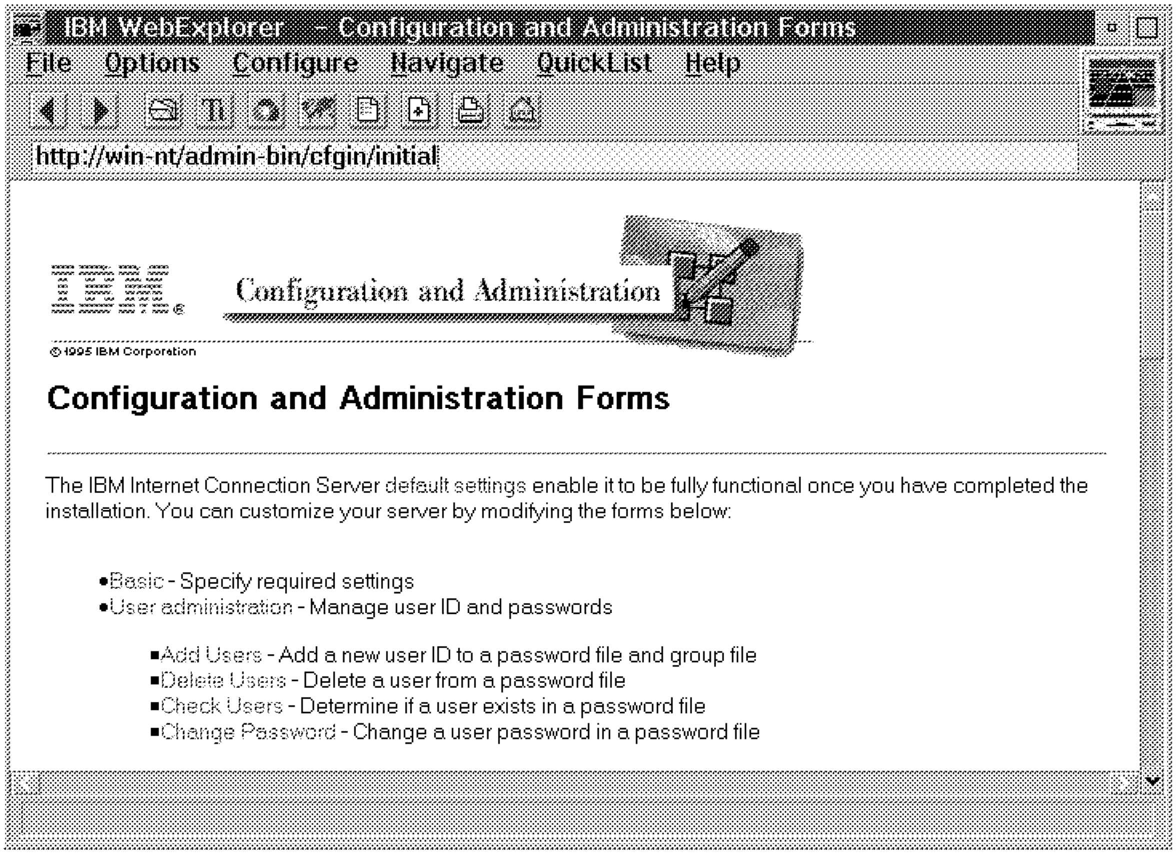
```
Protection  Intranet-Prot {
      Mask         @9.24.104.*
      PostMask     @9.24.104.*
      PutMask      @9.24.104.*
      GetMask      @9.24.104.*
      AuthType     @9.24.104.*
      ServerID     Internal_Services
}
Protect  /Internal/*  Intranet-Prot
```

Note that it is possible to group users defined in the password file in the group file, and both users defined in the password file and groups defined in the group file in the HTTPD configuration file. This basically means that there are many ways to achieve the same protection.

Also note that there is no protection by default. Therefore, if you set the Pass or Exec attribute for a file or directory, then anybody may request these files unless you define and set a protection. We thus recommend that you define, set and test the protection first without files. Then create the directory, place the files on the server and set the corresponding Pass and Exec directives.

It is possible to achieve better access control using Access Control List (ACL files). Please read the *Up and Running!* manual for more details about them.

4.6.2 Using the Configuration and Administration Forms


IBM WebExplorer - Configuration and Administration Forms
File
Options
Configure
Navigate
QuickList
Help
IBM
Configuration and Administration
©1995 IBM Corporation

http://win-nt/admin-bin/cfgin/initial

Configuration and Administration Forms

The IBM Internet Connection Server default settings enable it to be fully functional once you have completed the installation. You can customize your server by modifying the forms below:

- Basic - Specify required settings
- User administration - Manage user ID and passwords
  - Add Users - Add a new user ID to a password file and group file
  - Delete Users - Delete a user from a password file
  - Check Users - Determine if a user exists in a password file
  - Change Password - Change a user password in a password file

Figure 38. The Configuration and Administration Forms Entry Point

The IBM Internet Connection servers are shipped with Configuration and Administration forms on every platform. These forms are contained in HTML pages which call CGI scripts which in turn update the HTTPD configuration file.

The forms may be used from any form-enabled Web browser: IBM WebExplorer or Netscape Navigator, for example. By definition, access to the forms may be local or remote through the underlying IP network. You may restrict access to these pages to a limited number of hosts (IP address or hostname) and/or to a limited number of people (passwords). We describe how this is done later in this section.

Using Configuration and Administration forms is very convenient because you do not need to be at the server′s location to configure it. It also avoids having to install an additional FTP server alongside the Web server in order to update the configuration file remotely, which would be an additional security exposure. Furthermore, the forms are well documented, guide you throughout the configuration process, and tell you if the changes you have made have been processed successfully or not.

But this method of customizing also has some drawbacks. The first one is that it requires the Internet Connection server to be running. This means that the server has to be running before the security can be configured using the forms. We recommend that you never run a server before having set the security to the desired level.

Furthermore, this method potentially exposes your server more than you might like to. For example, we have found IBM Internet Connection servers where it was possible to access the Configuration and Administration forms from anywhere on the Internet with the default user ID and password! We have of course informed their owners of this security exposure. Never forget that the Internet contains people who own the same server software as you do, and some of them may know it better than you do.

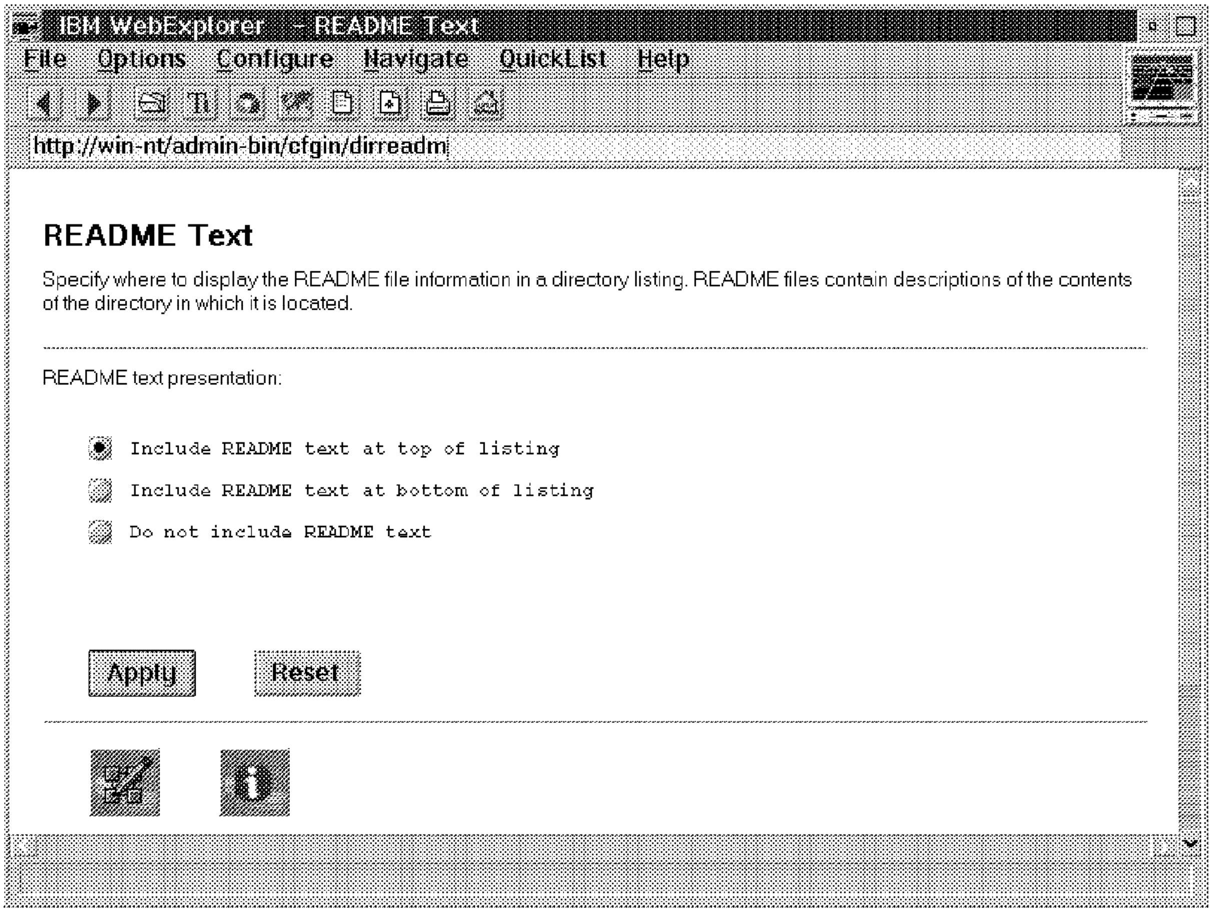
Another drawback of the Configuration and Administration forms is that they only allow you to modify the main server, that is the server whose configuration, name and location are the default ones. In the case of Windows NT this file is C:\WINNT35\httpd.cnf.

In conclusion, the forms are well-suited to making small punctual configuration changes, but not so well-suited for larger ones.

4.6.2.1 Example

Here is an example of how to use the Configuration and Administration forms. Before you use these forms make sure that the server′s and browser′s caching options are disabled. If you do not disable them, then the forms your browser displays may not reflect the server′s current configuration.

First of all, let us change the README Text position, using the form shown in the figure below:



IBM WebExplorer - README Text
File Options Configure Navigate QuickList Help
http://win-nt/admin-bin/cfgin/dirreadm

README Text

Specify where to display the README file information in a directory listing. README files contain descriptions of the contents of the directory in which it is located.

README text presentation:

● Include README text at top of listing

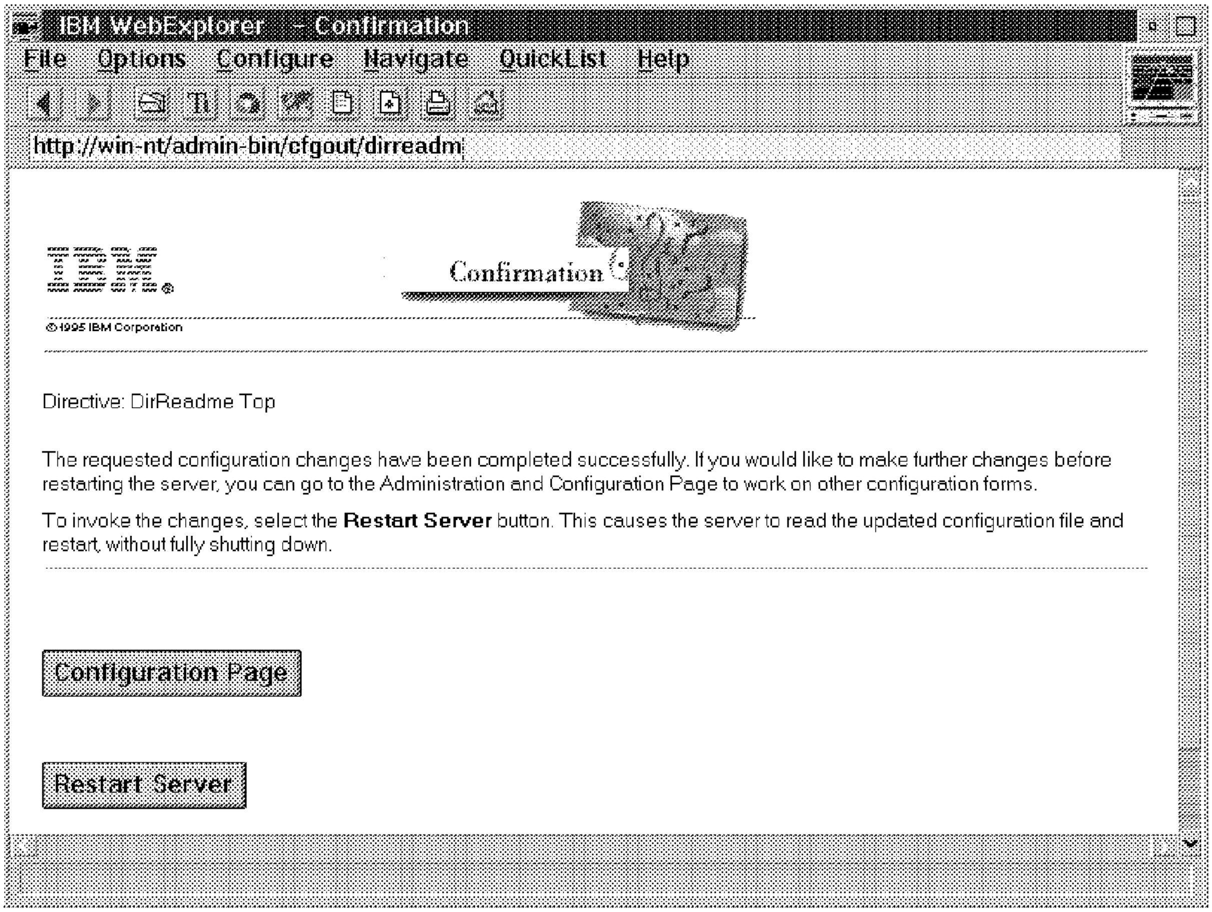
○ Include README text at bottom of listing

○ Do not include README text

Figure 39. Example of a Form

To do so, load the page, fill out the form according to your wishes, and select **Apply** to apply the changes. Select **Reset** to reset the form according to your server′s current settings.

If your changes were successfully applied, then you will be taken to a successful confirmation page as shown below.


IBM WebExplorer - Confirmation
File Options Configure Navigate QuickList Help
http://win-nt/admin-bin/cfgout/dirreadm
IBM
©1995 IBM Corporation
Confirmation
Configuration Page
Restart Server

Directive: DirReadme Top

The requested configuration changes have been completed successfully. If you would like to make further changes before restarting the server, you can go to the Administration and Configuration Page to work on other configuration forms.

To invoke the changes, select the **Restart Server** button. This causes the server to read the updated configuration file and restart, without fully shutting down.

Figure 40. A Successful Confirmation Page

Select **Restart** to restart your server and activate the changes you just made, or **Configuration Page** to make further changes.

Note that if you do not restart your server, then the forms you will load may not reflect the current server settings.

Furthermore, some configuration changes require that you reboot the server before they will take effect, as is the case for the reception of a certificate (described later).

If the changes could not be made, then the server will send you a negative confirmation page. In this case, correct the error and retry the operation.

4.6.3 How We Configured Our Server

We have made it our practice when configuring our servers to edit the configuration files. Our justification for doing this is based upon the good security reasons which we described earlier. For the servers on which we did not delete the Configuration and Administration forms we did the following: (We explain how it is done later in this section.)

- Changed the default user ID and password to access the forms
- Restricted the hosts from which it is possible to access the forms

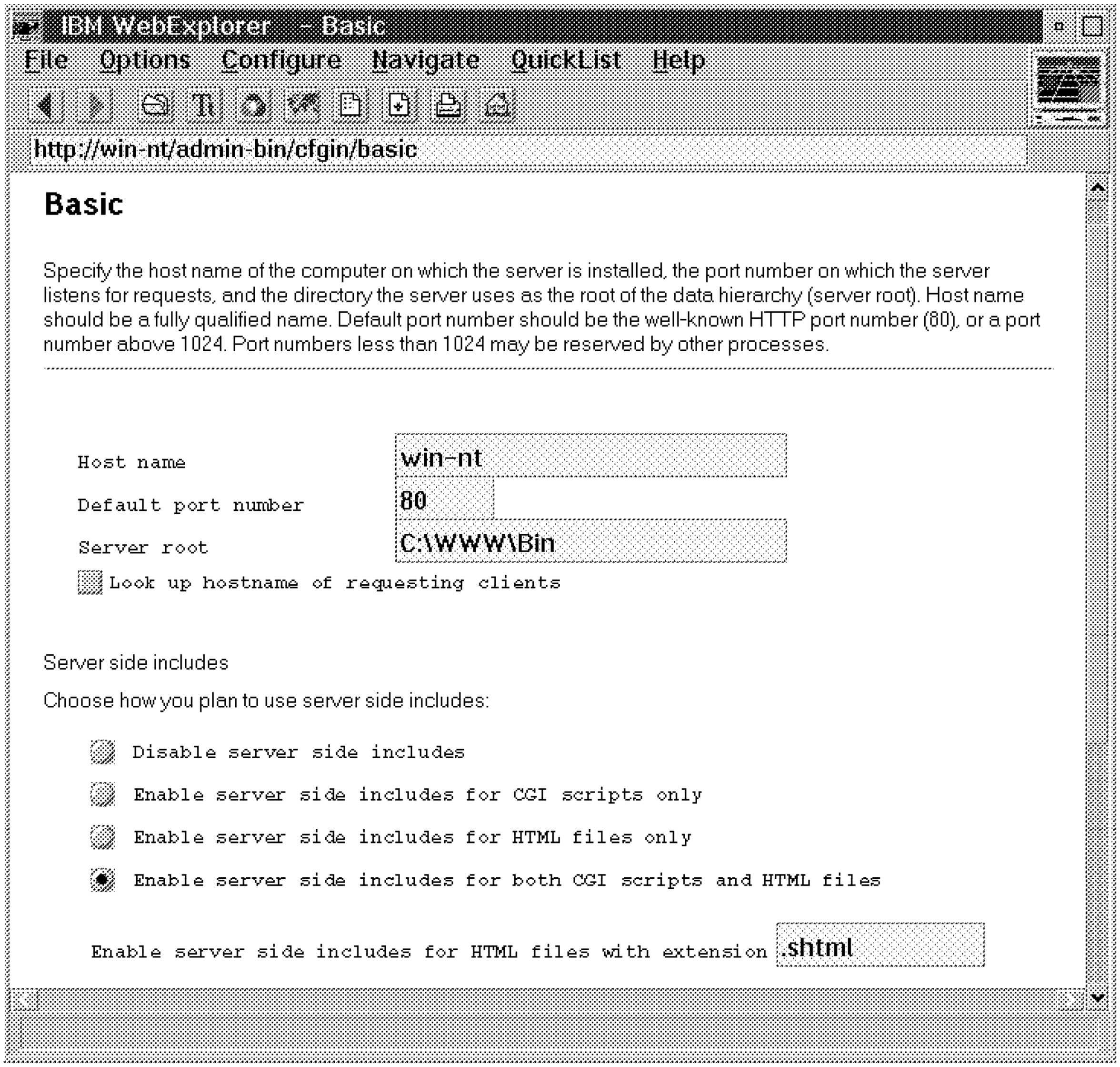
- Changed the forms′ URL

We learned a great deal about the HTTPD configuration file directives from using the forms. We advise you to use them if you are not familiar with the Internet Connection servers.

In the next subsections we describe the most important directives contained in the HTTPD configuration file of an IBM Internet Connection server. We first show the Configuration and Administration form and, where appropriate, show the corresponding default directives and examples of how these directives may be modified. Note that, in most cases, the directive has explanatory notes associated with it in the HTTPD configuration file.

4.6.3.1 Basic

Here is the Basic form.



http://win-nt/admin-bin/cfgin/basic

Basic

Specify the host name of the computer on which the server is installed, the port number on which the server listens for requests, and the directory the server uses as the root of the data hierarchy (server root). Host name should be a fully qualified name. Default port number should be the well-known HTTP port number (80), or a port number above 1024. Port numbers less than 1024 may be reserved by other processes.

Host name: win-nt
Default port number: 80
Server root: C:\WWW\Bin
☐ Look up hostname of requesting clients

Server side includes

Choose how you plan to use server side includes:

○ Disable server side includes
○ Enable server side includes for CGI scripts only
○ Enable server side includes for HTML files only
● Enable server side includes for both CGI scripts and HTML files

Enable server side includes for HTML files with extension: .shtml

Figure 41. The Basic Form

As you can see in Figure 42 on page 106, the directives have explanatory comments associated with them.

For more information regarding the use of server-side includes and the Common Gateway Interface (CGI), please refer to 5.9.4, “Server-Side Includes - Gods of Greek Mythology” on page 209 and 6.1.5, “Linking to a CGI Script - Calendar” on page 228, respectively.

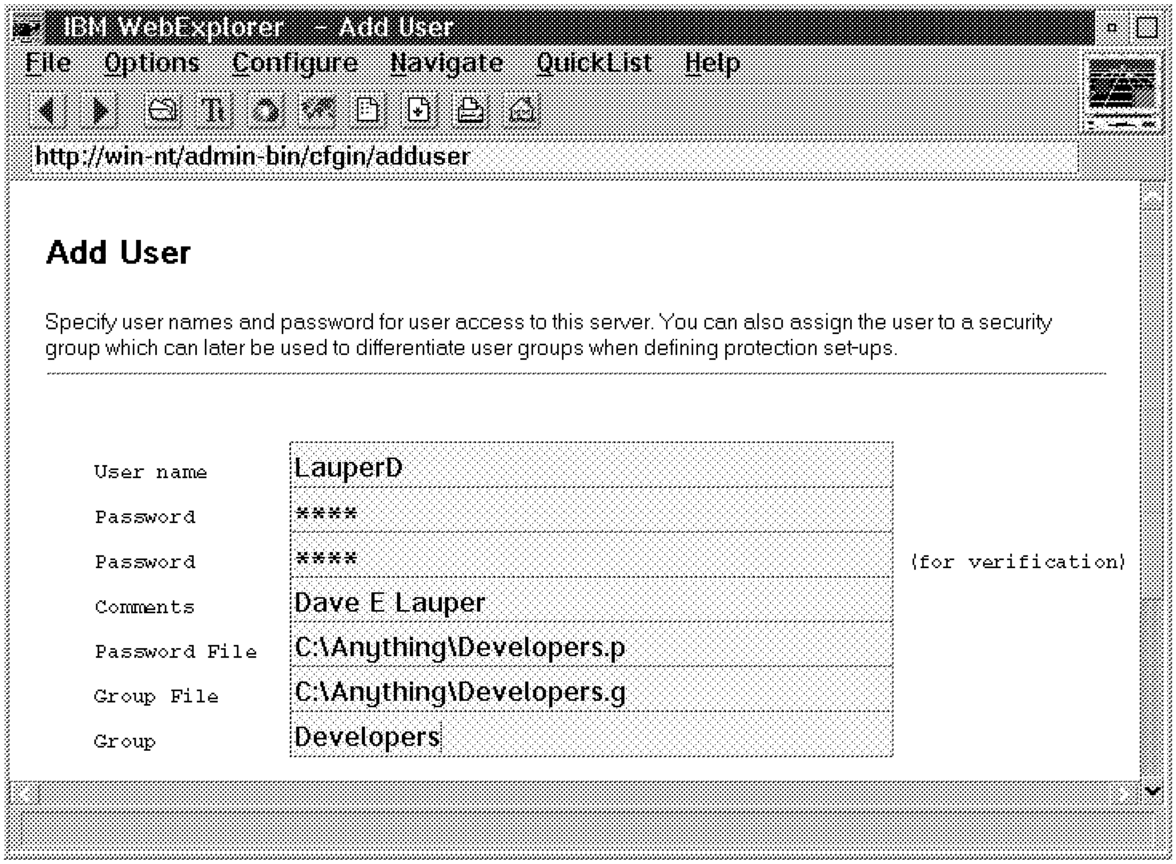
Here are the default directives that correspond to the Basic form.

```
#
#       Set ServerRoot to point to the directory where you unpacked this
#       distribution, or wherever you want your server to have its home.
#       Default:  <none>
#       Syntax:  ServerRoot    <path, including drive letter>
#       There can only be one value per keyword.
#
ServerRoot C:\WWW\Bin

#
#        Specify the fully-qualified hostname, including the domain.  You can
#        use an alias (if you have one set up) instead of the machine′s real name
#        name so that clients will not be able to see the real host name.
#        Default:  <none>
#        Syntax:  Hostname   <fully-qualified host name>
#
Hostname win-nt
#
#       Don′t lookup hostnames of clients
#
DNS-Lookup     Off

#
#       If you are not root you have to
#       use a port above 1024; good defaults are 8000, 8001, 8080
#       Default value for HTTP:  80
#       Syntax:  Port  <number>
#       There can only be one value per keyword.
#
Port          80

#
#  imbeds:
#    Use this directive to enable server side includes.
#    Syntax:  imbeds   <on/off/files/cgi><.suffix>
#             <.suffix> is optional - if present, it limits which files are
#             parsed for server side includes.
#    Default: on .shtml
#    If more than one imbeds directive is specified, the last one is used
#
imbeds on .shtml

#
#     The following are vendor-specific CAD-formats commonly
#     used at CERN and in HEP institutes:
#
AddType  .htmls    text/x-ssi-html               8bit    1.0 # Server-side includes
AddType  .shtml    text/x-ssi-html               8bit    1.0 # Server-side includes
```

Figure 42. The Directives Corresponding to the Basic Form

4.6.3.2 User Administration

Here is the Add User form.



http://win-nt/admin-bin/cfgin/adduser

Add User

Specify user names and password for user access to this server. You can also assign the user to a security group which can later be used to differentiate user groups when defining protection set-ups.

User name	LauperD	
Password	****	
Password	****	(for verification)
Comments	Dave E Lauper	
Password File	C:\Anything\Developers.p	
Group File	C:\Anything\Developers.g	
Group	Developers	

Figure 43. The Add User Form

The Add User form does not correspond to directives of the HTTPD configuration file. Rather, it interfaces to the HTADM administrative tool. The syntax of HTADM is as follows:

```
Administrative tool for IBM Internet Connection server access authorization

Usage:
        htadm -adduser pwfile ]user ]password ]real name[[[
        htadm -deluser pwfile ]user[
        htadm -passwd  pwfile ]user ]password[[
        htadm -check   pwfile ]user ]password[[
        htadm -create  pwfile
```

Figure 44. The Syntax of the HTADM Command

To create the same user as we did with the Add User form using HTADM, do the following:

- Create the C:\Anything directory. Note that the Add User form cannot create directories. Therefore, the group and password files it creates can only be placed in existing directories.

- Create the C:\Anything\Developers.p password file by running:

  ```
  htadm -create C:\Anything\Developers.p
  ```

- Add a password for Dave E Lauper by running:

  ```
  htdam -adduser C:\Anything\Developers.p LauperD Dave E Lauper
  ```

- Add LauperD to the Developers group of the C:\Anything\Developers.g group file as listed below:

```
Developers: LauperD
```

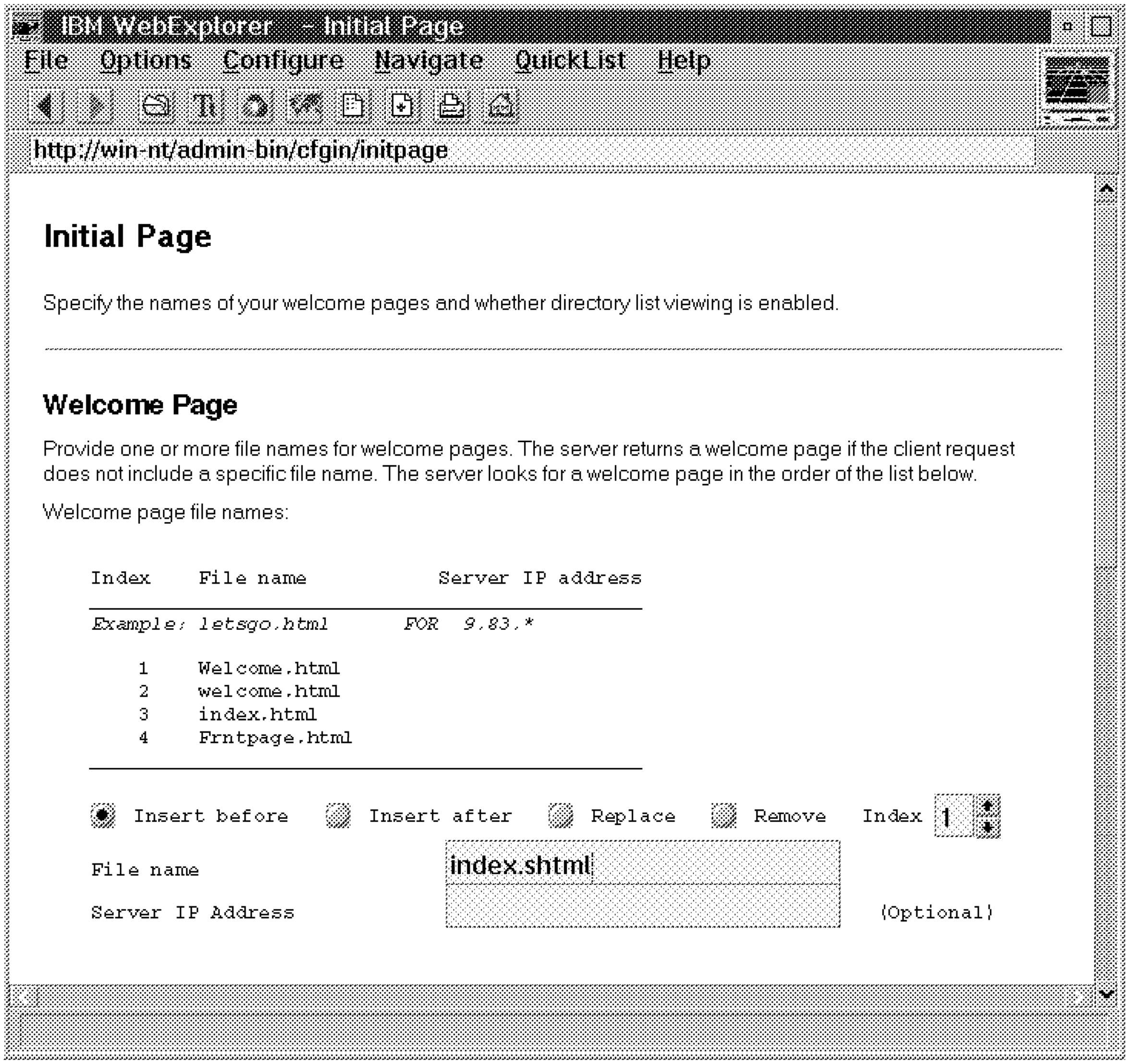
Figure 45. The C:\Anything\Developers.g Group File

The other user administration forms and uses of the HTADM tool are very similar to the above example. Please refer to the *Internet Connection Server Up and Running!* manual for further details.

We advise you to give inconspicuous names and extensions to your group and password files, and to place them in discreet directories, as we did in our example. This will prevent unauthorized people, who may have (in one way or another) gained access to the server, from finding the files too easily.

4.6.3.3 Initial Page - Welcome Page

Here is the Initial Page - Welcome Page form.



http://win-nt/admin-bin/cfgin/initpage

Initial Page

Specify the names of your welcome pages and whether directory list viewing is enabled.

Welcome Page

Provide one or more file names for welcome pages. The server returns a welcome page if the client request does not include a specific file name. The server looks for a welcome page in the order of the list below.

Welcome page file names:

Index	File name	Server IP address
Example:	*letsgo.html*	*FOR 9.83.**
1	Welcome.html	
2	welcome.html	
3	index.html	
4	Frntpage.html	

● Insert before ○ Insert after ○ Replace ○ Remove Index 1

File name index.shtml

Server IP Address (Optional)

Figure 46. The Initial Page - Welcome Page Form

As you can see from the form, we are adding a directive to change the order in which the server will look for and present a welcome page.

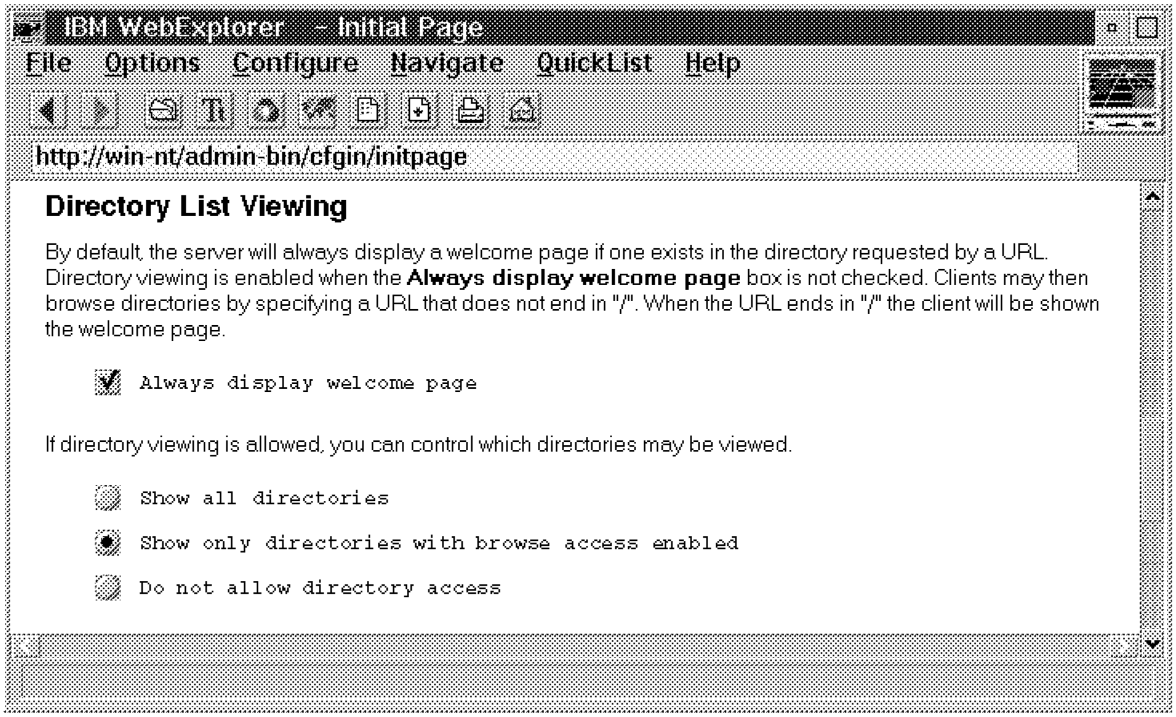
Here are the default directives that correspond to the Initial Page - Welcome Page form.

```
#
#        Specify the default document to be displayed to the client
#        when only a directory name is specified in the URL.
#        The first Welcome statement has precedence.
#        Defaults:  Welcome.html, welcome.html, index.html
#        Syntax:  Welcome <filename.html>
#        This directive may be defined multiple times in the
#        configuration file.
#
Welcome    Welcome.html
Welcome    welcome.html
Welcome    index.html
Welcome    Frntpage.html
```

Figure 47. The Directives Corresponding to the Initial Page - Welcome Page Form

4.6.3.4 Initial Page - Directory List Viewing Form

Here is the Initial Page - Directory List Viewing form.


IBM WebExplorer - Initial Page
File Options Configure Navigate QuickList Help

http://win-nt/admin-bin/cfgin/initpage

Directory List Viewing

By default, the server will always display a welcome page if one exists in the directory requested by a URL. Directory viewing is enabled when the **Always display welcome page** box is not checked. Clients may then browse directories by specifying a URL that does not end in "/". When the URL ends in "/" the client will be shown the welcome page.

☑ Always display welcome page

If directory viewing is allowed, you can control which directories may be viewed.

○ Show all directories

● Show only directories with browse access enabled

○ Do not allow directory access

Figure 48. The Initial Page - Directory List Viewing Form

We advise you to always display a welcome page. This will prevent the display of the contents of your server′s directories. A directory listing is not aesthetically pleasing, and Web users should only encounter them by choice (selecting a hyperlink, for example). Furthermore, this information reveals the structure of your server to the Internet. This information may be useful, especially to competitors and hackers.

For the same reason, we advise you not to allow directory access (DirAccess set to off). If you really need to serve files in an FTP-like style, then only show directories with browse access enabled (DirAccess set to Selective). This will hide the contents of the directories in which you did not place the .www_browsable file.

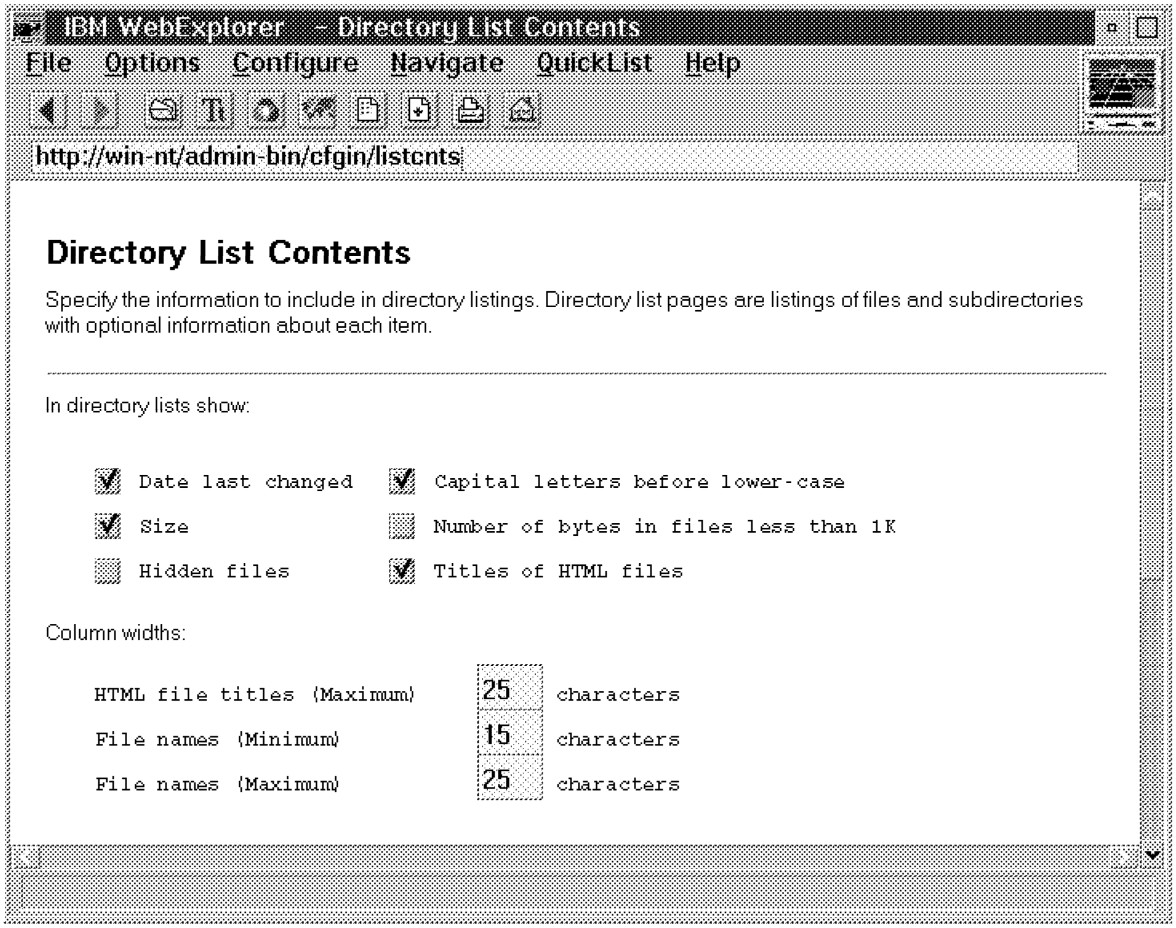
Here are the default directives that correspond to the Initial Page - Directory List Viewing form.

```
#
#         Indicate if the absence of a trailing slash in the URL will
#         provide a directory listing or the default welcome page.
#         Default:  On
#         Syntax:  AlwaysWelcome <on/off>
#
AlwaysWelcome  On

#
#          Enable/disable or selective directory browsing
#          Default:  On
#          Syntax:  DirAccess <on/off/selective>
#          One value per keyword is allowed.  One keyword/value pair is
#          allowed.
DirAccess   On
```

Figure 49. The Directives Corresponding to the Initial Page - Directory List Viewing Form

4.6.3.5 Directory List Contents

Here is the Directory List Contents form.


IBM WebExplorer - Directory List Contents
File Options Configure Navigate QuickList Help

http://win-nt/admin-bin/cfgin/listcnts

Directory List Contents

Specify the information to include in directory listings. Directory list pages are listings of files and subdirectories with optional information about each item.

In directory lists show:

☑ Date last changed ☑ Capital letters before lower-case

☑ Size □ Number of bytes in files less than 1K

□ Hidden files ☑ Titles of HTML files

Column widths:

HTML file titles (Maximum)	25	characters
File names (Minimum)	15	characters
File names (Maximum)	25	characters

Figure 50. The Directory List Contents Form

We advise you not to display hidden files.

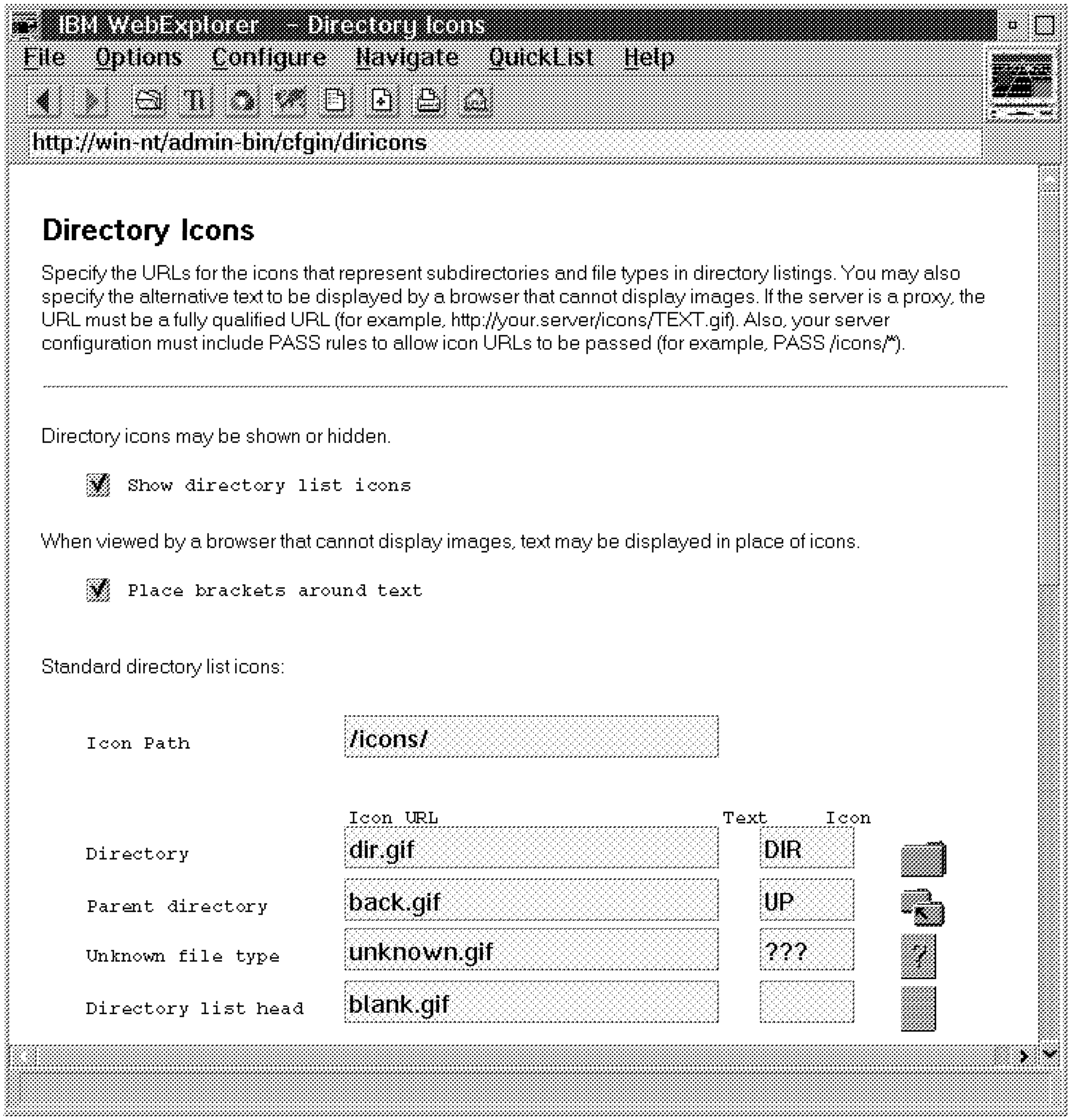
Here are the default directives that correspond to the Directory List Contents form.

```
#
#       Control the appearance of the directory listing.
#       Defaults:  DirShowIcons        On
#                  DirShowDate         On
#                  DirShowSize         On
#                  DirShowDescription  On
#                  DirShowBrackets     On
#                  DirShowCase         On
#                  DirShowHidden       On
#                  DirShowBytes        Off
#       Syntax:  <directive> <on/off>
#       Only one value per keyword is allowed.  Only one keyword/value
#       pair is allowed.
#
DirShowDate         On
DirShowSize         On
DirShowDescription  On
DirShowCase         On
DirShowHidden       On
DirShowBytes        Off

#
#       Specify the maximum width for the description text in
#       directory listings.
#       Default:  DirShowMaxDescrLength 25
#       Syntax:  DirShowMaxDescrLength  <num>
#       Only one value per keyword is allowed.  Only one keyword/value
#       pair is allowed.
#
DirShowMaxDescrLength  25

#
#       Specify the minimum width for the filename field for
#       directory listings.
#       Default:  DirShowMinLength  15
#       Syntax:  DirShowMinLength  <num>
#       Only one value per keyword is allowed.  Only one keyword/value
#       pair is allowed.
DirShowMinLength  15

#
#       Specify the maximum width for the filename field for
#       directory listings.
#       Default:  DirShowMaxLength  25
#       Syntax:  DirShowMaxLength  <num>
#       Only one value per keyword is allowed.  Only one keyword/value
#       pair is allowed.
#
DirShowMaxLength  25
```

Figure 51. The Directives Corresponding to the Directory List Contents Form

4.6.3.6 Directory Icons - Directory

Here is the Directory Icons - Directory form.



IBM WebExplorer - Directory Icons
File Options Configure Navigate QuickList Help

http://win-nt/admin-bin/cfgin/diricons

Directory Icons

Specify the URLs for the icons that represent subdirectories and file types in directory listings. You may also specify the alternative text to be displayed by a browser that cannot display images. If the server is a proxy, the URL must be a fully qualified URL (for example, http://your.server/icons/TEXT.gif). Also, your server configuration must include PASS rules to allow icon URLs to be passed (for example, PASS /icons/*).

Directory icons may be shown or hidden.

☑ Show directory list icons

When viewed by a browser that cannot display images, text may be displayed in place of icons.

☑ Place brackets around text

Standard directory list icons:

Icon Path: /icons/

	Icon URL	Text
Directory	dir.gif	DIR
Parent directory	back.gif	UP
Unknown file type	unknown.gif	???
Directory list head	blank.gif	

Figure 52. The Directory Icons - Directory Form

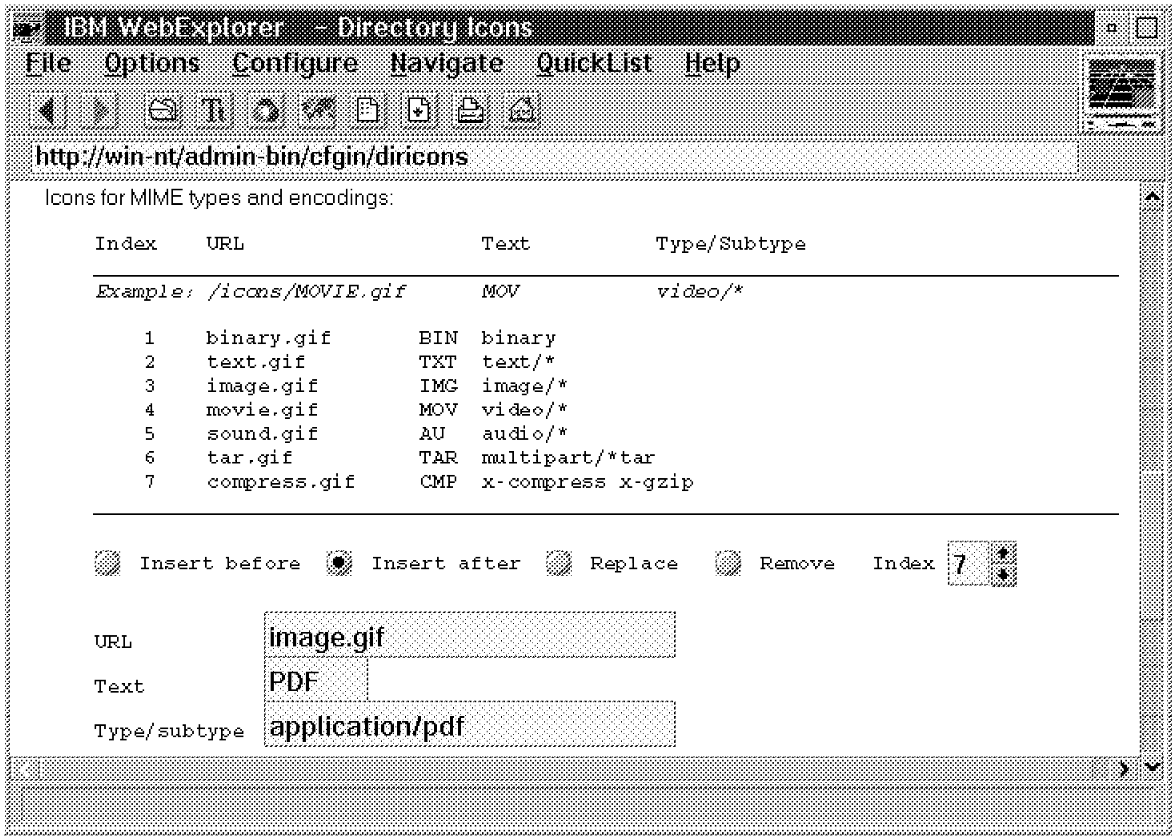
Here are the default directives that correspond to the Directory Icons - Directory form.

```
#
#       Control the appearance of the directory listing.
#       Defaults:  DirShowIcons        On
#                  DirShowDate         On
#                  DirShowSize         On
#                  DirShowDescription  On
#                  DirShowBrackets     On
#                  DirShowCase         On
#                  DirShowHidden       On
#                  DirShowBytes        Off
#       Syntax:  <directive> <on/off>
#       Only one value per keyword is allowed.  Only one keyword/value
#       pair is allowed.
#
DirShowIcons        On
DirShowBrackets     On

#
#       Specify path for the standard icons included in directory listings
#       Default: IconPath <ServerRoot>/icons/*
#       Syntax:  IconPath <icon directory path>
#
IconPath /icons/

#
#       Specify blank icon URL for directory listing
#       Default:  default shown below
#       Syntax:  AddBlankIcon <icon URL><ALT text>
#
#       Specify directory icon URL for directory listing
#       Default:  default shown below
#       Syntax:  AddDirIcon <icon URL><ALT text>
#
#       Specify parent directory icon URL for directory listing
#       Default:  default shown below
#       Syntax:  AddParentIcon <icon URL><ALT text>
#
#       Specify unknown icon URL for directory listing
#       Default:  default shown below
#       Syntax:  AddUnknownIcon <icon URL><ALT text>
#
AddBlankIcon   blank.gif
AddDirIcon     dir.gif          DIR
AddParentIcon  back.gif         UP
AddUnknownIcon unknown.gif      ???
```

Figure 53. The Directives Corresponding to the Directory Icons - Directory Form

4.6.3.7 Directory Icons - MIME Types

Here is the Directory Icons - MIME Types form.



http://win-nt/admin-bin/cfgin/diricons

Icons for MIME types and encodings:

```
Index   URL                     Text          Type/Subtype
---------------------------------------------------------------
Example: /icons/MOVIE.gif       MOV           video/*

    1   binary.gif       BIN   binary
    2   text.gif         TXT   text/*
    3   image.gif        IMG   image/*
    4   movie.gif        MOV   video/*
    5   sound.gif        AU    audio/*
    6   tar.gif          TAR   multipart/*tar
    7   compress.gif     CMP   x-compress x-gzip
---------------------------------------------------------------
```

○ Insert before ● Insert after ○ Replace ○ Remove Index 7

URL image.gif

Text PDF

Type/subtype application/pdf

Figure 54. The Directory Icons - MIME Types Form

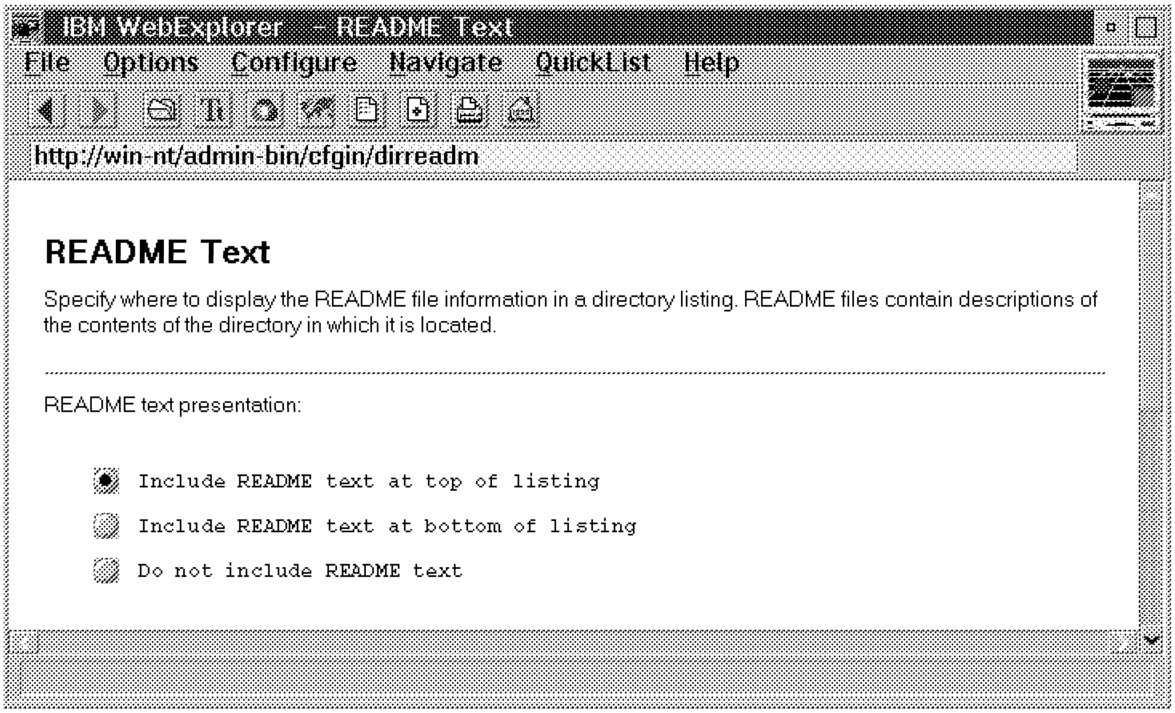
Here are the default directives that correspond to the Directory Icons - MIME Types form.

```
#
#       Bind icon URL to a MIME content-type or content-encoding
#       Default:  default set of icons shown below
#       Syntax:  AddIcon <icon URL><ALT text><template>
#
AddIcon        binary.gif       BIN  binary
AddIcon        text.gif         TXT  text/*
AddIcon        image.gif        IMG  image/*
AddIcon        movie.gif        MOV  video/*
AddIcon        sound.gif        AU   audio/*
AddIcon        tar.gif          TAR  multipart/*tar
AddIcon        compress.gif     CMP  x-compress x-gzip
```

Figure 55. The Directives Corresponding to the Directory Icons - MIME Types Form

4.6.3.8 README Text

Here is the README Text form.


IBM WebExplorer - README Text
File Options Configure Navigate QuickList Help
http://win-nt/admin-bin/cfgin/dirreadm

README Text

Specify where to display the README file information in a directory listing. README files contain descriptions of the contents of the directory in which it is located.

README text presentation:

● Include README text at top of listing

○ Include README text at bottom of listing

○ Do not include README text

Figure 56. The README Text Form

We strongly advise you to place README files wherever users will encounter directory listings on your server. These files are informative to the user and will save them time.

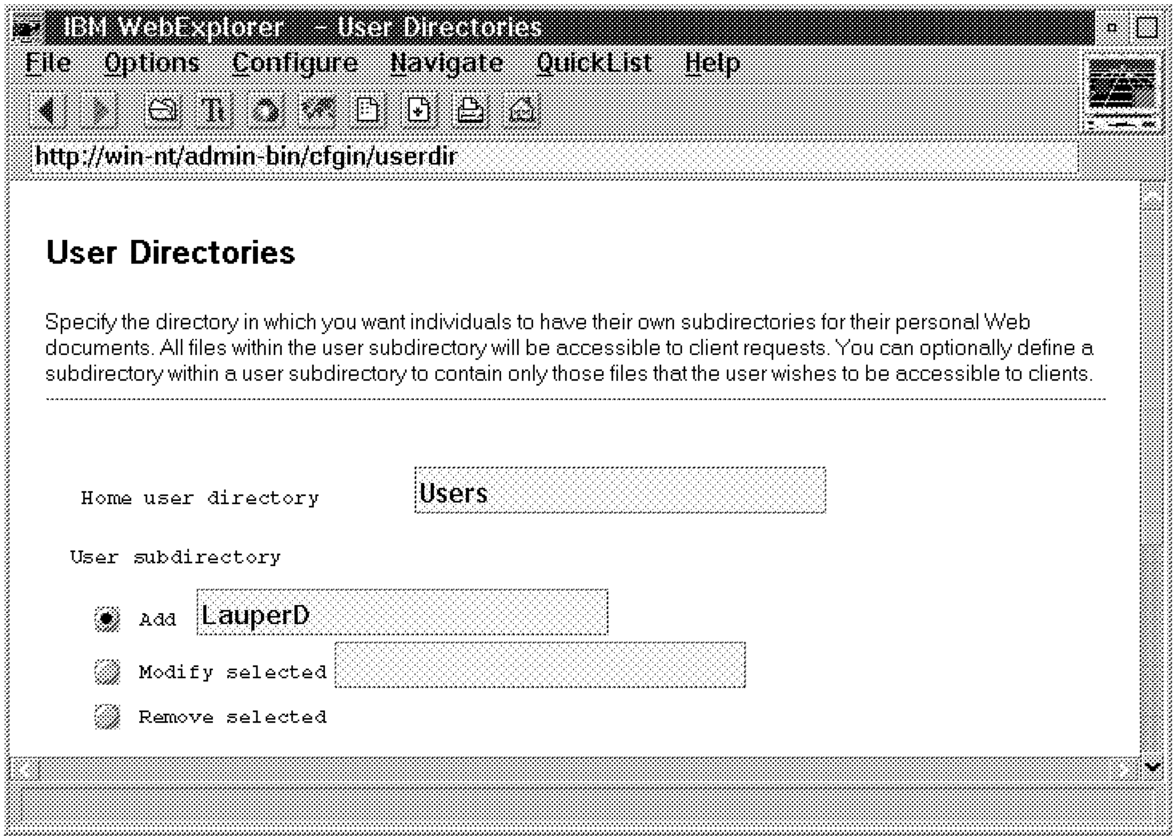
Here are the default directives that correspond to the README Text form.

```
#
#       Configure/disable readme feature for directory browsing.
#       Default:  top
#       Syntax:  DirReadme <top/bottom/off>
#       One value per keyword is allowed.  One keyword/value pair is
#       allowed.
DirReadme  top
```

Figure 57. The Directives Corresponding to the README Text Form

4.6.3.9 User Directories

Here is the User Directories form.


IBM WebExplorer - User Directories
File Options Configure Navigate QuickList Help

http://win-nt/admin-bin/cfgin/userdir

User Directories

Specify the directory in which you want individuals to have their own subdirectories for their personal Web documents. All files within the user subdirectory will be accessible to client requests. You can optionally define a subdirectory within a user subdirectory to contain only those files that the user wishes to be accessible to clients.

Home user directory: Users

User subdirectory

● Add: LauperD

○ Modify selected

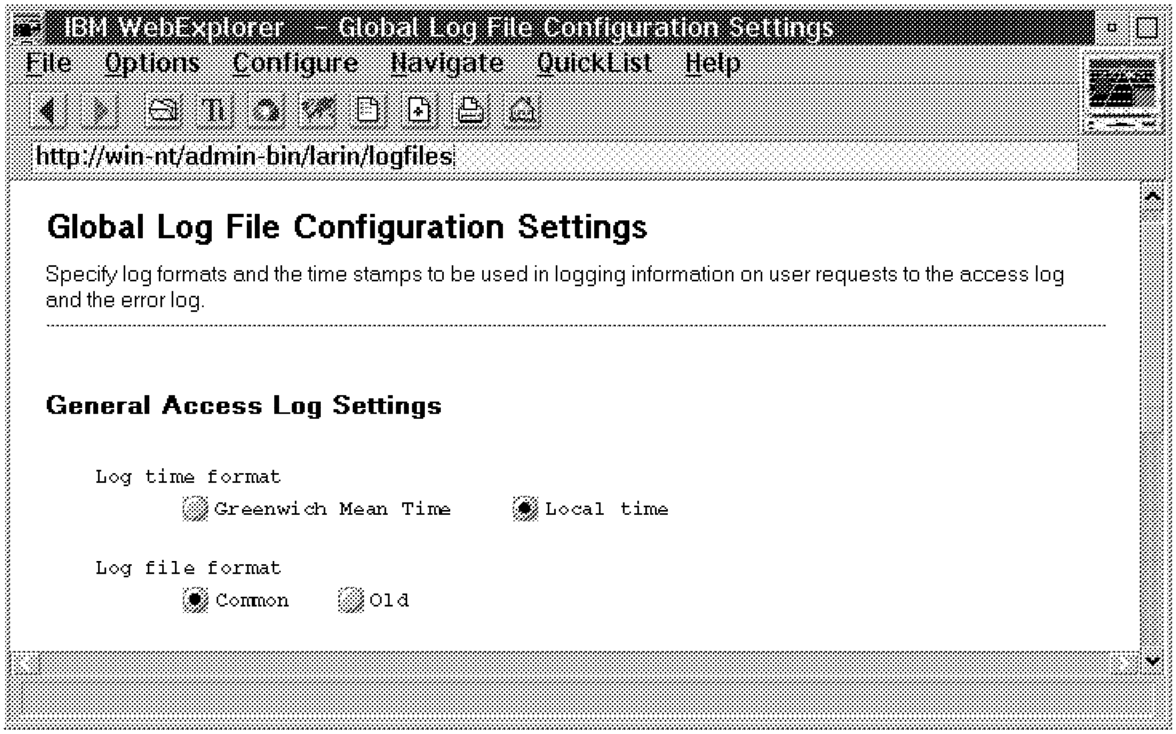
○ Remove selected

Figure 58. The User Directories Form

There are no default directives that correspond to the User Directories form.

4.6.3.10 Global Log File Configuration Settings

Here is the Global Log File Configuration Settings form.



http://win-nt/admin-bin/larin/logfiles

Global Log File Configuration Settings

Specify log formats and the time stamps to be used in logging information on user requests to the access log and the error log.

General Access Log Settings

Log time format
○ Greenwich Mean Time ● Local time

Log file format
● Common ○ Old

Figure 59. The Global Log File Configuration Settings Form

We advise you to use the common log file format, since it can be processed using most available log analysis programs as well as using the tools shipped with the Internet Connection server. For more information about logging, please refer to 4.6.8, “Logging Procedures” on page 148.

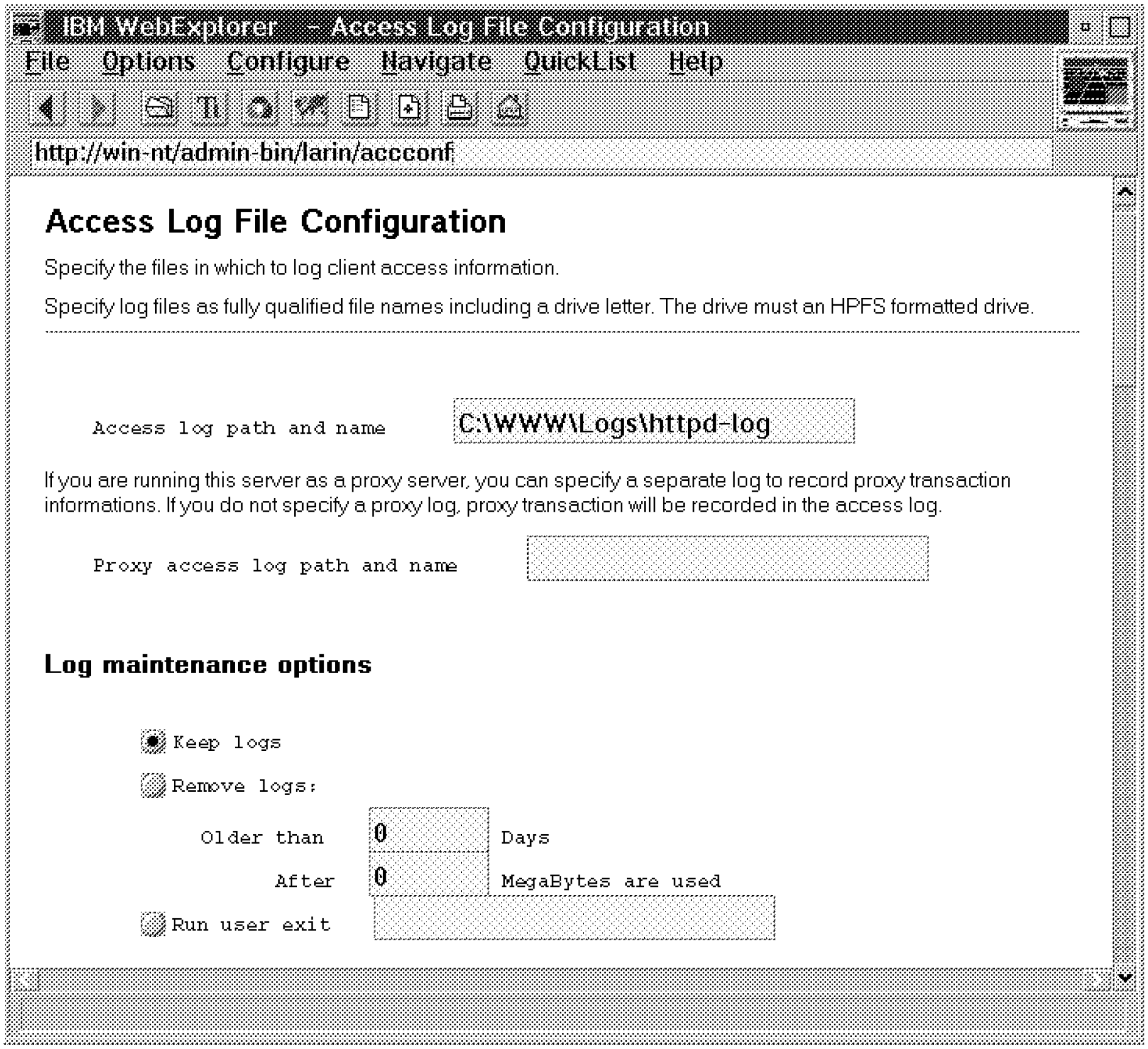
Here are the default directives that correspond to the Global Log File Configuration Settings form.

```
#
#       Default:  LocalTime
#       Syntax:  LogTime  <GMT | LocalTime>
# Example:
# LogTime       LocalTime
#
LogTime       LocalTime

#
#       Default:  Common
#       Syntax:  LogFormat  <Old | Common>
# Example:
# LogFormat     Common
#
LogFormat     Common
```

Figure 60. The Directives Corresponding to the Log File Configuration Settings Form

4.6.3.11 Access Log File Configuration

Here is the Access Log File Configuration - Maintenance form.


IBM WebExplorer - Access Log File Configuration
File Options Configure Navigate QuickList Help

http://win-nt/admin-bin/larin/accconf

Access Log File Configuration

Specify the files in which to log client access information.

Specify log files as fully qualified file names including a drive letter. The drive must an HPFS formatted drive.

Access log path and name C:\WWW\Logs\httpd-log

If you are running this server as a proxy server, you can specify a separate log to record proxy transaction informations. If you do not specify a proxy log, proxy transaction will be recorded in the access log.

Proxy access log path and name

Log maintenance options

● Keep logs

○ Remove logs:

Older than 0 Days

After 0 MegaBytes are used

□ Run user exit

Figure 61. The Access Log File Configuration - Maintenance Form

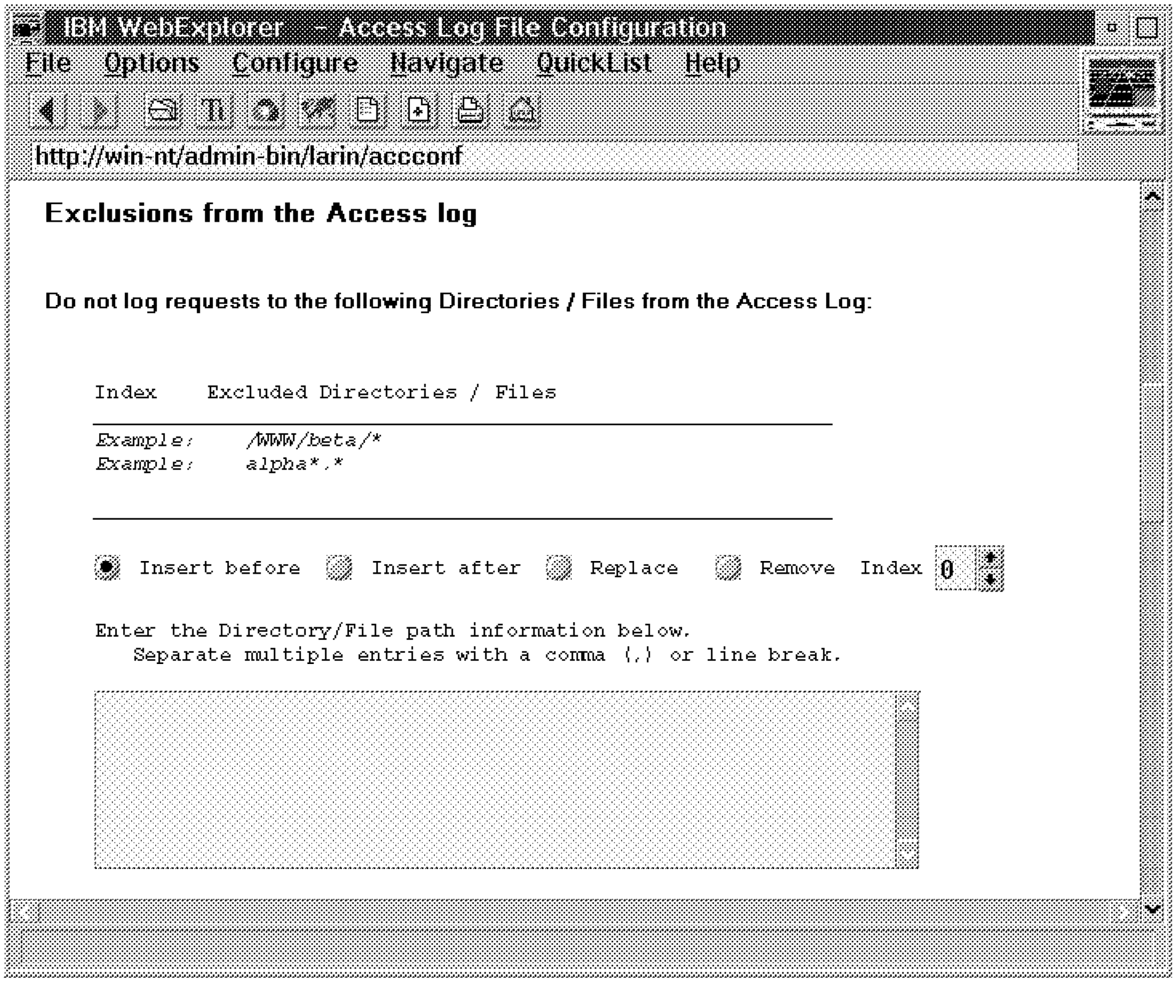
Here are the default directives that correspond to the Access Log File Configuration - Maintenance form.

```
#       Logging; if you want logging, uncomment these lines and specify
#       locations for your access and error logs
#       Default:  <none>
#       Syntax:  AccessLog <filename>
# Example:
# AccessLog     d:\www\daemon\etc\httlog
AccessLog C:\WWW\Logs\httpd-log

# AccessLogArchive directive:
# Enables the purge options (purge) or the user exit option (userexit)
# or does not do either (none) for Access logs.
# When selected, the purge action or userexit action will take place at
# midnight immediately after the previous day′s logs have been closed
# and the new day′s logs have been opened.
# If the userexit option is specified, the name and location of the user exit
# that is called must be specified following the userexit option parameter
#        Default: none
#        Syntax: AccessLogArchive <none | purge | userexit user_exit_spec>
# Examples:
# AccessLogArchive none
# AccessLogArchive purge
# AccessLogArchive userexit c:\usercode\movelogs.exe -d -o -g
AccessLogArchive none

# AccessLogExpire directive:
#
# Sets the age limit, in days,  for access log files. It is only a
# number, no additional # parameters (such as the text ″days″) is
# expected or handled.  Any access log files older than this date
# will be erased.  If set to zero then no expiration date exists.
# The file creation date as reported by the operating system is used
# to determine the date - the suffix of the filename (such as
# httpd-log.Mar2296) is not used to determine file age.
#       Default: 0
#       Syntax: AccessLogExpire <time-spec>
# Example:
# AccessLogExpire 30
AccessLogExpire 0

# AccessLogSizeLimit directive:
#
# Sets the sum total size limit, in megabytes, for the access logs.
# It is only a number, no additional parameters (such as the text
# ″meg″) is expected or handled.  If set to zero then no maximum size
# to the Access Logs is enforced. This directive takes effect after
# AccessLogExpire has been applied.  If the sum total size of access
# log files is larger than the limit assigned, files will be deleted
# starting with the oldest file until within the limit.
#        Default: 0
#        Syntax: AccessLogSizeLimit <size-spec>
# Example:
# AccessLogSizeLimit 20
AccessLogSizeLimit 0
```

Figure 62. The Directives Corresponding to the Access Log File Configuration - Maintenance Form

4.6.3.12 Access Log File Configuration - File Exclusions

Here is the Access Log File Configuration - File Exclusions form.


IBM WebExplorer - Access Log File Configuration
File Options Configure Navigate QuickList Help

http://win-nt/admin-bin/larin/accconf

Exclusions from the Access log

Do not log requests to the following Directories / Files from the Access Log:

```
Index    Excluded Directories / Files
Example:    /WWW/beta/*
Example:    alpha*.*
```

● Insert before ○ Insert after ○ Replace ○ Remove Index 0

Enter the Directory/File path information below.
Separate multiple entries with a comma (,) or line break.

Figure 63. The Access Log File Configuration - File Exclusions Form

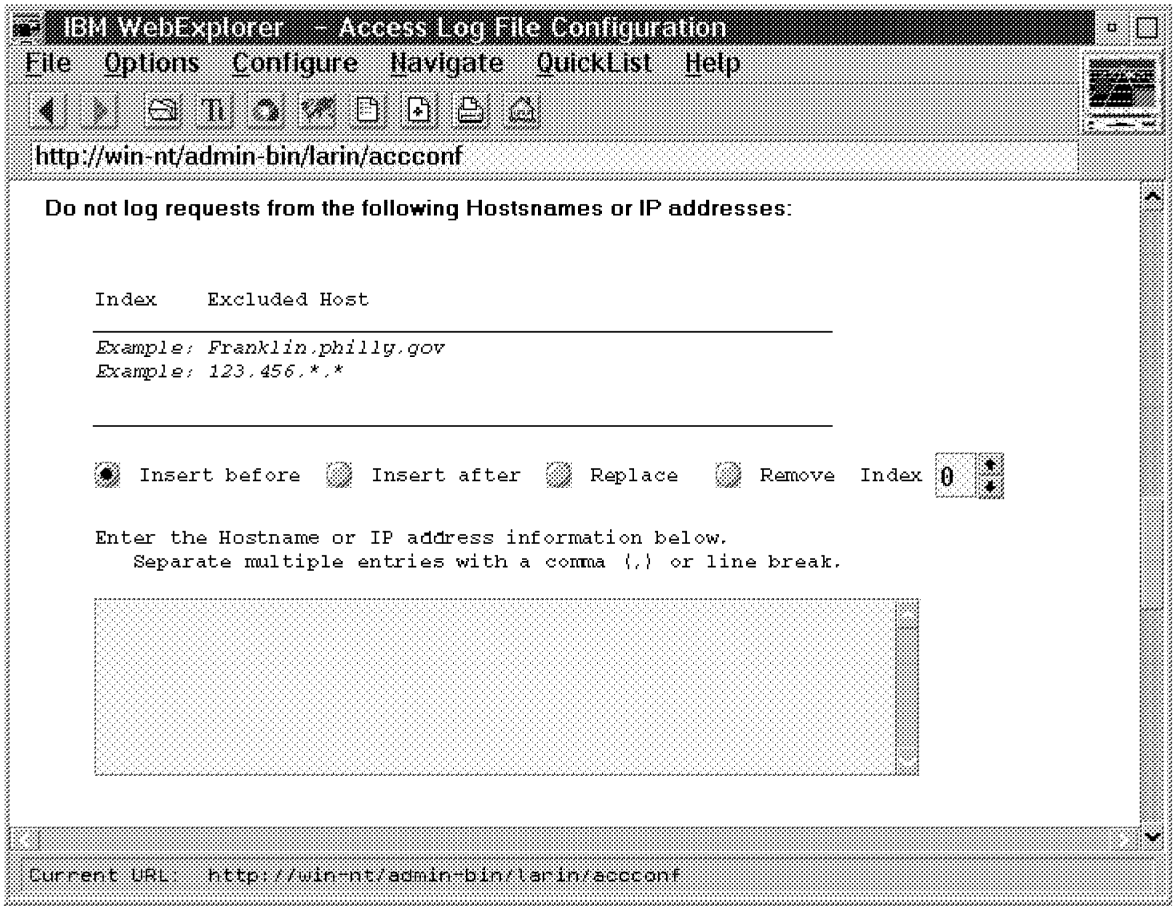
Here are the default directives that correspond to the Access Log File Configuration - File Exclusions form.

```
#
# AccessLogExcludeURL directive
#
# A filter to exclude request URLs matching a given template
#     Default <none>
#     Syntax  AccessLogExcludeURL <URL template>
# Examples:
# AccessLogExcludeURL  *.gif
# AccessLogExcludeURL  \Freebies\*
```

Figure 64. The Directives Corresponding to the Access Log File Configuration - File Exclusions Form

4.6.3.13 Access Log File Configuration - Host Exclusions

Here is the Access Log File Configuration - Host Exclusions form.


IBM WebExplorer - Access Log File Configuration
File Options Configure Navigate QuickList Help
http://win-nt/admin-bin/larin/accconf

Do not log requests from the following Hostsnames or IP addresses:

Index Excluded Host

Example: Franklin.philly.gov
Example: 123.456..**

● Insert before ○ Insert after ○ Replace ○ Remove Index 0

Enter the Hostname or IP address information below.
Separate multiple entries with a comma (,) or line break.

Current URL: http://win-nt/admin-bin/larin/accconf

Figure 65. The Access Log File Configuration - Host Exclusions Form

Note you have to set the DNS-Lookup directive to On in order to be able to specify hostnames here. Refer to Figure 42 on page 106 for more information.

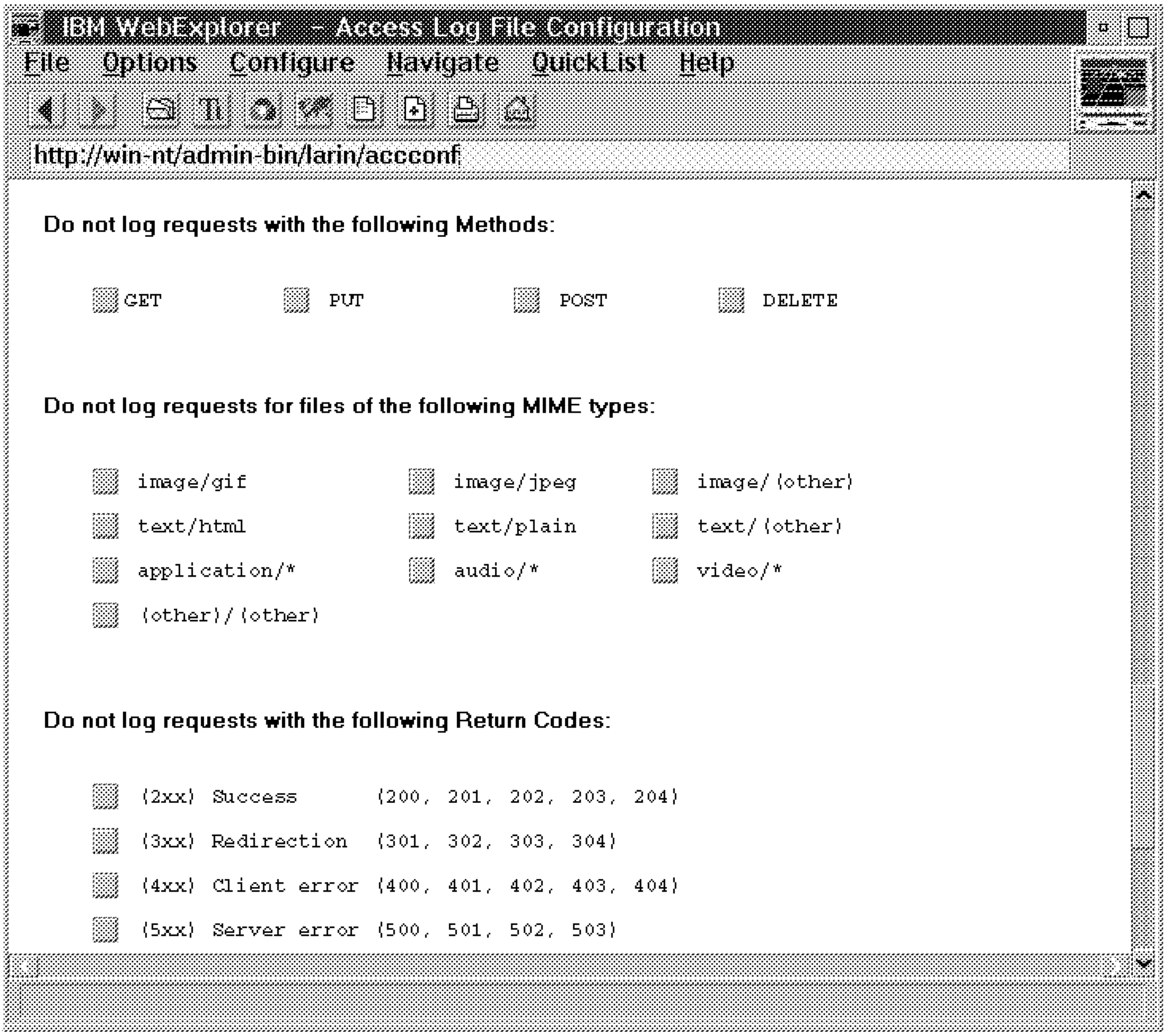
Here are the default directives that correspond to the Access Log File Configuration - Host Exclusions form.

```
#  NoLog directive:
#
#      Suppress access log entries for host matching a given IP address or
#      hostname. Wild cards ”*” may be used.  This directive may be used
#      multiple times within the configuration file.
#      Default:  <none>
#      Syntax:  NoLog  <hostnames and IP addresses>
# Examples:
# NoLog    128.141.*.*
# NoLog    *.location.company.com
# Nolog    *.*.*.com
# NoLog    *.ch  *.fr  *.it
#
```

Figure 66. The Directives Corresponding to the Access Log File Configuration - Host Exclusions Form

4.6.3.14 Access Log File Configuration - Other Exclusions

Here is the Access Log File Configuration - Other Exclusions form.



http://win-nt/admin-bin/larin/accconf

Do not log requests with the following Methods:

□ GET □ PUT □ POST □ DELETE

Do not log requests for files of the following MIME types:

□ image/gif □ image/jpeg □ image/(other)
□ text/html □ text/plain □ text/(other)
□ application/* □ audio/* □ video/*
□ (other)/(other)

Do not log requests with the following Return Codes:

□ (2xx) Success (200, 201, 202, 203, 204)
□ (3xx) Redirection (301, 302, 303, 304)
□ (4xx) Client error (400, 401, 402, 403, 404)
□ (5xx) Server error (500, 501, 502, 503)

Figure 67. The Access Log File Configuration - Other Exclusions Form

We recommend that you log as much information as possible. This will allow you to identify server and security problems faster.

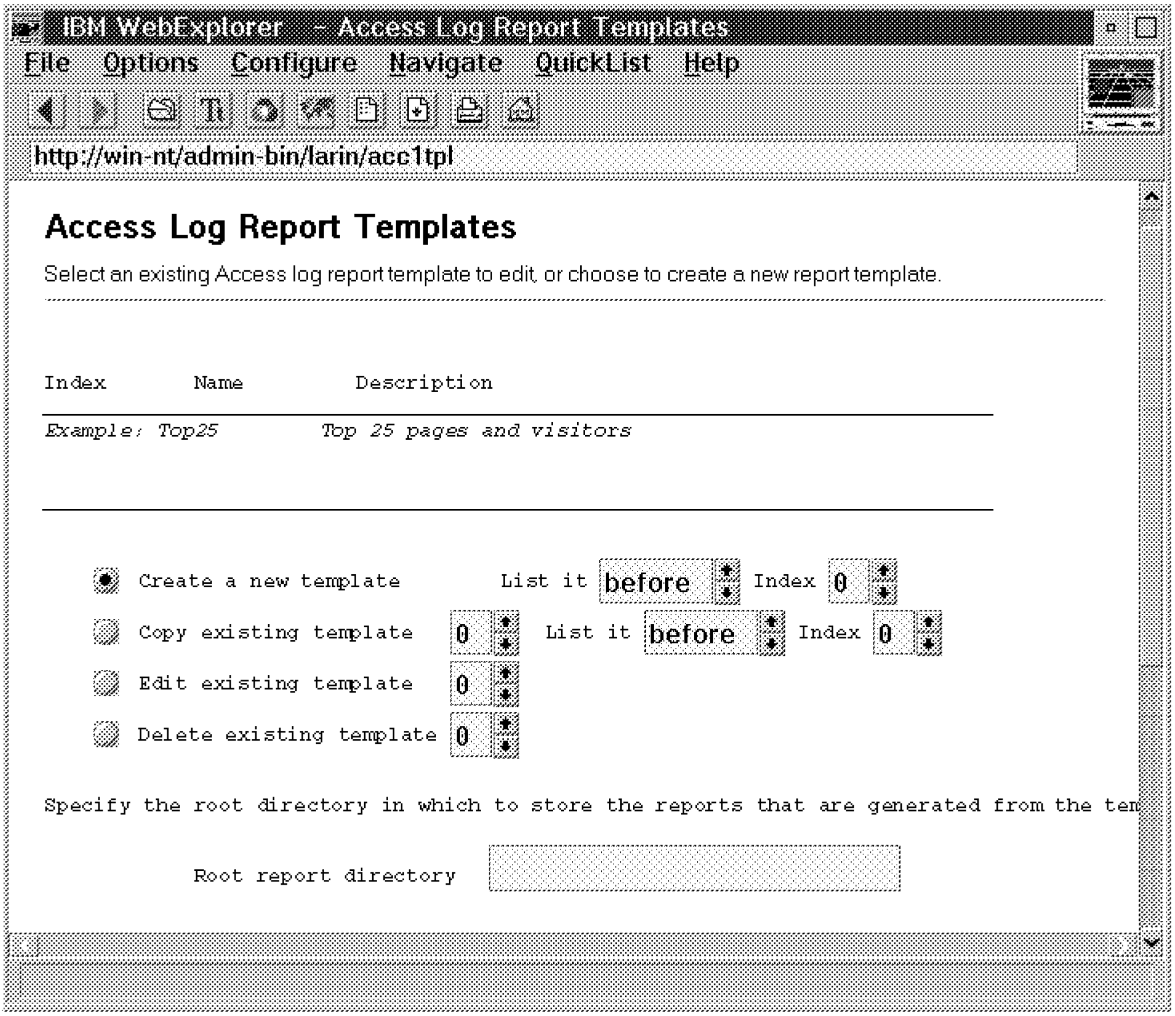
Here are the default directives that correspond to the Access Log File Configuration - Other Exclusions form.

```
# AccessLogExcludeMethod directive:
#
# A filter to exclude requests of a given method
#       Default: <none>
#       Syntax AccessLogExcludeMethod <Get | Put | Post | Delete>
# Examples:
# AccessLogExcludeMethod Get
# AccessLogExcludeMethod Put
# AccessLogExcludeMethod Post

# AccessLogExcludeReturnCode directive:
#
# A filter to exclude requests with a given return code range
# (200s, 300s, 400s, 500s)
#       Default: <none>
#       Syntax: AccessLogExcludeReturnCode <200 | 300 | 400 | 500>
# Examples:
# AccessLogExcludeReturnCode 300
# AccessLogExcludeReturnCode 400

# AccessLogExcludeMimeType directive:
#
# A filter to exclude requests for files of a given mime type.
#       Default: <none>
#       Syntax: AccessLogExcludeMimeType <text/html | text/plain |
#                                         text/other | image/gif |
#                                         image/jpeg | image/(other) |
#                                         application/* |
#                                         audio/* | video/* |
#                                         (other)/other)>
# Examples:
# AccessLogExcludeMimeType text/html
# AccessLogExcludeMimeType text/plain
```

Figure 68. The Directives Corresponding to the Access Log File Configuration - Other Exclusions Form

4.6.3.15 Error Log File Configuration

Here is the Error Log File Configuration form.



http://win-nt/admin-bin/larin/acc1tpl

Access Log Report Templates

Select an existing Access log report template to edit, or choose to create a new report template.

Index	Name	Description
Example:	*Top25*	*Top 25 pages and visitors*

● Create a new template List it before Index 0

○ Copy existing template 0 List it before Index 0

○ Edit existing template 0

○ Delete existing template 0

Specify the root directory in which to store the reports that are generated from the tem

Root report directory

Figure 69. The Error Log File Configuration Form

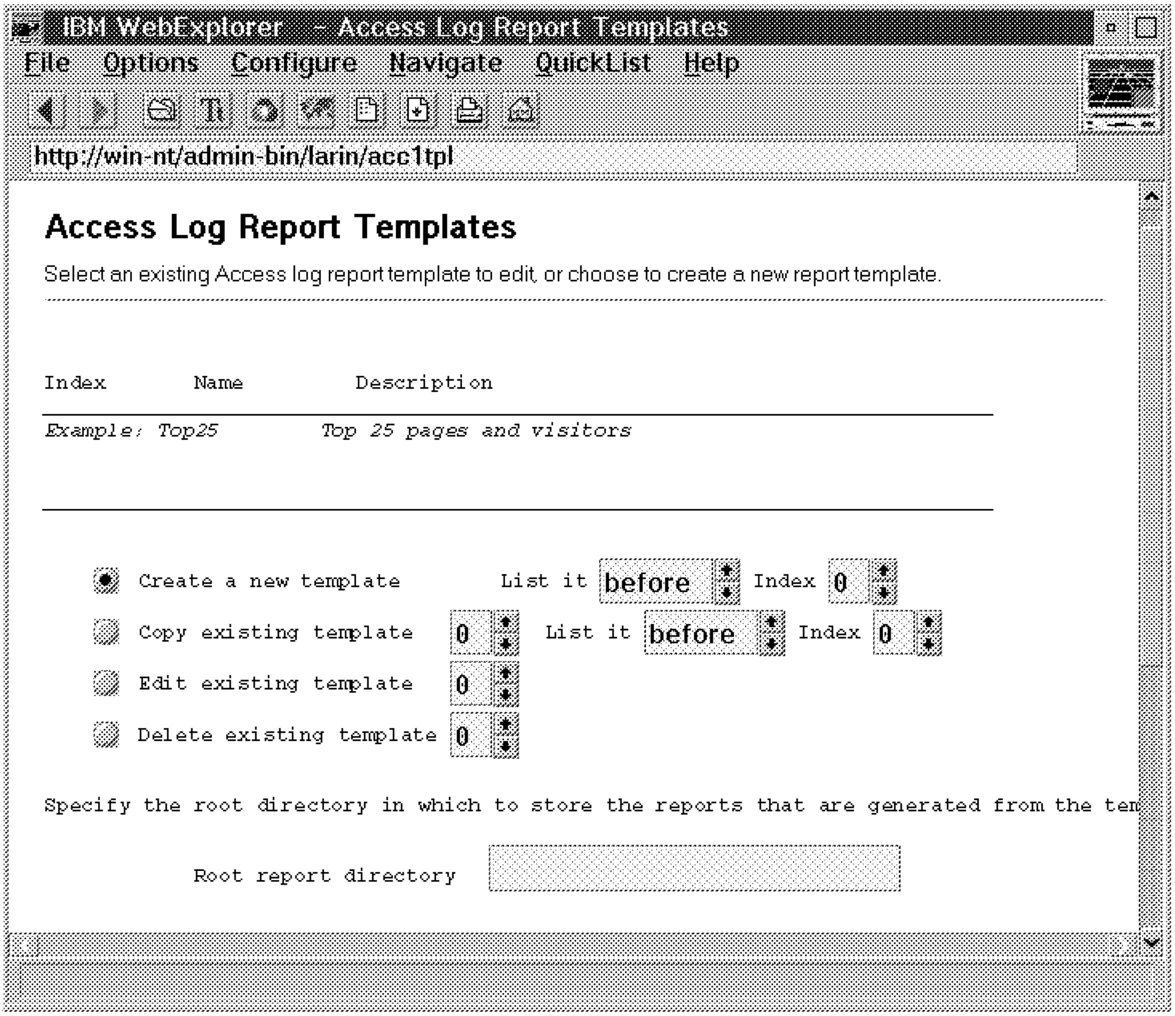
Here are the default directives that correspond to the Error Log File Configuration form.

```
#
#       Default:  <none>
#       Syntax:  ErrorLog <filename>
# Example:
# ErrorLog      d:\www\daemon\etc\htterr
#
ErrorLog C:\WWW\Logs\httpd-error

# ErrorLogArchive directive:
# Enables the purge options (purge) or the user exit option (userexit)
# or does not do either (none) for Error logs. When selected, the
# purge action or userexit action will take place at midnight
# immediately after the previous day′s logs have been closed and the
# new day′s logs have been opened.
# If the userexit option is specified, the name and location of the user exit
# that is called must be specified following the userexit option parameter
#        Default: none
#        Syntax: ErrorLogArchive <none | purge | userexit user_exit_spec>
# Example:
# ErrorLogArchive none
# ErrorLogArchive purge
# ErrorLogArchive userexit c:\usercode\movelogs.exe -d -o -g
ErrorLogArchive none

# ErrorLogExpire
# Sets the age limit, in days, for the error logs. It is only a number,
# no additional parameters (such as the text ″days″) is expected or
# handled.  Any error log files older than this date will be erased.
# If set to zero then no expiration date exists.  The file creation
# date as reported by the operating system is used to determine the
# date - the suffix of the filename (such as httpd-log.Mar2296) is not
# used to determine file age.
#       Default: 0
#       Syntax: ErrorLogExpire <time-spec>
# Example:
# ErrorLogExpire 30
#
ErrorLogExpire 0

# ErrorLogSizeLimit
# Sets the sum total size limit, in megabytes, for the error logs.  It
# is only a number, no additional parameters (such as the text ″meg″)
# is expected or handled.  If set to zero then no maximum size to the
# Error Logs is enforced. This directive take effect after
# ErrorLogExpire has been applied.  If the sum total size of error log
# files is larger than the limit assigned, files will be deleted
# starting with the oldest file until within the limit.
#        Default: 0
#        Syntax: ErrorLogSizeLimit <size-spec>
# Example:
# ErrorLogSizeLimit 20
ErrorLogSizeLimit 0
```

Figure 70. The Directives Corresponding to the Error Log File Configuration Form

4.6.3.16 Access Log Report Template

Here is the Access Log Report Template form.


IBM WebExplorer - Access Log Report Templates
File Options Configure Navigate QuickList Help
http://win-nt/admin-bin/larin/acc1tpl

Access Log Report Templates

Select an existing Access log report template to edit, or choose to create a new report template.

Index	Name	Description
Example:	*Top25*	*Top 25 pages and visitors*

● Create a new template List it before Index 0

○ Copy existing template 0 List it before Index 0

○ Edit existing template 0

○ Delete existing template 0

Specify the root directory in which to store the reports that are generated from the tem

Root report directory

Figure 71. The Access Log Report Template Form

The logging feature of the IBM Internet Connection Servers is addressed in detail in 4.6.8, “Logging Procedures” on page 148.

4.6.3.17 OS/400 Logging

In this section we introduce the directives that control logging in the HTTP Server of OS/400. For more information about the HTTP Server and Internet Connection for OS/400, please refer to 4.5.3, “AS/400” on page 76.

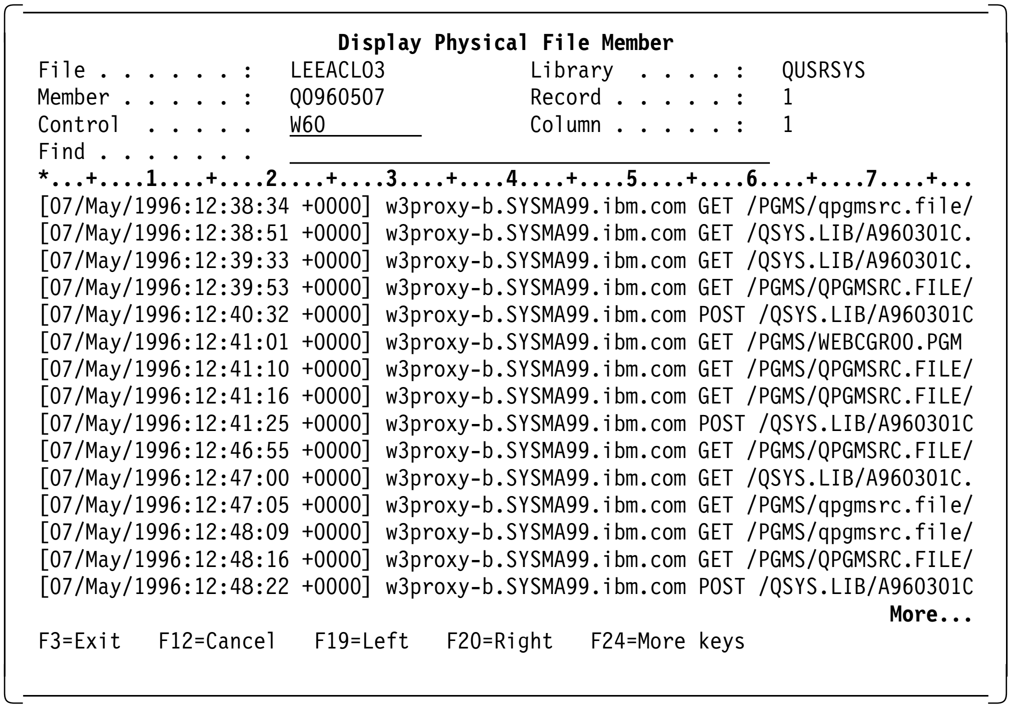
The following is a list of HTTP Server directives that control logging in the OS/400 HTTP Server environment.

AccessLog *log-file-name {max-size}*

The AccessLog directive specifies the log-file-name of the access log file. When the HTTP server is started, a member is automatically created in QUSRSYS/log-file-name with a member name based on time and date.

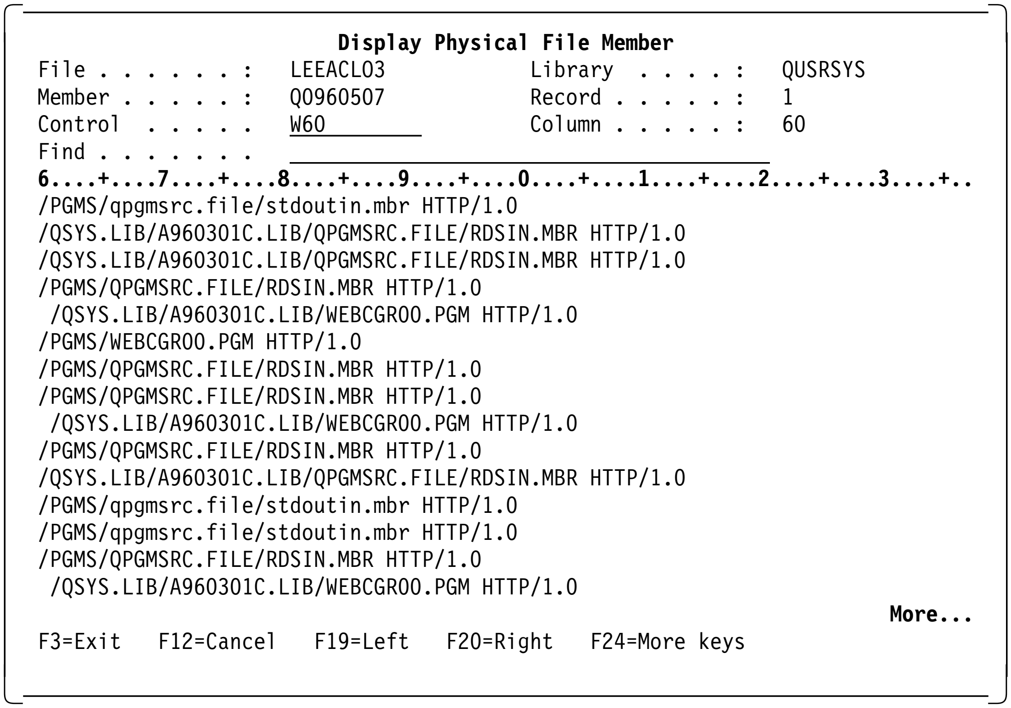
The optional max-size parameter specifies the maximum size of the file in kilobytes. If 0 is specified, the log file keeps growing until STRTCPSVR SERVER(*HTTP) RESTART(*YES) is executed. The default value is 2 Mb.

If this directive is not explicitly specified in the HTTP server configuration, no access logging is performed.



```
                        Display Physical File Member
File . . . . . . :   LEEACL03           Library  . . . . :   QUSRSYS
Member . . . . . :   Q0960507           Record . . . . . :   1
Control  . . . . .   W60                Column . . . . . :   1
Find . . . . . . .
*...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+...
[07/May/1996:12:38:34 +0000] w3proxy-b.SYSMA99.ibm.com GET /PGMS/qpgmsrc.file/
[07/May/1996:12:38:51 +0000] w3proxy-b.SYSMA99.ibm.com GET /QSYS.LIB/A960301C.
[07/May/1996:12:39:33 +0000] w3proxy-b.SYSMA99.ibm.com GET /QSYS.LIB/A960301C.
[07/May/1996:12:39:53 +0000] w3proxy-b.SYSMA99.ibm.com GET /PGMS/QPGMSRC.FILE/
[07/May/1996:12:40:32 +0000] w3proxy-b.SYSMA99.ibm.com POST /QSYS.LIB/A960301C
[07/May/1996:12:41:01 +0000] w3proxy-b.SYSMA99.ibm.com GET /PGMS/WEBCGR00.PGM
[07/May/1996:12:41:10 +0000] w3proxy-b.SYSMA99.ibm.com GET /PGMS/QPGMSRC.FILE/
[07/May/1996:12:41:16 +0000] w3proxy-b.SYSMA99.ibm.com GET /PGMS/QPGMSRC.FILE/
[07/May/1996:12:41:25 +0000] w3proxy-b.SYSMA99.ibm.com POST /QSYS.LIB/A960301C
[07/May/1996:12:46:55 +0000] w3proxy-b.SYSMA99.ibm.com GET /PGMS/QPGMSRC.FILE/
[07/May/1996:12:47:00 +0000] w3proxy-b.SYSMA99.ibm.com GET /QSYS.LIB/A960301C.
[07/May/1996:12:47:05 +0000] w3proxy-b.SYSMA99.ibm.com GET /PGMS/qpgmsrc.file/
[07/May/1996:12:48:09 +0000] w3proxy-b.SYSMA99.ibm.com GET /PGMS/qpgmsrc.file/
[07/May/1996:12:48:16 +0000] w3proxy-b.SYSMA99.ibm.com GET /PGMS/QPGMSRC.FILE/
[07/May/1996:12:48:22 +0000] w3proxy-b.SYSMA99.ibm.com POST /QSYS.LIB/A960301C
                                                                       More...
F3=Exit   F12=Cancel   F19=Left   F20=Right   F24=More keys
```

Figure 72. Typical Access Log Member (Left)



```
                        Display Physical File Member
File . . . . . . :   LEEACL03           Library  . . . . :   QUSRSYS
Member . . . . . :   Q0960507           Record . . . . . :   1
Control  . . . . .   W60                Column . . . . . :   60
Find . . . . . . .
6....+....7....+....8....+....9....+....0....+....1....+....2....+....3....+..
/PGMS/qpgmsrc.file/stdoutin.mbr HTTP/1.0
/QSYS.LIB/A960301C.LIB/QPGMSRC.FILE/RDSIN.MBR HTTP/1.0
/QSYS.LIB/A960301C.LIB/QPGMSRC.FILE/RDSIN.MBR HTTP/1.0
/PGMS/QPGMSRC.FILE/RDSIN.MBR HTTP/1.0
 /QSYS.LIB/A960301C.LIB/WEBCGR00.PGM HTTP/1.0
/PGMS/WEBCGR00.PGM HTTP/1.0
/PGMS/QPGMSRC.FILE/RDSIN.MBR HTTP/1.0
/PGMS/QPGMSRC.FILE/RDSIN.MBR HTTP/1.0
 /QSYS.LIB/A960301C.LIB/WEBCGR00.PGM HTTP/1.0
/PGMS/QPGMSRC.FILE/RDSIN.MBR HTTP/1.0
/QSYS.LIB/A960301C.LIB/QPGMSRC.FILE/RDSIN.MBR HTTP/1.0
/PGMS/qpgmsrc.file/stdoutin.mbr HTTP/1.0
/PGMS/qpgmsrc.file/stdoutin.mbr HTTP/1.0
/PGMS/QPGMSRC.FILE/RDSIN.MBR HTTP/1.0
 /QSYS.LIB/A960301C.LIB/WEBCGR00.PGM HTTP/1.0
                                                                       More...
F3=Exit   F12=Cancel   F19=Left   F20=Right   F24=More keys
```

Figure 73. Typical Access Log Member (Right)

Access logging is a useful aid for analyzing who is accessing your system, how frequently your system is being accessed and which Web pages are being accessed. This is useful for both security auditing and marketing related analysis.

ErrorLog *file-name {max-size}*
The ErrorLog directive specifies the file name of the access log file. When the HTTP server is started, a member is automatically created in QUSRSYS/file-name with a member name based on time and date.

The optional max-size parameter specifies the maximum size of the file in kilobytes. If 0 is specified, the log file keeps growing until STRTCPSVR SERVER(*HTTP) RESTART(*YES) is executed. The default value is 2 Mb. If this directive is not explicitly specified in the HTTP server configuration, no access logging is performed.

LogFormat DDS **| Common**
The LogFormat directive defines which format the log files are saved in. Common file format is a source physical file with the total record length of 375 bytes. It is very hard to process using PDM because the SEU can handle only records up to 240 bytes long. DDS file format is divided into three fields and has a total record length of 375 bytes. We have included into library ITSOIC400 the object and source code for program PRTWWWLOG, which is a quite simple ILE RPG program that produces a print of both file formats.

LogTime LocalTime **| GMT**
Specifies how the log file entries are time stamped.

NoLog *template{ template{...}}*
The NoLog directive is used to disable the logging of HTTP server accesses for selected groups of client/browser host names and host addresses. It can be used to filter out log entries of your regular users and leave just those entries that may be considered security breaches, or those of interest for marketing reasons.

Examples

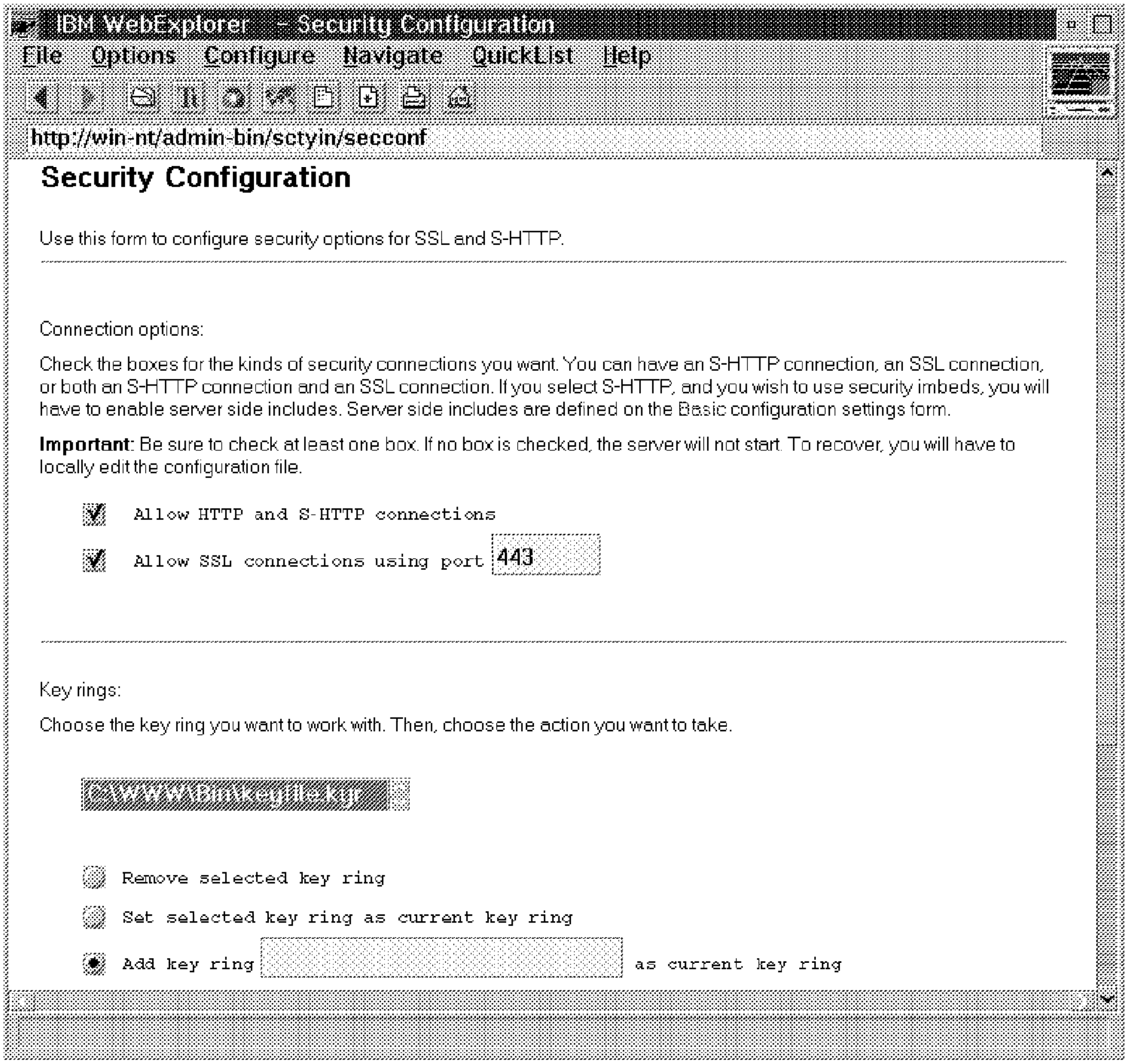
```
NoLog  9.180.*.*
NoLog  *.local.users.com
NoLog  *.abc  *.pqr  *.xyz
```

The template parameter is either a dotted decimal host address or a dotted Internet Host name. An asterisk ('*') may be used as a wild card for any of the terms.

This directive is relevant only when the AccessLog directive is specified. It may be specified multiple times in the HTTP server config and more than one template may be specified on each NoLog directive.

4.6.3.18 Security Configuration

Here is the Security Configuration form.



IBM WebExplorer - Security Configuration
File Options Configure Navigate QuickList Help

http://win-nt/admin-bin/sctyin/secconf

Security Configuration

Use this form to configure security options for SSL and S-HTTP.

Connection options:

Check the boxes for the kinds of security connections you want. You can have an S-HTTP connection, an SSL connection, or both an S-HTTP connection and an SSL connection. If you select S-HTTP, and you wish to use security imbeds, you will have to enable server side includes. Server side includes are defined on the Basic configuration settings form.

Important: Be sure to check at least one box. If no box is checked, the server will not start. To recover, you will have to locally edit the configuration file.

☑ Allow HTTP and S-HTTP connections

☑ Allow SSL connections using port 443

Key rings:

Choose the key ring you want to work with. Then, choose the action you want to take.

C:\WWW\Bin\keyfile.kyr

○ Remove selected key ring

○ Set selected key ring as current key ring

● Add key ring [ ] as current key ring

Figure 74. The Security Configuration Form

Note that both sslmode and normalmode directive cannot be set to off. In that case the server would not accept secure or nonsecure HTTP transactions, that is it would accept no transactions at all.

Here are the default directives that correspond to the Security Configuration form.

```
#
#  Security directives.
#
#  sslmode:
#     Use this directive to turn on/off SSL security using the port specified by the
#     sslport directive.
#     Syntax:  sslmode <on/off>
#     Default:  on
#
sslmode    on

#
#  sslport:
#     Use this directive to specify the port that should be used for SSL transactions
#     Syntax:  sslport <port number>
#     Default:  443
#
sslport   443

#
#  normalmode:
#     Use this directive to turn on/off non-secure transactions over the port
#     specify with the port directive.
#     Syntax:  normalmode:  <on/off>
#     Default:  on
#
normalmode  on

#  keyfile:
#    Use this directive to specify the names of key files that are available. Mu;tiple
#    keyfile directives are allowed.
#    Syntax:  keyfile   <filename.kyr>
#    Default: keyfile.kyr
#    Note:  The last file in the list is the file that the server will use.
#
```

Figure 75. The Directives Corresponding to the Security Configuration Form

4.6.4 Key and Certificate Management

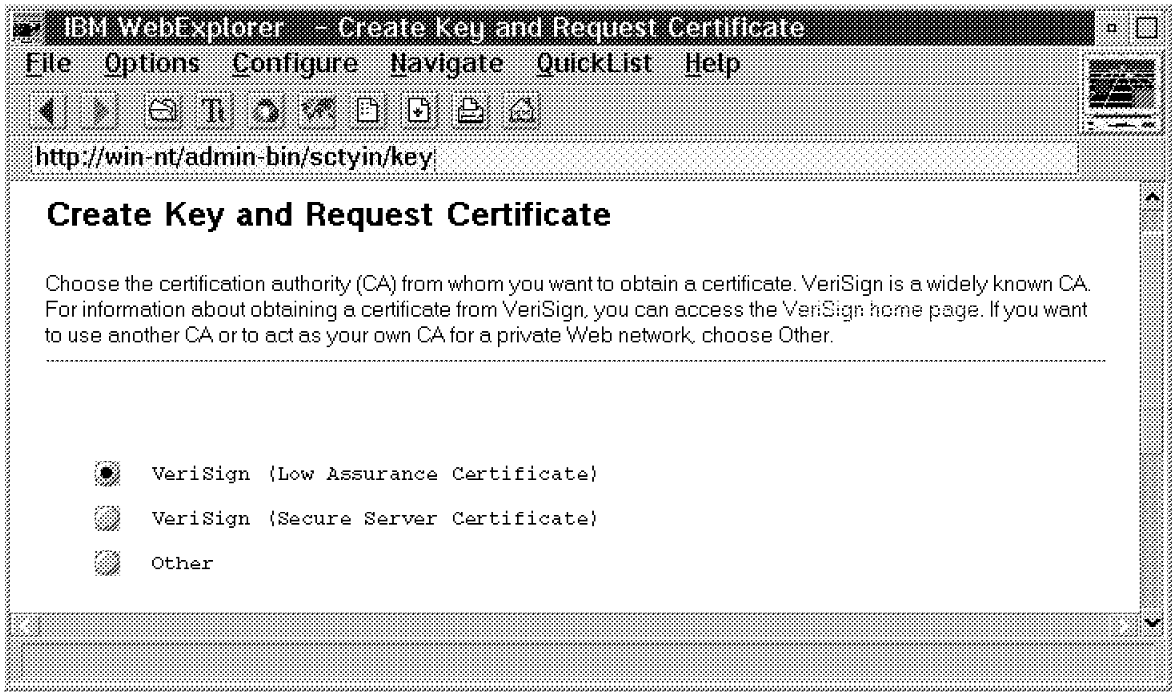
This section describes how to request a certificate and to create a public-private key pair for your IBM Internet Connection Server. This will enable the server to use the Secure Sockets Layer (SSL) in its communications with browsers.

This procedure is the same whether you chose Verisign or another certification authority. In the example described here we request and install a VeriSign Low Assurance Certificate.

The Administration and Configuration forms are the best way to do this. However, we advise you to disconnect your server from any unsecure network and not to exchange sensible data before and while you configure your server.

The first step is to link to the Create Key and Request Certificate form as shown in Figure 76 on page 134. The default URL of this page, relative to the server root is:

/admin-bin/sctyin/key


IBM WebExplorer - Create Key and Request Certificate
File Options Configure Navigate QuickList Help

http://win-nt/admin-bin/sctyin/key

Create Key and Request Certificate

Choose the certification authority (CA) from whom you want to obtain a certificate. VeriSign is a widely known CA. For information about obtaining a certificate from VeriSign, you can access the VeriSign home page. If you want to use another CA or to act as your own CA for a private Web network, choose Other.

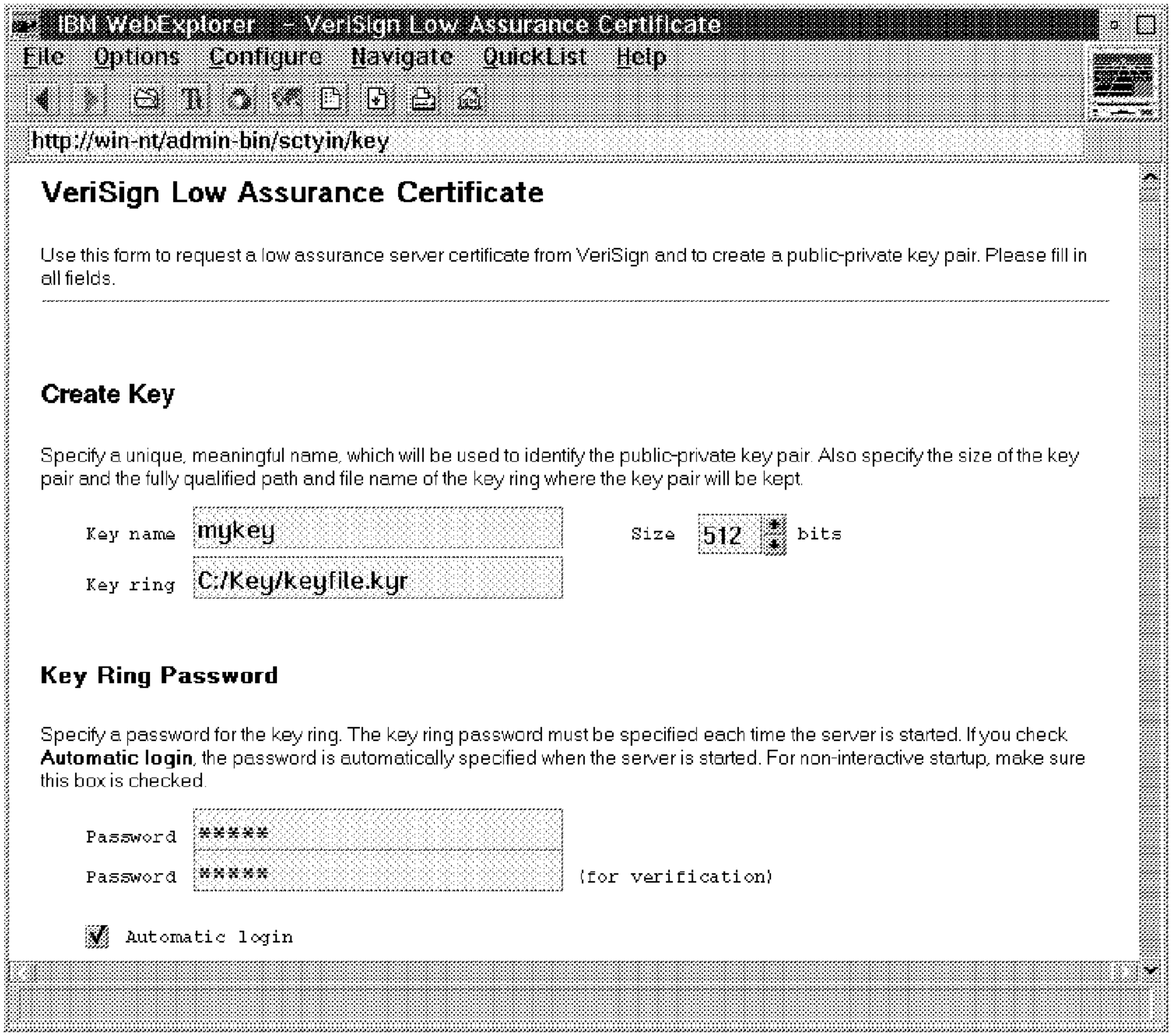
● VeriSign (Low Assurance Certificate)

○ VeriSign (Secure Server Certificate)

○ Other

Figure 76. The Create Key and Request Certificate Form

From the form shown above, select your certification authority. In our case we opted for VeriSign and their Low Assurance Certificate. We completed the forms that were presented to us as shown in Figure 77 on page 135, Figure 78 on page 136, and Figure 78 on page 136.


IBM WebExplorer - VeriSign Low Assurance Certificate
File Options Configure Navigate QuickList Help
http://win-nt/admin-bin/sctyin/key

VeriSign Low Assurance Certificate

Use this form to request a low assurance server certificate from VeriSign and to create a public-private key pair. Please fill in all fields.

Create Key

Specify a unique, meaningful name, which will be used to identify the public-private key pair. Also specify the size of the key pair and the fully qualified path and file name of the key ring where the key pair will be kept.

Key name mykey Size 512 bits

Key ring C:/Key/keyfile.kyr

Key Ring Password

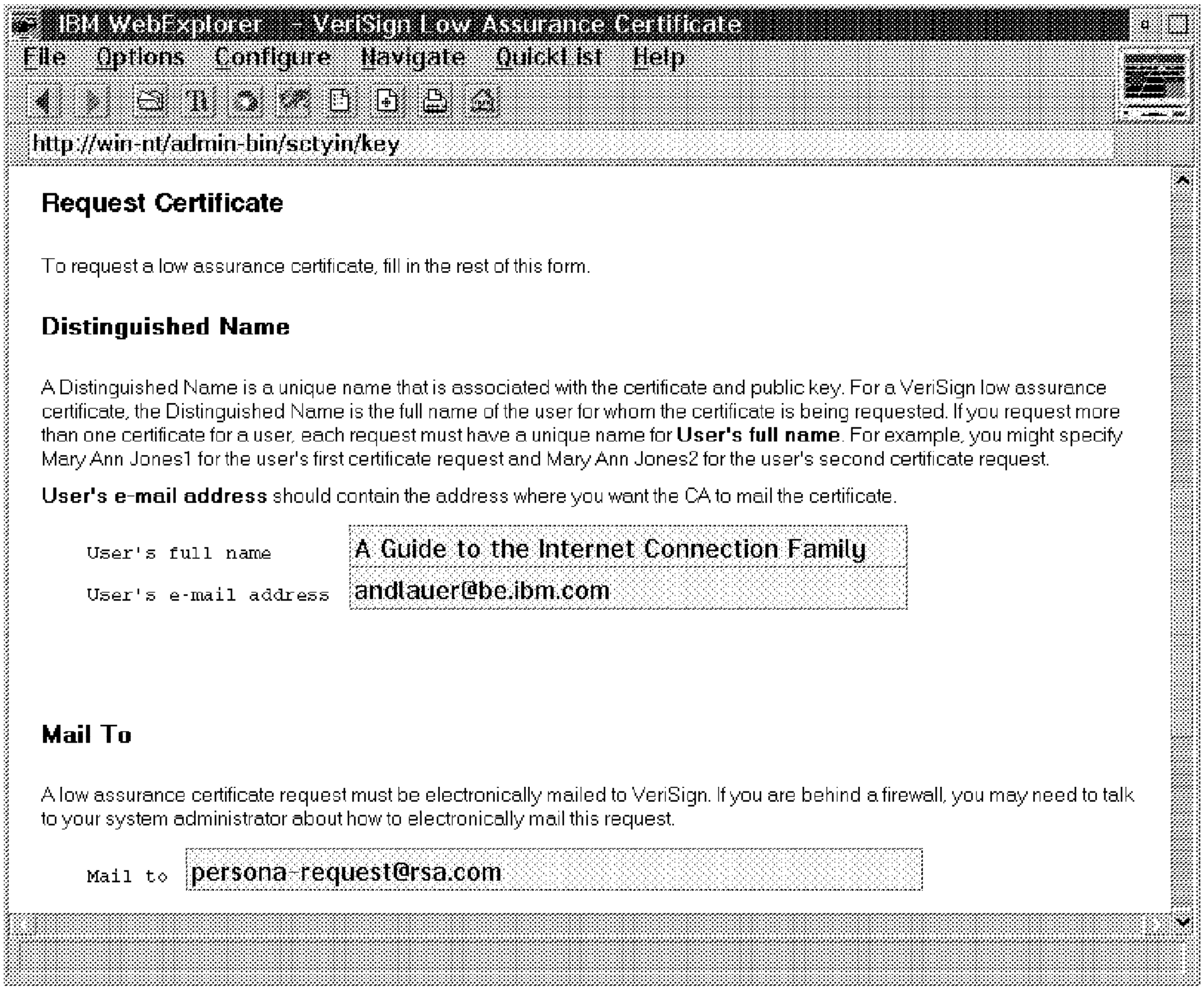
Specify a password for the key ring. The key ring password must be specified each time the server is started. If you check **Automatic login**, the password is automatically specified when the server is started. For non-interactive startup, make sure this box is checked.

Password *****

Password ***** (for verification)

☑ Automatic login

Figure 77. VeriSign Low Assurance Certificate Form (Part 1)


IBM WebExplorer - VeriSign Low Assurance Certificate
File Options Configure Navigate QuickList Help

http://win-nt/admin-bin/sctyin/key

Request Certificate

To request a low assurance certificate, fill in the rest of this form.

Distinguished Name

A Distinguished Name is a unique name that is associated with the certificate and public key. For a VeriSign low assurance certificate, the Distinguished Name is the full name of the user for whom the certificate is being requested. If you request more than one certificate for a user, each request must have a unique name for **User's full name**. For example, you might specify Mary Ann Jones1 for the user's first certificate request and Mary Ann Jones2 for the user's second certificate request.

User's e-mail address should contain the address where you want the CA to mail the certificate.

User's full name: A Guide to the Internet Connection Family

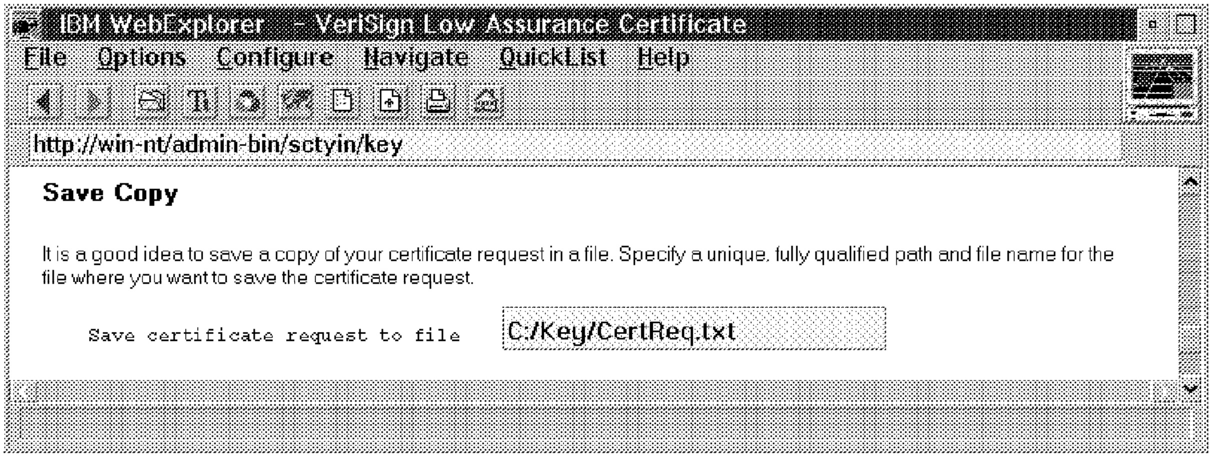
User's e-mail address: andlauer@be.ibm.com

Mail To

A low assurance certificate request must be electronically mailed to VeriSign. If you are behind a firewall, you may need to talk to your system administrator about how to electronically mail this request.

Mail to: persona-request@rsa.com

Figure 78. VeriSign Low Assurance Certificate Form (Part 2)


IBM WebExplorer - VeriSign Low Assurance Certificate
File Options Configure Navigate QuickList Help

http://win-nt/admin-bin/sctyin/key

Save Copy

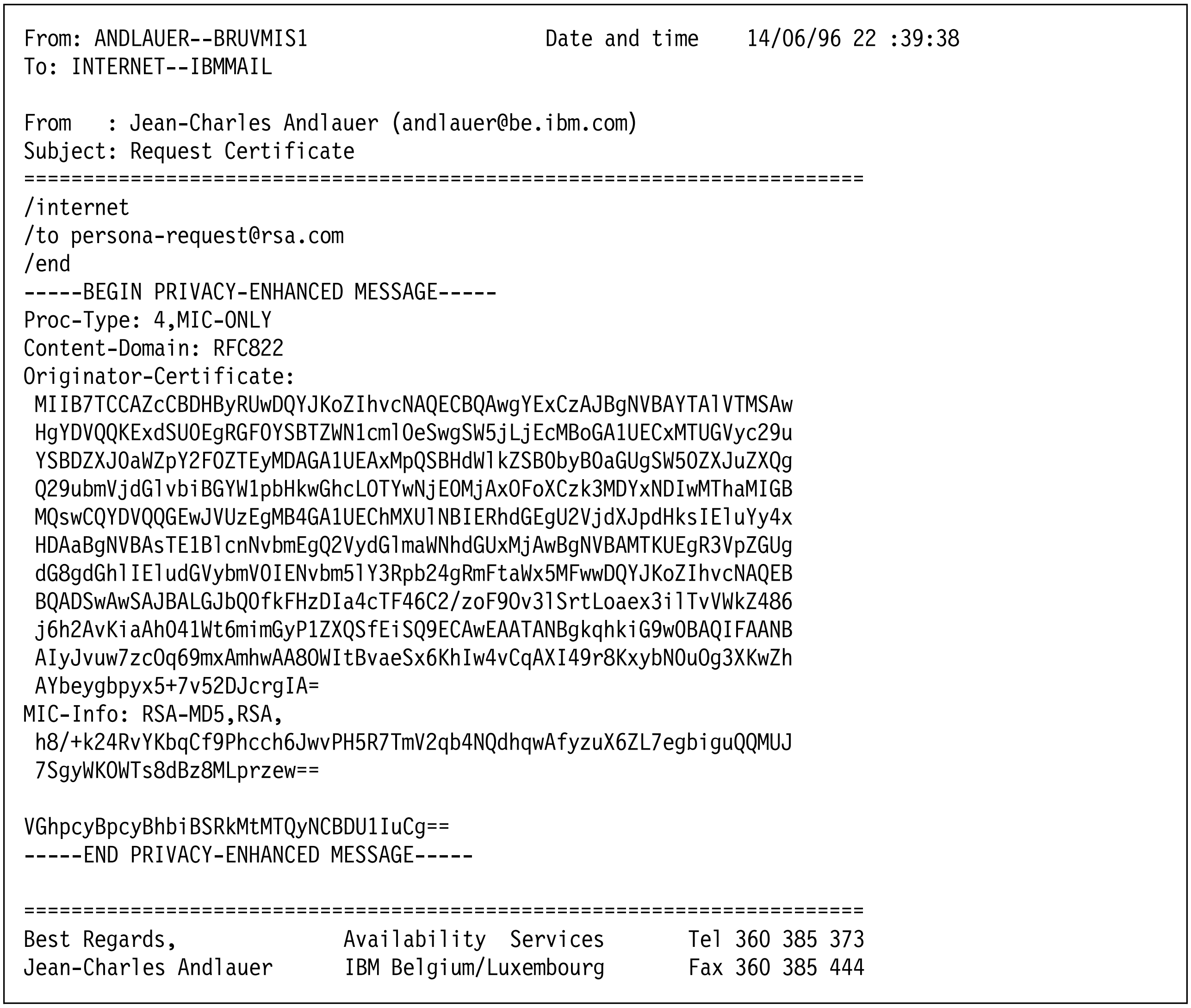
It is a good idea to save a copy of your certificate request in a file. Specify a unique, fully qualified path and file name for the file where you want to save the certificate request.

Save certificate request to file: C:/Key/CertReq.txt

Figure 79. VeriSign Low Assurance Certificate Form (Part 3)

As it was not possible to send or receive e-mail from our Internet Connection Server host, submitting this form ended with a `Certificate request was not mailed` error.

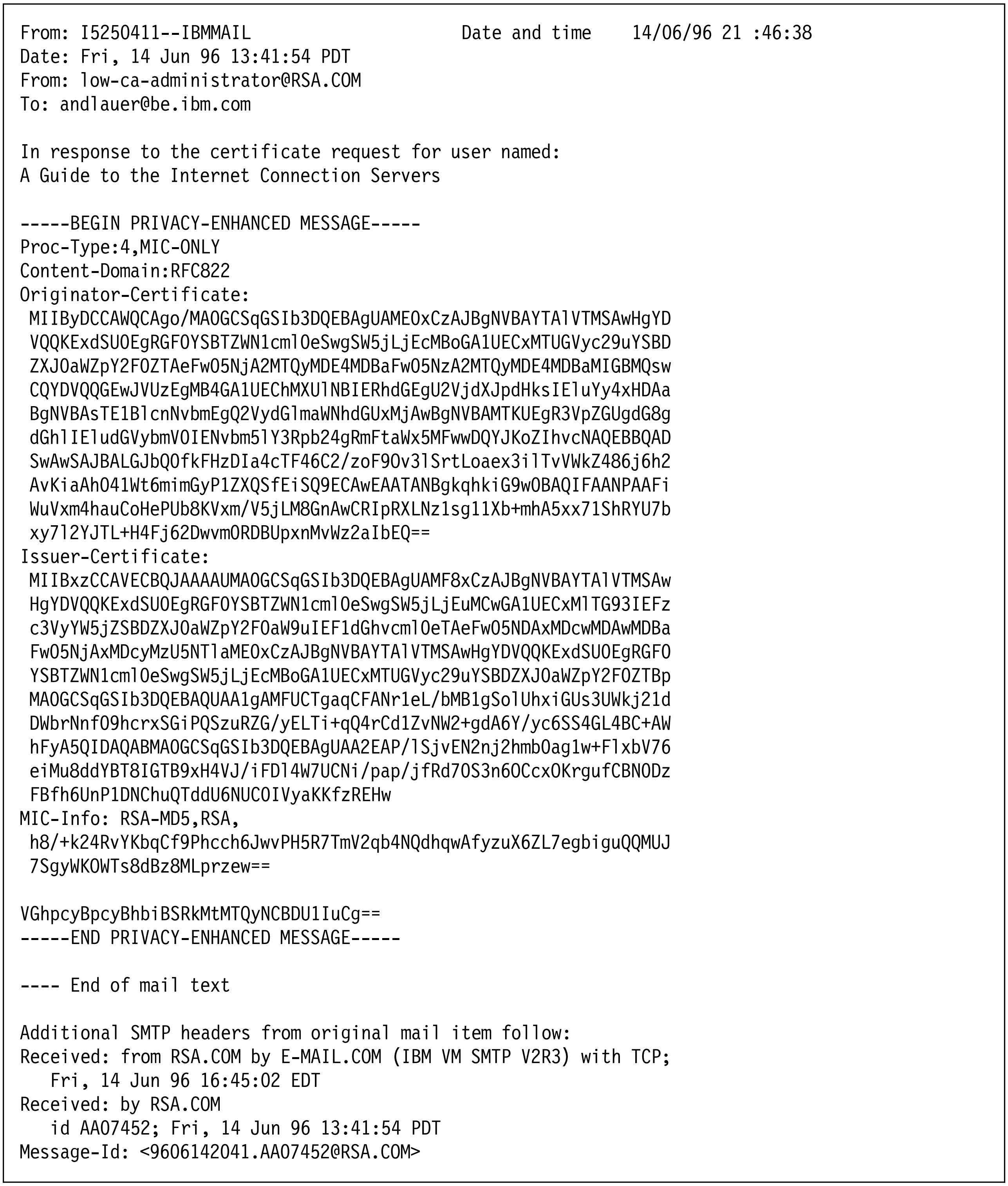
But we had made sure the certificate request would be saved in the C:/Key/CertReq.txt file (see Figure 79 on page 136). We then took the content of this file and placed it in an e-mail to persona-request@rsa.com, as shown in Figure 80. If you also do this, make sure that the characters in the string are not translated in any way while you copy the request from the file to the e-mail, and when this e-mail is sent from your host to the recipient.



```
From: ANDLAUER--BRUVMIS1                    Date and time    14/06/96 22 :39:38
To: INTERNET--IBMMAIL

From   : Jean-Charles Andlauer (andlauer@be.ibm.com)
Subject: Request Certificate
=======================================================================
/internet
/to persona-request@rsa.com
/end
-----BEGIN PRIVACY-ENHANCED MESSAGE-----
Proc-Type: 4,MIC-ONLY
Content-Domain: RFC822
Originator-Certificate:
 MIIB7TCCAZcCBDHByRUwDQYJKoZIhvcNAQECBQAwgYExCzAJBgNVBAYTAlVTMSAw
 HgYDVQQKExdSUOEgRGFOYSBTZWN1cmlOeSwgSW5jLjEcMBoGA1UECxMTUGVyc29u
 YSBDZXJOaWZpY2FOZTEyMDAGA1UEAxMpQSBHdWlkZSBObyBOaGUgSW5OZXJuZXQg
 Q29ubmVjdGlvbiBGYW1pbHkwGhcLOTYwNjEOMjAxOFoXCzk3MDYxNDIwMThaMIGB
 MQswCQYDVQQGEwJVUzEgMB4GA1UEChMXUlNBIERhdGEgU2VjdXJpdHksIEluYy4x
 HDAaBgNVBAsTE1BlcnNvbmEgQ2VydGlmaWNhdGUxMjAwBgNVBAMTKUEgR3VpZGUg
 dG8gdGhlIEludGVybmVOIENvbm5lY3Rpb24gRmFtaWx5MFwwDQYJKoZIhvcNAQEB
 BQADSwAwSAJBALGJbQOfkFHzDIa4cTF46C2/zoF9Ov3lSrtLoaex3ilTvVWkZ486
 j6h2AvKiaAhO41Wt6mimGyP1ZXQSfEiSQ9ECAwEAATANBgkqhkiG9wOBAQIFAANB
 AIyJvuw7zcOq69mxAmhwAA8OWItBvaeSx6KhIw4vCqAXI49r8KxybNOuOg3XKwZh
 AYbeygbpyx5+7v52DJcrgIA=
MIC-Info: RSA-MD5,RSA,
 h8/+k24RvYKbqCf9Phcch6JwvPH5R7TmV2qb4NQdhqwAfyzuX6ZL7egbiguQQMUJ
 7SgyWKOWTs8dBz8MLprzew==

VGhpcyBpcyBhbiBSRkMtMTQyNCBDU1IuCg==
-----END PRIVACY-ENHANCED MESSAGE-----

=======================================================================
Best Regards,              Availability  Services       Tel 360 385 373
Jean-Charles Andlauer      IBM Belgium/Luxembourg       Fax 360 385 444
```

Figure 80. Certificate Request to persona-request@rsa.com

After a while we received the reply presented in Figure 81 on page 138.

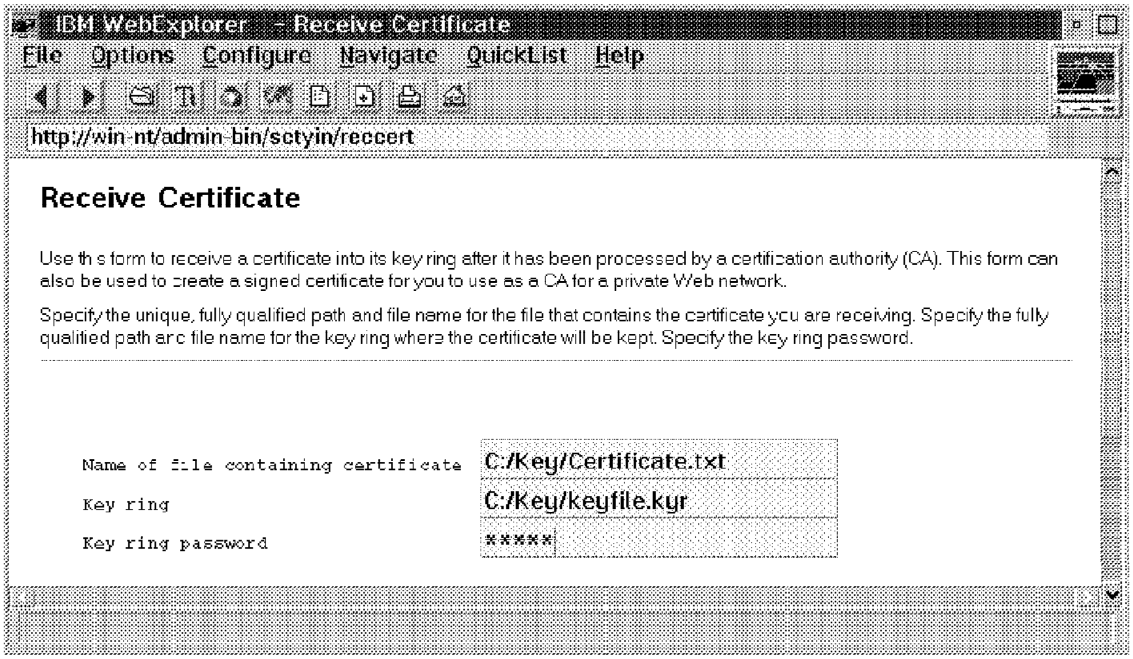


```
From: I5250411--IBMMAIL                   Date and time    14/06/96 21 :46:38
Date: Fri, 14 Jun 96 13:41:54 PDT
From: low-ca-administrator@RSA.COM
To: andlauer@be.ibm.com

In response to the certificate request for user named:
A Guide to the Internet Connection Servers

-----BEGIN PRIVACY-ENHANCED MESSAGE-----
Proc-Type:4,MIC-ONLY
Content-Domain:RFC822
Originator-Certificate:
 MIIByDCCAWQCAgo/MAOGCSqGSIb3DQEBAgUAMEOxCzAJBgNVBAYTAlVTMSAwHgYD
 VQQKExdSUOEgRGFOYSBTZWN1cmlOeSwgSW5jLjEcMBoGA1UECxMTUGVyc29uYSBD
 ZXJOaWZpY2FOZTAeFwO5NjA2MTQyMDE4MDBaFwO5NzA2MTQyMDE4MDBaMIGBMQsw
 CQYDVQQGEwJVUzEgMB4GA1UEChMXUlNBIERhdGEgU2VjdXJpdHksIEluYy4xHDAa
 BgNVBAsTE1BlcnNvbmEgQ2VydGlmaWNhdGUxMjAwBgNVBAMTKUEgR3VpZGUgdG8g
 dGhlIEludGVybmVOIENvbm5lY3Rpb24gRmFtaWx5MFwwDQYJKoZIhvcNAQEBBQAD
 SwAwSAJBALGJbQOfkFHzDIa4cTF46C2/zoF9Ov3lSrtLoaex3ilTvVWkZ486j6h2
 AvKiaAhO41Wt6mimGyP1ZXQSfEiSQ9ECAwEAATANBgkqhkiG9wOBAQIFAANPAAFi
 WuVxm4hauCoHePUb8KVxm/V5jLM8GnAwCRIpRXLNz1sg11Xb+mhA5xx71ShRYU7b
 xy7l2YJTL+H4Fj62DwvmORDBUpxnMvWz2aIbEQ==
Issuer-Certificate:
 MIIBxzCCAVECBQJAAAAUMAOGCSqGSIb3DQEBAgUAMF8xCzAJBgNVBAYTAlVTMSAw
 HgYDVQQKExdSUOEgRGFOYSBTZWN1cmlOeSwgSW5jLjEuMCwGA1UECxMlTG93IEFz
 c3VyYW5jZSBDZXJOaWZpY2FOaW9uIEF1dGhvcmlOeTAeFwO5NDAxMDcwMDAwMDBa
 FwO5NjAxMDcyMzU5NTlaMEOxCzAJBgNVBAYTAlVTMSAwHgYDVQQKExdSUOEgRGFO
 YSBTZWN1cmlOeSwgSW5jLjEcMBoGA1UECxMTUGVyc29uYSBDZXJOaWZpY2FOZTBp
 MAOGCSqGSIb3DQEBAQUAA1gAMFUCTgaqCFANr1eL/bMB1gSolUhxiGUs3UWkj21d
 DWbrNnfO9hcrxSGiPQSzuRZG/yELTi+qQ4rCd1ZvNW2+gdA6Y/yc6SS4GL4BC+AW
 hFyA5QIDAQABMAOGCSqGSIb3DQEBAgUAA2EAP/lSjvEN2nj2hmbOag1w+FlxbV76
 eiMu8ddYBT8IGTB9xH4VJ/iFDl4W7UCNi/pap/jfRd7OS3n6OCcxOKrgufCBNODz
 FBfh6UnP1DNChuQTddU6NUCOIVyaKKfzREHw
MIC-Info: RSA-MD5,RSA,
 h8/+k24RvYKbqCf9Phcch6JwvPH5R7TmV2qb4NQdhqwAfyzuX6ZL7egbiguQQMUJ
 7SgyWKOWTs8dBz8MLprzew==

VGhpcyBpcyBhbiBSRkMtMTQyNCBDU1IuCg==
-----END PRIVACY-ENHANCED MESSAGE-----

---- End of mail text

Additional SMTP headers from original mail item follow:
Received: from RSA.COM by E-MAIL.COM (IBM VM SMTP V2R3) with TCP;
   Fri, 14 Jun 96 16:45:02 EDT
Received: by RSA.COM
   id AA07452; Fri, 14 Jun 96 13:41:54 PDT
Message-Id: <9606142041.AA07452@RSA.COM>
```

Figure 81. Certificate from low-ca-administrator@rsa.com

We then placed the certificate, that is the part of the reply placed between the Begin and End Privacy-Enhanced Message lines inclusive, on our server′s host in the C:\Key\Certificate.txt file.

All that is left to do is to receive the certificate in its key ring, using the Receive Certificate form of the Configuration and Administration forms, as shown in Figure 82 on page 139.


IBM WebExplorer - Receive Certificate
File Options Configure Navigate QuickList Help

http://win-nt/admin-bin/sctyin/reccert

Receive Certificate

Use th s form to receive a certificate into its key ring after it has been processed by a certification authority (CA). This form can also be used to create a signed certificate for you to use as a CA for a private Web network.

Specify the unique, fully qualified path and file name for the file that contains the certificate you are receiving. Specify the fully qualified path arc file name for the key ring where the certificate will be kept. Specify the key ring password.

Name of file containing certificate	C:/Key/Certificate.txt
Key ring	C:/Key/keyfile.kyr
Key ring password	*****

Figure 82. Receive Certificate Form

Once this is done successfully, use the Key Management forms to manage your keys.

In order to activate the changes described above it is necessary to shut down and restart your server again. (Restarting the server from the graphical user interface or the Administration and Configuration forms is not enough.)

After restart you are ready to serve files securely through SSL.

4.6.5 Proxy Functions

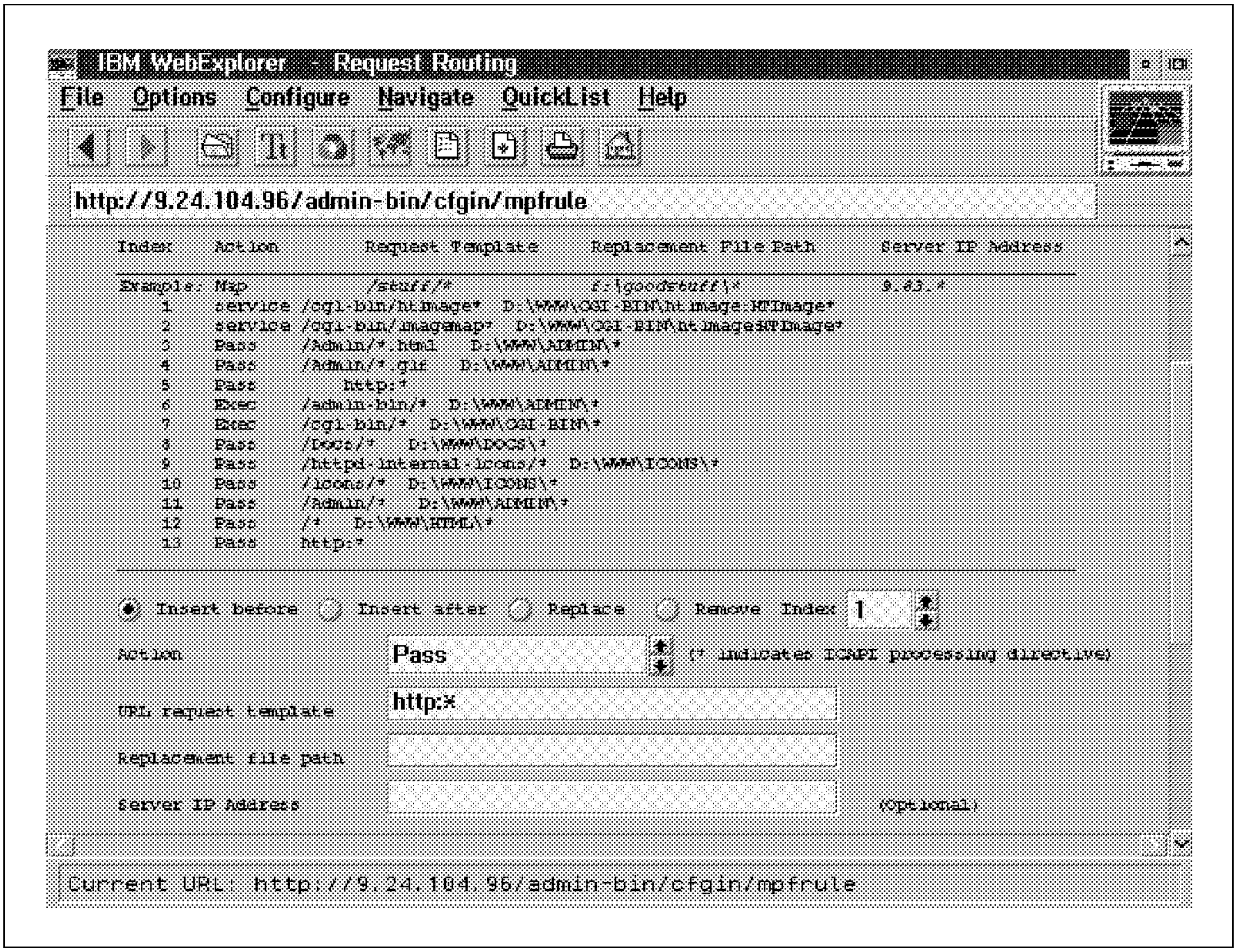
The Internet Connection server can be configured to retrieve documents on behalf of the Web browsers which it is serving. As such it is referred to as a proxy server. In addition, you can configure the proxy server to cache the retrieved documents locally.

Most Web browsers also support the local caching of documents so a part of your planning will be to decide where and when to cache. Some important choices to consider are as follows:

- Location of cache - server, browser or both
- Size of cache
- Which documents to cache
- How long to keep cached documents

In this section we use Configuration and Administration Forms to set up the Internet Connection server as a caching proxy server.

4.6.5.1 Resource Mapping - Request Routing


IBM WebExplorer - Request Routing
File Options Configure Navigate QuickList Help

http://9.24.104.96/admin-bin/cfgin/mpfrule

```
Index     Action        Request Template     Replacement File Path      Server IP Address

Example:  Map           /stuff/*             f:\goodstuff\*             9.83.*
    1     service /cgi-bin/htimage*  D:\WWW\CGI-BIN\htimage:HTImage*
    2     service /cgi-bin/imagemap*  D:\WWW\CGI-BIN\htimageHTImage*
    3     Pass     /Admin/*.html   D:\WWW\ADMIN\*
    4     Pass     /Admin/*.gif   D:\WWW\ADMIN\*
    5     Pass         http:*
    6     Exec     /admin-bin/*  D:\WWW\ADMIN\*
    7     Exec     /cgi-bin/*  D:\WWW\CGI-BIN\*
    8     Pass     /Docs/*   D:\WWW\DOCS\*
    9     Pass     /httpd-internal-icons/*  D:\WWW\ICONS\*
   10     Pass     /icons/*  D:\WWW\ICONS\*
   11     Pass     /Admin/*   D:\WWW\ADMIN\*
   12     Pass     /*   D:\WWW\HTML\*
   13     Pass     http:*
```

● Insert before ○ Insert after ○ Replace ○ Remove Index 1

Action: Pass (* indicates ICAPI processing directive)

URL request template: http:*

Replacement file path:

Server IP Address: (Optional)

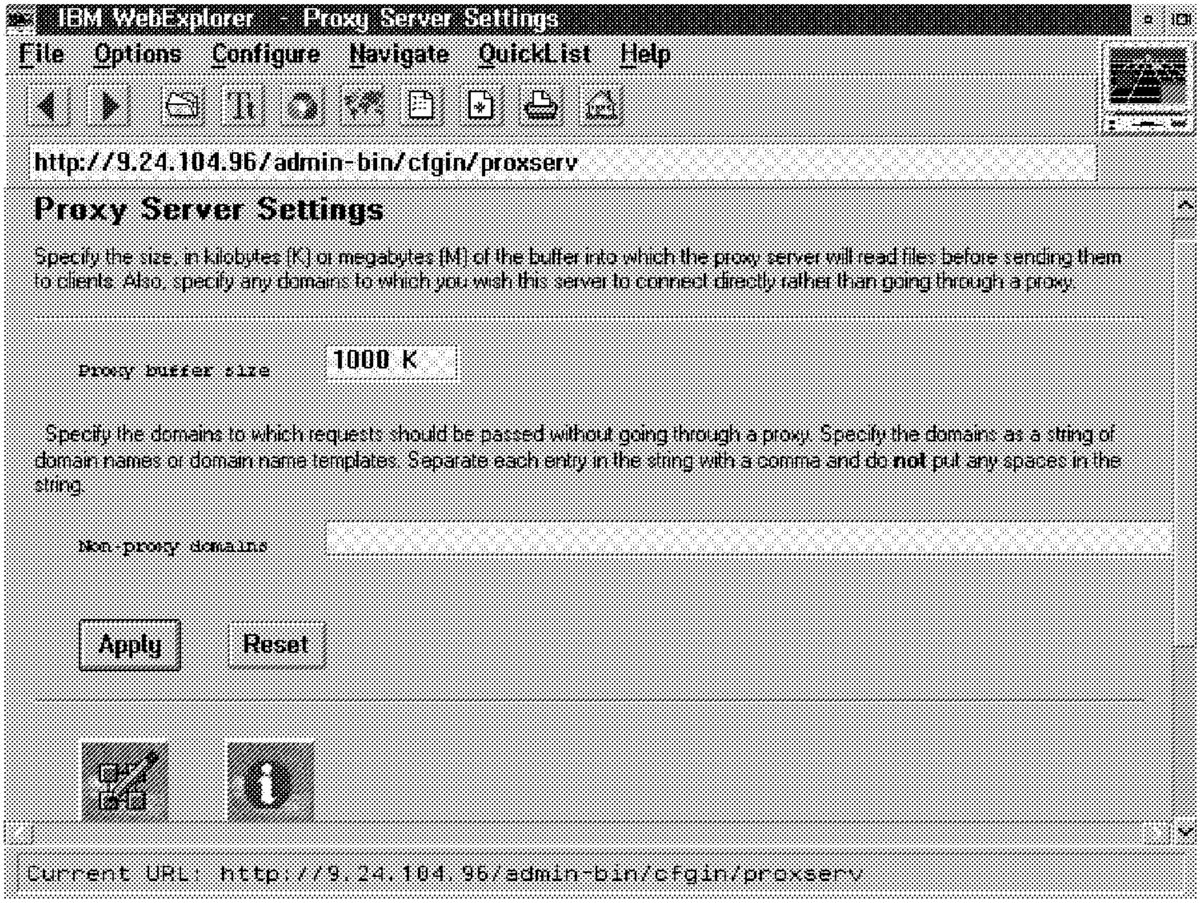
Current URL: http://9.24.104.96/admin-bin/cfgin/mpfrule

Figure 83. Request Routing

The first step is to set up rules for dealing with requests received at the server. In our case we did the following:

- Started up Configuration and Administration Forms
- Selected Request Routing and obtained the panel shown in Figure 83.
- In this panel, we specified the following:
  - A Pass rule for all HTTP requests. As you can see, ″wild card″ characters are permissible.

4.6.5.2 Proxy Server Settings


IBM WebExplorer - Proxy Server Settings
File Options Configure Navigate QuickList Help
Apply
Reset

http://9.24.104.96/admin-bin/cfgin/proxserv

Proxy Server Settings

Specify the size, in kilobytes (K) or megabytes (M) of the buffer into which the proxy server will read files before sending them to clients. Also, specify any domains to which you wish this server to connect directly rather than going through a proxy.

Proxy buffer size 1000 K

Specify the domains to which requests should be passed without going through a proxy. Specify the domains as a string of domain names or domain name templates. Separate each entry in the string with a comma and do **not** put any spaces in the string.

Non-proxy domains

Current URL: http://9.24.104.96/admin-bin/cfgin/proxserv

Figure 84. Proxy Server Settings

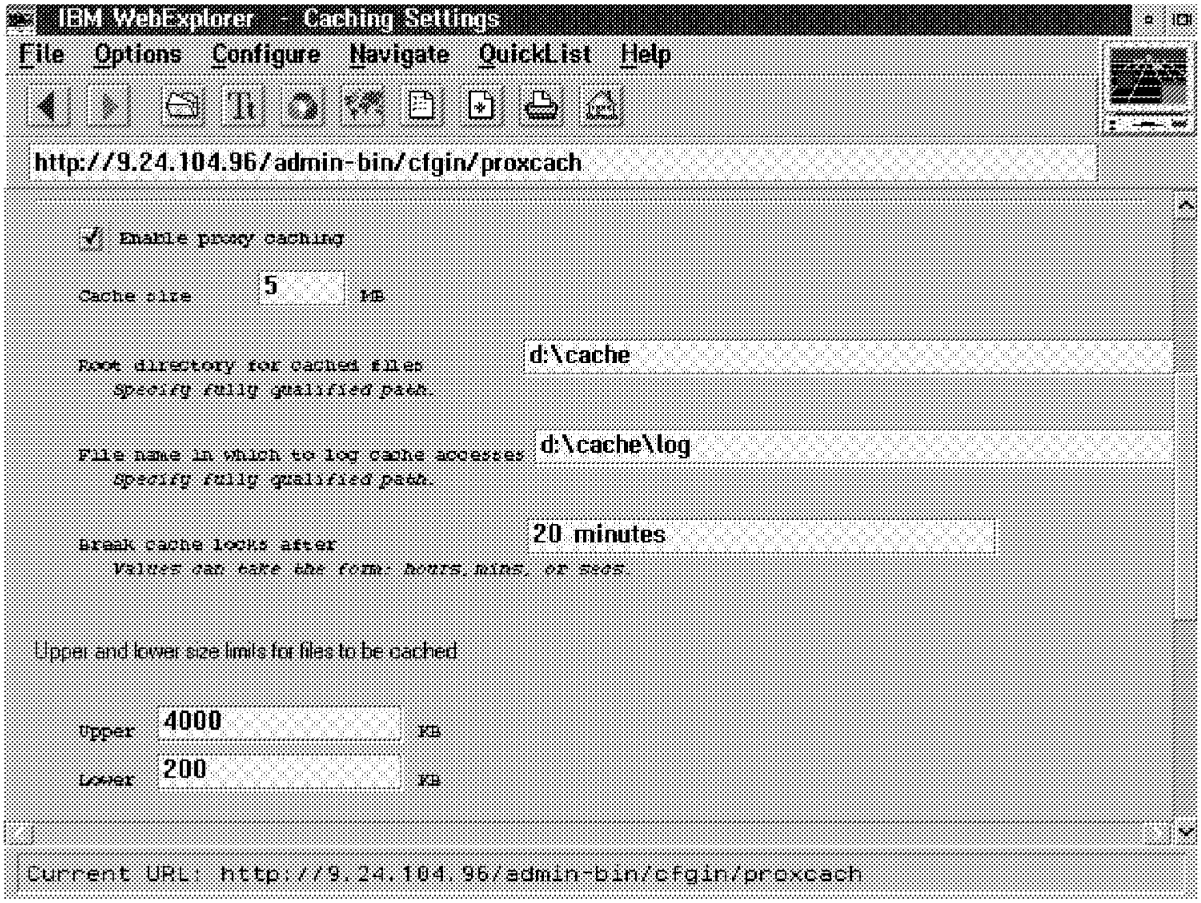
The next step is to specify the proxy buffer size and any non-proxy domains. We did this as follows:

- Selected Proxy Server Settings and obtained the panel shown in Figure 84.
- In this panel, we specified the following:
  - A proxy buffer with a size of 1000 KB
  - No non-proxy domains

    The non-proxy domain option is most useful if you are using multiple proxies. Let us suppose that there were a second proxy on a firewall which protected the intranet from the Internet. Let us further suppose that this proxy would forward requests bound for the Internet to the firewall proxy. In this case it would be appropriate to limit the forwarding of requests to the firewall proxy by specifying non-proxy domains within our intranet.

    Since we are not specifying multiple proxies, we have no need to specify non-proxy domains.

4.6.5.3 Caching Settings



http://9.24.104.96/admin-bin/cfgin/proxcach

☑ Enable proxy caching

Cache size 5 MB

Root directory for cached files d:\cache
Specify fully qualified path.

File name in which to log cache accesses d:\cache\log
Specify fully qualified path.

Break cache locks after 20 minutes
Values can take the form: hours, mins, or secs.

Upper and lower size limits for files to be cached

Upper 4000 KB

Lower 200 KB

Current URL: http://9.24.104.96/admin-bin/cfgin/proxcach

Figure 85. Caching Settings

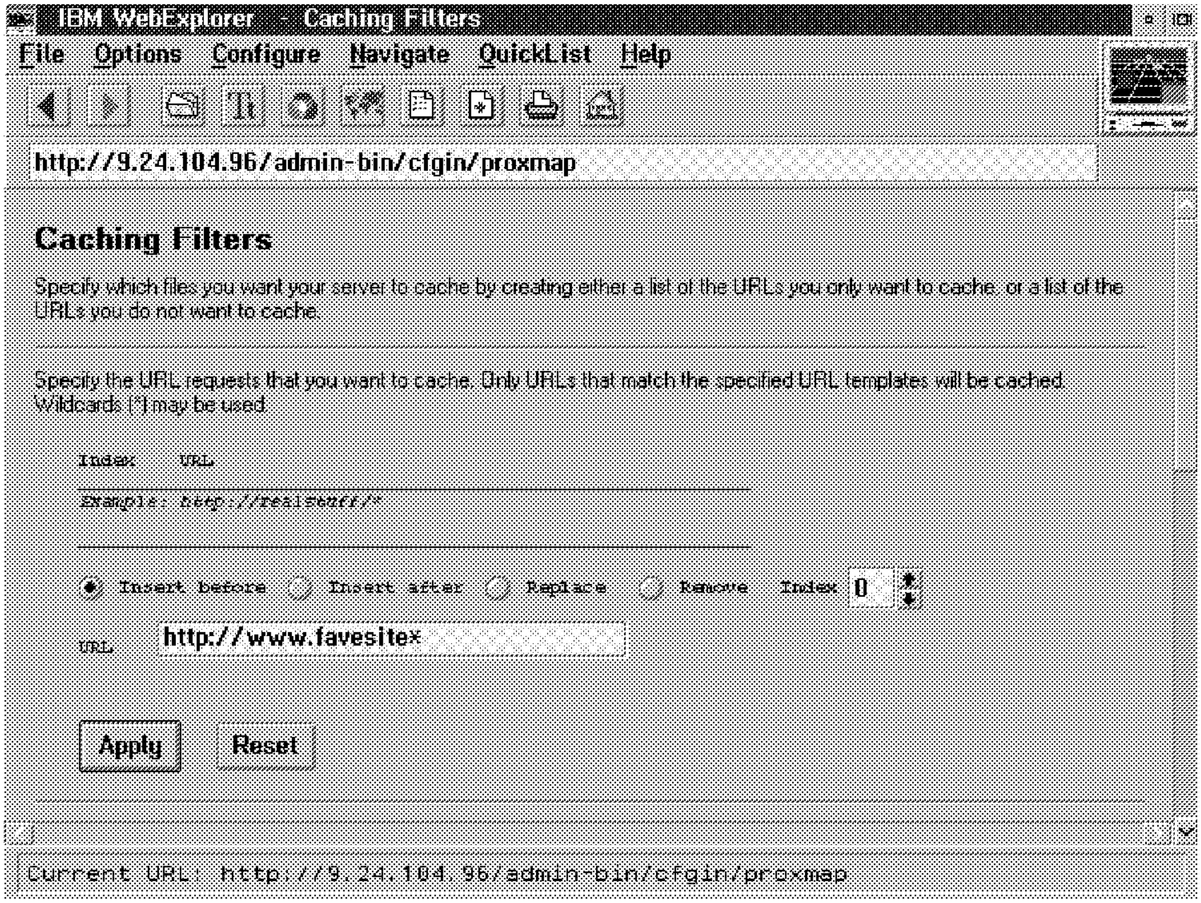
The next step is to enable proxy caching and related parameters. We did this as follows:

- Selected Caching Settings and obtained the panel shown in Figure 85.
- In this panel, we specified the following:
  - Enable proxy caching (by clicking on the button)
  - A cache size of 5 MB
  - A caching root directory of d:\cache
  - A cache log file of d:\cache\log
  - Other parameters were allowed to default

  The caching root directory d:\cache\log will be empty to start with but, once Web browsers start making use of the proxy server, it will start to fill up.

  For example, suppose that a browser makes a request for the IBM home page at www.ibm.com. The proxy server will create additional directories off of the root for HTTP requests and, in particular, those destined for www.ibm.com. Hence, all HTML files and associated images that are retrieved from www.ibm.com will be cached in the d:\cache\http\www.ibm.com directory.

4.6.5.4 Caching Filters



http://9.24.104.96/admin-bin/cfgin/proxmap

Caching Filters

Specify which files you want your server to cache by creating either a list of the URLs you only want to cache, or a list of the URLs you do not want to cache.

Specify the URL requests that you want to cache. Only URLs that match the specified URL templates will be cached. Wildcards (*) may be used.

Index URL

*Example: http://realstuff/**

● Insert before ○ Insert after ○ Replace ○ Remove Index 0

URL http://www.favesite*

Apply Reset

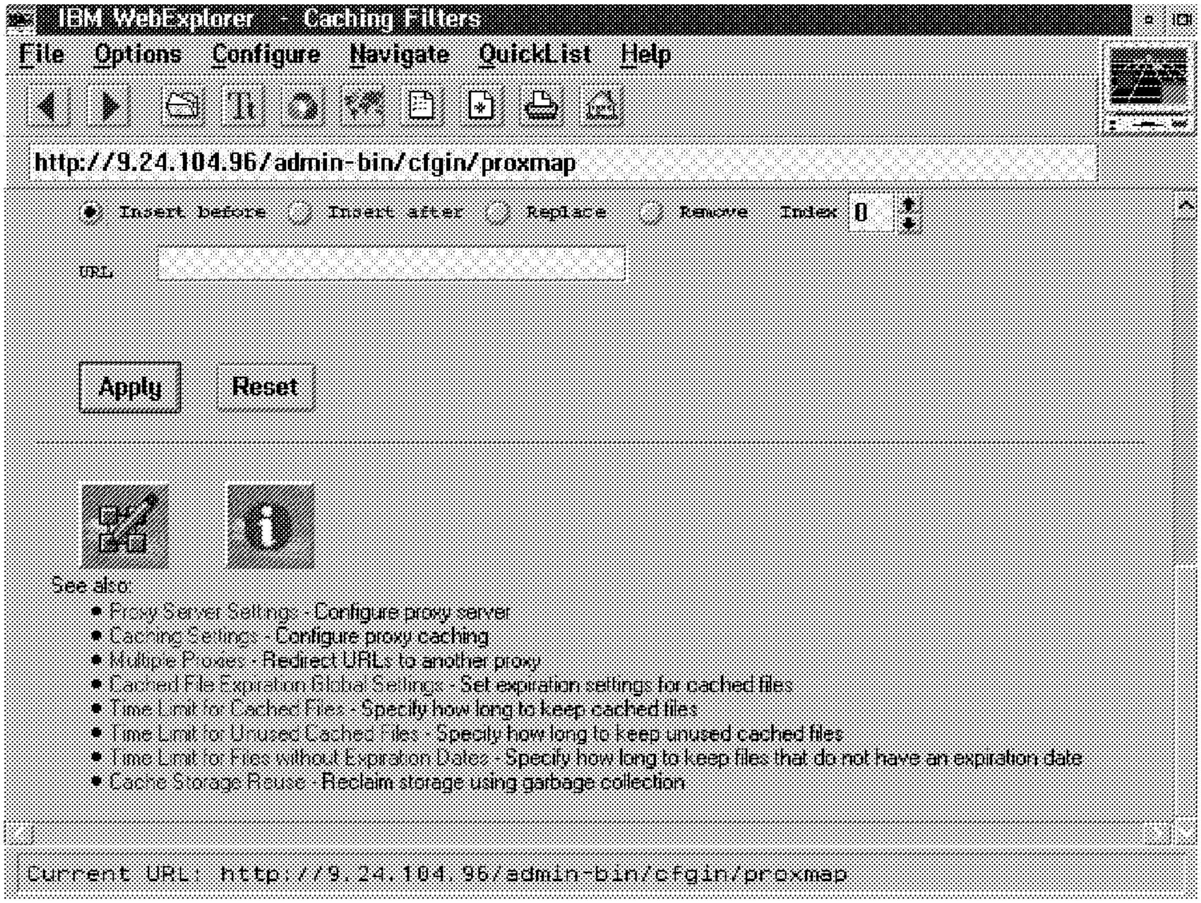
Current URL: http://9.24.104.96/admin-bin/cfgin/proxmap

Figure 86. Caching Filters - 1 of 2

The next step is to optionally specify caching filters that will determine from which URLs documents will be retrieved by the proxy server and cached locally. We did this as follows:

- Selected Caching Filters and obtained the panel shown in Figure 86.
- In this panel, we specified the following:
  - Requests for URLs starting http://www.favesite should be cached

  As you have no doubt guessed, this is a made up example. If you do not specify any caching filters then requests for all HTTP URLs will be cached. We chose initially in this example to cache only HTTP requests (see 4.6.5.1, “Resource Mapping - Request Routing” on page 140). However, other protocols, FTP for example, are also supported.



http://9.24.104.96/admin-bin/cfgin/proxmap

● Insert before ○ Insert after ○ Replace ○ Remove Index 0

URL

See also:

- Proxy Server Settings - Configure proxy server
- Caching Settings - Configure proxy caching
- Multiple Proxies - Redirect URLs to another proxy
- Cached File Expiration Global Settings - Set expiration settings for cached files
- Time Limit for Cached Files - Specify how long to keep cached files
- Time Limit for Unused Cached Files - Specify how long to keep unused cached files
- Time Limit for Files without Expiration Dates - Specify how long to keep files that do not have an expiration date
- Cache Storage Reuse - Reclaim storage using garbage collection

Current URL: http://9.24.104.96/admin-bin/cfgin/proxmap

Figure 87. Caching Filters - 2 of 2

At this stage we had completed the main steps in setting up our Internet Connection server as a caching proxy. In Figure 87 we see that there are further options that allow us to set time limits for cache files, specify multiple proxies, etc.

All that remained at this stage was to click on the Apply button to make these changes. As always where configuration changes are concerned it was also necessary to restart the server to put the changes into effect.

4.6.6 Tuning the Server

Server performance will be influenced by the following directives:

- MinActiveThreads
- MaxActiveThreads
- CacheLocalFile
- IdleThreadTimeout

Each time the server receives a request from a client, it uses one or two threads to perform the requested action. One thread is used if the server is not performing DNS lookup. The server first checks to see if any threads are available. If so, the server uses available threads to process the request. If no threads are available and the maximum number of active threads has not been reached, the server starts new threads to process the request. If the maximum number of active threads has been reached, the server holds the request until threads are available. When a request finishes, the threads it was using become idle. As long as idle threads do not expire, they are available for the server to use again.

The following are the meanings of the performance-related directives:

- **MinActiveThreads**

Specify the minimum number of threads to keep available or active. Use this directive to set the minimum number of threads that you want the server to either be using or have available to use. The server will not close threads below this minimum even if the threads are idle. Generally, the more power your machine has, the higher the value you should use for this directive. If your machine starts to spend too much time on overhead tasks, such as swapping memory, try reducing this value.

Default: MinActiveThreads 20

- **MaxActiveThreads**

Specify the maximum number of threads to keep available or active. Use this directive to set the maximum number of threads that you want to have active at one time. If the maximum is reached, the server holds new requests until another request finishes and threads become available. Generally, the more power your machine has, the higher the value you should use for this directive. If your machine starts to spend too much time on overhead tasks, such as swapping memory, try reducing this value.

Default: MaxActiveThreads 40

- **CacheLocalFile**

Specify files you want to load in memory at startup. Use this directive to specify the names of files you want to load into the server′s memory each time you start the server. You can have multiple occurrences of this directive in the configuration file. Include a separate directive for each file you want to load into memory. By keeping your most frequently requested files loaded in the server′s memory, you can improve your server′s response time for those files. For example, if you load your server′s welcome page into memory at startup, the server can handle requests for the page much more quickly than if it had to read the file from a disk. Keep in mind that for each file you load into memory, you are making that amount of memory unavailable for other uses.

Before responding to a request for a file that is stored in memory, the server checks to see if the file has changed since the server was started. If the file has changed, the server responds to the request with the updated file and deletes the old version from its memory. To load the new file into memory, you need to restart the server.

- **IdleThreadTimeout**

Specify length of time to keep idle threads available. Use this directive to specify how long the server should keep an idle thread available. A thread becomes idle after the last request to use it completes. If the number of threads already available or active is greater than the value on MinActiveThreads and the server does not use the thread again within the specified time, the server closes the idle thread. Specify the time value in any combination of hours, minutes or seconds. If you use the default value of forever, the server does not close any idle threads.

Default: IdleThreadTimeout forever

- **ServerPriority**

Specify the priority you want your server to have on your system. Use this directive to specify a priority class for the Internet Connection Server. The operating system uses priority classes to determine which processes have priority over others. Valid values are:

- 0 - No priority
- 1 - Maximum priority as a normal process
- 2 - Maximum priority as a foreground server process

You may want to use a value of 0 if the machine of your Internet Connection Server is running on and is also processing other types or requests.

Default: ServerPriority 1

4.6.6.1 ACL Files

Use this directive to specify whether access control list files will be checked for file protection. Set this directive to never or protect only for better server performance. The format of the directive is:

```
UseACLs setting
```

The value of setting may be:

- always

The server will always look for an ACL file on every file request.

- protect only

The server will only look for an ACL file when the file request is for a file that is covered by a protection statement.

- never

The server will never look for an ACL file on request.

Examples:

```
UseACLs protect only
UseACLs always
```

4.6.6.2 Meta Files

Use this directive to specify whether meta files will be used by the server. Set this directive to off for better server performance. The format of the directive is either on or off.

```
UseMetaFiles on
UseMetaFiles off
```

The default is on.

4.6.7 Backup Procedures

Backup is based on the concept of protecting data from being lost or destroyed. The concept of backup and restore is based on one golden rule: your backup policies must be able to provide the data that is required to meet your recovery requirements. You cannot back up data without giving any thought to the restore process. The restore process, and its requirements, must drive the backup method.

In preparing backup procedures we have to consider the following:

- Human error
- Media failure
- Outage minimization
- Disaster recovery

4.6.7.1 Backup Requirements

You should make a backup of the ICS files as soon as possible after you install the ICS for the first time. This ensures that you can recover any portion of the ICS files that is accidentally erased or corrupted.

You can choose from two ways to back up the ICS related files: full backup and incremental backup. A full backup may take longer to run than an incremental backup, however, recovery time may be faster with a full backup since incremental backups do not need to be applied. If you want to back up automatically, you may use a storage manager such as Adstar Distributed Storage Manager (ADSM). In addition, you have to consider the backup cycle for the files. If you update any files related to the ICS, you should back up them as soon as possible.

At least the following files must be backed up to another machine or another media such as tapes or diskettes:

- Httpd.conf configuration file
- Password files
- Group files
- ACL files
- Content files
- Access/error log files

The three main options that are available for backing up these files are as follows:

- Back up the files to another media using standard shell commands such as cp/xcopy, tar and cpio
- Back up the files individually using ADSM
- Back up the files to a remote machine using FTP or NFS

Note: Files are often bundled together in single files by utilities like pax, tar, and cpio. When these files are bundled into a single file it is called an archive file. To save backup space and backup time, archive files can also be compressed with utilities like compress, pkzip, and lha, or the utilities by using -z option.

4.6.8 Logging Procedures

The Internet Connection server logs different types of information in various files, according to your configuration.

In this section, we describe to you the format of those files and we provide you with some guidelines on administering them. Finally, we discuss some analysis tools.

4.6.8.1 Log Files Format and Location

The server starts new log files each day at midnight if it is running. Otherwise the server starts new log files the first time you start it on a given day. When creating the file, the server uses the file names you specified in the configuration file or the predefined names, depending on the log file and appends a date suffix. The date suffix is in the format:

```
Mmmddyy
```

where Mmm is the first three letters of the month (calendar name), dd is the day of the month and yy is the last two digits of the year; for example:

```
httpd-log.Jun0696
```

Note that MVS appends an additional suffix which corresponds to the time of the first request/error on the server; for example:

```
httplog.Jun0696.0531
```

Also, note that several different files will be created if you restart your server. On the other hand, no log files are created if your server is not accessed during the day.

As you will see in the following sections, dedicated directives allow you to set up the log files′ locations. It is up to you to decide the directory (or directories) in which you want to store the log files. You must pay attention to the fact that the log files tend to grow quickly and you would probably not appreciate it if your server filled up one of your vital file systems (for example, the root (/) on an AIX system).

There is no universal law as to where the log files should be stored and how much disk space should be dedicated to them. This will depend greatly on how many disks are installed on your system, how they are partitioned, etc. But it also depends on the traffic to and from your Web server and the structure of the Web site. A service with many short pages will generate more log entries than a service with less pages, even if the amount of information sent to the client is equivalent. Pages with many images (GIF, JPEG, etc) will also generate more log entries than plain text files.

Only experience will provide you with information on how much space you should consider dedicating to your log files for your specific site. To some extent this will depend upon how often you will be backing up and purging the files. Based upon our experience we concluded that we should allow or allocate in excess of 100 MB of disk space.

Please refer to 4.6.8.8, “Log Files Administration” on page 153 for guidelines on how to administer the log files.

The information logged relates to incoming requests, internal server errors, CGI script errors, requests for cached files (if your server acts as a proxy server).

If your server is Release 4.1 (or higher), it will also log information on the referrer and the agent of incoming requests.

Each entry in the log files contains the time. You can choose between the local time and Greenwich Mean Time formats. See the LogTime directive in the configuration file. The time format is common to all log files:

```
DD/Mon/YYYY:hh:mm:ss TTTTT
```

That is date, month (calendar name), year, hour (24-hour format), minutes, seconds and time zone.

4.6.8.2 Access and Cache Log Files Format

The access log file and the cache log file can have two different formats:

- The common format which is the one used by most Web servers
- The old format which was used with early versions of Web servers from CERN

Since you will probably run a generic statistics program, you most likely will want to use the common format, which is the default (see the LogFormat directive in the configuration file).

The common format is:

```
REMOTE-ADDRESS REMOTE-IDENT REMOTE-USER [TIME] ″REQUEST″ STATUS BYTES
```

where:

REMOTE-ADDRESS is the address of the requester (that is the client′s host machine). If the server is set to reverse resolve, the IP address of the client, and if the address was successfully resolved, this information is the name of the host, otherwise it is the IP address of the host. This information is equivalent to that passed in REMOTE_HOST and REMOTE_ADDR environment variables to CGI programs.

REMOTE-IDENT is in theory the name of the user as retrieved from the client′s identd server; in practice, this mechanism has not so far been implemented so far in the Internet Connection servers so that this field is always set to - (hyphen). This information is equivalent to the REMOTE_IDENT variable that is passed to CGI programs.

REMOTE-USER is the name of the user that was provided during the authentication process if one occurred; if no name was provided, this field is set to - (hyphen). This information is equivalent to the REMOTE_USER variable that is passed to CGI programs.

TIME is the time in common or old format according to the LogTime directive.

REQUEST is the first line of the request (generally the request method followed by the URL and the HTTP version as defined in the HTTP protocol).

STATUS is the status code of the request as defined in the HTTP protocol.

BYTES is the number of bytes returned to the requester in the response (not including the HTTP header).

Here are some examples of access log entries:

```
jeansan.itso.ral.ibm.com - - [12/Jun/1996:09:45:38 +0500] \
          "GET /cgi/environ HTTP/1.0" 200 439
9.67.43.8 - webadmin [12/Jun/1996:16:42:31 +0500] \
          "GET /admin-bin/cfgin/initial HTTP/1.0" 200 6183
9.24.104.27 - - [12/Jun/1996:16:54:19 +0500] "GET /junk HTTP/1.0" 404 335
```

(Note that the \ character indicates that the entry is to be continued on the next line.)

The first entry shows an append when the DNS-Lookup directive is set to On. The second entry shows an append corresponding to an access to a protected resource thus requiring an authentication. The last entry shows an append for an error condition. (The object requested could not be found.)

The access log file name is specified in the AccessLog directive in the configuration file. There is a default path name which depends on the platform on which the server is running.

Here are examples of settings:

```
AccessLog /var/log/www/httpd-log
AccessLog D:\WWW\LOG\HTTP-LOG
AccessLog /usr/local/ServerRoot/logs/httplog
```

for AIX, OS/2 (or Windows NT) and MVS respectively.

The cache log file name is specified in the CacheAccessLog directive in the configuration file. By default, the proxy server does not log cache access requests; you have to include this directive in your configuration file.
Here are examples of settings:

```
CacheAccessLog /var/log/www_cache/httpd_cache.log
CacheAccessLog D:\WWW\LOG\CACHE-LOG
CacheAccessLog /usr/local/ServerRoot/logs/httpcache
```

for AIX, OS/2 (or Windows NT) and MVS respectively. Note that you can specify that you do not want to log access requests made from specific hosts or domains that match a given template. See the NoLog directive in the configuration file:

```
NoLog 9.* (if DNS-Lookup is Off)
NoLog *.ibm.com (if DNS-Lookup is On)
```

4.6.8.3 Logging Access Facilities

The new logging access feature in Internet Connection Server V4.1 introduces flexibility.

At the time of writing, the Internet Connection Server V4.1 is not yet generally available so the functions may change slightly.

You have the choice to log everything or selected information in the access log files (access and proxy cache). Filters for URL, request method, MIME type and return code can be defined to determine what level of information will be logged in the access log.

By default, all of the information is logged to the access log files. The following directives in the configuration setup define the filter settings:

- AccessLogExcludeURL
- AccessLogExcludeMethod

- AccessLogExcludeReturnCode
- AccessLogExcludeMimeType

However, unless you have disk space restrictions, you will most likely keep all information logged and set up report filter rules (see 4.6.8.9, “Log Files Analysis” on page 154). The more information that is logged, the more accurate and exhaustive will be the resulting reports.

4.6.8.4 Error Log File Format

The Internet Connection Server logs errors in a separate file. It logs internal errors and HTTP protocol errors.

Internal errors are logged using the following format:

```
[TIME] TEXT
```

where:

TIME is the time in common or old format according to the LogTime directive.

TEXT is a text phrase describing the internal server error.

Internal server errors include service messages, warning and error reports. The text is self explanatory and should help you with problem diagnosis.

The HTTP protocol errors are logged using the following format:

```
[TIME] [STATUS] [REQUESTER] DESCRIPTION
```

where:

TIME is the time in common or old format according to the LogTime directive.

STATUS is the status returned in the response to the requester; this is expressed in text (MULTI FAILED, NOT AUTHORIZED, etc.).

REQUESTER describes the identity of the requester including host name and, if available, the referrer URL. (Not all browsers send this information.)

DESCRIPTION gives more information on the nature of the error; for example the forbidden or unauthorized URL.

Here are some examples of error log entries:

```
Auto Login file for Key Ring was not found.
Secure Server was started by SRC function and requires an Auto Login file.
[12/Jun/1996:16:54:19 +0500] [MULTI FAILED] [host: 9.24.104.27] /junk
```

The two first entries are internal server errors; they were appended at server startup and indicate that the resources required for security facilities were not found. The last entry corresponds to an HTTP protocol error; the object requested could not be found (error code 404).

The error log file name is specified in the *ErrorLog* directive in the configuration file. There is a default path name which depends on the platform on which the server is running. The following are examples of settings for AIX, OS/2 (or Windows NT) and MVS respectively:

```
ErrorLog /var/log/www/error-log
ErrorLog D:\WWW\LOG\ERROR-LOG
ErrorLog /usr/local/ServerRoot/logs/errorlog
```

4.6.8.5 CGI Script Errors Log File

According to the Common Gateway Interface, what the program writes on STDOUT (standard output) is sent as is to the requester. But the CGI program may also write output to STDERR. All the messages sent to STDERR (standard error) are logged in the CGI error log file.

There is no prototype for error messages. Therefore the CGI error log file has no specific format. It is up to the CGI scripts developer to define a common format if required for maintenance purposes.

Note that for debug purposes this error log file is very useful. Debugging messages sent to STDERR will be appended to the CGI error file.

The CGI error file name and file path are predefined. This file is located in the same directory as the error log file and is named cgi_error.

This file is not created when running Internet Connection Server for MVS. In any case, the system log gives all required information on execution errors.

4.6.8.6 Referer Log File Format

The referer log file exists only on Internet Connection Server V4.1 (or higher). The server records in this file referer information using the following format:

```
[TIME] ″REFERER″
```

where:

TIME is the time in common or old format according to the LogTime directive.

REFERER is the last location of the browser, equivalent to the REFERER_URL variable passed to CGI programs. Note that this information is not sent by all browsers and, for those that do, only in certain circumstances.

Here are some examples of referer log entries:

```
[12/Jun/1996:08:00:49 +0500]    ″http://rs60004.itso.ral.ibm.com/shapes/″
[12/Jun/1996:08:01:14 +0500]    ″″
```

The first entry is a standard one in that the browser sent the information related to the last location. The second entry corresponds to a case where the information was not given by the browser.

The referer log file name and file path are predefined. This file is located in the same directory as the access log file and is named referer_log.

4.6.8.7 Agent Log File Format

The agent log file exists only in Internet Connection Server V4.1 (or higher). The server records the user agent information in this file using the following format:

```
[TIME] ″AGENT″
```

where:

TIME is the time (as described above).

AGENT is the name of the browser; equivalent to the HTTP_USER_AGENT variable passed to CGI programs.

Here are some examples of agent log entries:

```
[04/Jun/1996:11:05:49 +0500] ″IBM WebExplorer DLL /v960311 Beta″
[04/Jun/1996:12:47:35 +0500] ″Mozilla/3.0b3 (X11; I; AIX 1)″
```

The agent log file name and file path are predefined. This file is located in the same directory as the access log file and is named agent_log.

4.6.8.8 Log Files Administration

No matter where the log files are stored, they tend to grow. You need a mechanism to manage the logs.

The mechanism will make sure that the log files are:

- Sent to an archive
- Compressed if they are going to be stored for some time on the system
- Removed when they are old

The mechanism will be implemented according to your platform. You should consider activating it daily or at least weekly. The corresponding procedure will be launched using the scheduling tools available on your platform (cron for AIX, AT for Windows NT, Automated Operations Control for MVS).

This backup and purge procedure will be generally preceded by the log analysis procedure. You will find a discussion on log analysis in 4.6.8.9, “Log Files Analysis” on page 154.

In order to help you with log file maintenance, Internet Connection Server V4.1 introduces an automatic log removal capability.

Access log and error logs can be purged based on their age (in days) or their collective size (in megabytes). For each log, you have the option of turning purge on or off or of setting up a user exit program.

```
AccessLogArchive none/purge/userexit
ErrorLogArchive none/purge/userexit
```

Examples:

```
AccessLogArchive none
ErrorLogArchive purge
ErrorLogArchive userexit C:\WWWCODE\MVLOGS.EXE
```

When selected, the purge actions or userexit action will take place at midnight immediately after the previous day′s logs have been closed and the new day′s logs have been opened.

If you turn on the purge option, you can set up the thresholds for purging to occur. Files are removed based on the settings defined for the AccessLogExpire and AccessLogSizeLimit directives. If any logs are older than the AccessLogExpire setting, they are remote. The size of the remaining logs is then calculated; if the sum total of the remaining log exceeds the value of the AccessLogSizeLimit directive, the server will remove log files, starting with the oldest one, until the sum total size of the logs falls below the value of the AccessLogSizeLimit directive.

For example, to remove files older than 14 days without any size consideration, the settings would be:

```
AccessLogArchive purge
AccessLogExpire  14
AccessLogSizeLimit 0
```

If you activate this purge mechanism (or any other purge mechanism), pay attention that you do not remove the files before they are analyzed.

If you select the user exit option, the server will call on a daily basis the exit specified. To compress error log files older than 7 days, the settings would be as follows on a UNIX platform:

```
ErrorLog /var/log/www/httpd-errors
ErrorLogArchive userexit /usr/local/server_root/maintenance/compress_log
```

where /usr/local/server_root/maintenance/compress_log is a basic shell script:

```
#!/bin/ksh
# compress_log
#
# This script is invoked as an user exit by the server or htlogrep binary
# It is invoked with the list of log files to handle as arguments
#
# compress_log compresses log files older than 3 days

for file in $*
do
        find $file -mtime +3 -exec compress {} \;
done
```

Note that the user exit program will be invoked with the log files′ names as an argument. In the previous example, compress_log is invoked using the following command:

```
/usr/local/server_root/compress_log /var/log/www/httpd-errors*
```

4.6.8.9 Log Files Analysis

Analyzing the log files is a job that all Webmasters should consider doing regularly and certainly before the files are erased for disk space considerations.

Analyzing the log files means on the one hand detecting internal errors, link failures, etc., and on the other hand monitoring the usage of the Web site.

The error log file, as we have seen in 4.6.8.1, "Log Files Format and Location" on page 148, is a list of all the troubles your server has and also the troubles people have accessing the pages on your site. Keeping an eye on this log file is mandatory and will probably be the job of the Webmaster or systems administrator. Because browsing this file with the naked eye is not realistic and because you would probably like to know immediately what is going on, you should consider using software to monitor your system (daemons, disk space, CPU utilization, etc.). The file monitoring capability of this tool will check for updates and error messages and generate alerts if required.

The most important job for the Webmaster is probably access log analysis.

By monitoring the usage of your Web site, you will find out:

- How many people are accessing your Web site
- Which pages are the most popular
- Where people are coming from

- Which browser the clients are using
- How many bytes were sent

Depending on the level of information you expect from the analysis of the log files, you will probably decide either to develop your own analysis programs or to use one of the tools available on the Internet or simply to use the new log reporting feature implemented in Internet Connection Server V4.1.

4.6.8.10 Internet Connection Server Reporting Feature

New functions related to logging and reporting activities have been implemented in Internet Connection Server V4.1. At the time of writing, the product is not yet available so that the functions may change somewhat. Nevertheless, it is worth having a look at these functions since their usage will avoid having to implement an additional tool.

These new facilities allow users to create HTML reports for the entire server, or for specific files and/or directories on the server.

You can create your own templates for server access reports. The resulting reports give daily, weekly, monthly and yearly statistics on:

- Requestor IP/Hostname
- Requested URL/Filename

All the access reports will be stored in a unique directory according to the value specified in AccessReportRoot in your configuration file.

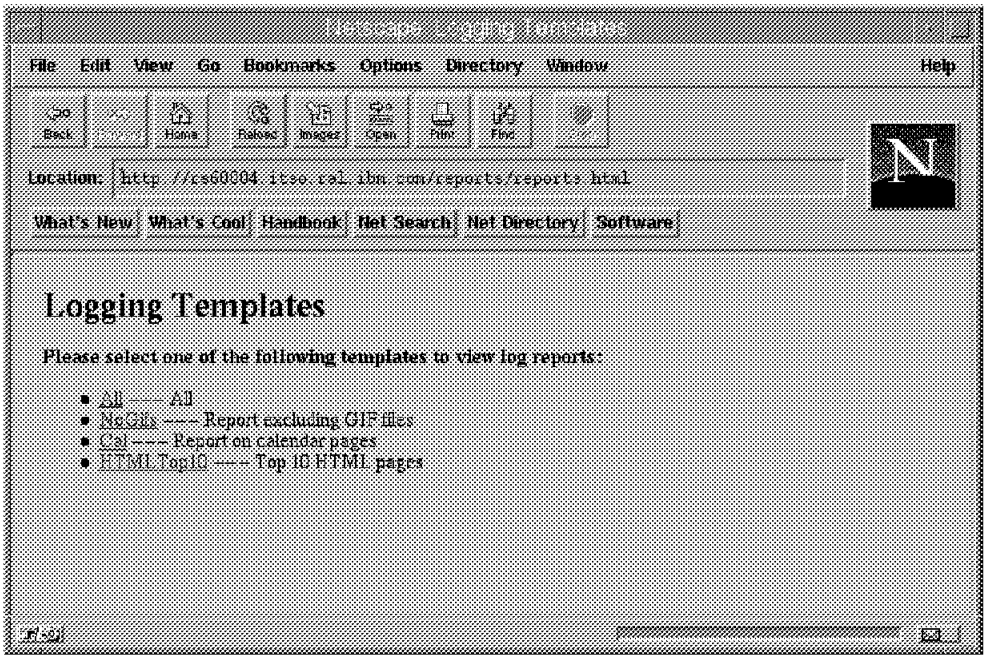
Example:

```
AccessReportRoot D:\WWW\REPORTS
```

The syntax of a template is:

```
AcessReportTemplate TemplateName {
     AccessReportDescription   Report Description
     (template subdirective)
     (template subdirective)
     etc
}
```

Although the AccessReportDescription is optional, we recommend that you use it for readability reasons. In fact, this description appears on the front page generated by the reporting tool as shown in Figure 88 on page 156.



http://rs60004.itso.ral.ibm.com/reports/reports.html

Logging Templates

Please select one of the following templates to view log reports:

- All ---- All
- NoGifs ---- Report excluding GIF files
- Cal ---- Report on calendar pages
- HTML Top10 ---- Top 10 HTML pages

Figure 88. Logging Templates

The subdirectives that can be used in a report template definition to create customized reports are:

- AccessReportTopList to specify the top number of items on which to report (All for all items, which is a default, otherwise a number.)
- AccessReportExcludeURL to exclude access log records with URLs matching a given template
- AccessReportIncludeURL to include access log records with URLs matching a given template
- AccessReportExcludeHostName to exclude access log records with hostnames matching a given template
- AccessReportIncludeHostName to include access log records with hostnames matching a given template
- AccessReportExcludeMethod to exclude access log records with a given method
- AccessReportExcludeReturnCode to exclude access log records with given range (Valid values are 200, 300, 400 and 500.)

You can create as many report templates as you wish. The only known limits are your creativity and the number of files that you agree to create in your report directory. (The log reporting tool is quite prolific.)

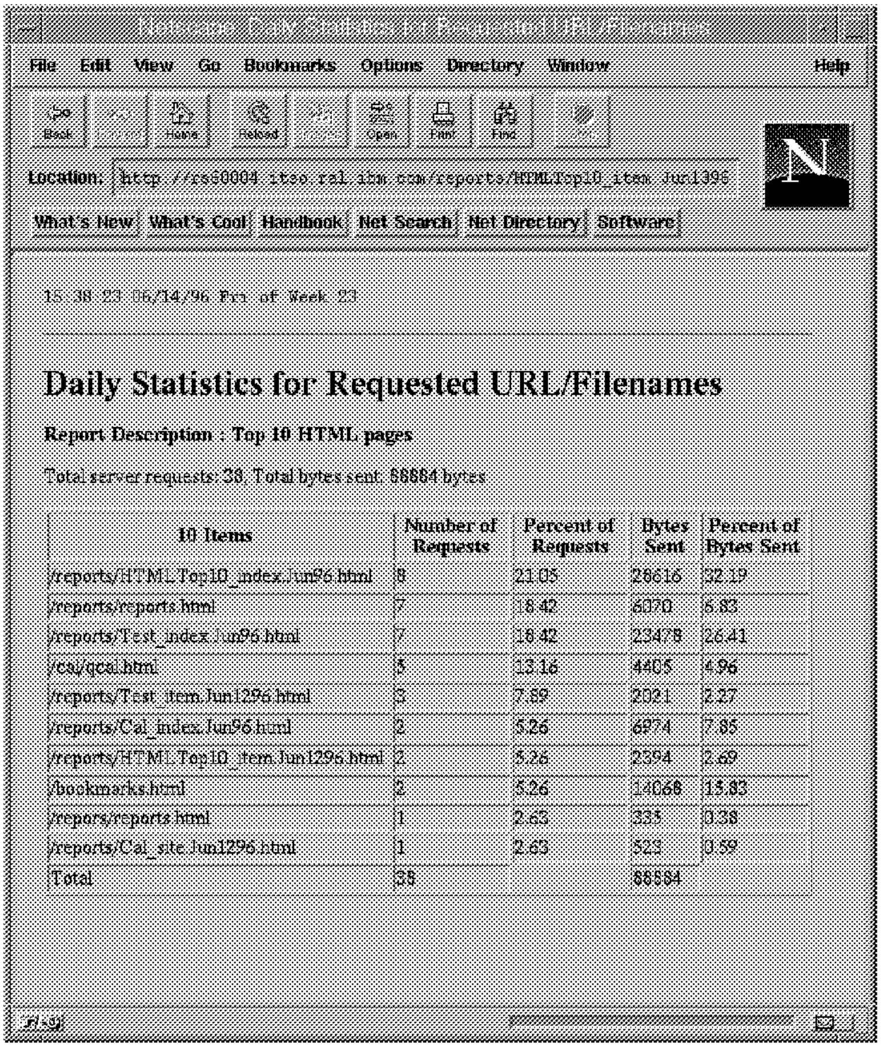
Let us take a look at a simple scenario.

In order to focus on the design of your more accessed pages you want to determine the top 10 HTML pages of your Web server. But the users in your intranet do not make ″standard″ use of your server and you want your report to reflect the customers′ accesses.

The resulting report template configuration would appear as:

```
AccessReportTemplate HTMLTop10 {
      AccessReportDescription Top 10 HTML pages
      AccessReportTopList     10
      AccessReportIncludeURL  *.html
      AccessReportExcludeHostname  9.*
      AccessReportExcludeHostname  *.ibm.com
}
```

and you would obtain a HTML page such as the one shown in Figure 89.



http://rs60004.itso.ral.ibm.com/reports/HTMLTop10_item.Jun1396

15:38:23 06/14/96 Fri of Week 23

Daily Statistics for Requested URL/Filenames

Report Description : Top 10 HTML pages

Total server requests: 38, Total bytes sent: 88884 bytes

10 Items	Number of Requests	Percent of Requests	Bytes Sent	Percent of Bytes Sent
/reports/HTMLTop10_index.Jun96.html	8	21.05	28616	32.19
/reports/reports.html	7	18.42	6070	6.83
/reports/Test_index.Jun96.html	7	18.42	23478	26.41
/cal/qcal.html	5	13.16	4405	4.96
/reports/Test_item.Jun1296.html	3	7.89	2021	2.27
/reports/Cal_index.Jun96.html	2	5.26	6974	7.85
/reports/HTMLTop10_item.Jun1296.html	2	5.26	2394	2.69
/bookmarks.html	2	5.26	14068	15.83
/repors/reports.html	1	2.63	335	0.38
/reports/Cal_site.Jun1296.html	1	2.63	523	0.59
Total	38		88884	

Figure 89. Daily Statistics for URL/File Names

The Log Reports are generated automatically daily at midnight immediately after the previous day's logs have been closed and the new day's logs have been opened, that is at the same time as log purge or userexit actions.

However, you can test your templates by running the htlogrep utility that performs exactly the same actions as the server each night.

The htlogrep executable is located as follows:

- \WWW\BIN\HTLOGREP.EXE on OS/2 and Windows NT
- /usr/sbin/htlogrep on AIX

The htlogrep command is usually invoked without any argument. It will process the log files of the previous day and generate the requested reports.

4.6.8.11 Log Tools

Because the access log files are produced in a common format, many of the tools you will find on the Internet are applicable.

The following URL gives many references to existing tools. These range from free, basic analysis tools up to powerful commercial products.

```
http://www.yahoo.com/Computers_and_Internet/Internet
                    /World_Wide_Web/HTTP/Servers/Log_Analysis_Tools
```

A parameter that will certainly impact your choice is of course your environment: both platform and available programming languages. Most programs are developed in Perl or C; some come compiled, others require a compiler.

Most of the tools produce HTML pages; but you will also find tools that create and maintain databases (such as PressView for Windows NT).

In order to show you what kind of output you can expect from those tools, we will discuss two of the tools available on Internet. These tools have been implemented and tested successfully on an AIX platform. Because our experimental server was not as solicited as a production server should be, we will show you output on real-life servers.

Our choice was at random (or nearly since we have only tested free software or software for which an evaluation copy was available). We do not intend to promote one product more than another one. It is simply that writing on all existing tools is beyond the scope of this book.

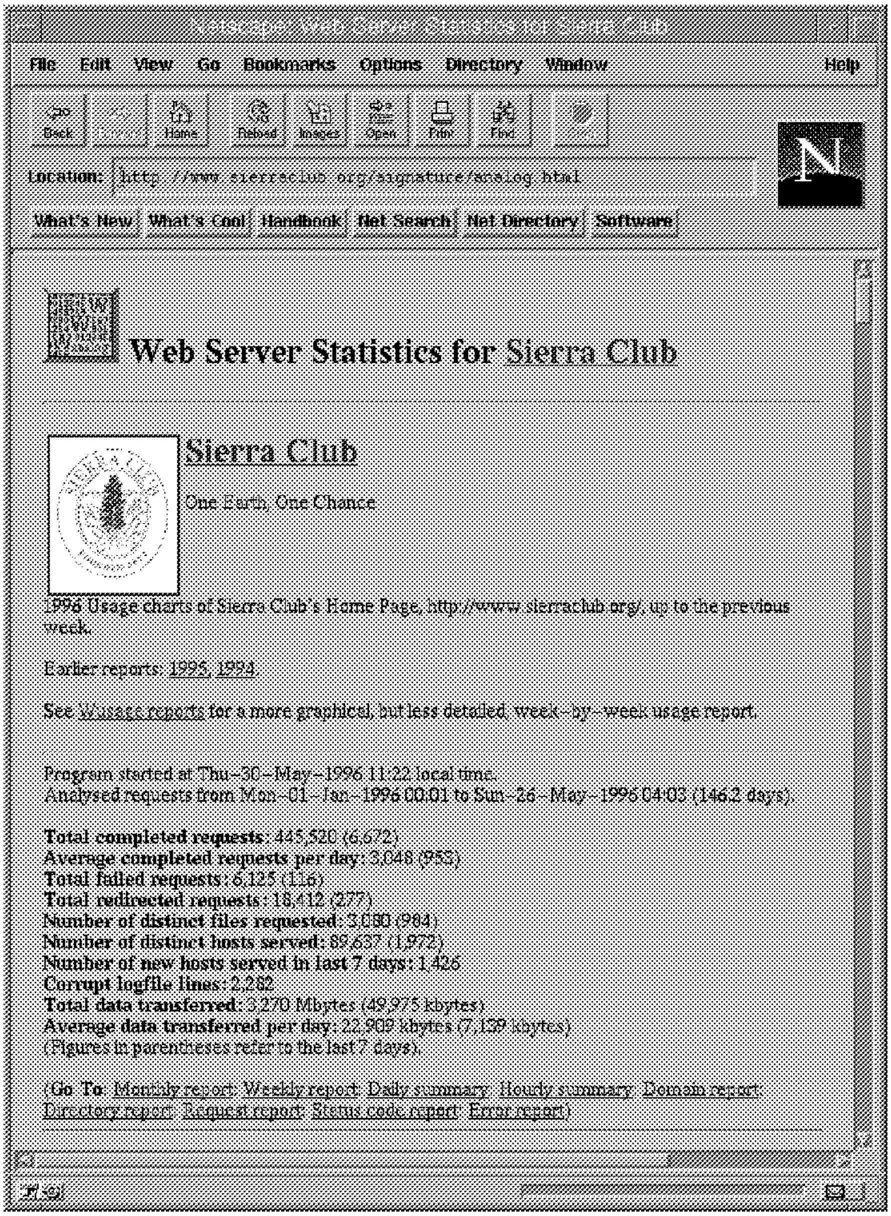
Analog: This log file analysis program is free and works on any UNIX and DOS machine:

- It is written in C, thus requiring a C compiler.
- It produces ASCII output as well as HTML. The HTML code produced does not use tables or image-stretching properties so it works on any browser.
- It has a form interface allowing users to select which options they want via a Web page.

Refer to the following URL:

```
http://www.statslab.com.ac.uk/~ sret1/analog
```

You will find here an example of the output that analog has produced at the Sierra Club Web site. It is partial output only since the full report is really detailed and long. This first page of statistics (Figure 90 on page 159) is interesting since it gives valuable information like the total of requests and the total of data transferred. It may help you in sizing your Web site and in providing solid numbers to make claims about your site.



http://www.sierraclub.org/signature/analog.html

Web Server Statistics for Sierra Club

Sierra Club

One Earth, One Chance

1996 Usage charts of Sierra Club's Home Page, http://www.sierraclub.org/, up to the previous week.

Earlier reports: 1995, 1994.

See Wusage reports for a more graphical, but less detailed, week-by-week usage report.

Program started at Thu-30-May-1996 11:22 local time.
Analysed requests from Mon-01-Jan-1996 00:01 to Sun-26-May-1996 04:03 (146.2 days).

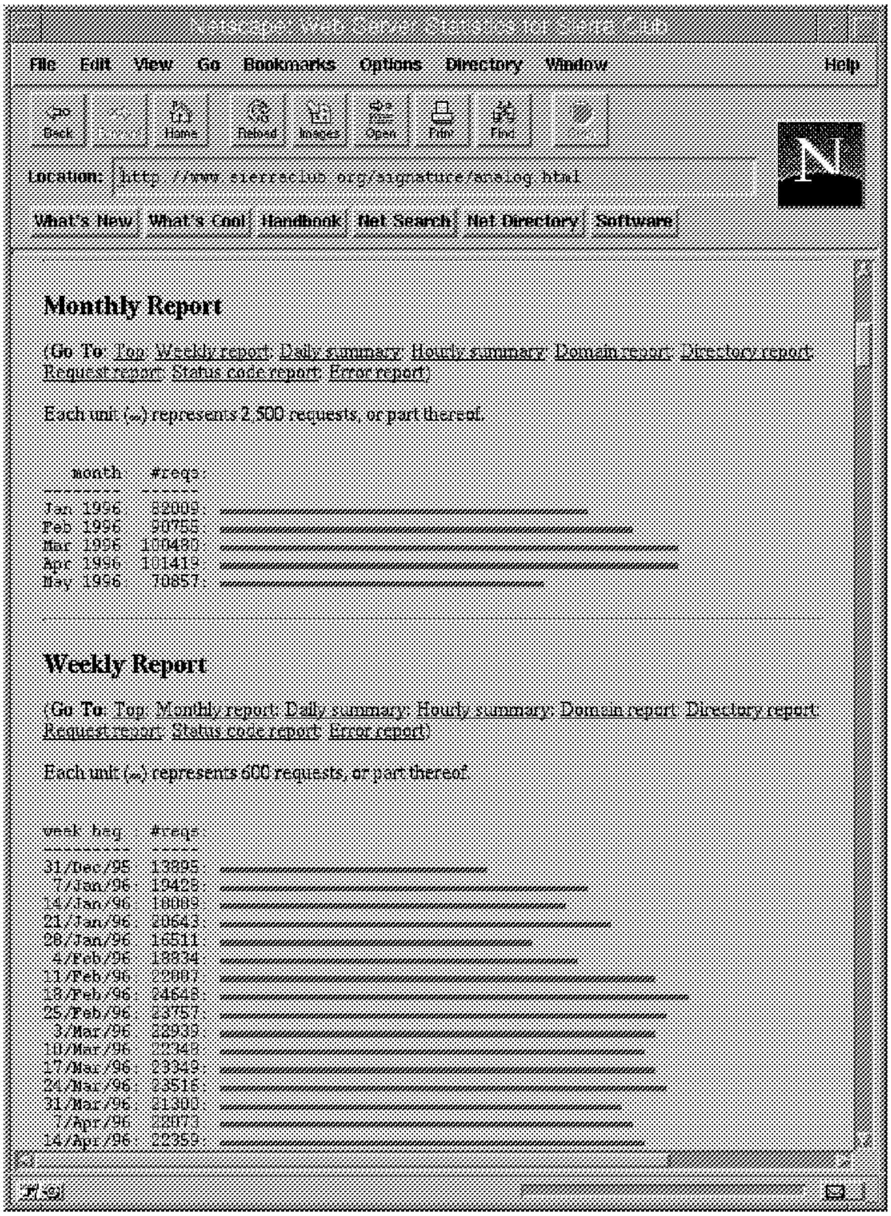
Total completed requests: 445,520 (6,672)
Average completed requests per day: 3,048 (953)
Total failed requests: 6,125 (116)
Total redirected requests: 18,412 (277)
Number of distinct files requested: 3,080 (984)
Number of distinct hosts served: 89,637 (1,972)
Number of new hosts served in last 7 days: 1,426
Corrupt logfile lines: 2,282
Total data transferred: 3,270 Mbytes (49,975 kbytes)
Average data transferred per day: 22,909 kbytes (7,129 kbytes)
(Figures in parentheses refer to the last 7 days).

(**Go To**: Monthly report: Weekly report: Daily summary: Hourly summary: Domain report: Directory report: Request report: Status code report: Error report)

Figure 90. Analog - General Summary

For Web sites serving different servers (for different companies or departments) you can produce separate reports. Each report will have its general summary giving information on requests and data transferred.

The analog program can produce many reports and summaries as shown in Figure 91 on page 160.



Monthly Report

(**Go To**: Top: Weekly report: Daily summary: Hourly summary: Domain report: Directory report: Request report: Status code report: Error report)

Each unit (-) represents 2,500 requests, or part thereof.

```
   month:  #reqs:
---------  ------
Jan 1996:   82009:
Feb 1996:   90755:
Mar 1996:  100480:
Apr 1996:  101419:
May 1996:   70857:
```

Weekly Report

(**Go To**: Top: Monthly report: Daily summary: Hourly summary: Domain report: Directory report: Request report: Status code report: Error report)

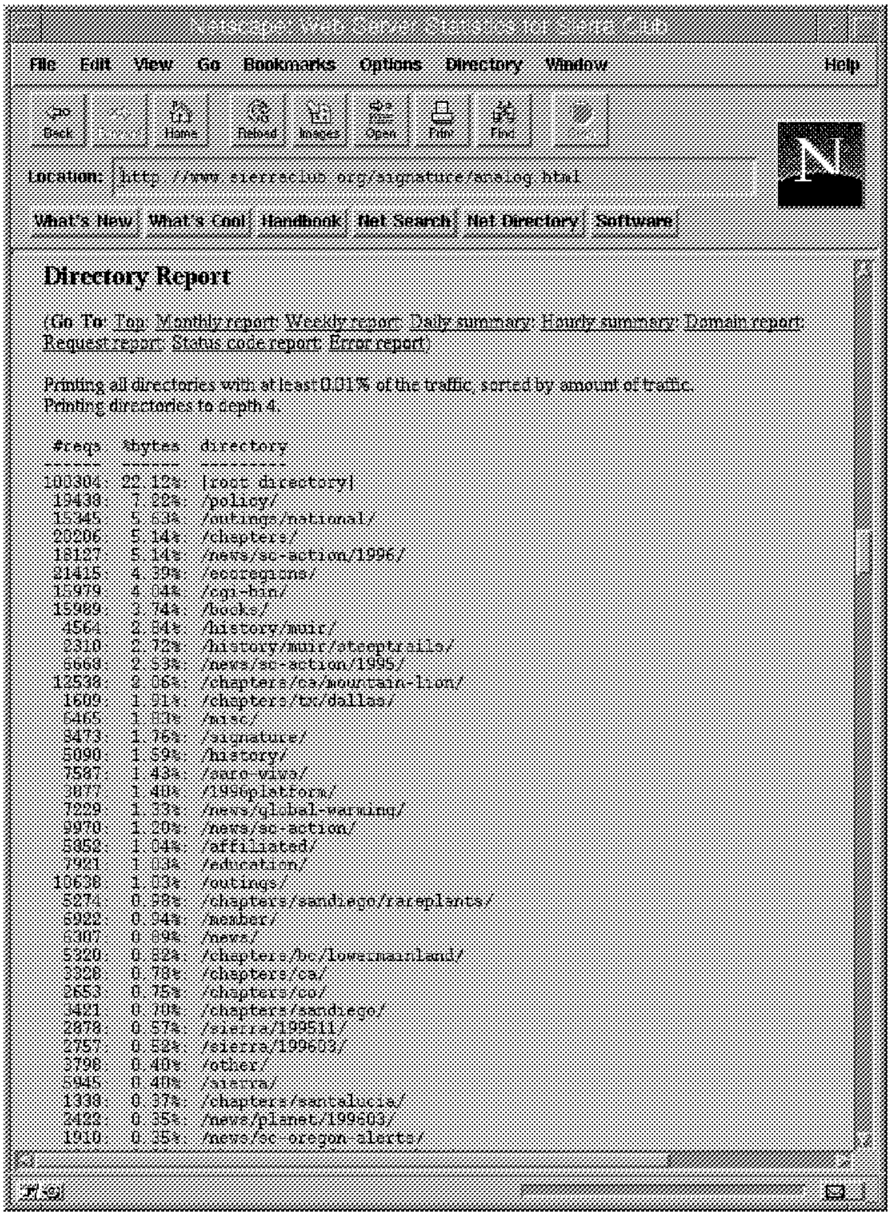
Each unit (-) represents 600 requests, or part thereof.

```
week beg.: #reqs:
---------  -----
31/Dec/95: 13895:
 7/Jan/96: 19428:
14/Jan/96: 18089:
21/Jan/96: 20643:
28/Jan/96: 16511:
 4/Feb/96: 18834:
11/Feb/96: 22887:
18/Feb/96: 24648:
25/Feb/96: 23757:
 3/Mar/96: 22939:
10/Mar/96: 22348:
17/Mar/96: 23349:
24/Mar/96: 23516:
31/Mar/96: 21300:
 7/Apr/96: 22073:
14/Apr/96: 22359:
```

Figure 91. Analog - Monthly and Weekly Reports

The default output will satisfy most users but you can set up an analog to produce other reports. The options are taken from a configuration file or from command line arguments. It′s really flexible.

Among the built reports, the directory report (Figure 92 on page 161) is particularly valuable. Using this, you would easily detect that only a few people are using a particular area of your site even if you thought that it would be perceived as the most interesting; it may be that people do not know of this area in which case you can advertise it.


Netscape: Web Server Statistics for Sierra Club
File Edit View Go Bookmarks Options Directory Window Help
Back Forward Home Reload Images Open Print Find Stop
Location: http://www.sierraclub.org/signature/analog.html
What's New What's Cool Handbook Net Search Net Directory Software

Directory Report

(Go To: Top: Monthly report: Weekly report: Daily summary: Hourly summary: Domain report: Request report: Status code report: Error report)

Printing all directories with at least 0.01% of the traffic, sorted by amount of traffic.
Printing directories to depth 4.

```
 #reqs  %bytes  directory
------  ------  ---------
100304: 22.12%: [root directory]
 19438:  7.22%: /policy/
 15345:  5.63%: /outings/national/
 20206:  5.14%: /chapters/
 18127:  5.14%: /news/sc-action/1996/
 21415:  4.39%: /ecoregions/
 15979:  4.04%: /cgi-bin/
 15989:  3.74%: /books/
  4564:  2.84%: /history/muir/
  2310:  2.72%: /history/muir/steeptrails/
  6668:  2.53%: /news/sc-action/1995/
 12538:  2.06%: /chapters/ca/mountain-lion/
  1609:  1.91%: /chapters/tx/dallas/
  6465:  1.83%: /misc/
  3473:  1.76%: /signature/
  5090:  1.59%: /history/
  7587:  1.43%: /saro-viva/
  3877:  1.40%: /1996platform/
  7229:  1.33%: /news/global-warming/
  9970:  1.20%: /news/sc-action/
  5852:  1.04%: /affiliated/
  7921:  1.03%: /education/
 10638:  1.03%: /outings/
  5274:  0.98%: /chapters/sandiego/rareplants/
  6922:  0.94%: /member/
  6307:  0.89%: /news/
  5320:  0.82%: /chapters/bc/lowermainland/
  3328:  0.78%: /chapters/ca/
  2653:  0.75%: /chapters/co/
  3421:  0.70%: /chapters/sandiego/
  2878:  0.57%: /sierra/199511/
  2757:  0.52%: /sierra/199603/
  3798:  0.40%: /other/
  5945:  0.40%: /sierra/
  1338:  0.37%: /chapters/santalucia/
  2423:  0.35%: /news/planet/199603/
  1910:  0.35%: /news/sc-oregon-alerts/
```

Figure 92. Analog - Directory Report

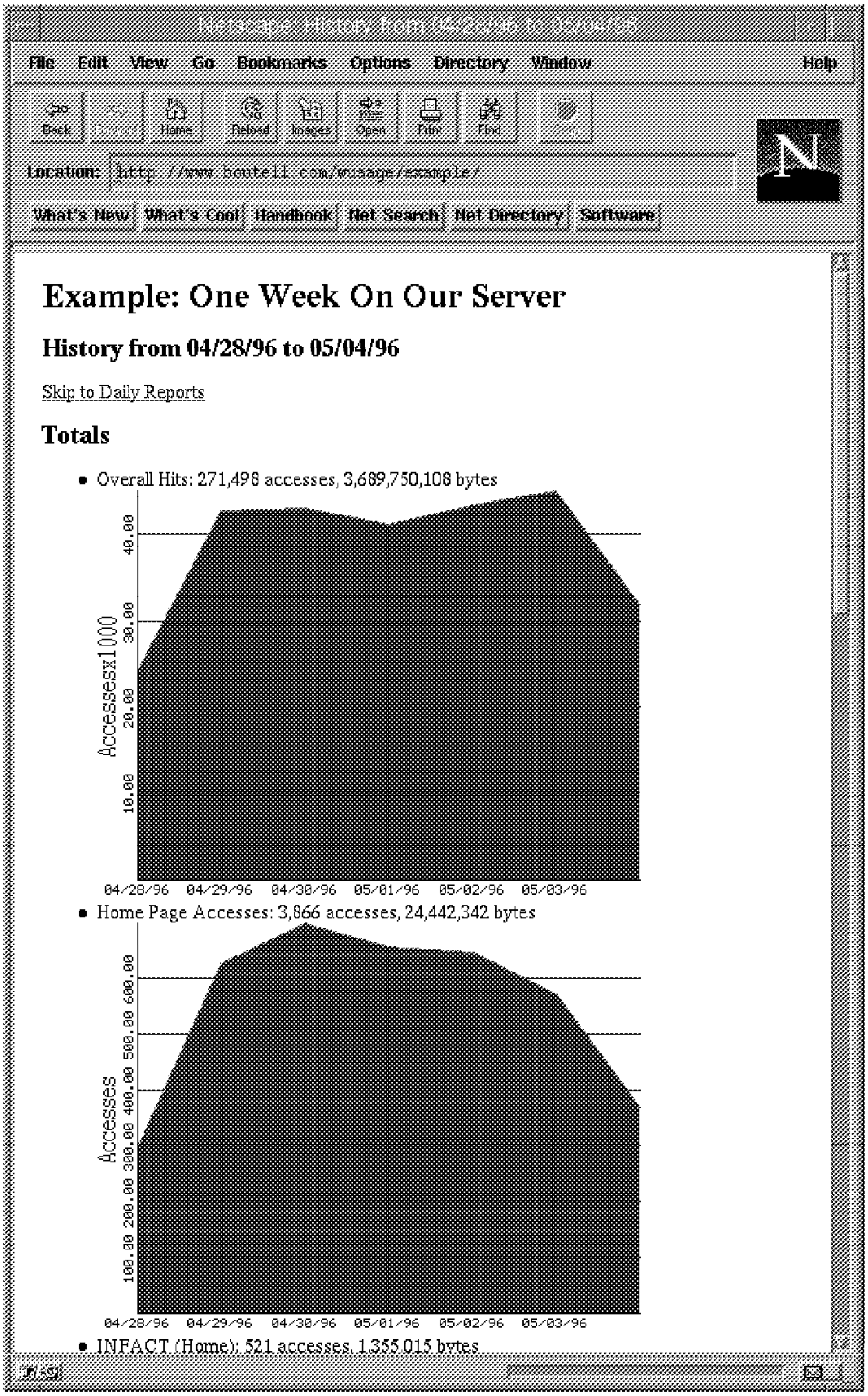
Also, from this report you can figure out which pages are the most accessed and focus more attention on them and use them to advertise new areas, new products, etc.

Wusage: Wusage is a product of Boutell.Com, Inc. A licence is required for the usage of this software.***:*** You can download an evaluation copy (expires after 30 days) by accessing the following URL:

```
http://www.boutell.com/wusage
```

It is a compiled program that will run on AIX, OS/2, Microsoft Windows, Windows 95 or Windows NT. It produces many statistics relating to the activity of your Web server including daily, weekly and monthly history reports as well as summary reports, hourly reports, popular documents reports, result codes reports, etc.

The following statistics samples were extracted from the Boutell Web server itself:



Example: One Week On Our Server

History from 04/28/96 to 05/04/96

Skip to Daily Reports

Totals

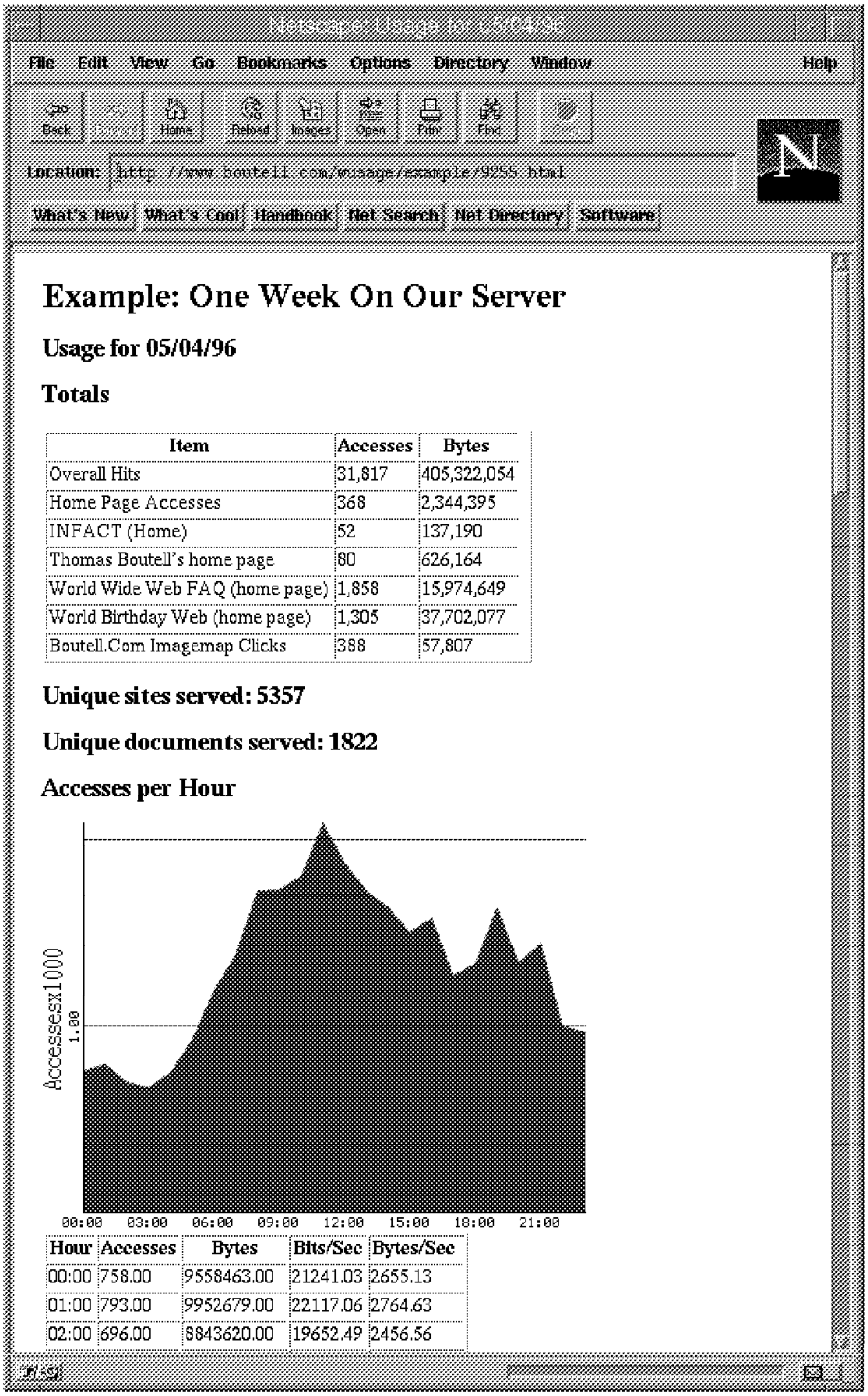
- Overall Hits: 271,498 accesses, 3,689,750,108 bytes

- Home Page Accesses: 3,866 accesses, 24,442,342 bytes

- INFACT (Home): 521 accesses, 1,355,015 bytes

Figure 93. Wusage - Total Reports

The report begins with a report on total accesses and bytes transmitted during the time period (a day, a week or a month according to the setting) for each of the totals specified in the configuration file (Figure 93). In this case, the period time is a week and the output type is a table. (It could have been a graph.)



Example: One Week On Our Server

Usage for 05/04/96

Totals

Item	Accesses	Bytes
Overall Hits	31,817	405,322,054
Home Page Accesses	368	2,344,395
INFACT (Home)	52	137,190
Thomas Boutell's home page	80	626,164
World Wide Web FAQ (home page)	1,858	15,974,649
World Birthday Web (home page)	1,305	37,702,077
Boutell.Com Imagemap Clicks	388	57,807

Unique sites served: 5357

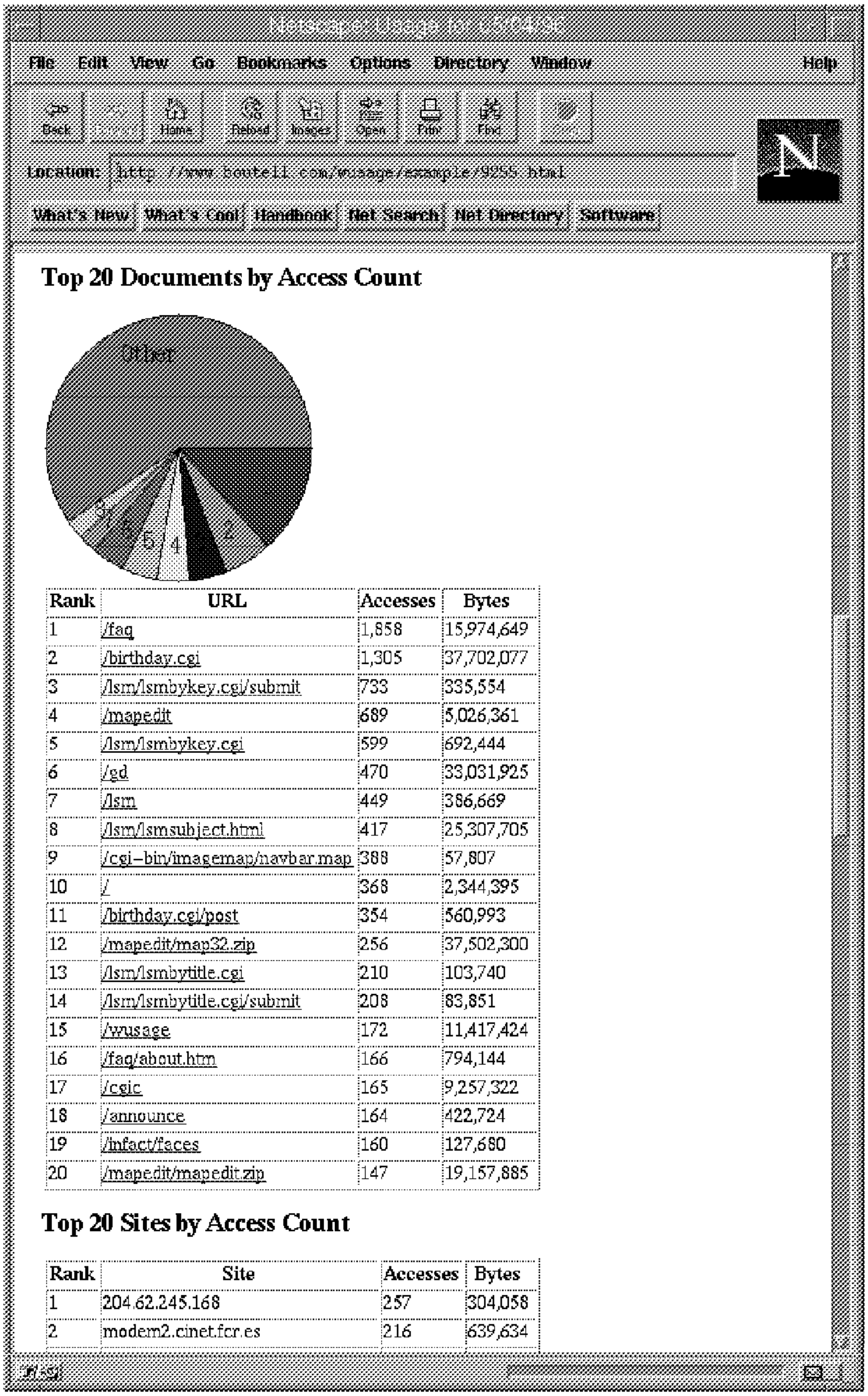
Unique documents served: 1822

Accesses per Hour

Hour	Accesses	Bytes	Bits/Sec	Bytes/Sec
00:00	758.00	9558463.00	21241.03	2655.13
01:00	793.00	9952679.00	22117.06	2764.63
02:00	696.00	8843620.00	19652.49	2456.56

Figure 94. Wusage - Day Report

Then there are time period reports: day, week and month reports such as in Figure 94.



Top 20 Documents by Access Count

Rank	URL	Accesses	Bytes
1	/faq	1,858	15,974,649
2	/birthday.cgi	1,305	37,702,077
3	/lsm/lsmbykey.cgi/submit	733	335,554
4	/mapedit	689	5,026,361
5	/lsm/lsmbykey.cgi	599	692,444
6	/gd	470	33,031,925
7	/lsm	449	386,669
8	/lsm/lsmsubject.html	417	25,307,705
9	/cgi-bin/imagemap/navbar.map	388	57,807
10	/	368	2,344,395
11	/birthday.cgi/post	354	560,993
12	/mapedit/map32.zip	256	37,502,300
13	/lsm/lsmbytitle.cgi	210	103,740
14	/lsm/lsmbytitle.cgi/submit	208	83,851
15	/wusage	172	11,417,424
16	/faq/about.htm	166	794,144
17	/cgic	165	9,257,322
18	/announce	164	422,724
19	/infact/faces	160	127,680
20	/mapedit/mapedit.zip	147	19,157,885

Top 20 Sites by Access Count

Rank	Site	Accesses	Bytes
1	204.62.245.168	257	304,058
2	modem2.cinet.fcr.es	216	639,634

Figure 95. Wusage - Popular Documents Report

The Top 20 Documents report in Figure 95 is a pie chart and a table featuring the most frequently accessed documents on the site. The table is ranked by total access in this case but could be ranked by total bytes instead.

It produces historical graphs, but also a tab-delimited ASCII file which any spreadsheet program can open directly.

If you turn on the notfound option, your reports will end with a list of the URLs that users unsuccessfully tried to access on your server. Sometimes, you may realize that you have accidentally removed or renamed a file.

Other utilities, such as pwebstats by the University of Melbourne (see URL http://www.unimelb.edu.au/pwebstats/pwebstats.html), produce sophisticated bars in addition to pie charts. It is to some extent a question of taste.

4.6.8.12 Log Analysis Administration

In all cases, you will have to run your analysis tool on a regular basis. It is up to you to decide whether you want daily, weekly or monthly reports. But it is advisable to run your tool often since your log files will grow larger and you will probably not be able to keep them for an infinite time.

Note that some tools, like wusage, work on compressed files or STDIN, thus allowing you to compress more regularly your files and run your analysis tool less often.

In order to automate the launch of the statistics program, you will have to set up automatic scheduling tools. Some examples are:

- Cron on AIX (or any UNIX platform)
- AT command on Windows NT
- Automated Operations Control (AOC) on MVS

4.6.9 AS/400 HTTP Server Configuration and Operation

Many of the features of Internet Connection for AS/400 are reliant upon the HTTP server. In this section, we look in detail at the HTTP server, its configuration, and its operation.

Note

The HTTP server configuration *must* be modified by the user before the AS/400 system can be used as a Web server. By default, access to any Web page is forbidden.

If the documents are served from the QDLS file system (folders), user profile QTMHHTTP *must* have a directory entry added using the ADDDIRE command. Likewise, user profile QTMHHTTP *must* have object authority to all of the objects that are served.

4.6.9.1 HTTP Server Configuration Commands

The following list contains TCP/IP configuration commands for the TCP/IP HTTP Server:

1. CFGTCPHTTP (Configure TCP/IP HTTP)

   This menu of options is used to configure the HTTP server.

2. CHGHTTPA (Change HTTP attributes)

   This command is used to change the HTTP server attributes, such as the number of servers to start, whether to autostart the server, and so on. It is also an option on the Configure TCP/IP HTTP menu.

3. WRKHTTPCFG (Work with HTTP configuration)

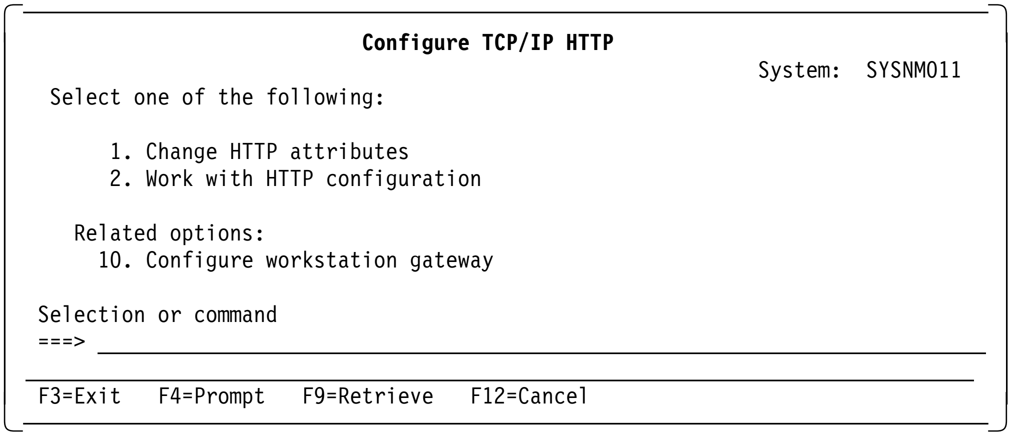
This command presents the user with a work-with display to allow HTTP administrators to add, change, display, or remove HTTP configuration options. The configuration format follows the CERN HTTP server configuration model and may look unfamiliar to many AS/400 users, but a traditional AS/400 work-with list display is used to interact with the file. It is also an option on the Configure TCP/IP HTTP menu.

There are several methods of starting the HTTP server once you have finished configuring it:

- The STRTCP command starts the server if the HTTP server attribute AUTOSTART is set to *YES.
- The STRTCPSVR SERVER(*HTTP) command starts the server if TCP/IP is already started and the HTTP server is not.
- The STRTCPSVR SERVER(*HTTP) RESTART(*HTTP) command restarts the HTTP server if both TCP/IP and the HTTP server are already started. This is useful when the configuration is changed and the HTTP server must be restarted in order for those changes to take effect. Since HTTP is basically a stateless protocol, this can be done while clients are accessing the AS/400 server. A client may just experience a slight delay while the server restarts.

4.6.9.2 The CFGTCPHTTP Display

The Configure TCP/IP HTTP display appears if the CFGTCPHTTP command is entered from the command line, or if CFGTCPAPP option 14 is selected.



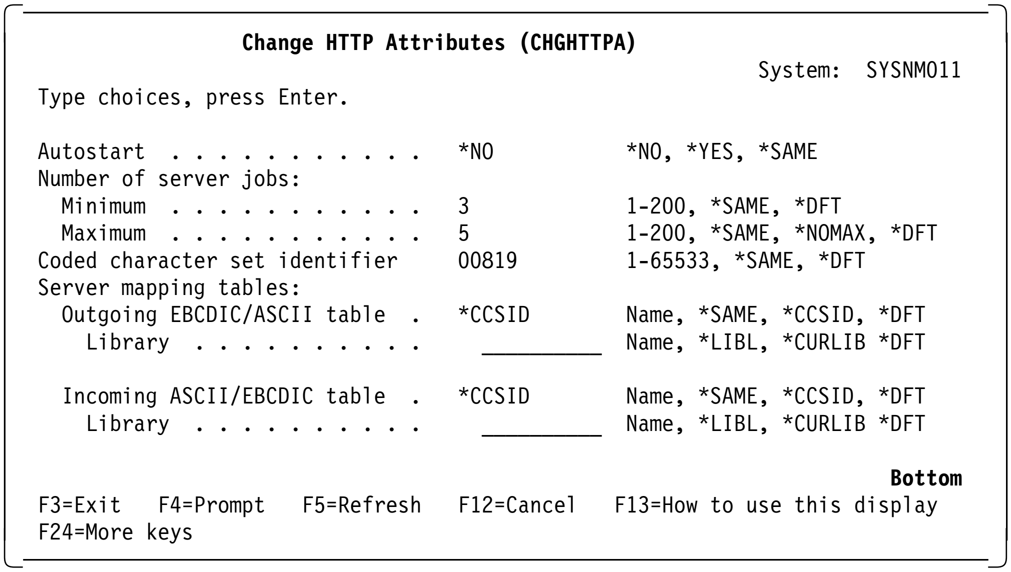
```
                           Configure TCP/IP HTTP
                                                          System:   SYSNM011
  Select one of the following:

       1. Change HTTP attributes
       2. Work with HTTP configuration

    Related options:
      10. Configure workstation gateway

 Selection or command
 ===> ___________________________________________________________________

 F3=Exit   F4=Prompt   F9=Retrieve   F12=Cancel
```

Figure 96. CFGTCPHTTP Display

- Option 1 - Prompts the CHGHTTPA command.
- Option 2 - Calls the WRKHTTPCFG command.
- Option 10- Calls the CFGTCPWSG command.

4.6.9.3 The CHGHTTPA Display

The Change HTTP Attributes display appears if the CHGHTTPA command is prompted from the command line or if CFGTCPHTTP option 1 is selected.



```
                  Change HTTP Attributes (CHGHTTPA)
                                                            System:  SYSNM011
 Type choices, press Enter.

 Autostart  . . . . . . . . . . .   *NO           *NO, *YES, *SAME
 Number of server jobs:
   Minimum  . . . . . . . . . . .   3             1-200, *SAME, *DFT
   Maximum  . . . . . . . . . . .   5             1-200, *SAME, *NOMAX, *DFT
 Coded character set identifier     00819         1-65533, *SAME, *DFT
 Server mapping tables:
   Outgoing EBCDIC/ASCII table  .   *CCSID        Name, *SAME, *CCSID, *DFT
     Library  . . . . . . . . . .     __________  Name, *LIBL, *CURLIB *DFT

   Incoming ASCII/EBCDIC table  .   *CCSID        Name, *SAME, *CCSID, *DFT
     Library  . . . . . . . . . .     __________  Name, *LIBL, *CURLIB *DFT

                                                                      Bottom
 F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
 F24=More keys
```

Figure 97. Change HTTP Attributes Display

The values that are used on the preceding display are the default values.

Let′s take time to look at some of the parameters.

Autostart (AUTOSTART)
Controls when the AS/400 system starts the HTTP server. Select *YES if you want the server to start whenever TCP/IP is started. If you use the STRTCPSRV command, this parameter is ignored and the HTTP server is started.

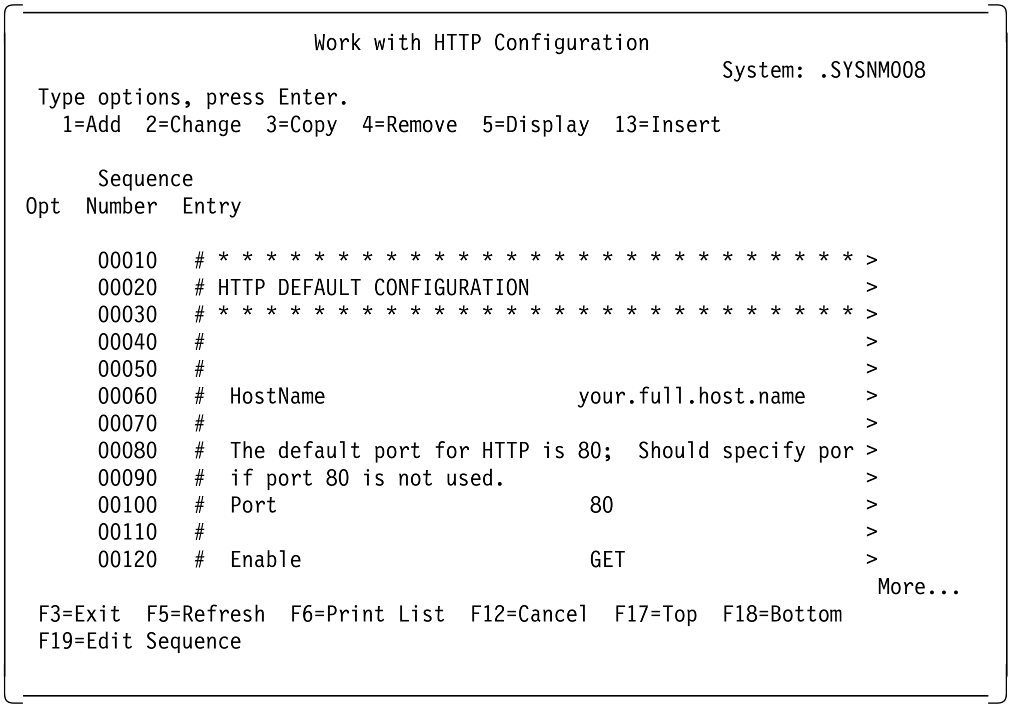
Number of Server Jobs (NBRSVR)
Use this parameter to specify the minimum number of HTTP server jobs to start when the HTTP server is started and the maximum number of HTTP server jobs to be used. If you set the maximum number of servers too high, this affects the performance of HTTP servers satisfying browser requests. You may want to increase this, however, if the HTTP servers are serving large documents that ″tie-up″ a server job for a long period.

Even if, for example, all of the HTTP servers are busy at a given moment, the new client requests coming into the HTTP server are queued. As soon as an HTTP server becomes available, the request is processed.

The remaining parameters are concerned with character translation and are similar to the parameters in Telnet attributes and FTP attributes. They are outside of the current discussion. For a complete explanation, see the *TCP/IP Configuration and Reference*.

4.6.9.4 The WRKHTTPCFG Display

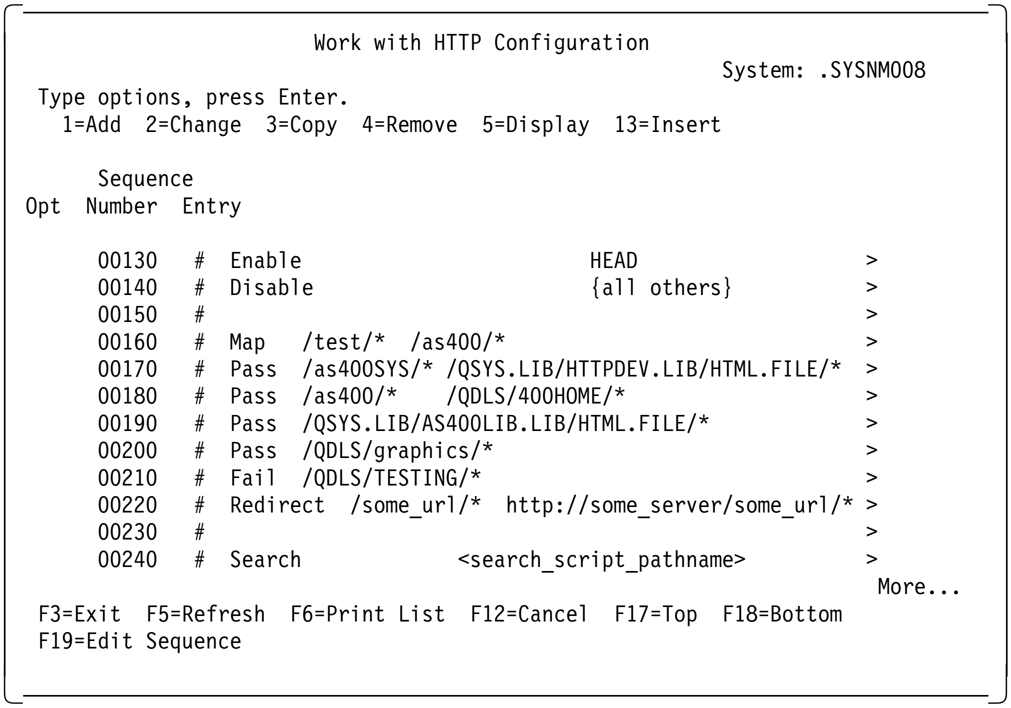
This display appears if CFGTCPHTTP option 2 is selected or if the WRKHTTPCFG command is entered from a command line.



```
                         Work with HTTP Configuration
                                                      System: .SYSNM008
 Type options, press Enter.
   1=Add  2=Change  3=Copy  4=Remove  5=Display  13=Insert

      Sequence
Opt  Number  Entry

      00010   # * * * * * * * * * * * * * * * * * * * * * * * * * * * >
      00020   # HTTP DEFAULT CONFIGURATION                            >
      00030   # * * * * * * * * * * * * * * * * * * * * * * * * * * * >
      00040   #                                                       >
      00050   #                                                       >
      00060   #  HostName                     your.full.host.name     >
      00070   #                                                       >
      00080   #  The default port for HTTP is 80;  Should specify por >
      00090   #  if port 80 is not used.                              >
      00100   #  Port                          80                     >
      00110   #                                                       >
      00120   #  Enable                        GET                    >
                                                                        More...
 F3=Exit  F5=Refresh  F6=Print List  F12=Cancel  F17=Top  F18=Bottom
 F19=Edit Sequence
```

Figure 98. WRKHTTPCFG Display

Note: The # character is used to comment out a line.



```
                         Work with HTTP Configuration
                                                      System: .SYSNM008
 Type options, press Enter.
   1=Add  2=Change  3=Copy  4=Remove  5=Display  13=Insert

      Sequence
Opt  Number  Entry

      00130   #  Enable                        HEAD                   >
      00140   #  Disable                       {all others}           >
      00150   #                                                       >
      00160   #  Map   /test/*  /as400/*                              >
      00170   #  Pass  /as400SYS/* /QSYS.LIB/HTTPDEV.LIB/HTML.FILE/*  >
      00180   #  Pass  /as400/*    /QDLS/400HOME/*                    >
      00190   #  Pass  /QSYS.LIB/AS400LIB.LIB/HTML.FILE/*             >
      00200   #  Pass  /QDLS/graphics/*                               >
      00210   #  Fail  /QDLS/TESTING/*                                >
      00220   #  Redirect  /some_url/*  http://some_server/some_url/* >
      00230   #                                                       >
      00240   #  Search            <search_script_pathname>           >
                                                                        More...
 F3=Exit  F5=Refresh  F6=Print List  F12=Cancel  F17=Top  F18=Bottom
 F19=Edit Sequence
```

Figure 99. More of the WRKHTTPCFG Display

When the HTTP server is started, this configuration information that is stored in file QUSRSYS/QATMHTTPC.CONFIG is used to process incoming client requests.

This configuration file can be saved and restored so before you make changes to a working HTTP server configuration, it may be worth backing it up.

You may find it easier to use the Source Entry Utility (SEU) to make lots of changes to the HTTP configuration file. If this is the case, then you should copy the HTTP configuration file member to a Source Physical File, modify it using SEU, then copy the changed file member back to QUSRSYS/QATMHTTPC. Time-wise, this is only worth doing if you are making major modifications to the file or do not feel comfortable with the HTTP server configuration interface. This method has one limitation. SEU can only handle files with the record length of 240 bytes which leaves 228 bytes for the actual data. When using SEU, you might get into difficulties with long path names in the directives.

An alternative way to edit the configuration file is to download the file to a PC using FTP and after editing it, upload it using FTP. Remember, though, that if you use any of these alternative ways to change the HTTP configuration, you should syntax check the file using the WRKHTTPCFG command.

Note: Before we discuss the configuration in detail, there are some considerations when working with HTTP configuration:

- Options 1, 2, 4 and 13 are only available if the user has *IOSYSCFG special authority, otherwise, only option 5 for display is allowed.
- Options 1, 2, 4 and 13 are only available to one user at a time. If the file is being modified by one user, only option 5 for display is available to subsequent users.
- Entry length is 640 characters, 55 of which are shown from the Work with HTTP Configuration display.
- Sequence numbers are reset to increments of 10 once configuration changes have been completed and processed.
- The configuration file is read sequentially.
- A valid entry comprises two parts, a directive and a value. The syntax should follow rules as defined by the CERN validation module.
- After configuration changes have been made, the file is checked by the CERN validation module and any invalid entries are highlighted.
- After modifying the HTTP Server configuration, the HTTP Server should be restarted. This can be accomplished without ending the server using the STRTCPSVR SERVER(*HTTP) RESTART(*HTTP) command.

4.6.9.5 HTTP Server Directives

AS/400 HTTP server directives determine how the HTTP server behaves. These directives are derived from the CERN configuration model and information about it can be viewed at URL:
`http://www.w3.org/hypertext/WWW/Daemon/User/Admin.html`

The directives supported for the AS/400 HTTP server include:

- General settings:
  - HostName
  - Port
  - Enable
  - Disable
- Mapping rules (also known as URL translation rules):
  - Map

  - Pass
  - Fail
  - Redirect
  - Exec

- File name suffix definitions (source type definitions for QSYS.LIB):
  - AddType
  - AddEncoding
  - AddLanguage
  - SuffixCaseSense

- Accessory programs (also known as accessory scripts):
  - Search
  - POST-Script

- Directory listings:
  - DirAccess
  - Welcome
  - AlwaysWelcome
  - DirReadme
  - DirShowDate
  - DirShowSize
  - DirShowBytes
  - DirShowOwner
  - DirShowDescription
  - DirShowMaxDescrLength

- Logging:
  - AccessLog
  - ErrorLog
  - LogFormat
  - LogTime
  - NoLog

- Timeouts:
  - InputTimeOut
  - OutputTimeOut
  - ScriptTimeOut

Note

The minimum directives that you need to serve a document or execute a CGI are a Pass or Exec directive.

Let us look in detail at the HTTP server directives and their use.

4.6.9.6 General Settings

Hostname *full.server.host.name*

The HostName directive is used to override the host name value set using the Change local domain and host names option of the AS/400 Configure TCP/IP (CFGTCP) command. The directive only applies to those circumstances where HTTP server generates URLs that are self referential. We leave this directive commented out (#).

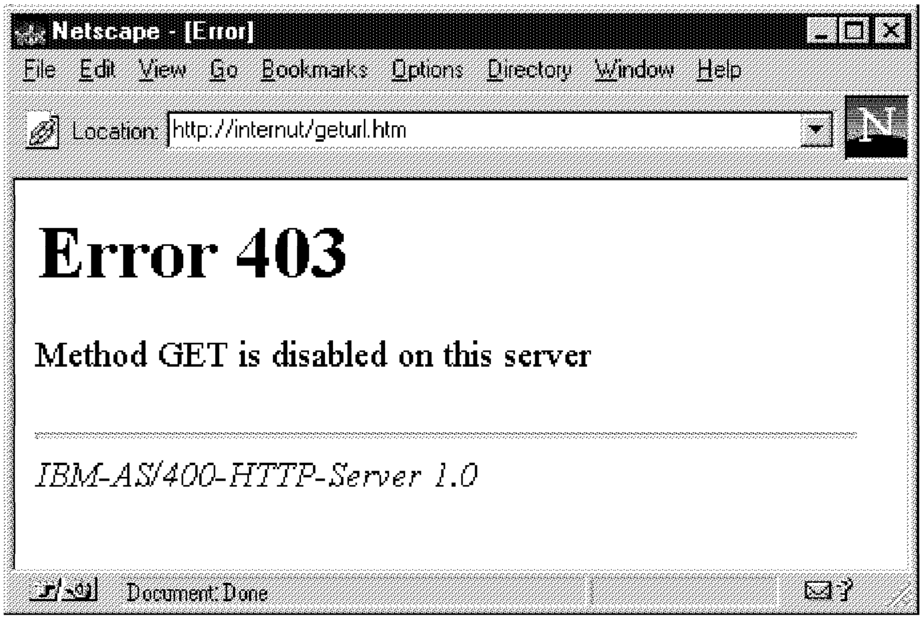
Port *port-no*

The Port directive is used to override the listening port value specified in the TCP/IP Services table for service www (accessible through the Configure TCP/IP (CFGTCP) Configure related tables option and then the Work with service table entries option). It may be useful to run the HTTP server on a non-standard port for testing purposes but without the need to change the Services Table. It is recommended that you use a port number greater than 1024 if you override the default HTTP sever port 80. To access the server with a HTTP server port of 1025, the required URL looks similar to this: `http://ServerName:1025/welcome.htm`

Enable & Disable *method-name*

The Enable and Disbale directives cause the HTTP server to accept or reject incoming request URIs based on the Method coded in the URI. These methods are described in the HTTP/1.0 specification. The AS/400 HTTP server supports GET, HEAD, and POST. All other methods are rejected.

In our HTTP server configuration, we changed the default values so that GET and POST were both enabled, as a simple browser request uses the GET method and many FORMS use the POST method. If a method is disabled, the HTTP server sends an error message to the Web browser informing the end user that a particular method is not enabled.


Netscape - [Error]
File Edit View Go Bookmarks Options Directory Window Help
Location: http://internut/geturl.htm
Document: Done

Error 403

Method GET is disabled on this server

IBM-AS/400-HTTP-Server 1.0

Figure 100. Method GET Disabled on Server

Note: The error message is sent by the AS/400 HTTP server in the form of an HTML document. The error message, therefore, is the same regardless of the type of Web browser.

Defaults

- Enable GET
- Enable HEAD

- Disable POST

If the methods are commented out, then the defaults values determine what is enabled or disabled. Therefore, there should be no value in adding a directive such as:

```
Enable GET
```

4.6.9.7 Mapping Rules

Mapping rules (which are also known as URL translation rules) allow the Web administrator to define a virtual hierarchy of Web resources that are presented by the HTTP server. This virtual hierarchy is independent of the physical file system (or systems) on which the resources are actually stored. This is very useful since:

1. It allows resources to be physically relocated without changing the apparent hierarchy and its implied associations.
2. The details of the underlying physical file systems can be hidden from client applications accessing the HTTP server.

The second reason is especially important for the HTTP server because of the differences between some of the AS/400 file systems and the tree structured hierarchical UNIX file systems upon which the HTTP protocol and the URL scheme are based.

4.6.9.8 URL Mapping Overview

The mapping directives Map, Pass, Fail, Redirect and Exec are used to specify a set of rules that define a process for translating and processing the path (and file) information contained in a URL that is received in a client request URL.

Each of the mapping directives may be specified multiple times and in any combination, and may be placed anywhere in the HTTP server configuration. When the HTTP server configuration is read during initialization, the mapping directives are stored, in order of appearance, in an internal mapping rule table.

The named object in each incoming URL is compared against each of the mapping directives in turn. If a directive applies, the specified translation or action is carried out. Depending on the directive that was matched, rule processing is either suspended or continues with the next rule using the newly translated URL.

4.6.9.9 Mapping Directive Descriptions and Examples

Note: It is important to note that the following directives are case sensitive. A template checks the input value for a match letter-by-letter. This has important security implications, especially for the Fail directive that we cover later in this section.

Map *template replacement*

If the URL matches the template, substitute the replacement string for the template and continue to the next rule using the resulting URL.

Example: Your Welcome.htm document is reached through the following path in the QDLS file system:

```
http://ServerName/QDLS/FOLDER1/WEBSTUFF/HTML/Welcome.htm
```

This is quite a complex URL for a client to remember and type so it makes sense to present a simpler, virtual structure to the client.

Using the Map directive, we can modify the HTTP server configuration as follows:

```
Map /info/*  /QDLS/FOLDER1/WEBSTUFF/HTML/*
```

Now the client can type a URL such as:

```
http://ServerName/info/Welcome.htm
```

The Map statement replaces the /info/* value with the /QDLS/FOLDER1/WEBSTUFF/HTML/* replacement string and the HTTP server continues with further rule processing using the resulting URL.

Pass *template {replacement}*

If the URL matches the template, rule processing is terminated.

If the optional replacement string is present, substitute it for the template string and complete the client request using the resultant URL.

Example: We want to allow Web clients to browse the document Welcome.htm; therefore, we *must* code a Pass statement:

```
Pass /info/Welcome.htm  /QDLS/FOLDER1/WEBSTUFF/HTML/Welcome.htm
```

When a Web client requests URL:

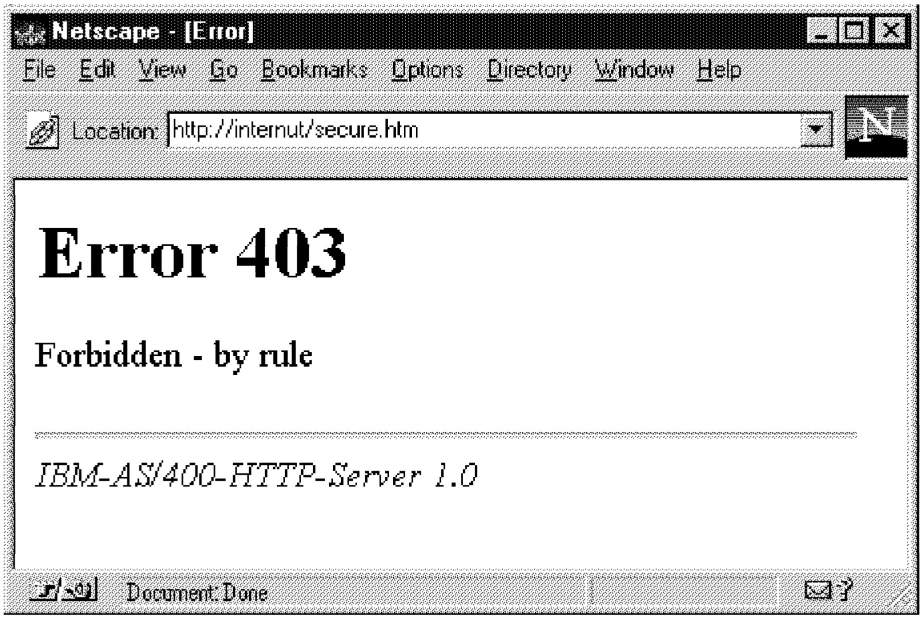
```
http://ServerName/info/Welcome.htm
```

The HTTP server Passes (or allows) the request. In this case, however, we used the optional replacement string to modify the request through the real file structure to the Welcome.htm document.

Note: A generic statement such as Pass /* is a simple way of allowing access to any document, file member, and so on, but the security implications are such that we recommend using specific Pass statements to only those objects that you want the public to see.

Fail *template*

If the URL matches the template rule, processing is terminated and access is refused. An error code is returned to the client.


Netscape - [Error]
File Edit View Go Bookmarks Options Directory Window Help
Location: http://internut/secure.htm
Document: Done

Error 403

Forbidden - by rule

IBM-AS/400-HTTP-Server 1.0

Figure 101. A Failed URL Request

Because these templates are case sensitive and some file systems are not, it is easier to secure objects using the Pass directive than the Fail. This is because you need to add a statement for every possible combination of upper and lower case letters in the object you are trying to secure.

Example: You want to refuse access to object secure.htm,so considering case sensitivity, you add two Fail statements as shown in the following example:

```
Fail  /info/secure.htm
```

```
Fail  /info/SECURE.HTM
```

You assume that you are secure since both upper and lowercase possibilities have been covered. It is, however, possible to access the document using a URL such as:

```
http://ServerName/info/SeCuRe.htm
```

Since the value SeCuRe.htm does not match either of your Fail statement templates, the request does not fail and the document is accessed.

Once again, we recommend using specific Pass statements and maybe, as an additional measure, a generic Fail /* catch-all at the bottom of your configuration file, although it should not be necessary if your Pass statements are explicit enough, as requests should fail by default.

Redirect *template replacement*

When documents, or entire virtual trees of documents, are moved from one server to another, the Redirect directive can be used to redirect the request to another server.

If the URL matches the template, rule processing is terminated. Substitute the replacement string for the template string and send a Redirect containing the resulting URL back to the requester.

Using this directive, the requester is transparently re-routed to the URL specified in the replacement string.

Example:

```
Redirect  /info/*  http://AnotherServer/NewDir/Welcome.htm
```

This redirects a request for any document in the /info directory to the Welcome.htm document on the host called AnotherServer.

Exec *template replacement*

A match with the template specified on an Exec directive indicates that the URL should be interpreted as referring a program to be executed by the HTTP sever on behalf of a client/browser.

On this directive, both template and replacement are required to end with a wildcard (asterisk *).

A template identifies the virtual path to a program (or you can think of this as a virtual directory that contains one or more of your programs). It is replaced with replacement which is the actual path to the program on this server.

Note: AS/400 executable programs are required to be program objects in a library in the QSYS.LIB file system. Therefore, the URL specified by replacement must be of the form /QSYS.LIB{/lib.LIB}/*.

The beginning of the text represented by the wildcard is assumed to be the actual name of the program.

Example

```
Exec  /info/cgipgm/*  /QSYS.LIB/as400cgi.LIB/*
```

The Exec directive maps the /info/cgipgm directory into the /QSYS.LIB/as400cgi.LIB library where, in this example, our CGI programs are kept.

Note: You should be aware that the position of directive statements in the HTTP server configuration is important. For example, a statement such as Pass /* positioned before a Map or Exec statement prevents rule processing from ever reaching them, as every incoming URL matches the generic catch-all Pass statement, and rule processing ends. Always ensure that any directive statements that you add do not adversely affect subsequent statements.

4.6.9.10 Accessory Programs

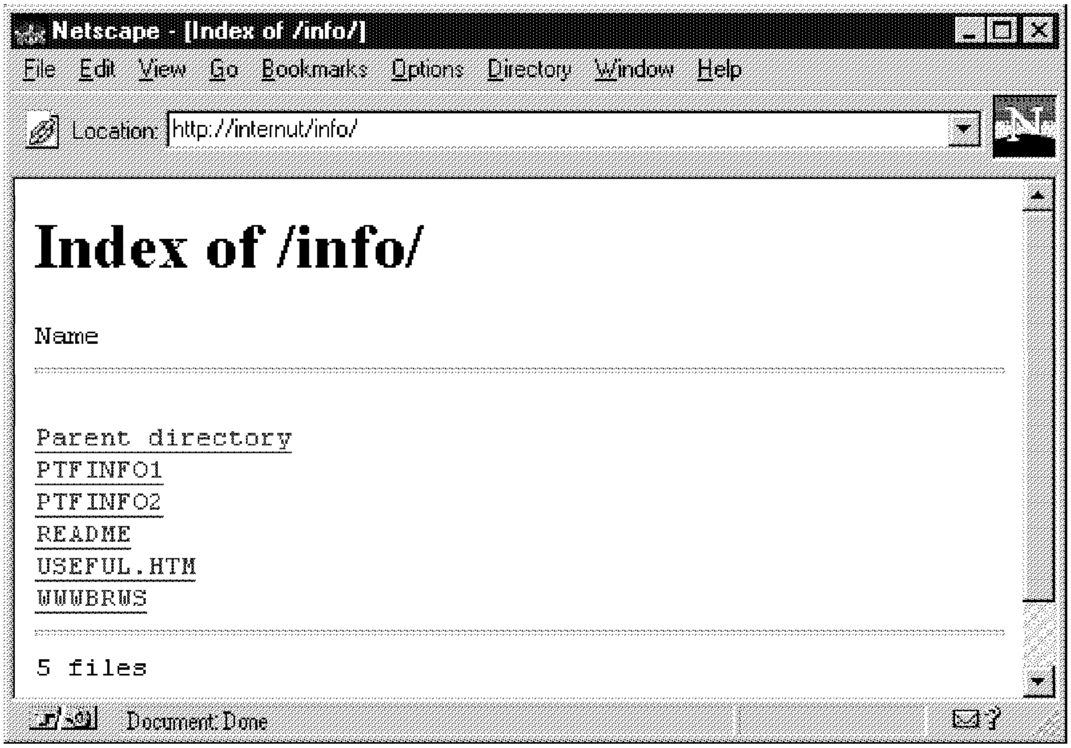
The Search and POST-script directives are used to specify a default program to use when a CGI search or a POST request are received but do not match any Exec directives.

We are not talking about this area in this book. For a detailed explanation, see the *TCP/IP Configuration and Reference*.

4.6.9.11 Directory Listings

DirAccess On | Off | Selective

This directive enables/disables the generation of a directory listing document for URLs that do not refer to a specific IFS file or QSYS.LIB member.



Index of /info/

Name

Parent directory
PTFINFO1
PTFINFO2
README
USEFUL.HTM
WWWBRWS

5 files

Figure 102. Typical Directory Listing Document

Note: The HTTP server′s decision to generate a directory listing is also affected by the Welcome and AlwaysWelcome directives.

On Generation of directory listings documents is enabled. It is enabled for URLs of the form /QSYS.LIB{/lib.LIB}/file.FILE. Other QSYS.LIB forms are rejected.

Off Generation of directory listing documents is disabled for all URLs.

Selective Generate a directory listing document when an entity named wwwbrws is present.

This enables the system administrator to selectively allow directory listings of specific files (in /QSYS.LIB{/lib.LIB}/file.FILE) and directories (IFS directories).

QSYS.LIB Create a file member named wwwbrws to allow the generation of a directory listing document of the file containing this member.

IFS Create a directory, a file object in the directory, a folder, or document within the folder called wwwbrws to generate a directory listing of the directory or folder containing the object wwwbrws. Figure 103 on page 178 shows folder LEETEST in QDLS that contains three HTML files and a file wwwbrws that allows the generation of a directory listing document for this folder.

Welcome *name*

The Welcome directive specifies the name of a file to be used when no file is specified on an incoming URL. It is a useful method of forcing a client to a particular document such as your home page.

When an incoming URL has no QSYS.LIB file, IFS file, or QDLS document specified in it, the HTTP server, depending upon the state of the AlwaysWelcome directive, scans the file or directory in the URL for a welcome file matching the name (or names) specified. If an object is found that matches a name, then it is returned to the client. If it does not find a matching object, then a directory listing document may be returned to the client in accordance with the setting of the DirAccess directive.

Table 8. HTTP Server Welcome Object Processing

Condition			**Result**
Welcome name exists	**AlwaysWelcome directive**	**URL has trailing ′/′ 1**	
Yes	ON (Default)	Ignored	Serve Welcome
Yes	OFF	Yes	Serve Welcome
Yes	OFF	No	Check DirAccess (see Table 9)
No	Ignored	Ignored	Check DirAccess (see Table 9)

1 Refers to integrated file system directory, AS/400 file or QDLS folder with no file specified. The URL must always point to a directory for welcome or directory processing.

Table 9. Directory Processing

Condition		**Result**
DirAccess directive	**′wwwbrws′ in directory**	
OFF (Default)	Ignored	Error 403
ON	Ignored	Serve Directory
Selective	Yes	Serve Directory and contents of README file if enabled with DirReadme
Selective	No	Error 403

It is possible to add multiple Welcome directives should you need a welcome page with a different name in a different file.

Note: Welcome processing is further controlled by the setting of the AlwaysWelcome directive.

AlwaysWelcome On | Off

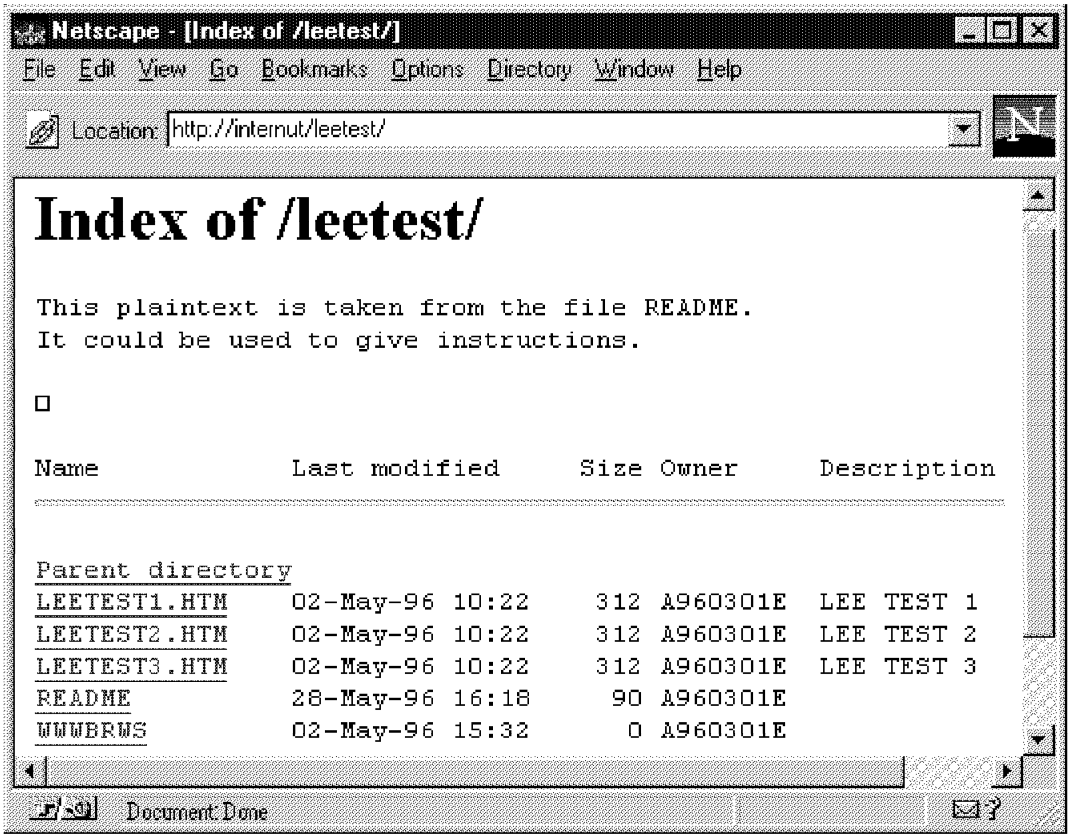
The AlwaysWelcome directive can be used to further control welcome file processing for URLs that do not refer to a specific resource.

Off If the URL ends with a trailing slash (/), the welcome file (if one exists) is returned to the client.

If the URL does not end with a trailing slash, the URL is processed in accordance with the setting of the DirAccess directive.

On The welcome file (if one exists) is always returned regardless of the presence or absence of a trailing slash in the URL.

The remaining directives in this section control the appearance of directory listing documents generated by the HTTP server. Figure 103 shows a directory listing of folder LEETEST in QDLS with directive DirReadme set to Top, DirShowDate, DirShowSize, DirShowBytes, DirShowOwner, DirShowDescription set On, and DirShowMaxDescrLenth set to 25 (the default).


Netscape - [Index of /leetest/]
File Edit View Go Bookmarks Options Directory Window Help
Location: http://internut/leetest/
Document: Done

Index of /leetest/

```
This plaintext is taken from the file README.
It could be used to give instructions.

□

Name                Last modified      Size Owner      Description
---------------------------------------------------------------------

Parent directory
LEETEST1.HTM        02-May-96 10:22     312 A960301E   LEE TEST 1
LEETEST2.HTM        02-May-96 10:22     312 A960301E   LEE TEST 2
LEETEST3.HTM        02-May-96 10:22     312 A960301E   LEE TEST 3
README              28-May-96 16:18      90 A960301E
WWWBRWS             02-May-96 15:32       0 A960301E
```

Figure 103. Directory Listing

See the following list for a brief explanation of these directives. The default setting for each directive is shown underlined.

DirReadme Top | Bottom | <u>Off</u>

This directive controls if and where README information is placed in a directory listing document.

If README is a member name in QSYS.LIB or a file name in the IFS, then the plain text it contains is displayed on the directory listing document.

DirShowDate On | <u>Off</u>

Controls whether the modification date is displayed for each entity in the listing.

DirShowSize On | Off
Controls whether the size is displayed for each entity in the listing.

DirShowBytes On **| Off**
Controls whether, for objects smaller than one kilobyte, the exact size in bytes is displayed.

DirShowOwner On | Off
Controls whether the owning user ID is displayed for each entity displayed in the listing.

DirShowDescription On **| Off**

Controls whether the descriptions of documents are included in the listing. For QSYS.LIB members, the member description is shown. For members that contain HTML and whose description is blank, or for HTML files in the IFS, the description is extracted, if possible, from the TITLE section of the document.

DirShowMaxDescrLength *len*
Sets the maximum number of characters allowed for description text.

The order in which the resultant values of these directives are displayed on the listing document is fixed regardless of the order the directives are entered in the HTTP server config.

4.6.9.12 Timeouts

These directives allow the HTTP server administrator to set time limits for the various parts of the request handling process. All three directives require a time-spec argument in one of the following forms:

- 2 minutes
- 10 minutes
- 1 hour

Whole minutes are the smallest increment that can be set.

InputTimeOut *time-spec*
The time to wait for a client to send the MIME header part of the request. (The message body, if there is one, is read during subsequent processing.)

Default: The default value is 2 minutes.

OutputTimeOut *time-spec*
The time to allow for sending the response back to the client. This time may need to be increased if large files are being served over slow links.

Default: The default value is 20 minutes.

ScriptTimeOut *time-spec*
The time to wait for a CGI program to finish. If the program does not finish in the allotted time, processing of the entire request is forcefully terminated and an error code returned to the client.

Default: The default value is 5 minutes.

4.6.9.13 Updates to Existing TCP/IP Commands

The following is a list of TCP/IP commands that are updated to support the TCP/IP server.

1. CFGTCPAPP (Configure TCP/IP Applications)

   The menu of options is updated to include an option to call the CFGTCPHTTP command.

2. STRTCPSVR (Start TCP/IP Servers)

   The SERVER parameter is updated to allow the special value *HTTP (Start HTTP server).

   A new *RESTART parameter is added, which is only valid for a server value of *HTTP. This gives the HTTP server a restart option, which is common with CERN-like HTTP servers and gives the customers a greater degree of control.

All HTTP server jobs run in batch mode under the QSYSWRK subsystem, and are started with the STRTCPSRV SERVER(*HTTP) command and ended with the ENDTCPSRV SERVER(*HTTP) command.

4.6.9.14 URI Interpretation

The AS/400 HTTP server acts on browser requests in accordance with the content of the request URI (Universal Resource Identifier) from the request line.

The following description shows how URLs for each file system are written and handled.

Requests for QSYS.LIB objects

Physical Files and *Source Physical Files*

The format for a physical file/source physical file URL is:

```
QSYS.LIB/library.LIB/file.FILE/member.MBR
```

If a member is identified, the member is sent after being translated from EBCDIC to ASCII. The member type *must* be HTML for the contents to be identified as HTML, otherwise the content is identified as plain text.

If the file is specified with no member, a list of the members is sent as an HTML document using the member descriptions as identifiers. If a member HEADER is present, it is added as a description at the beginning of a list. If a member README is present, it is added as a description at the end of the list.

Requests for QDLS objects

The format for a QDLS URL is:

```
QDLS/folder1/folder.../document
```

PCFILE objects in QDLS are sent ″as-is″ with the file extension determining the type. A list of types is provided with the server, and others may be added.

Program call (CGI interface)

The format for a program call is:

```
QSYS.LIB/library.LIB/program.PGM
```

The client can request that the HTTP server run a program. The program identified by the client is expected to produce an HTML document that the server can serve back to the requesting client. The Common Gateway Interface (CGI) specification defines how the server is expected to call such external programs, and how those programs should return the document that they produce. Refer to 6.1.5, “Linking to a CGI Script - Calendar” on page 228 and 6.1.9, “OS/400 CGI - ITSO Company” on page 253 for some CGI implementation examples.

Note: The CGI program must be in the QSYS.LIB file system which is the only file system from which you can execute programs on the AS/400 system.

DB2 database requests

DB2/www is an extension to the AS/400 HTTP server. The HTTP server recognizes requests for DB2/www when parsing the URL of a client′s request and passes such requests to the DB2/www server extension.

The format for a DB2/www URL is:

```
QSYS.LIB/library.LIB/db2www.PGM/{macro-file}/{command}
```

Maps Image map requests are another form of program call (CGI) requests.

The format for a image map request is:

```
QSYS.LIB/library.LIB/imagemap.PGM/{imagemap-spec}
```

HTML maps can be served by either a server-provided image mapping program (image map) or a user-provided image mapping program. Refer to 5.9.1, “Clickable Image Maps - Shapes” on page 196 and 5.9.2, “Clickable Image Maps - World Map” on page 200 for examples of implementing clickable image maps.

Note: The image map program must be in the QSYS.LIB file system which is the only file system from which you can execute programs on the AS/400 system.

Request for other file systems

Requests for objects from other file systems are processed using the Integrated File System.

- Root
- QOpenSys
- QLANSrv

Files from these file systems are considered to be binary files not requiring translation. They are sent as-is to the client.

Chapter 5. Establishing a Presence on the Internet

In this chapter we discuss the process whereby you can establish a presence on the Internet using an Internet Connection server.

To start with we might ask some basic questions:

- What may be presented on the Web?
- Why should I establish a presence on the Web?
- Can I save costs by having a Web presence?

5.1 What May Be Presented on the Web?

It is possible to use the multimedia capabilities of the Web to present many diverse types of information including the following:

- General administrative information:
  - E-mail, reports, memos
  - Announcement of organizational changes
  - Corporate policies
  - Posting of job openings
- Marketing and sales information:
  - Description of products/services
  - Catalog/information ordering
  - Pricing, features, functions and benefits
  - Photos of products and events
- Engineering information:
  - Drawings
  - Specifications
  - Manuals
- Transaction information:
  - Statistics
  - Database queries
- Software updates

5.2 Why Should I Establish a Presence on the Web?

This is an easy question to answer. There are millions of people around the world who connect to the Web and who are, therefore, your potential audience. And of course the Web population is growing all the time.

In addition, the following characteristics of the worldwide Internet are worth considering:

- Speed - Instant access anywhere in the world, no delivery times
- Availability - 24 hours a day, 7 days a week
- Cost - One monthly fee to the Internet provider
- Scope - Send/receive information to/from anyone, anywhere, anytime so long as a connection exists

5.3 Can I Save Costs by Having a Web Presence?

Providing information to your customers via the Web is highly cost-effective. Let′s look at some examples.

You need not send thousands of letters to your customers. Make or change just one page on your Web site and the information given to your customers is up to date.

Place all the items on your Web site that your customer may ask for. This saves you and your customer time and money.

Add a remark to your advertising pages on TV, newspaper etc. that advertises your Web site′s URL. There are no additional costs associated with doing this.

As a user of the Web, you can gain valuable commercial marketing information by watching the news groups or online forums on the Internet. It can be useful to be active in these forums, to give answers or raise questions regarding your products and services.

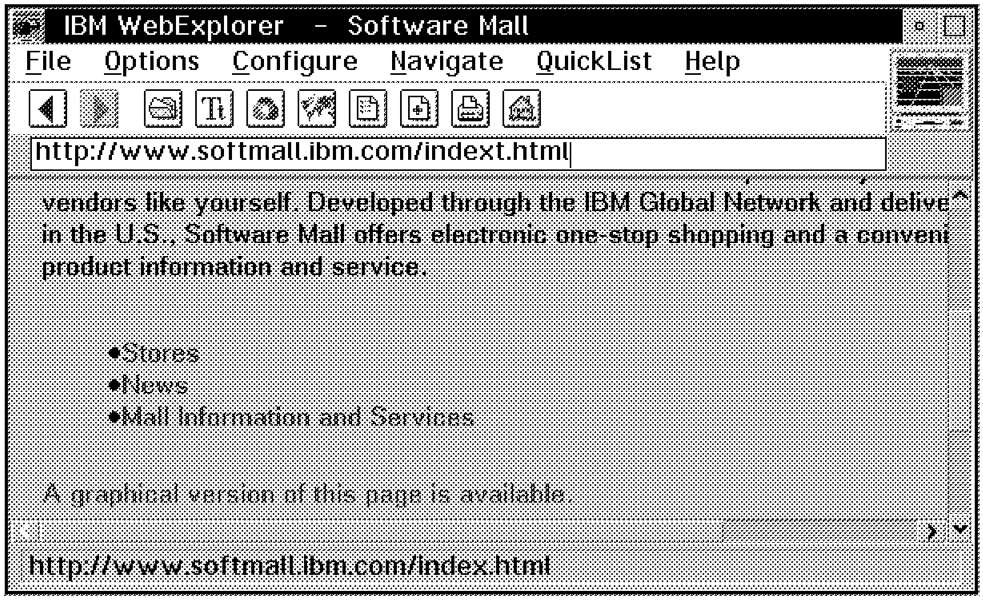
5.4 Designing Your Web Page

In the following sections we discuss the different types of Web pages that you can choose from. These are as follows:

- Text only
- Text and graphics
- Forms

5.4.1 Text and Graphics Web Pages

A text Web page contains no graphic elements. Such a Web page may be viewed by all kinds of browsers even those without graphic support.



http://www.softmall.ibm.com/indext.html

vendors like yourself. Developed through the IBM Global Network and delive
in the U.S., Software Mall offers electronic one-stop shopping and a conveni
product information and service.

- Stores
- News
- Mall Information and Services

A graphical version of this page is available.

http://www.softmall.ibm.com/index.html

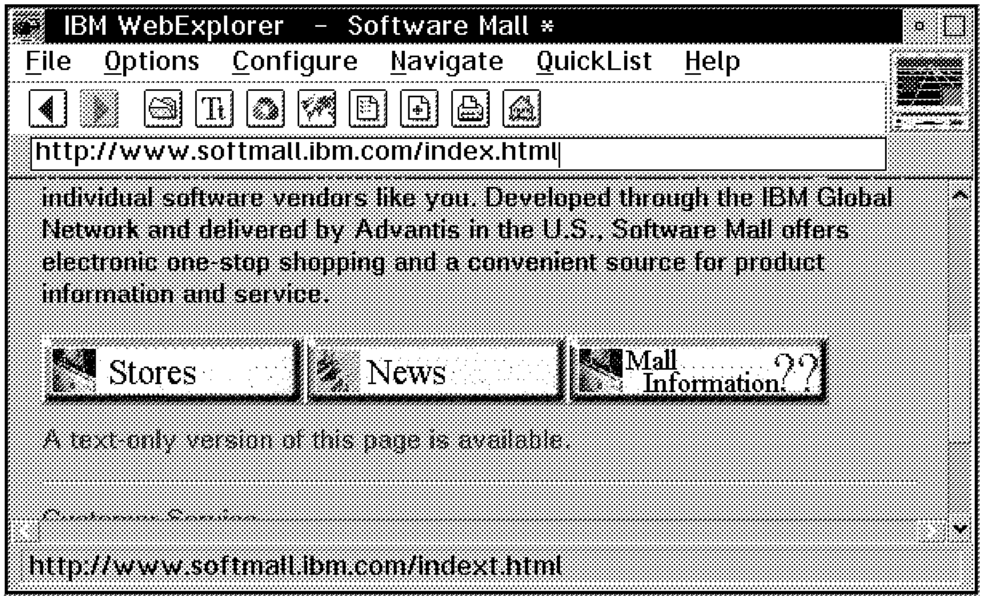
Figure 104. A Text-Only Web Page

The indext.html file generates the text-only page shown in Figure 104. The file indext.html is designed for browsers with a limited capability. By pointing to A graphical version of this page is available, file index.html will be invoked. This page is built with HTML that supports and invokes the graphical features shown in Figure 105.

Here is an excerpt from the indext.html file:

```
electronic one-stop shopping and a convenient source for product information
and service.
<p>
<ul>
<li><a href=”/stores/index.html”>Stores</a>
<li><a href=”/news.html”>News</a>
<li><a href=”/info.html”>Mall Information and Services</a>
</ul>

<p><a href=”/index.html”>A graphical version of this page is available.</

<hr>
<a href=”/mall/cs_mall.html”>Customer Service</a><br>
<!-- end ibmbody--------------------------------------------------------
<hr>
<h4><b>
```

Here is the result of selecting the hyperlink to index.html:


IBM WebExplorer - Software Mall *
File Options Configure Navigate QuickList Help
Stores
News
Mall Information ??

http://www.softmall.ibm.com/index.html

individual software vendors like you. Developed through the IBM Global Network and delivered by Advantis in the U.S., Software Mall offers electronic one-stop shopping and a convenient source for product information and service.

A text-only version of this page is available.

http://www.softmall.ibm.com/indext.html

Figure 105. A Graphic Web Page

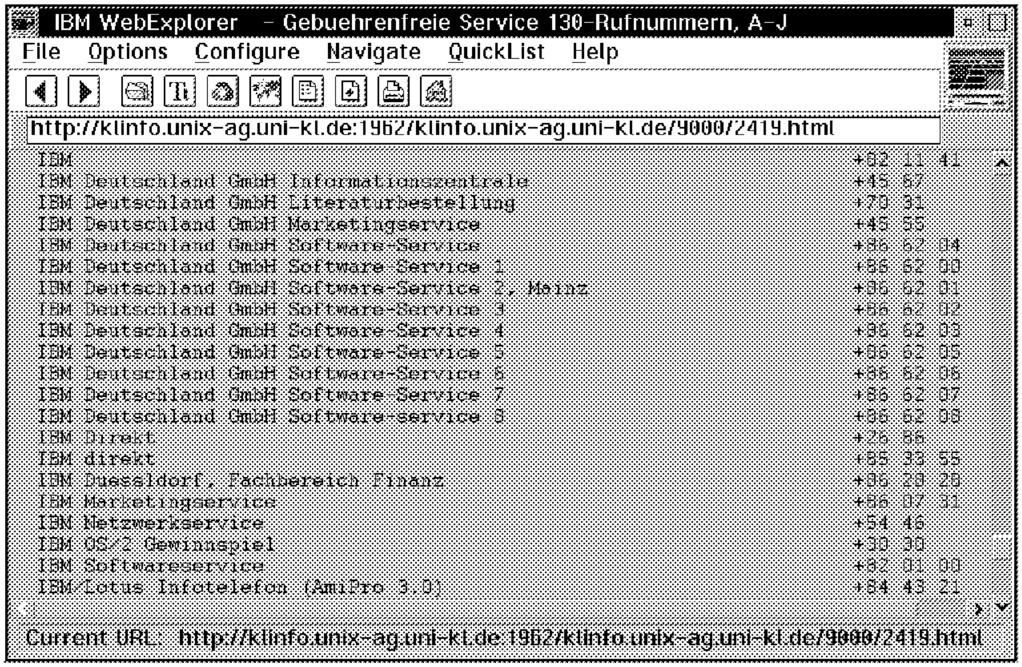
Here is an excerpt from the index.html file:

```
Electronic one-stop shopping and a convenient source for product information
and service. </p>
<p>
<a href=”/stores/index.html”>
   <img src=”/icons/store.gif” border=0 alt=”Stores”></a>
<a href=”/news.html”>
   <img src=”/icons/news.gif”  border=0 alt=”News”></a>
<a href=”/info.html”>
```

```
   <img src="/icons/info.gif"  border=0 alt="Info"></a>
</p>
<p>

<a href="/indext.html">A text-only version of this page is available.</a>

</p>
<hr>
<a href="/mall/cs_mall.html">Customer Service</a><br>
<!-- end ibmbody----------------------------------------------------------
<hr>
<h4><b>
```

By clicking on one of the three graphical icons (Stores, News or Mall Information) one of three further pages will be loaded (index, news or info HTML files).

Here is another text-based Web page:


IBM WebExplorer - Gebuehrenfreie Service 130-Rufnummern, A-J
File
Options
Configure
Navigate
QuickList
Help

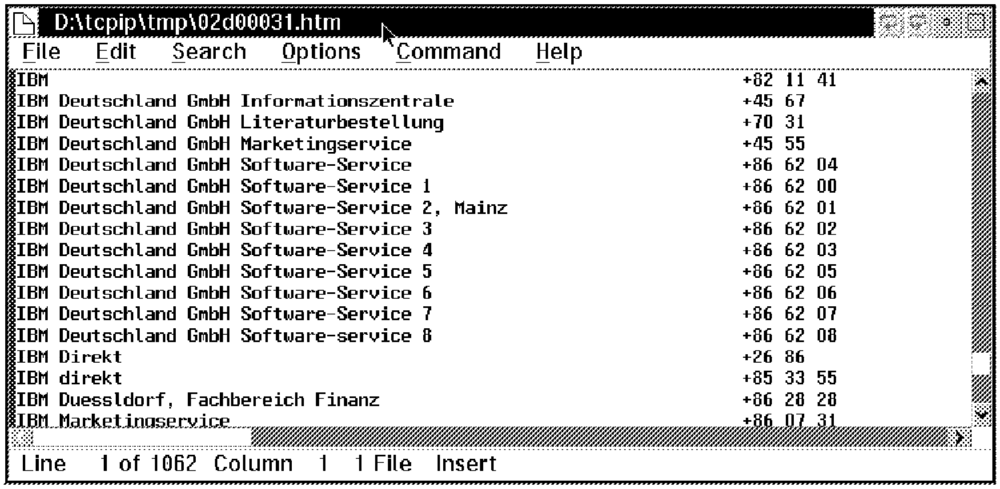
http://klinfo.unix-ag.uni-kl.de:1962/klinfo.unix-ag.uni-kl.de/9000/2419.html

```
IBM                                                      +82 11 41
IBM Deutschland GmbH Informationszentrale                +45 67
IBM Deutschland GmbH Literaturbestellung                 +70 31
IBM Deutschland GmbH Marketingservice                    +45 55
IBM Deutschland GmbH Software-Service                    +86 62 04
IBM Deutschland GmbH Software-Service 1                  +86 62 00
IBM Deutschland GmbH Software-Service 2, Mainz           +86 62 01
IBM Deutschland GmbH Software-Service 3                  +86 62 02
IBM Deutschland GmbH Software-Service 4                  +86 62 03
IBM Deutschland GmbH Software-Service 5                  +86 62 05
IBM Deutschland GmbH Software-Service 6                  +86 62 06
IBM Deutschland GmbH Software-Service 7                  +86 62 07
IBM Deutschland GmbH Software-service 8                  +86 62 08
IBM Direkt                                               +26 86
IBM direkt                                               +85 33 55
IBM Duesseldorf, Fachbereich Finanz                      +86 28 28
IBM Marketingservice                                     +86 07 31
IBM Netzwerkservice                                      +54 46
IBM OS/2 Gewinnspiel                                     +30 30
IBM Softwareservice                                      +82 01 00
IBM/Lotus Infotelefon (AmiPro 3.0)                       +84 43 21
```

Current URL: http://klinfo.unix-ag.uni-kl.de:1962/klinfo.unix-ag.uni-kl.de/9000/2419.html

Figure 106. ASCII Web Page

Here we are viewing the HTML associated with Figure 106. This we did by selecting the **View HTML** option from the File pull-down menu of WebExplorer.

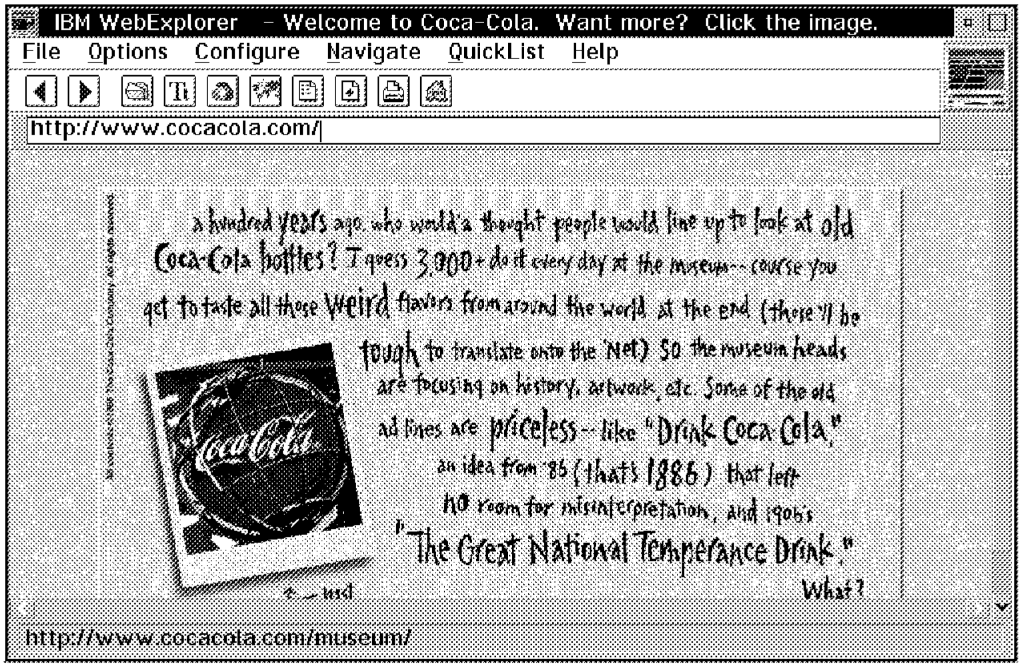


```
IBM                                                          +82 11 41
IBM Deutschland GmbH Informationszentrale                    +45 67
IBM Deutschland GmbH Literaturbestellung                     +70 31
IBM Deutschland GmbH Marketingservice                        +45 55
IBM Deutschland GmbH Software-Service                        +86 62 04
IBM Deutschland GmbH Software-Service 1                      +86 62 00
IBM Deutschland GmbH Software-Service 2, Mainz               +86 62 01
IBM Deutschland GmbH Software-Service 3                      +86 62 02
IBM Deutschland GmbH Software-Service 4                      +86 62 03
IBM Deutschland GmbH Software-Service 5                      +86 62 05
IBM Deutschland GmbH Software-Service 6                      +86 62 06
IBM Deutschland GmbH Software-Service 7                      +86 62 07
IBM Deutschland GmbH Software-service 8                      +86 62 08
IBM Direkt                                                   +26 86
IBM direkt                                                   +85 33 55
IBM Duessldorf, Fachbereich Finanz                           +86 28 28
IBM Marketingservice                                         +86 07 31
```

Line 1 of 1062 Column 1 1 File Insert

Figure 107. Viewing the HTML

The Web page shown in Figure 108 consists of a background file and an image file.


IBM WebExplorer - Welcome to Coca-Cola. Want more? Click the image.
File
Options
Configure
Navigate
QuickList
Help
Coca-Cola
← last

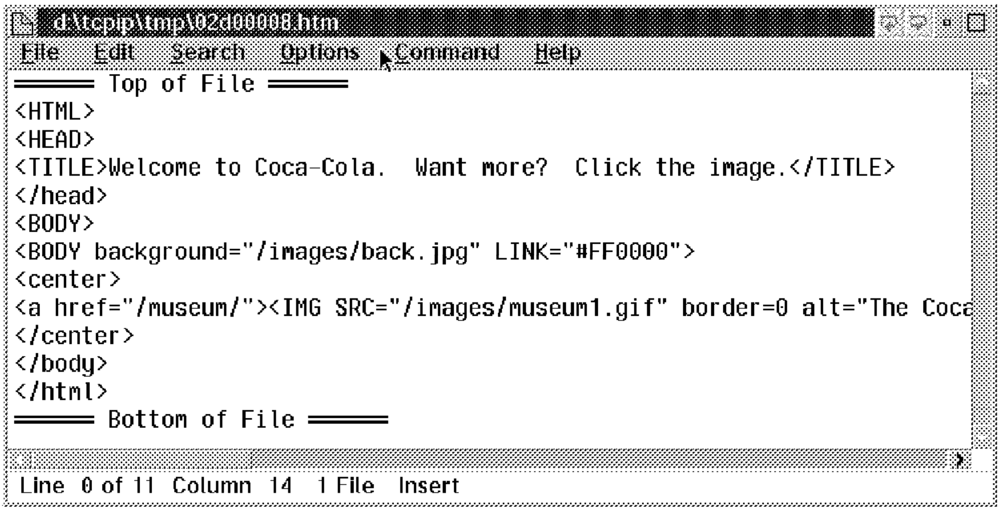
http://www.cocacola.com/

a hundred years ago, who would'a thought people would line up to look at old Coca-Cola bottles? I guess 3,000+ do it every day at the museum-- course you get to taste all those weird flavors from around the world at the end (there'll be tough to translate onto the 'Net) So the museum heads are focusing on history, artwork, etc. Some of the old ad lines are priceless -- like "Drink Coca Cola," an idea from '86 (that's 1886) that left no room for misinterpretation, and 1906's "The Great National Temperance Drink." What?

http://www.cocacola.com/museum/

Figure 108. Graphic Web Page

Here we are viewing the HTML associated with Figure 109 on page 188.


d:\tcpip\tmp\02d00008.htm
File Edit Search Options Command Help

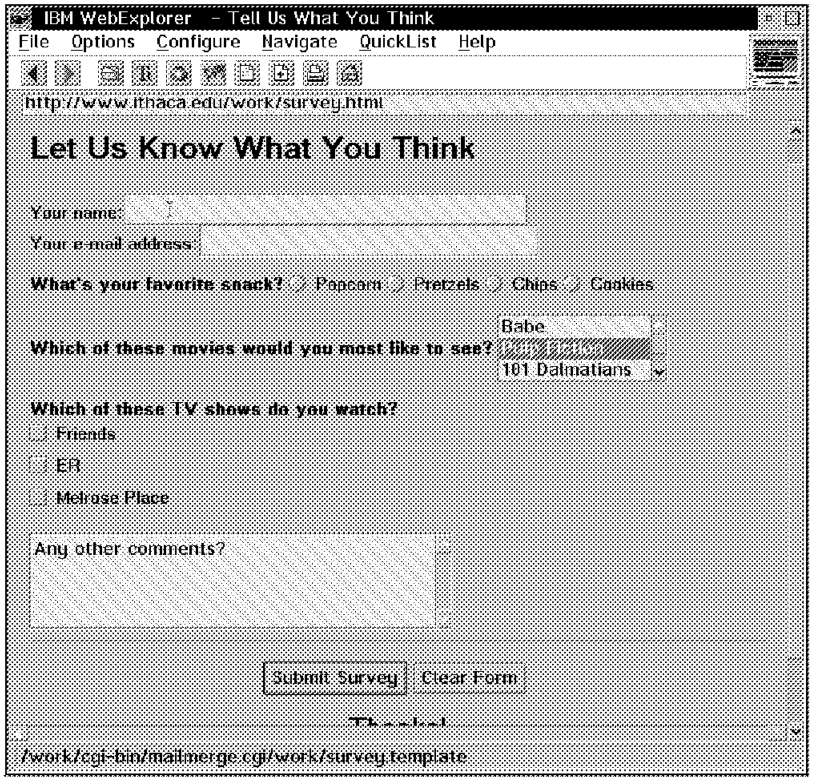
```
═════ Top of File ═════
<HTML>
<HEAD>
<TITLE>Welcome to Coca-Cola.  Want more?  Click the image.</TITLE>
</head>
<BODY>
<BODY background="/images/back.jpg" LINK="#FF0000">
<center>
<a href="/museum/"><IMG SRC="/images/museum1.gif" border=0 alt="The Coca
</center>
</body>
</html>
═════ Bottom of File ═════
```

Line 0 of 11 Column 14 1 File Insert

Figure 109. HTML Document of Graphic Web Page

5.4.2 Form Web Pages

This type of Web page allows you to fill in text in preformatted fields and submit the form to an electronic address. Here is an example:


IBM WebExplorer - Tell Us What You Think
File Options Configure Navigate QuickList Help
http://www.ithaca.edu/work/survey.html
Submit Survey
Clear Form
/work/cgi-bin/mailmerge.cgi/work/survey.template

Let Us Know What You Think

Your name:

Your e-mail address:

What's your favorite snack? ○ Popcorn ○ Pretzels ○ Chips ○ Cookies

Which of these movies would you most like to see? Babe / 101 Dalmatians

Which of these TV shows do you watch?

□ Friends

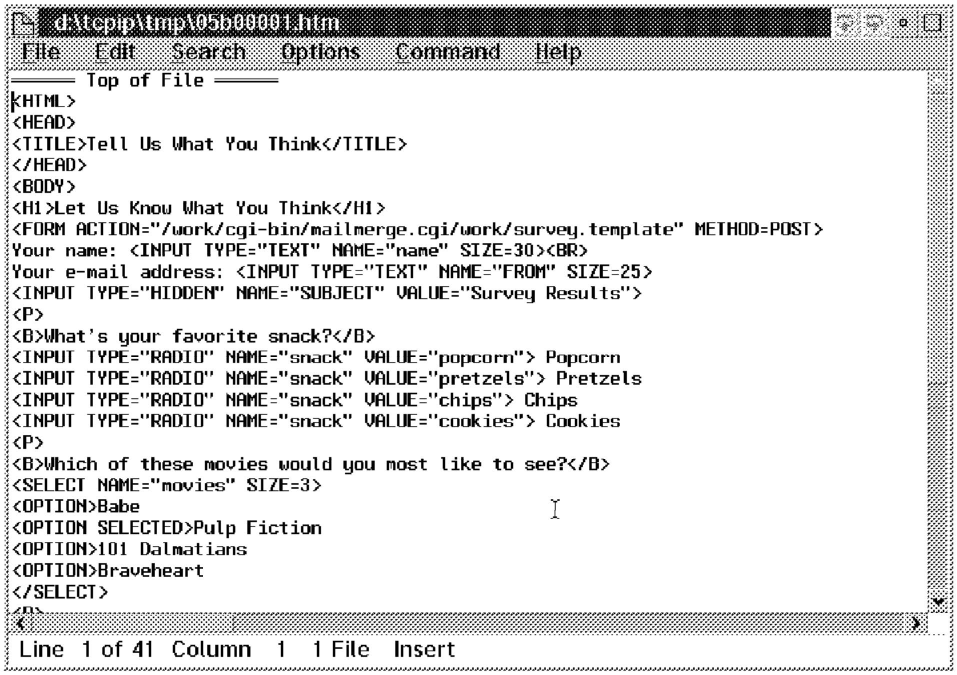
□ ER

□ Melrose Place

Any other comments?

Figure 110. Web Page with Forms Elements

Here we are viewing the HTML associated with Figure 110.


d:\tcpip\tmp\05b00001.htm
File Edit Search Options Command Help

```
===== Top of File =====
<HTML>
<HEAD>
<TITLE>Tell Us What You Think</TITLE>
</HEAD>
<BODY>
<H1>Let Us Know What You Think</H1>
<FORM ACTION="/work/cgi-bin/mailmerge.cgi/work/survey.template" METHOD=POST>
Your name: <INPUT TYPE="TEXT" NAME="name" SIZE=30><BR>
Your e-mail address: <INPUT TYPE="TEXT" NAME="FROM" SIZE=25>
<INPUT TYPE="HIDDEN" NAME="SUBJECT" VALUE="Survey Results">
<P>
<B>What's your favorite snack?</B>
<INPUT TYPE="RADIO" NAME="snack" VALUE="popcorn"> Popcorn
<INPUT TYPE="RADIO" NAME="snack" VALUE="pretzels"> Pretzels
<INPUT TYPE="RADIO" NAME="snack" VALUE="chips"> Chips
<INPUT TYPE="RADIO" NAME="snack" VALUE="cookies"> Cookies
<P>
<B>Which of these movies would you most like to see?</B>
<SELECT NAME="movies" SIZE=3>
<OPTION>Babe
<OPTION SELECTED>Pulp Fiction
<OPTION>101 Dalmatians
<OPTION>Braveheart
</SELECT>
```

Line 1 of 41 Column 1 1 File Insert

Figure 111. HTML Document with Form Elements

5.5 Searching the Internet

In this section we look at what is available to assist in searching for information on the Internet.

5.5.1 Search Engines

There are many search engines on the Internet. Among the most popular are:

- AltaVista

  `http://www.altavista.digital.com`

- InfoMarket

  `http://www.infomarket.ibm.com`

- Lycos

  `http://www.lycos.com`

- Magellan

  `http://www.mckinley.com`

- Netscape

  `http://www.netscape.com`

- Search

  `http://www.search.com`

- WebCrawler

  `http://webcrawler.com`

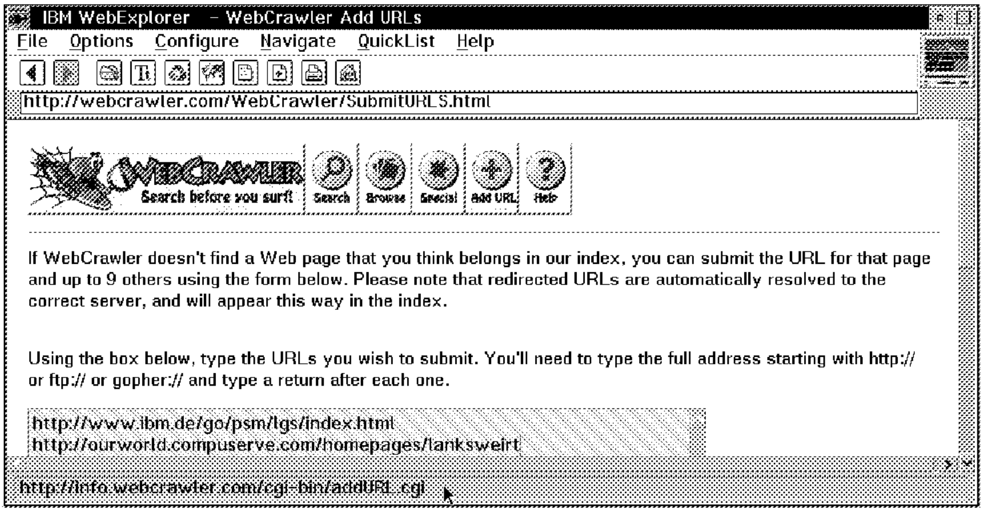
- Yahoo

  `http://www.yahoo.com`

Each program has its own way to start searching. All provide help facilities which provide further information. It is very useful to know how to limit the search and get better results.

The search machines will often provide a tool to add your document to the search machine′s database. Thus, you are given the opportunity to promote or advertise any intellectual capital that you would like to make freely available on the Web.

In Figure 112 on page 191, we see how it is possible to add URLs to the WebCrawler index database.


IBM WebExplorer - WebCrawler Add URLs
File Options Configure Navigate QuickList Help
WebCrawler
Search before you surf!
Search
Browse
Special
Add URL
Help

http://webcrawler.com/WebCrawler/SubmitURLS.html

If WebCrawler doesn't find a Web page that you think belongs in our index, you can submit the URL for that page and up to 9 others using the form below. Please note that redirected URLs are automatically resolved to the correct server, and will appear this way in the index.

Using the box below, type the URLs you wish to submit. You'll need to type the full address starting with http:// or ftp:// or gopher:// and type a return after each one.

http://www.ibm.de/go/psm/lgs/index.html
http://ourworld.compuserve.com/homepages/lanksweirt

http://info.webcrawler.com/cgi-bin/addURL.cgi

Figure 112. Adding Documents to the WebCrawler Index

InfoMarket Search on the other hand provides a *cryptolope* mechanism which enables articles, literature and other forms of intellectual capital to be available at a price. Only by paying for the viewing rights, using a secure Web browser, are you able to obtain a document.

A collection of WWW search programs is available at:

```
http://cui.unige.ch/meta-index.html
http://ds2.internic.net/tools
```

We found lists with the best Web pages at:

```
http://www.primenet.com/donnlee/best95.html
http://www.bergen.gov/AAST/Wow/wow.html
```

For registering your server you could start at:

```
http://www.w3.org/hypertext/DataSources/WWW/Geographical_generation/
new-servers.html
```

5.5.2 Worms, Spiders and Robots

These tools automate the process of searching Web servers.

Disadvantages:

- Server and network performance will be decreased when using these tools.
- The tools cannot usually evaluate the quality of the information that they find.

Here are some examples of such tools with their associated URLs:

- Aliweb

```
http://www.nexor.co.uk/public/aliweb/aliweb.html
```

- Swish

```
http://www.eit.com/software/swish/swish.html
```

- Wais

```
http://www.eit.com/software/wwwwais/wwwwais.html
```

In addition, a list of WWW robots, wanderers and spiders can be found at:

```
http://info.webcrawler.com/mak/projects/robots/robots.html
```

5.6 Using URLs

A Uniform Resource Locator (URL) is used to point to a specific file on an Internet host. URLs can also point to queries and documents stored within databases.

Different uses for the URL:

- File URL

  URLs to a file server are specified as follows:

  ```
  file://ftp.tamtam.com/public/files/rire.txt
  ```

  Top level is: files://ftp.tamtam.com/.

  Public directory level is files://ftp.tamtam.com/pub.

- Gopher URL

  URLs to a gopher server are specified as follows:

  ```
  gopher://os2info.austin.ibm.com
  ```

- News URL

  URLs to a newsgroup are specified as follows:

  ```
  news:comp.os.os2.misc
  ```

  Note that not every browser supports this news command.

- WWW URL

  URLs to a Web server are specified as follows:

  ```
  http://www.baba.com/~ doris
  ```

  The tilde (~) indicates that the resource is in the specified home directory of user doris.

A reference to URLs and more can be found at:

```
http://www.w3.org/pub/WWW/Addressing
```

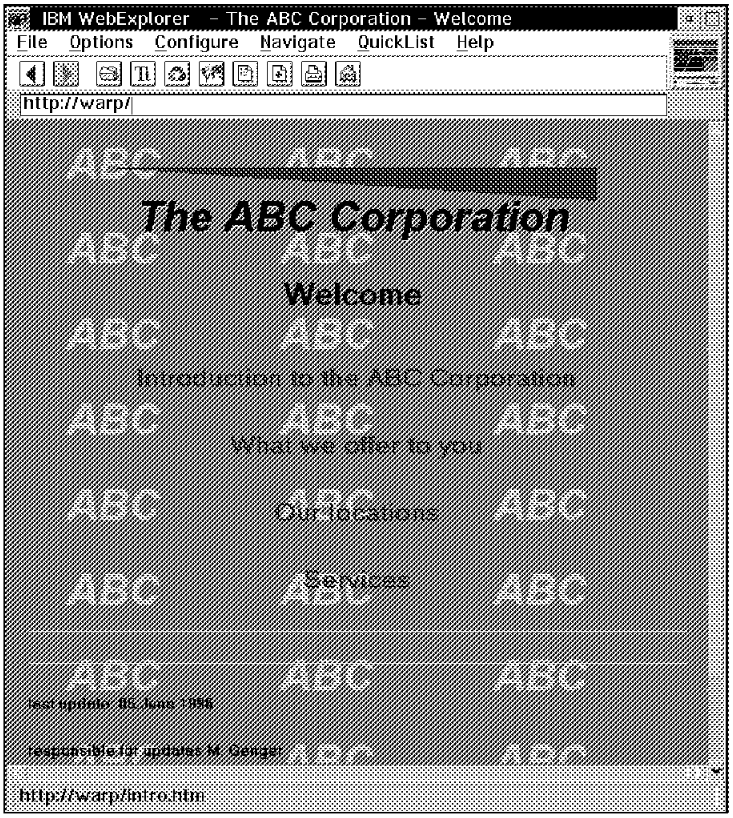
A URL normally indicates the absolute position of a resource. Therefore, selecting the URL will take you straight to the resource without needing to take a step-by-step path through a host′s directory structure.

5.7 Example of Building a Presence on the Internet

In this section we introduce an example of building a Web server presence on the Internet. You will learn among other things how easy it is to build HTML pages.

5.7.1 The ABC Corporation Pages

The home page of the fictitious ABC corporation is depicted in Figure 113 on page 193.



The ABC Corporation

Welcome

Introduction to the ABC Corporation

What we offer to you

Our locations

Services

last update: 05 June 1996

responsible for updates M. Genger

http://warp/intro.htm

Figure 113. The Home Page of the ABC Corporation

We set up this home page as follows:

We created two gif files:

- abcback.gif
- abcmast.gif

File abcback.gif is the file which repeats ABC in the background. File abcmast.gif contains the header The ABC Corporation and a graphic triangle.

Next we created The ABC Corporation home page and named it abcmast.htm. The HTML contained in abcmast.htm is as follows:

```
<HTML>
<HEAD>
<TITLE>The ABC Corporation - Welcome</TITLE>
</HEAD>
<BODY BACKGROUND="abcback.gif" BGCOLOR=00FF00>
<CENTER>
<H1>
<IMG ALT="The ABC Corporation" SRC="abcmast.gif"><BR>
Welcome
</Center>
<li>
<Center>
<H2> <A HREF="intro.htm">Introduction to the ABC Corporation</A></H2>
<li>
<H2> <A HREF="wwoffer.htm">What we offer to you</A></H2>
```

```
<li>
<H2> <A HREF="ourloc.htm">Our locations</A></H2>
<li>
<H2> <A HREF="services.htm">Services</A></H2>
</Center>
</H1>
<HR>
<HR>
</CENTER>
</BODY>

<H6> last update: 05.June 1996 </H6>
<H6> responsible for updates M. Genger </H6>
<H6> <A HREF="mailto:mgenger@vnet.ibm.com"> send mail to M. Genger </H6>
</HTML>
```

Within the document, there are three links to other documents:

- What we offer to you
- Our locations
- Services

and a mailto function to the person responsible for the document. Please notice that, in this example, all documents are stored in the same directory on the server.

If you select What we offer to you, you will be hyperlinked to the wwoffer.htm document.

Document wwoffer.htm contains the following HTML:

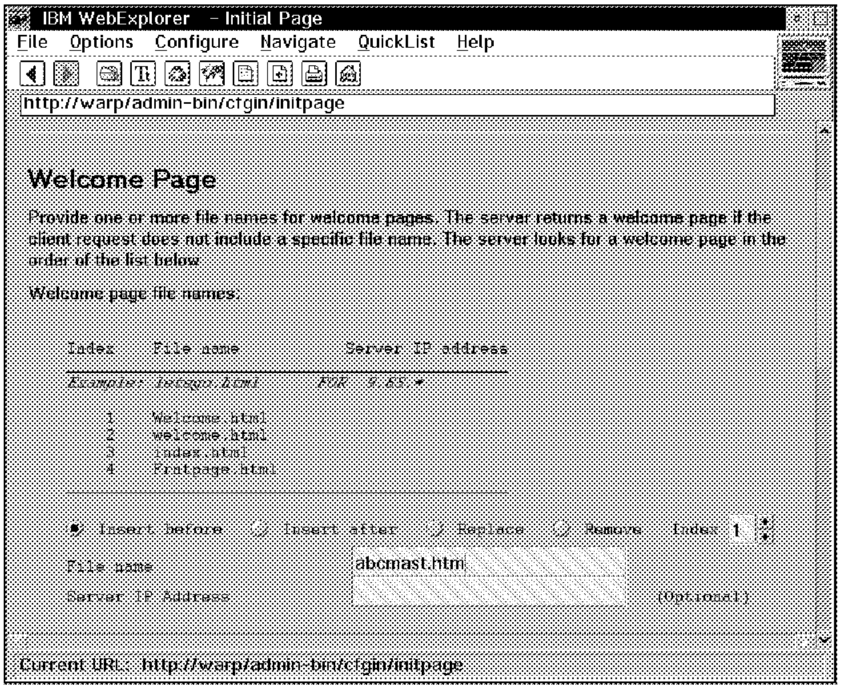
```
<HTML>
<HEAD>
<TITLE>Introduction to the ABC Corporation </TITLE>
</HEAD>
<BODY BACKGROUND="abcback.gif" BGCOLOR=00FF00>
<CENTER>
<br>
<ul>
<H2> We offer:  </H2>
</br>
<ul><br>
<H2> Delivery all over the world </H2>
<ul><br>
<H2> Best quality at lowest price   </H2>
<ul><br>
<H2> Fullservice</H2>
</CENTER>
```

The other two pages, Our locations and Services, are set up in a similar way.

Next we copied the HTML pages to our server directory:

```
D:/www/html
```

To provide access to the new pages we changed the configuration sequence of the HTML pages and added abcmast.htm as the first page to be shown (see Figure 114 on page 195). Do not forget to restart the server to make the changes effective.


IBM WebExplorer - Initial Page
File Options Configure Navigate QuickList Help

http://warp/admin-bin/cfgin/initpage

Welcome Page

Provide one or more file names for welcome pages. The server returns a welcome page if the client request does not include a specific file name. The server looks for a welcome page in the order of the list below

Welcome page file names:

Index	File name	Server IP address
Example:	*letsgo.html*	[illegible]
1	Welcome.html	
2	welcome.html	
3	index.html	
4	Frntpage.html	

● Insert before ○ Insert after ○ Replace ○ Remove Index 1

File name: abcmast.htm

Server IP Address: (Optional)

Current URL: http://warp/admin-bin/cfgin/initpage

Figure 114. The Welcome Page Configuration Panel

To get access to the Configuration and Administration panel on the server add frntpage.html to the URL of your server, as follows:

```
http://warp/frntpage.html
```

Bear in mind that when the server receives requests that do not point to a specific directory, it tries to serve the request from the document root directory. As a result of this, if you want your home page to be returned for requests that do not specify a directory or file name, you need to have your home page in your document root directory.

The document root directory is the directory you specified as your HTML directory during installation. If you used the installation defaults the document root directory is c:/www/html.

5.8 Changing Your Document Root Directory

If you are creating a new set of HTML documents for your server you might want to change your document root directory.

From the Configuration and Administration Forms page click **Request routing**. On the request routing form do the following:

- Select the **Replace** button.
- Change the index field to the number of the pass rule that has /* as its URL request template.
- Change the action field to pass.
- Enter /* in the URL request template field.
- Enter your new document root directory in the replacement file path field.
- Click **Apply**.

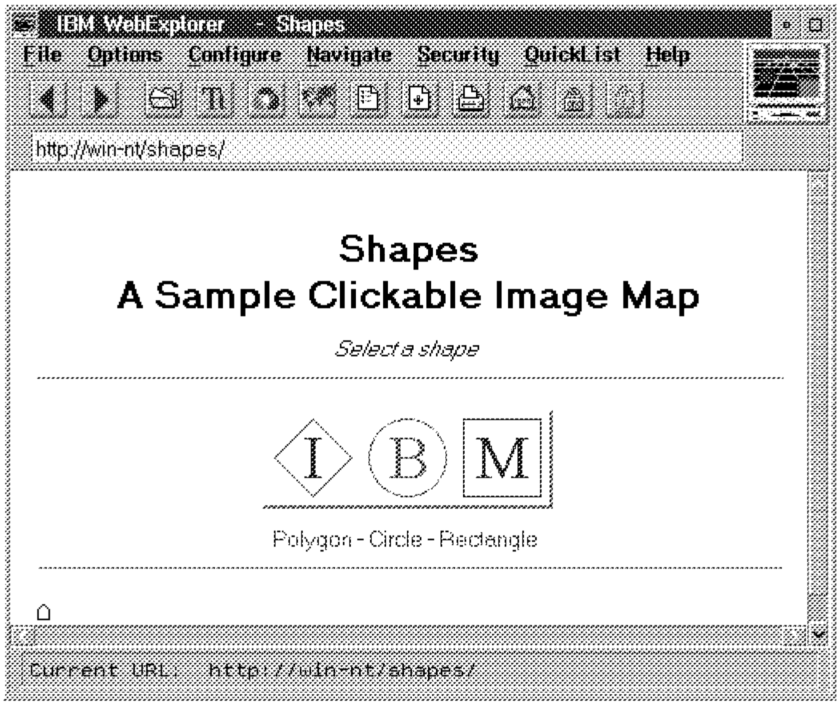
5.9 Enhancing Your Web Server

In this section we look at examples of using clickable image maps, animation and server-side includes to enhance the appeal and usability of your Web server.

5.9.1 Clickable Image Maps - Shapes

Our first example is a clickable image map. This example is remarkable for two reasons. The first is that it requires no programming at all, since the htimage service it involves is shipped with the Internet Connection Server package. The second is that htimage is called in combination with an IMG tag, and not as a stand-alone URL or as a form action.

Users will see something like Figure 115 when they link to our Shapes home page if their browser can display GIF graphics.



Shapes
A Sample Clickable Image Map

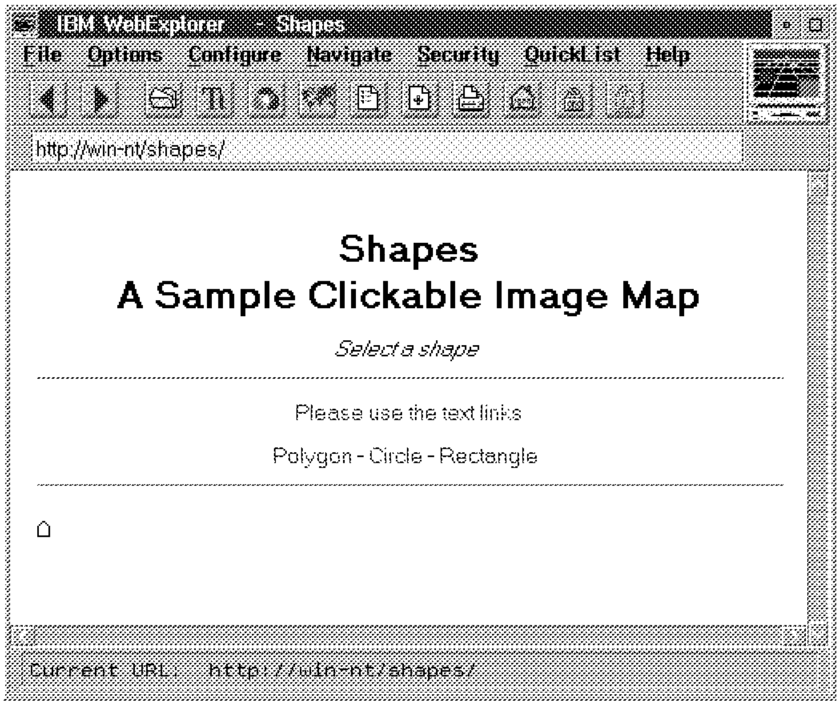
Select a shape

Polygon - Circle - Rectangle

Current URL: http://win-nt/shapes/

Figure 115. The Shapes Home Page, When Graphics are Displayed

If not, their browser will display something like Figure 116 on page 197.



http://win-nt/shapes/

Shapes
A Sample Clickable Image Map

Select a shape

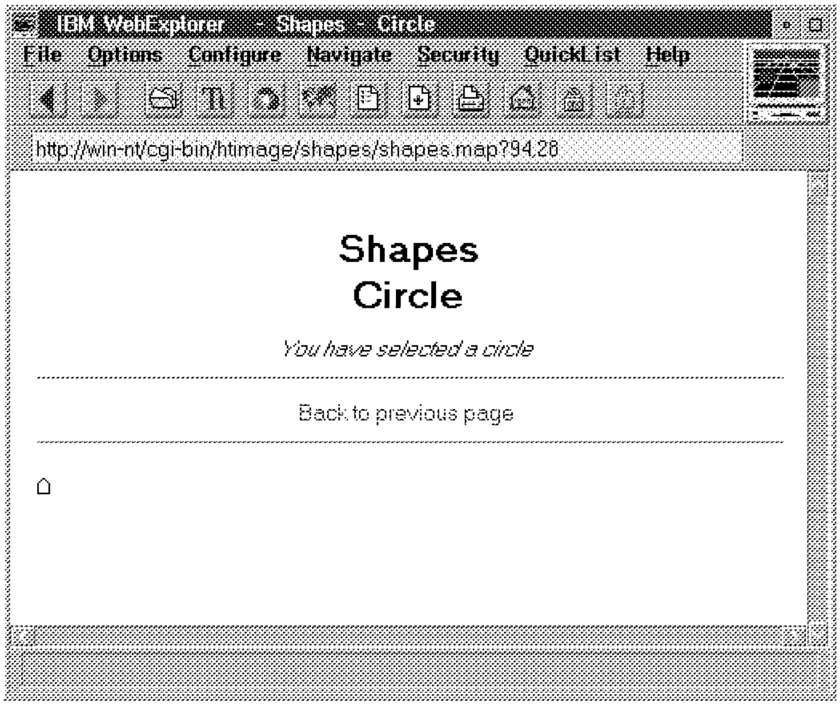
Please use the text links

Polygon - Circle - Rectangle

Current URL: http://win-nt/shapes/

Figure 116. The Shapes Home Page, When Graphics Are Not Displayed

In the first case, when graphics are supported, clicking in one of the shapes represented on the image will take the user to the corresponding page. One of these pages is represented in Figure 117.


IBM WebExplorer - Shapes - Circle
File Options Configure Navigate Security QuickList Help

http://win-nt/cgi-bin/htimage/shapes/shapes.map?94,28

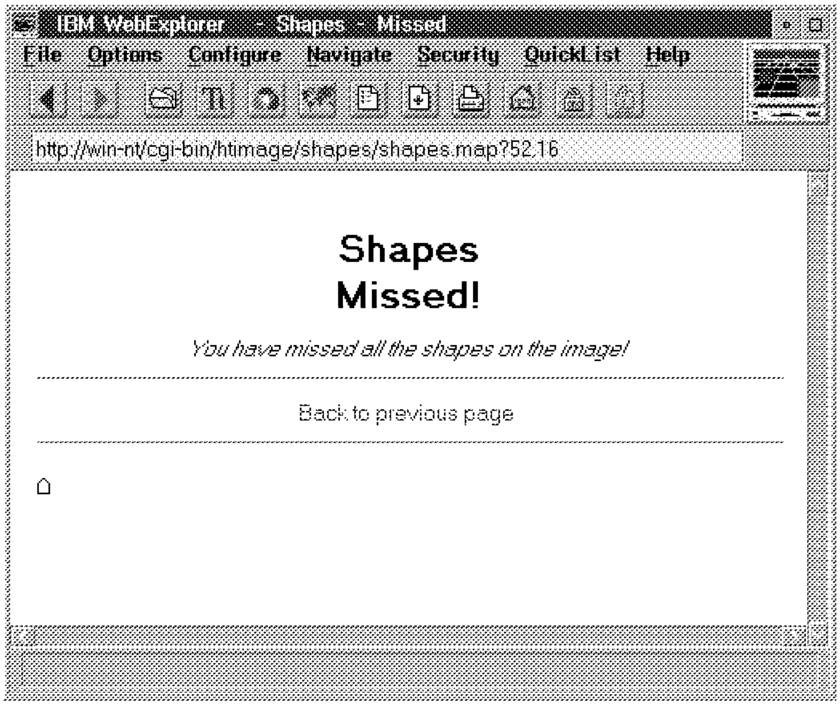
Shapes
Circle

You have selected a circle

Back to previous page

Figure 117. The Page Displayed if the User Clicked on the Circle

Furthermore, if users click on the image but not inside a shape, they will be notified as in Figure 118 on page 198.



IBM WebExplorer - Shapes - Missed
File Options Configure Navigate Security QuickList Help

http://win-nt/cgi-bin/htimage/shapes/shapes.map?52,16

Shapes

Missed!

You have missed all the shapes on the image!

Back to previous page

Figure 118. The Page Displayed if the User Clicked Outside the Defined Shapes

Whether the user′s browser accepts GIF graphics or not, clicking on the text link will always take the user to the relevant page.

5.9.1.1 The Associated HTML

The HTML source of our Shapes home page is as follows:

```
<HTML>
<HEAD>
<TITLE>Shapes</TITLE>
</HEAD>
<BODY BGCOLOR=FFFFFF>
<CENTER>
<H1>Shapes<BR>A Sample Clickable Image Map</H1>
<I>Select a shape</I>
<HR>
<A HREF=″/cgi-bin/htimage/shapes/shapes.map″>
<IMG ALT=″Please use the text links″ ISMAP SRC=″/shapes/shapes.gif″>
</A>
<P>
<A HREF=″/shapes/polygon.html″>Polygon</A> -
<A HREF=″/shapes/circle.html″>Circle</A> -
<A HREF=″/shapes/rectangle.html″>Rectangle</A>
<HR>
</CENTER>
</BODY>
</HTML>
```

Figure 119. The HTML Source of the Shapes Home Page

The essential htimage mechanism implemented in this page is included in the three following lines:

```
<A HREF=″/cgi-bin/htimage/shapes/shapes.map″>
<IMG ALT=″Please use the text links″ ISMAP SRC=″/shapes/shapes.gif″>
</A>
```

Let′s describe them, starting with the IMG tag:

- SRC specifies where the graphic is located.
- The ALT attribute (Please use the text links) is very important in this example, since the graphic is the key to our page. If the users′ browsers cannot display the graphic, the users will not be able to click on it. It is therefore very important to tell the users that the page offers a clickable image map, and to guide them to alternate ways of linking to further pages.

  We then simply added the lines:

  ```
  <A HREF=″/shapes/polygon.html″>Polygon</A> -
  <A HREF=″/shapes/circle.html″>Circle</A> -
  <A HREF=″/shapes/rectangle.html″>Rectangle</A>
  ```
- The ISMAP attribute indicates that the graphic is an image map.

Let us now examine the anchor tag:

```
<A HREF=″/cgi-bin/htimage/shapes/shapes.map″>...</A>
```

When the graphic is clicked upon, the browser will send the http://server/cgi-bin/htimage/shapes/shapes.map URL to the server. The directive in the HTTPD configuration file that indicates to the server that it should execute htimage/shapes/shapes.map is:

```
Exec /cgi-bin/* C:\WWW\CGI-Bin\*
```

We have developed this application on an Internet Connection Server for Windows NT, that includes htimage as a service.

The default HTTPD configuration file includes the following line:

```
Service /cgi-bin/htimage* C:\WWW\CGI-Bin\htimage:HTIMAGE*
```

This simply means that htimage should be called in combination with the map whose URL is located after cgi-bin/htimage. This URL is relative to the server root as base and in our case the map′s URL is /shapes/shapes.map.

Let us now take a look at the graphic and the corresponding map. The graphic is a 177 pixels wide and 59 pixels high GIF represented in Figure 115 on page 196. The map file is a text file whose lines have the following format:

```
region-identifier [region definition] URL
```

Our shapes.map map file is the following:

```
default                                  /shapes/missed.html
polygon   (29, 5) (53,29) (29,53) (5,29) /shapes/polygon.html
circle    (88,29) 24                     /shapes/circle.html
rectangle (123,5) (171,53)               /shapes/rectangle.html
```

Figure 120. The shapes.map Map File

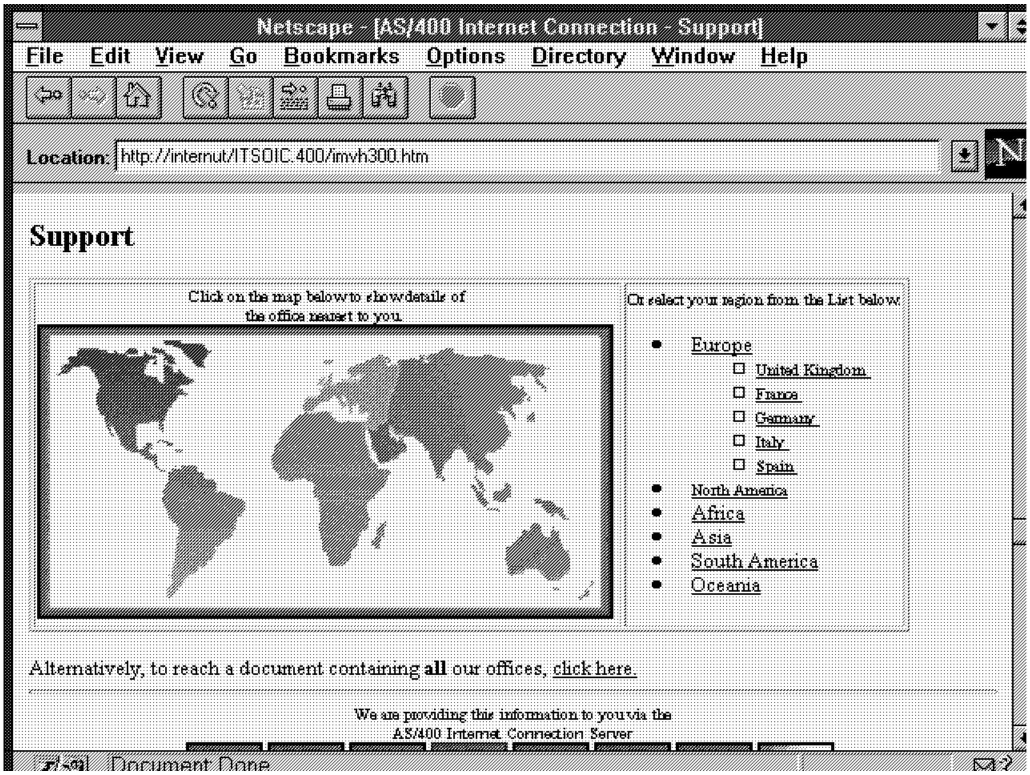
- The default region is the region that is not covered by any of the defined regions.
- A polygon is defined by its corners. Htimage allows polygons with up to 100 corners.
- A circle is defined by its center and its radius.
- A rectangle is defined by two diagonally opposite corners.

All coordinates are given relative to the upper left corner of the picture, which is (0,0).

5.9.2 Clickable Image Maps - World Map

Suppose you want people to be able to choose a car, check if a certain model is available, and make reservations from your Web server. What would be better than to show them a map of the country and allow them to click on any location and make the desired reservation? That is what a clickable image map can do. It can have many other uses also. For example, it helps users better understand a picture or diagram by allowing them to click on a component represented in the graphic to see an explanation. It also lets users move down to finer levels of detail on a map of a campus, building, or whatever you want.

Clickable image maps obviously work only with graphically oriented Web browsers. You should always try to provide alternate text-based methods of accessing the same information.



Location: http://internut/ITSOIC.400/imvh300.htm

Support

Click on the map below to show details of the office nearest to you.

Or select your region from the List below.

- Europe
  - □ United Kingdom
  - □ France
  - □ Germany
  - □ Italy
  - □ Spain
- North America
- Africa
- Asia
- South America
- Oceania

Alternatively, to reach a document containing **all** our offices, click here.

We are providing this information to you via the AS/400 Internet Connection Server

Figure 121. Clickable World Map. This figure shows a clickable map and an alternate text-based method of accessing the same information.

Image map requests are another form of program call or CGI. You can request image maps from either a server-provided image mapping program or a user-provided image mapping program. The easiest way to provide this level of function for your Web application is to take advantage of a server-provided mapping program such as the one found on the AS/400 system. This IBM written CGI-BIN application is called QTMHIMAG in the QTCP library.

If an image map request is made from a user-provided image program, it is handled as a standard program call request that must meet the CGI interface. We do not show an example of the user-provided image mapping, as this is more work than it is worth.

There are several steps you need to take when creating a server-provided image map:

- Plan a clickable image map.
- Map the hotspots in a map file.
- Reference your clickable image map in HTML.
- Configure the HTTP server.

Let′s take a look at each of these steps in detail.

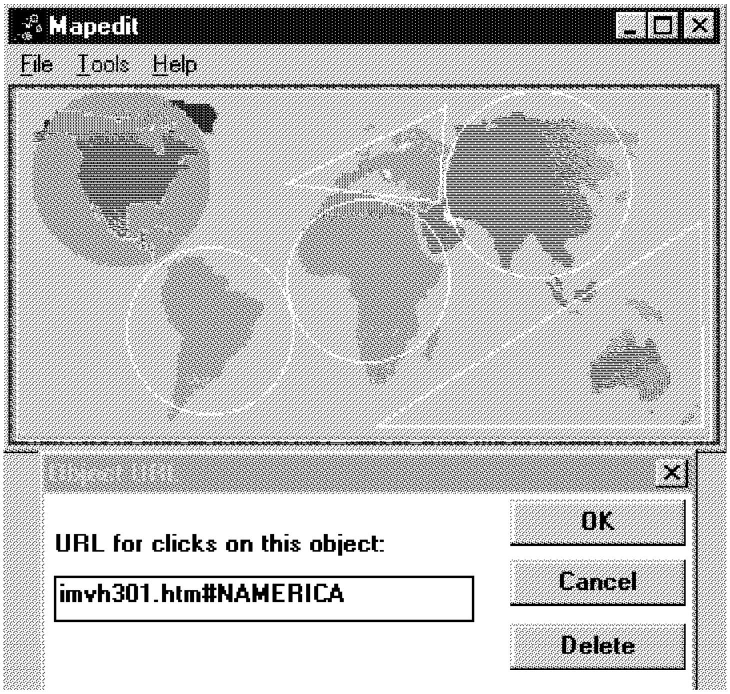
5.9.2.1 Planning the Clickable Image Map

First, determine the image you are going to use. As for any inline images, the graphics need to be in GIF format for the widest portability. Then define where you are going to map the clickable areas or hotspots.

Depending on the size and make up of the image you are mapping, you can use polygons, rectangles, or circles to identify hotspots.

5.9.2.2 Mapping the Hotspots in a Map File

The next step is to develop an image map file such as the one seen previously. You need to create a separate image map file for each clickable image map on your Web server. Each line in an image map file represents a hotspot by defining an area within the graphic and the corresponding URL to be returned if someone clicks on that area with their Web browser. By using a program on your PC such as MAPEDIT, hotspots on the image are formatted in CERN by: Shape, Coordinate-1, Coordinate-2 ... Coordinate-n, URL.



URL for clicks on this object:

imvh301.htm#NAMERICA

Figure 122. Defining Hotspots and Associated URLs. This figure shows North America in a circle shape mapped to its associated URL.

This map is made up of circles and triangular shaped polygons. A rectangular shape was mapped last for the entire image as a default area.

Shapes are defined by the following:

- A circle - For a circle (a pair of coordinates for the center and a single value for the radius).
- A polygon - For a polygon (The number of coordinates depend on the number of sides defined for the polygon.)
- A rectangle - For a rectangle (with two coordinate pairs: upper-left and lower-right).

Coordinates are x,y pairs counting in pixels from the upper left-hand corner of the image. The URL can be either an actual URL pointing to any resource on the World Wide Web, or it can point to another image or document on your server.

```
       *************** Beginning of data ********************************
001.00 circle (101,123) 42 /ITSOIC.400/imvh301.htm#SAMERICA
002.00 circle (56,46) 45 /ITSOIC.400/imvh301.htm#NAMERICA
003.00 circle (180,98) 41 /ITSOIC.400/imvh301.htm#AFRICA
004.00 circle (265,50) 47 /ITSOIC.400/imvh301.htm#ASIA
005.00 poly (218,10) (215,58) (138,49) /ITSOIC.400/imvh301.htm#EUROPE
006.00 poly (346,68) (184,170) (347,170) /ITSOIC.400/imvh301.htm#OCEANIA
007.00 rect (4,3) (351,177) /ITSOIC.400/imvh301.htm#OCEANIA
008.00 default /ITSOIC.400/imvh301.htm#OCEANIA
       ****************** End of data **********************************
```

Figure 123. Image Map Coordinates. This example shows circular, polygon, and rectangular shapes and their defined URLs.

The server takes the coordinates from a user′s mouse click and steps through the map file to determine if the click is within any hotspots. As soon as the first match is found, the corresponding URL is used to redirect a document to the user′s Web browser. If no matches are found, a default URL (that you specify in the image map file) is returned.

Notice that the last line in the file is a default statement. An image map file must *always* contain a default statement. This is required so that if a user clicks on a part of the image that has not been mapped, the user is sent to a defined default area.

It is normal to mix and match circles, polygons, and rectangles in the same map file. Although you should try to minimize overlapping hotspots in the image map, if there are any, the first match is the one used.

5.9.2.3 Referencing Your Clickable Image Map in HTML

The final step in creating an image map is to tell the Web browser accessing the HTML document that the inline image it displays is a clickable image map. In practice, what you do is create a link where this image is the link trigger and the URL invokes the image map script and passes it the map name. For example:

```
<A HREF=″http://servername/imagemap.pgm/imagemap.file″>
<IMG SRC=″image.gif″> ISMAP></A>
```

5.9.2.4 Configuring Your Clickable Image Map in the HTTP Server

In order to configure the HTTP server on the AS/400 system, we reference the previous example. Our imagemap.pgm is qtcp/qtmhimag and our image maps are found in the imagemap.file. Here is how our URL looks using the AS/400 system to serve image maps:

```
<A HREF=″http://servername/qsys.lib/qtcp.lib/qtmhimag.pgm 1
/qsys.lib/itsoic400.lib/imagemap.file/mapk.mbr″> 2
<IMG SRC=″imvg300.gif″ ISMAP></A> 3
```

1 This section of the URL represents the server name and also where the image mapping program resides on the AS/400 system.

2 This section tells the server where the actual image map file is located on the AS/400 system.

3 This section shows the source of the image or GIF file that is to be mapped and ISMAP tells the server this is an image map.

We can shorten and simplify the URL by taking advantage of the HTTP mapping rules. For example, we use the following URL to do the same thing:

```
<A HREF="/cgi-bin/imagemap/map1.mbr">
<IMG SRC="world.gif" ISMAP></A>
```

Where you see /cgi-bin/imagemap, we used the HTTP Map configuration statement to reference the longer URL. The following lines must be added to the HTTP configuration to make it work:

```
Map /cgi-bin/imagemap/*  /qsys.lib/qtcp.lib/qtmhimag.pgm
/qsys.lib/itsoic400.lib/imagemap.file/*

Pass /qsys.lib/itsoic400.lib/imagemap.file/*

Exec /qsys.lib/qtcp.lib/*
```

In order to access the HTTP configuration file, type WRKHTTPCFG on an AS/400 command line. Add the Map, Pass, and Exec statements noted previously. By using the Map directive in our HTTP configuration, we are able to use shorter URLs in our HTML documents. The Pass statement is there in order to pass the image map files from the server. There must also be an Exec statement in order to call the qtmhimag program from the qtcp library.

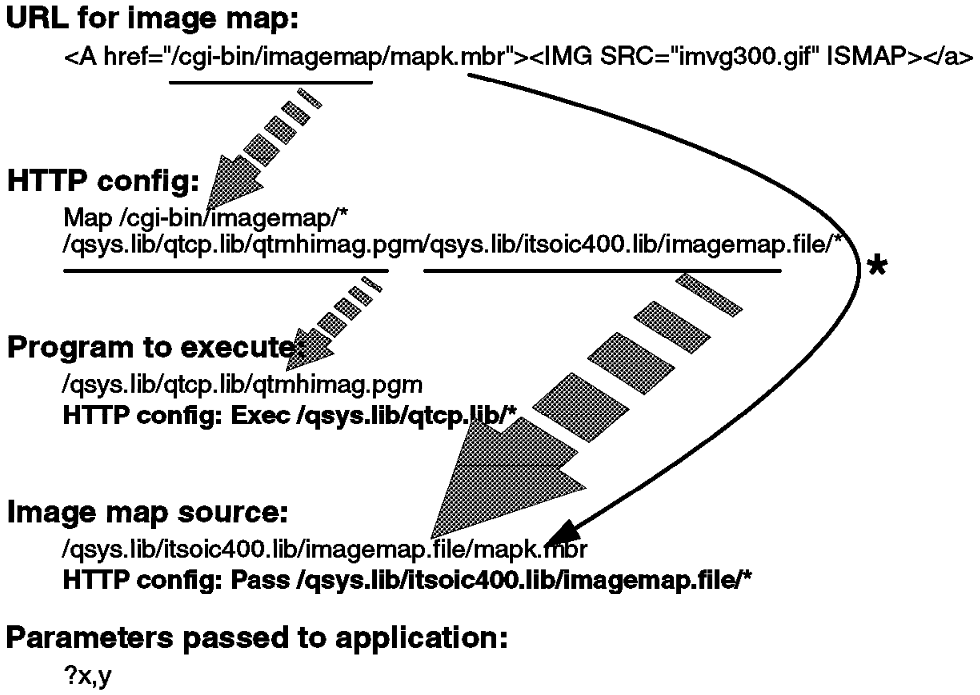

URL for image map:
<A href="/cgi-bin/imagemap/mapk.mbr"><IMG SRC="imvg300.gif" ISMAP></a>
HTTP config:
Map /cgi-bin/imagemap/*
/qsys.lib/qtcp.lib/qtmhimag.pgm/qsys.lib/itsoic400.lib/imagemap.file/*
*
Program to execute:
/qsys.lib/qtcp.lib/qtmhimag.pgm
HTTP config: Exec /qsys.lib/qtcp.lib/*
Image map source:
/qsys.lib/itsoic400.lib/imagemap.file/mapk.mbr
HTTP config: Pass /qsys.lib/itsoic400.lib/imagemap.file/*
Parameters passed to application:
?x,y

Figure 124. Mapping of Image Map URL to HTTP Configuration

Note

More information on image mapping and image map requests on the AS/400 system is available in the *TCP/IP Configuration and Reference, Version 3*, SC41-3420.

5.9.3 Animated Pages - Love All, Serve All

HTML gives you the possibility to serve hypertext pages to people using various hardware, operating systems and browsers. You should therefore take care that most, if not all, users can view all the information located on the pages you design.

Here are some typical examples of what you should not do, and why:

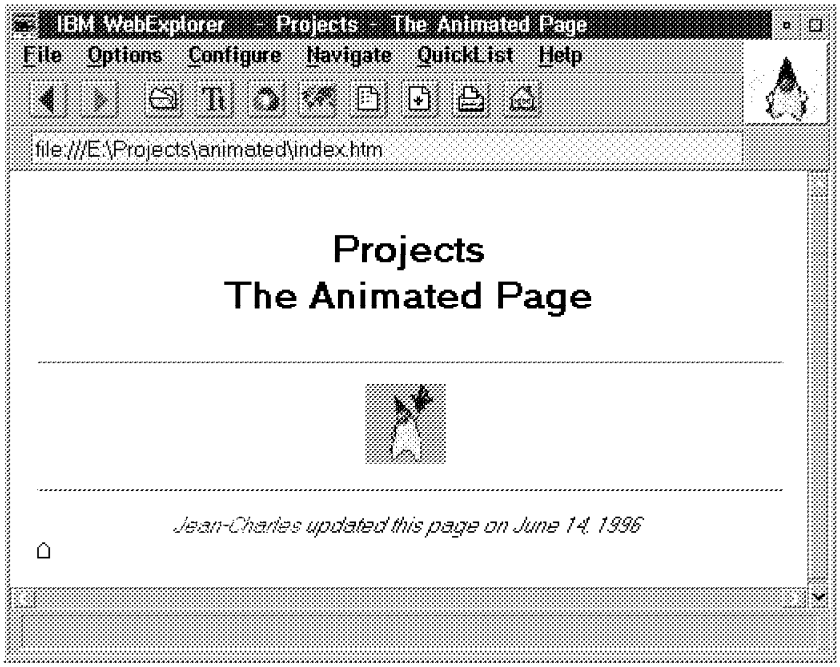
- Do not place all the links to all your pages only in a clickable image map or Java applet in your home page. People linking to your page with a browser that does not support graphics or Java will only see a crippled home page and may not be able to link further.
- Do not use browser-specific tags or features extensively. What if someone can not or does not wish to use this particular browser?
- Do not locate critical information in tables only. Tables are a convenient way of organizing your pages, and almost every browser now supports the TABLE tag. Some however do not; think about them.

If you have to or want to ignore the advice above, you are recommended to always serve alternate pages in standard HTML. Minimize the version of HTML required to view these alternate pages as much as possible, so as to comply with most browsers.

In this section we introduce a little example of how you can include an animation in a page so that most browsers will be able to display it. Depending on their browser, the users will see:

- A Java applet
- An animated image
- A static image
- Text

Let′s take a look at some different options.

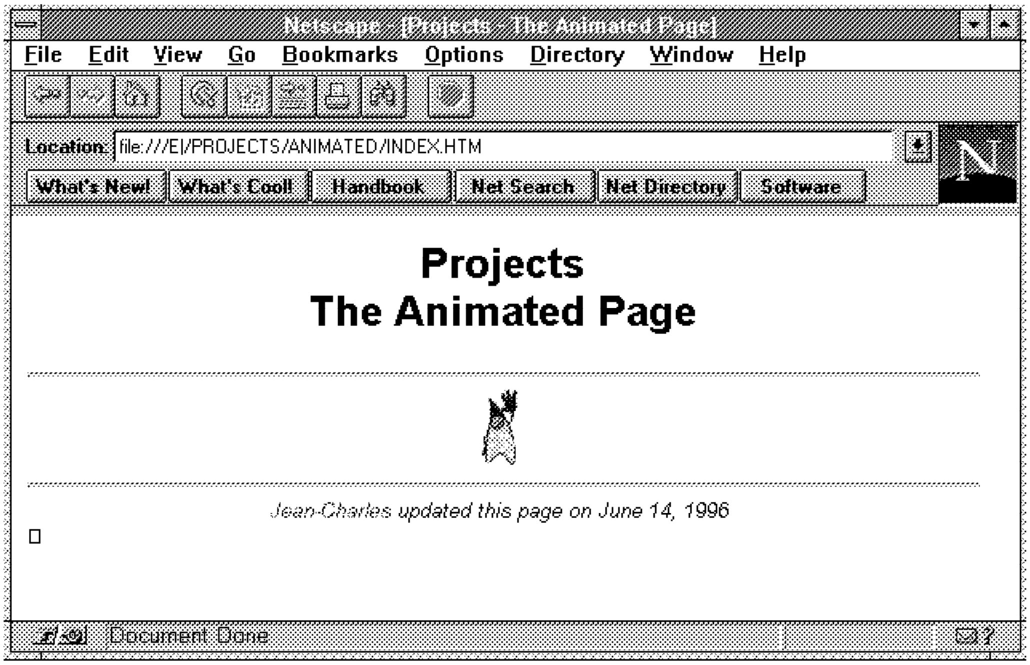


file:///E:\Projects\animated\index.htm

Projects
The Animated Page

Jean-Charles updated this page on June 14, 1996

Figure 125. A Java Applet with IBM WebExplorer Java Technology Demo



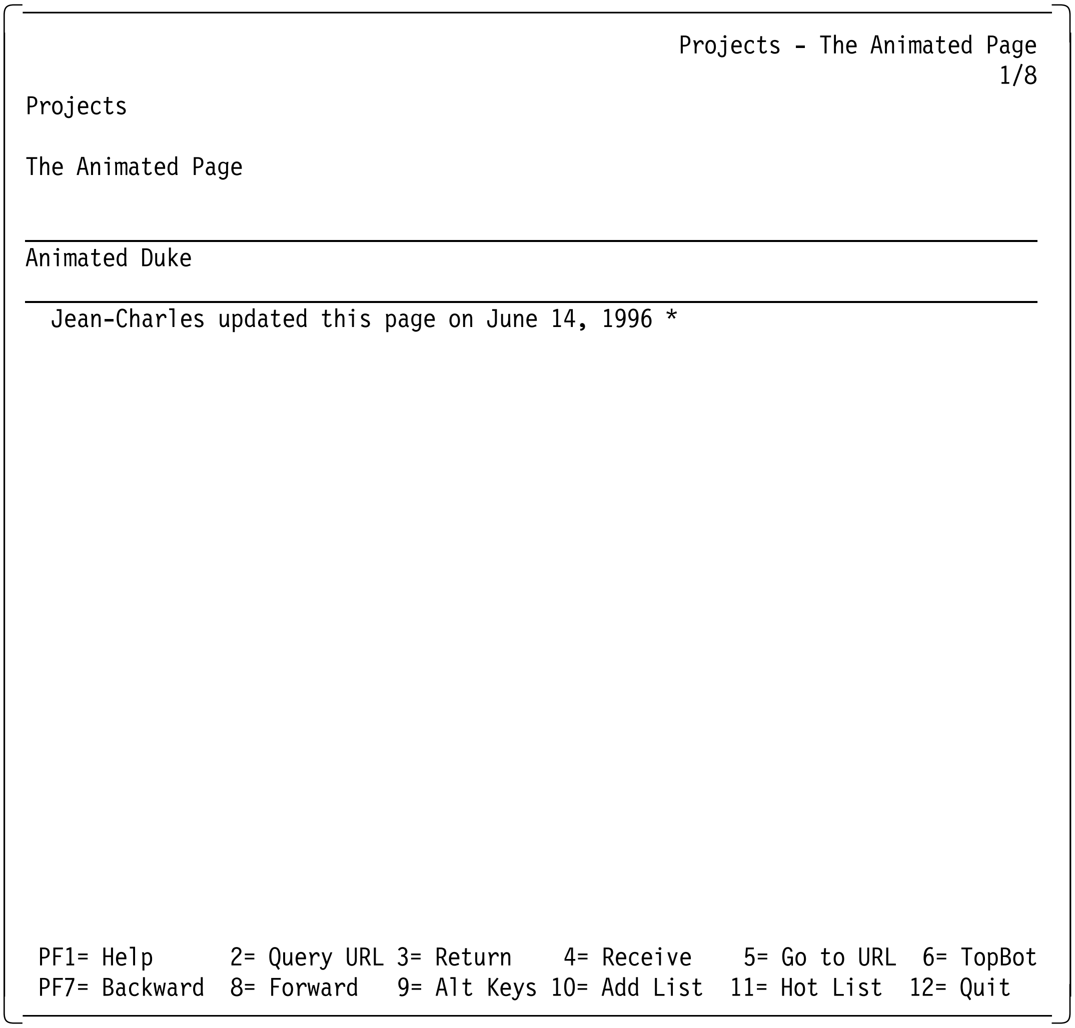
Location: file:///E|/PROJECTS/ANIMATED/INDEX.HTM

Projects
The Animated Page

Jean-Charles updated this page on June 14, 1996

Document Done

Figure 126. An Animated Image with Netscape Navigator 2.0 for Windows 3.1



```
                                           Projects - The Animated Page
                                                                    1/8
Projects

The Animated Page

Animated Duke

  Jean-Charles updated this page on June 14, 1996 *















 PF1= Help      2= Query URL 3= Return    4= Receive    5= Go to URL  6= TopBot
 PF7= Backward  8= Forward   9= Alt Keys 10= Add List  11= Hot List  12= Quit
```

Figure 127. A Line of Text with Charlotte Browser for VM

5.9.3.1 The Associated HTML

The HTML source of our page is listed below:

```
<HTML>
<HEAD>
<TITLE>Projects - The Animated Page</TITLE>
</HEAD>
<BODY BGCOLOR=FFFFFF>
<ANIMATE>
<FRAME SRC=″duke01.gif″><FRAME SRC=″duke01.gif″>
<FRAME SRC=″duke02.gif″><FRAME SRC=″duke02.gif″>
<FRAME SRC=″duke03.gif″><FRAME SRC=″duke03.gif″>
<FRAME SRC=″duke04.gif″><FRAME SRC=″duke04.gif″>
<FRAME SRC=″duke05.gif″><FRAME SRC=″duke05.gif″>
<FRAME SRC=″duke06.gif″><FRAME SRC=″duke06.gif″>
<FRAME SRC=″duke07.gif″><FRAME SRC=″duke07.gif″>
<FRAME SRC=″duke08.gif″><FRAME SRC=″duke08.gif″>
<FRAME SRC=″duke09.gif″><FRAME SRC=″duke09.gif″>
<FRAME SRC=″duke10.gif″><FRAME SRC=″duke10.gif″>
</ANIMATE>
<CENTER>
<H1>Projects<BR>The Animated Page</H1>
<HR>
<APPLET CODE=″Animator.class″ WIDTH=50 HEIGHT=50>
<param name=imagesource value=″.″>
<param name=endimage value=10>
<param name=pause value=200>
<IMG ALT=″Animated Duke″ SRC=″T.gif″>
</APPLET>
<HR>
<I><A HREF=″mailto:andlauer@be.ibm.com″>Jean-Charles</A> updated this page on
June 14, 1996</I>
</CENTER>
</BODY>
</HTML>
```

Figure 128. The HTML of the Animated Page

The ANIMATE Tag: As you can see we included the ANIMATE tag, which is only supported by the IBM WebExplorer. This tag allows you to change the browsers default animation by adding a personalized one that will remain until the browser is stopped or another animation is loaded.

The syntax of the ANIMATE tag is simply:

```
<ANIMATE>
<FRAME SRC=url>
</ANIMATE>
```

The graphics referenced by the FRAME tag should be 50 by 50 pixels JPEGs or noninterlaced GIFs without transparency.

Using the ANIMATE tag is rather safe, since all the requirements are handled by HTML tags. Browsers that do not support them should thus disregard them.

The APPLET Tag: The APPLET tag imbeds a Java applet into the page, when it is viewed by a Java-enabled browser. Here we simply included the Animator applet, distributed by Sun as a sample Java application.

As was the case for the ANIMATE tag, all the information required by the APPLET tag is included in HTML tags. If a browser does not support them, it does not interpret them.

However, the APPLET tag also allows you to set an alternate HTML source between the begin and end tags. In our case it is:

```
<IMG ALT="Animated Duke" SRC="T.gif">
```

This alternate source is interpreted only if the browser does not support Java (the APPLET tag). It is disregarded if the browser supports Java (the APPLET tag).

Animated GIFs: The GIF graphical image format allows you to code sequences of images and thus animations. Tools to create such animated graphics are now available on most platforms.

For example, if the viewer used to display such a GIF does not support animated GIFs then it will only display the first image as does the IBM WebExplorer. Therefore make sure that the first image is meaningful.

Our page thus includes an alternate animation to the Animator applet for the browsers that support animated GIFs. This alternative is a simple graphic for the browser that accepts GIFs, but not the animated ones.

Finally, the Animated Duke text is provided in all other cases.

The ALT Attribute of the IMG Tag: Never forget to set the ALT attribute of your IMG tags so that non-graphical browsers are taken care of.

5.9.4 Server-Side Includes - Gods of Greek Mythology

The Internet Connection servers all feature server-side includes. Server-side includes allow you to add dynamic data into your HTML pages. Typical examples of such data are the current time, the date a document was updated, or the size of a file.

Furthermore, server-side includes allow to you include any other document into your HTML pages, as text or other HTML files. This feature is very useful to rationally organize your server, as our scenario demonstrates.

We planned to set up pages dedicated to the gods of Greek mythology on our server. Our first concern was to offer the most detailed information as soon as possible.

This data was available in the form of 17 text files comprising one summary and 16 descriptions, one of which is listed in Figure 129 on page 210.

```
HEBE, the goddess of youth, was the daughter of Zeus and
Hera.  Hebe served for a long time as cupbearer to the gods,
serving them their nectar and ambrosia.  She was replaced in
this office by the Trojan prince Ganymede.
According to one story, she resigned as cupbearer to the
gods upon her marriage to the hero Hercules, who had just
been deified.  In another, she was dismissed from her position
because of a fall she suffered while in attendance on the
gods.
```

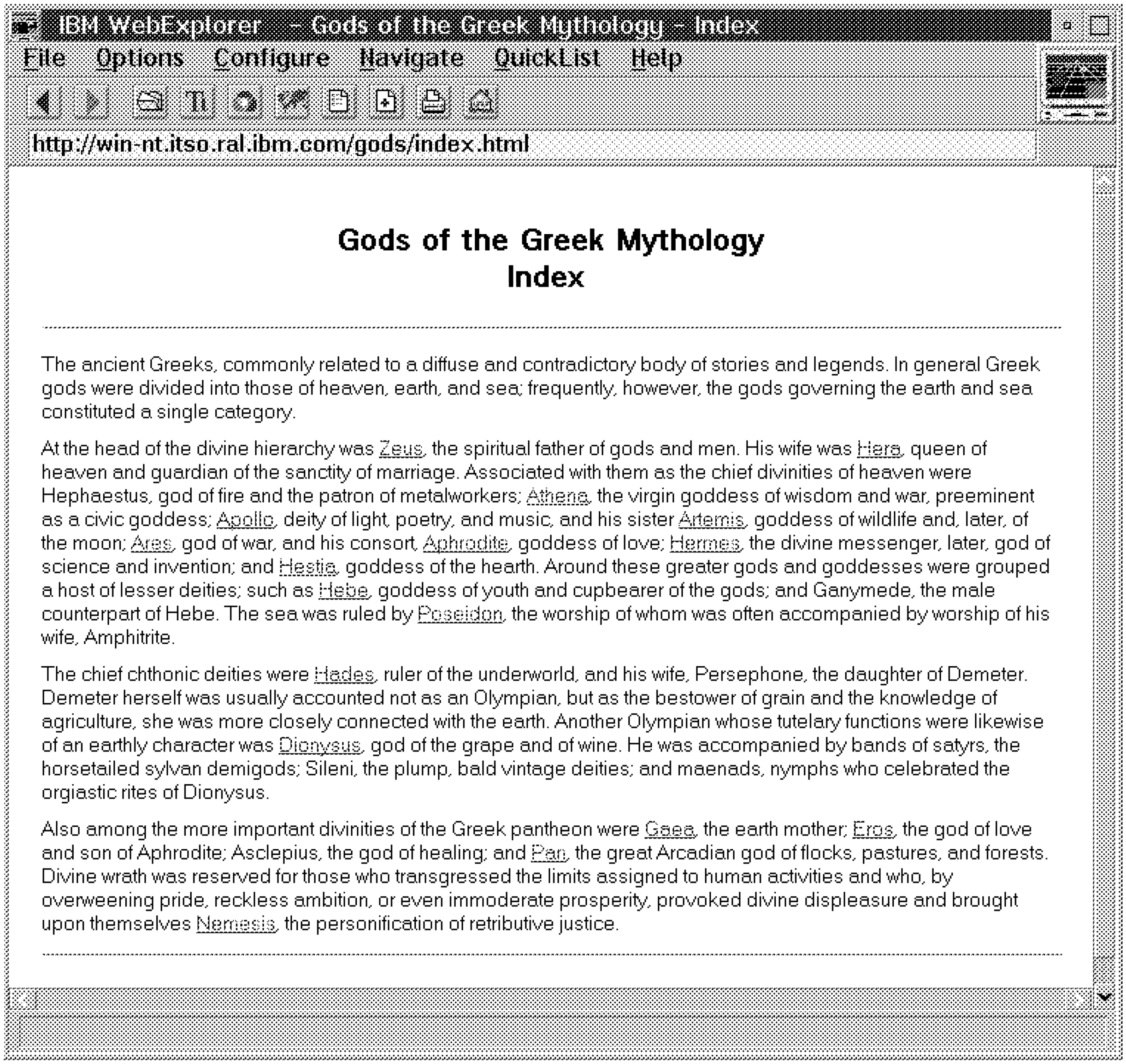
Figure 129. Hebe, A Text File

This organization corresponded to our needs. We therefore simply tagged the text files to transform them into basic HTML files, as shown in Figure 130, and installed them on our server.

```
<HTML>
<HEAD>
<TITLE>Gods of the Greek Mythology - Hebe</TITLE>
</HEAD>
<BODY BGCOLOR=FFFFFF>
<CENTER>
<H1>Gods of the Greek Mythology<BR>Hebe</H1>
<HR>
</CENTER>
HEBE, the goddess of youth, was the daughter of Zeus and
Hera.  Hebe served for a long time as cupbearer to the gods,
serving them their nectar and ambrosia.  She was replaced in
this office by the Trojan prince Ganymede.
<P>According to one story, she resigned as cupbearer to the
gods upon her marriage to the hero Hercules, who had just
been deified. In another, she was dismissed from her position
because of a fall she suffered while in attendance on the
gods.
<CENTER>
<HR>
</CENTER>
</BODY>
</HTML>
```

Figure 130. Hebe, A Basic HTML File

We used the approach described above to a create an index page. This page, as viewed with an IBM WebExplorer, is shown in Figure 131 on page 211.


IBM WebExplorer - Gods of the Greek Mythology - Index
File Options Configure Navigate QuickList Help

http://win-nt.itso.ral.ibm.com/gods/index.html

Gods of the Greek Mythology
Index

The ancient Greeks, commonly related to a diffuse and contradictory body of stories and legends. In general Greek gods were divided into those of heaven, earth, and sea; frequently, however, the gods governing the earth and sea constituted a single category.

At the head of the divine hierarchy was Zeus, the spiritual father of gods and men. His wife was Hera, queen of heaven and guardian of the sanctity of marriage. Associated with them as the chief divinities of heaven were Hephaestus, god of fire and the patron of metalworkers; Athena, the virgin goddess of wisdom and war, preeminent as a civic goddess; Apollo, deity of light, poetry, and music, and his sister Artemis, goddess of wildlife and, later, of the moon; Ares, god of war, and his consort, Aphrodite, goddess of love; Hermes, the divine messenger, later, god of science and invention; and Hestia, goddess of the hearth. Around these greater gods and goddesses were grouped a host of lesser deities; such as Hebe, goddess of youth and cupbearer of the gods; and Ganymede, the male counterpart of Hebe. The sea was ruled by Poseidon, the worship of whom was often accompanied by worship of his wife, Amphitrite.

The chief chthonic deities were Hades, ruler of the underworld, and his wife, Persephone, the daughter of Demeter. Demeter herself was usually accounted not as an Olympian, but as the bestower of grain and the knowledge of agriculture, she was more closely connected with the earth. Another Olympian whose tutelary functions were likewise of an earthly character was Dionysus, god of the grape and of wine. He was accompanied by bands of satyrs, the horsetailed sylvan demigods; Sileni, the plump, bald vintage deities; and maenads, nymphs who celebrated the orgiastic rites of Dionysus.

Also among the more important divinities of the Greek pantheon were Gaea, the earth mother; Eros, the god of love and son of Aphrodite; Asclepius, the god of healing; and Pan, the great Arcadian god of flocks, pastures, and forests. Divine wrath was reserved for those who transgressed the limits assigned to human activities and who, by overweening pride, reckless ambition, or even immoderate prosperity, provoked divine displeasure and brought upon themselves Nemesis, the personification of retributive justice.

Figure 131. Index, A Basic HTML File Viewed with an IBM WebExplorer

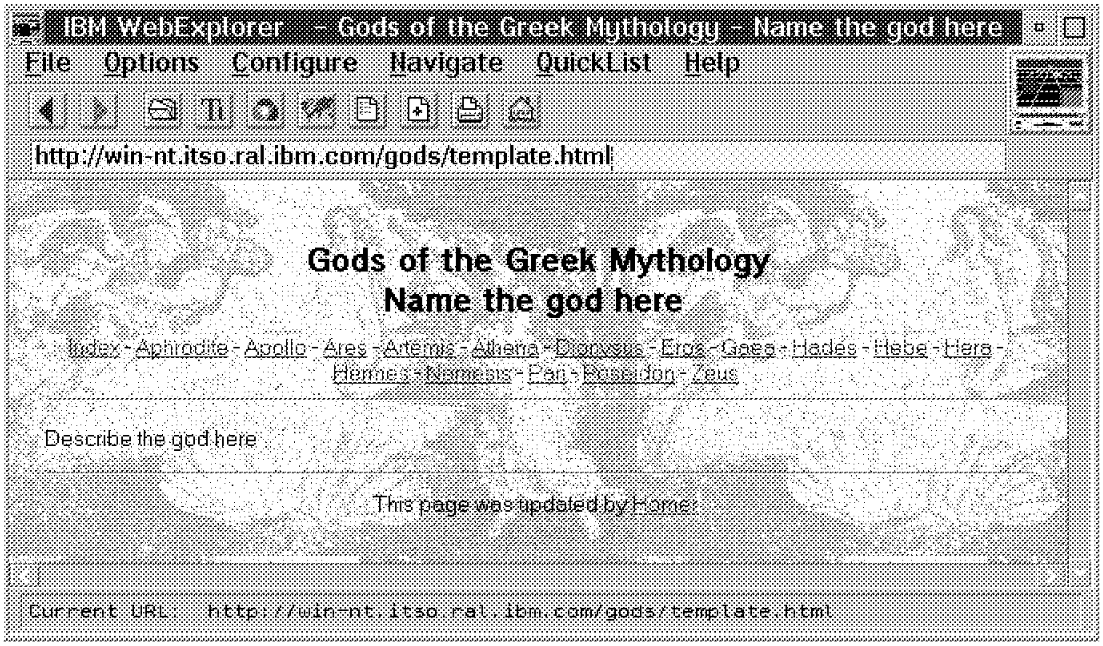
Our Gods of Greek Mythology pages had only been on the Web for a short while when users already complained that:

- It was impossible to complain about these pages because they did not link to an e-mail address.
- Going from one god to another required returning to the index.
- The pages were far too sober for such an attractive subject.

We therefore designed a page template that responded to these demands. The source of this template is listed in Figure 132 on page 212. Figure 133 on page 213 shows how this template looks when it is viewed with an IBM WebExplorer.

```
<HTML>
<HEAD>
<TITLE>Gods of the Greek Mythology - Name the god here</TITLE>
</HEAD>
<BODY BACKGROUND=”venus.jpg”>
<CENTER>
<H1>
Gods of the Greek Mythology<BR>
Name the god here
</H1>
<A HREF=”index.html”>Index</A> -
<A HREF=”aphrodite.html”>Aphrodite</A> -
<A HREF=”apollo.html”>Apollo</A> -
<A HREF=”ares.html”>Ares</A> -
<A HREF=”artemis.html”>Artemis</A> -
<A HREF=”athena.html”>Athena</A> -
<A HREF=”dionysus.html”>Dionysus</A> -
<A HREF=”eros.html”>Eros</A> -
<A HREF=”gaea.html”>Gaea</A> -
<A HREF=”hades.html”>Hades</A> -
<A HREF=”hebe.html”>Hebe</A> -
<A HREF=”hera.html”>Hera</A> -
<A HREF=”hermes.html”>Hermes</A> -
<A HREF=”nemesis.html”>Nemesis</A> -
<A HREF=”pan.html”>Pan</A> -
<A HREF=”poseidon.html”>Poseidon</A> -
<A HREF=”zeus.html”>Zeus</A>
<HR>
</CENTER>
Describe the god here
<CENTER>
<HR>
This page was updated by <A HREF=”mailto:Homer@Simpsonian.org”>Homer</A>
</CENTER>
</BODY>
</HTML>
```

Figure 132. Page Template - HTML Source



IBM WebExplorer - Gods of the Greek Mythology - Name the god here
File
Options
Configure
Navigate
QuickList
Help

http://win-nt.itso.ral.ibm.com/gods/template.html

Gods of the Greek Mythology
Name the god here

Index - Aphrodite - Apollo - Ares - Artemis - Athena - Dionysus - Eros - Gaea - Hades - Hebe - Hera - Hermes - Nemesis - Pan - Poseidon - Zeus

Describe the god here

This page was updated by Homer

Current URL http://win-nt.itso.ral.ibm.com/gods/template.html

Figure 133. The Page Template, Viewed with an IBM WebExplorer

We now had to solve the following problem: how could we easily apply this template to all the existing descriptions in such a way that any modification to the template would also impact the descriptions?

The answer was clear: server-side includes. The advantage of server-side includes is that they allow you to include files in HTML documents.

Our first step then was to activate the server-side includes; we did this by adding the following AddType directive to our HTTPD configuration file:

```
AddType .shtml text/x-ssi-html 8bit 1.0
```

This directive enables the interpretation of the server-side includes in documents whose extension is shtml. As we see later, server-side includes are placed inside HTML comments, and are therefore not interpreted in documents whose extension is not specified in such an AddType directive. For example, this is the case for files whose extension is html where the default HTTPD configuration file is in use.

Then we modified the page template, as shown in Figure 134 on page 214, and saved it as template.shtml.

```
<HTML>
<HEAD>
<TITLE>Gods of the Greek Mythology - <!--#echo var=NAME --></TITLE>
</HEAD>
<BODY BACKGROUND=″venus.jpg″>
<CENTER>
<H1>
Gods of the Greek Mythology<BR>
<!--#echo var=NAME -->
</H1>
<A HREF=″index.shtml″>Index</A> -
<A HREF=″aphrodite.shtml″>Aphrodite</A> -
<A HREF=″apollo.shtml″>Apollo</A> -
<A HREF=″ares.shtml″>Ares</A> -
<A HREF=″artemis.shtml″>Artemis</A> -
<A HREF=″athena.shtml″>Athena</A> -
<A HREF=″dionysus.shtml″>Dionysus</A> -
<A HREF=″eros.shtml″>Eros</A> -
<A HREF=″gaea.shtml″>Gaea</A> -
<A HREF=″hades.shtml″>Hades</A> -
<A HREF=″hebe.shtml″>Hebe</A> -
<A HREF=″hera.shtml″>Hera</A> -
<A HREF=″hermes.shtml″>Hermes</A> -
<A HREF=″nemesis.shtml″>Nemesis</A> -
<A HREF=″pan.shtml″>Pan</A> -
<A HREF=″poseidon.shtml″>Poseidon</A> -
<A HREF=″zeus.shtml″>Zeus</A>
<HR>
</CENTER>
<!--#include file=″&NAME;\.html″ -->
<CENTER>
<HR>
This page was updated on <!--#flastmod file=″&NAME;\.html″ -->
by <A HREF=″mailto:Homer@Simpsonian.org″>Homer</A>
</CENTER>
</BODY>
</HTML>
```

Figure 134. template.shtml, the Modified Template

Because the extension of our template is shtml, the server-side includes it contains will be interpreted.

The syntax of a server-side include is the following:

```
<!--#command tag=value ... -->
```

Note that, as we mentioned earlier, it is included in an HTML comment.

Our template contains three different server-side includes:

- `<!--#echo var=NAME -->`

  The echo server-side include displays the value of a variable. In our case, it is the variable NAME whose value is the name of the god. We describe how this variable is set later in this section.

- `<!--#include file=″&NAME;\.html″ -->`

  The include server-side include includes a file in the output. We want to include the description of a given god, which is located in a file whose name is simply the god′s name, and whose extension is html; zeus.html for example. ″&NAME;\.html″ creates this string. &NAME; echoes the NAME variable that contains the god′s name. \. is a period. Please read the Internet Connection server′s manual for the complete list of escape characters.

- `<!--#flastmod file=″&NAME;\.html″ -->`

  The flastmod server-side include displays the last time a document was modified. Here we display the last time the description of the god was modified.

Now we have to set the NAME variable, which is what users do when they browse a page such as the one listed below:

```
<!--#global var=NAME value=hebe -->
<!--#include file=″template\.shmtl″ -->
```

Figure 135. Invoking Hebe with Hebe.shtml

Here we set the global variable NAME to hebe with the global server-side include. NAME will keep this value throughout all the included documents. If we had used the set server-side include, NAME would have been set to hebe in the current document only.

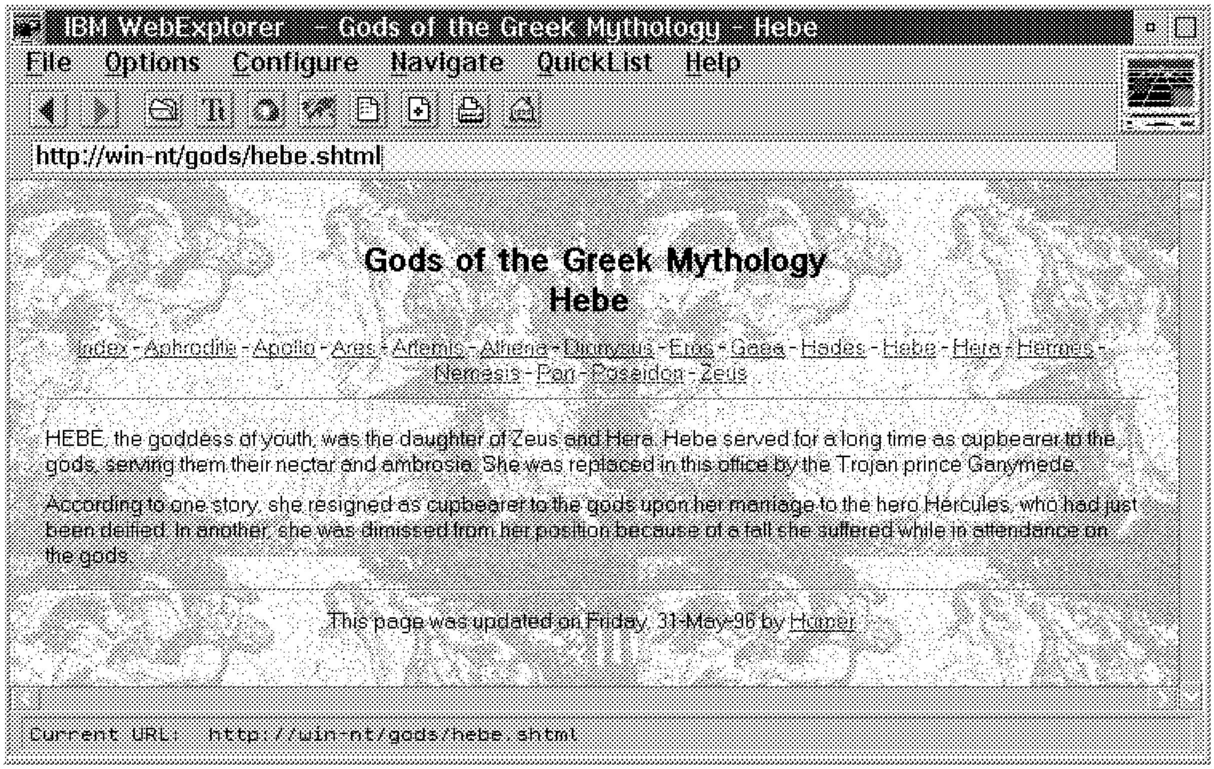
All we have to do now is modify the description of the gods so that they fit in the new template. We simply stripped the HTML, HEAD, BODY and H1 tags, to obtain something like we see in Figure 136.

```
HEBE, the goddess of youth, was the daughter of Zeus and
Hera.  Hebe served for a long time as cupbearer to the gods,
serving them their nectar and ambrosia.  She was replaced in
this office by the Trojan prince Ganymede.
<P>According to one story, she resigned as cupbearer to the
gods upon her marriage to the hero Hercules, who had just
been deified. In another, she was dismissed from her position
because of a fall she suffered while in attendance on the
gods.
```

Figure 136. Hebe, A Modified HTML File

We have now achieved our goal: we separated the contents of our pages from their layout, and it has become an easy matter to spread any change in the layout to all of our pages.

Here is how this looks when it is browsed.



http://win-nt/gods/hebe.shtml

Gods of the Greek Mythology
Hebe

Index - Aphrodite - Apollo - Ares - Artemis - Athena - Dionysus - Eros - Gaea - Hades - Hebe - Hera - Hermes - Nemesis - Pan - Poseidon - Zeus

HEBE, the goddess of youth, was the daughter of Zeus and Hera. Hebe served for a long time as cupbearer to the gods, serving them their nectar and ambrosia. She was replaced in this office by the Trojan prince Ganymede.

According to one story, she resigned as cupbearer to the gods upon her marriage to the hero Hercules, who had just been deified. In another, she was dimissed from her position because of a fall she suffered while in attendance on the gods.

This page was updated on Friday, 31-May-96 by Homer

Current URL: http://win-nt/gods/hebe.shtml

Figure 137. Hebe, The Definitive Result

5.9.4.1 Conclusion

As this example shows, the server-side includes are very useful to organize the data that is hosted on your server rationally. Server-side includes however have a major drawback: they use a lot of the server′s resources. In fact, the server parses the files containing server-side includes before it sends them to the browser.

You should therefore consider alternatives to server-side includes if you intend to set up heavy duty, large scale or long term Internet servers. At the time we were writing this book, the trend was still to use CGI scripts, because of the large range of possibilities that they permit. But it is certain that the use of server APIs will dominate all the other programming techniques, because they combine flexibility and performance. For more about the use of CGIs and APIs refer to Chapter 6, "Establishing an Active Presence on the Internet" on page 217.

Chapter 6. Establishing an Active Presence on the Internet

In Chapter 5, “Establishing a Presence on the Internet” on page 183, we discussed how to develop a Web server that can offer static documents and links to other pages or servers. You can develop a more active presence on the Internet by using CGI scripts and API extensions to add function to your server.

In case you are not familiar with the CGI and API acronyms, here is a reminder:

- CGI refers to Common Gateway Interface
- API refers to Application Program Interface

CGI programs are often called CGI scripts, but as you see in the examples below, you can develop your own CGI programs in many languages, not only in scripting languages. The reason they are referred to as scripts is historical in that they were originally developed in sh, bash, and perl on UNIX platforms.

Referring to an API out of context has no real meaning. You will find APIs in all domains including network, Windows environments, etc. In our context, API is a generic term used to refer to the available interfaces in the Web environment. This covers in our case IBM ICAPI, but also Netscape′s NSAPI and Microsoft′s ISAPI.

In this chapter we generically refer to CGI applications and API applications; only where necessary do we refer to a particular API, ICAPI for example. Whatever term you use to refer to them, CGIs and APIs should be considered as extensions to your server to increase its capabilities.

The following examples show you how CGIs and APIs can animate your Web server. The intent is not to teach you any particular language but to explain to you what kind of extension you can expect from scripts and, using basic examples, how you should implement them on your Internet Connection server.

None of these examples requires an in-depth knowledge of any particular language. However, in order to show the large choice of tools available to you, each of the CGI samples has been developed in a different language: Perl, Korn Shell, C, REXX and batch.

6.1 CGI Scripts

There are many things we can do using CGI scripts:

- Provide help in redirecting to other URLs
- Build interactive images
- Build on-the-fly images
- Process forms

6.1.1 Selecting Your Programming Language

The principle of Common Gateway Interface is that you should be able to use any programming language. You choose the one you will be using according to:

- The platform on which your server is running
- The task your application has to perform
- Your programming skills

6.1.1.1 Your Server Platform

The operating system on which your server is running is probably the decisive factor in your choice of a programming language.

Not all programming languages are available on every platform. For example, there is no port of Visual Basic for AIX, OS/2 or MVS. This fact is not only essential when you plan to develop intranet or Internet applications, but also if you consider migrating your server to another platform. Imagine you have set up a server that has become so popular that it has outgrown the resources of the Windows NT host on which you have installed it. Because the Internet Connection Servers are ported from the same code, you can easily migrate your server to a more powerful AIX or MVS system, unless you programmed your applications in a platform-specific programming language, such as Visual Basic.

Furthermore, some languages are more suited to an operating system than others. This is typically the case of C for AIX and REXX for OS/2. We advise you to use a standard language which is supported on most platforms rather than exotic flavors of rare but nevertheless powerful languages. This will assure you of better support and will allow you to share the experience and sometimes even the applications of other developers. Check your favorite search engine and your news server to find them.

6.1.1.2 The Purpose of Your Application

Another important criterion in selecting a programming language is the purpose of your application. Not all languages are suited to every application. For example, a batch file under Windows NT is all it takes to switch to a different page depending on the browser used to view it. However, DOS commands are clearly inappropriate to query and update complex databases. Therefore, make sure the programming language you choose allows you to do want you want it to do, and even a little more. A good way of finding out if it does is to search the Internet for examples of applications similar to the ones you want to create.

6.1.1.3 Your Programming Skills

The two previous criteria may still leave you with a choice among several programming languages. In this case, use a programming language that you are familiar with. This will allow you to develop safe and reliable applications easily. After all, you are developing potentially exposed applications. You need to have sufficient knowledge of the language to ensure that your CGI scripts are reliable and do not expose your server to hackers and other undesirables. Furthermore, you want to deliver the relevant information continuously and safely for your network. This will be much easier if you are comfortable with your programming environment.

6.1.1.4 Response Time

The response times of your application may determine whether you will use an interpreted or a compiled programming language. If the required response times are to be small then you will want to opt for a compiled language. Some languages, such as REXX, may be run interpreted or compiled, thus offering both the easy testing and debugging of an interpreted language, and the speed of a compiled language.

6.1.2 Programming Languages

In this section we list some of the programming languages with which it is possible to develop CGI scripts. Select the one you will use based upon the above criteria.

A complete description of these languages would exceed the scope of this book so we do not attempt it.

Furthermore, updated descriptions of the languages most commonly used on the Internet are available on the Internet. We recommend that you consult these descriptions before you start a large project. A good starting point is Yahoo which can be found at:

```
http://www.yahoo.com/Computers_and_Internet/Programming_Languages/
```

Please refer to Table 10 for a summary of available languages by platform.

Table 10. CGI Programming Languages by Platform

	Windows NT	OS/2	AIX	HP-UX	Solaris	MVS
Scripting Languages	DOS, batch files	OS/2, batch files, command files	Shell Scripts (Bourne, Korn, C, bash, etc.)	Shell Scripts (Bourne, Korn, C, bash, etc.)	Shell Scripts (Bourne, Korn, C, bash, etc.)	OMVS POSIX Shell Script
C	Freeware	Freeware	Operating System, Freeware, Commercial	Operating System, Freeware	Operating System, Freeware	Commercial
Perl	Freeware	Freeware	Freeware	Freeware	Freeware	Freeware
REXX	Evaluation, Commercial	Operating System	Freeware, Shareware, Commercial	Freeware, Shareware	Freeware, Shareware	Operating System
NetRexx	Not Available	Freeware	Not Available	Not Available	Not Available	Not Available
Java	Not Available	Freeware	Freeware	Freeware	Freeware	Not Available

Notice that Perl is available on all platforms for which there is an Internet Connection server. This explains why Perl is one of the most popular CGI programming languages.

Java however is now becoming the Internet programming language, because of its adaptation to the Internet. Although Java is mainly used in applets imbedded into HTML documents, it is possible to write stand-alone Java programs that can thus be used as CGI scripts. We provide an example of this in Appendix A, “Two Simple Browsers and One Simple Server” on page 299.

6.1.3 Using Environment Variables - Welcome

This example demonstrates the use of the server′s environment variables, and involves minimal programming.

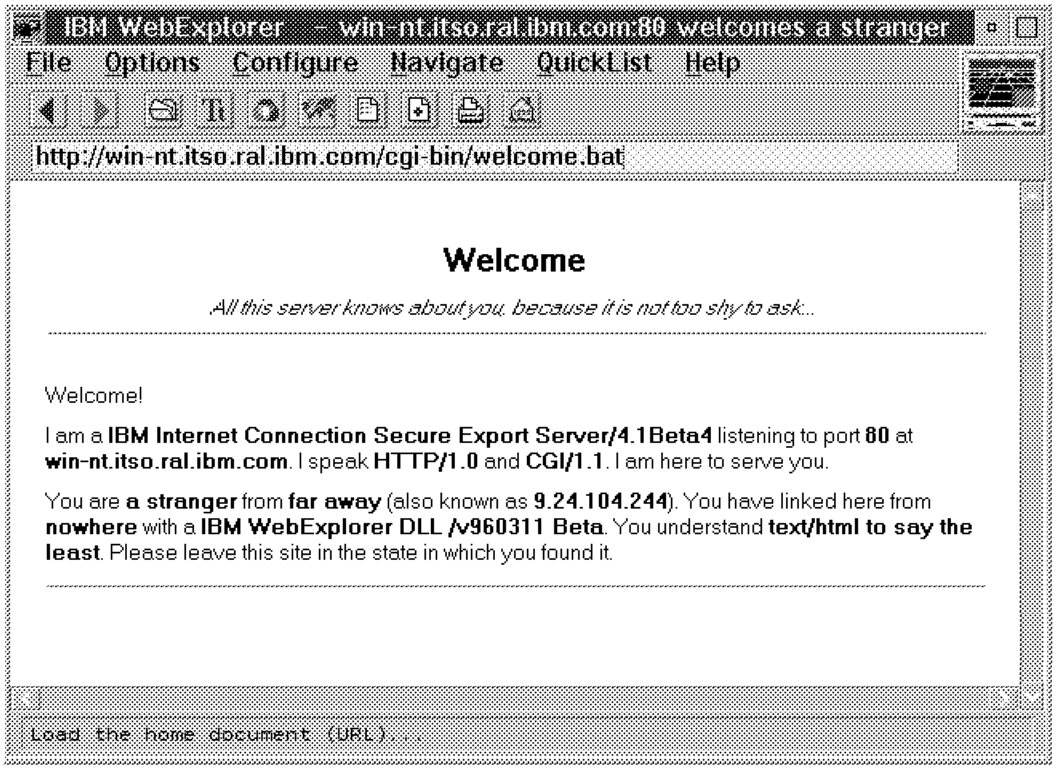
As we mentioned before, an HTTP transaction is divided into four steps:

- The browser connects to the server.
- The browser sends a request to the server.
- The server responds to the browser.
- The connection is closed.

Also, as we already stated, the browser′s request may contain some surplus fields in addition to the HTTP request. Most of these fields describe the browser and its capabilities. The server places the most common fields in environment variables that are accessible to any application triggered by the browser′s request. This is the case for CGI scripts or programs using the server′s API.

In this example the Welcome CGI script returns a page containing some information about the browser used to display it. This information has been previously stored by the server in the form of environment variables.

In Figure 138, Figure 139 on page 221, Figure 140 on page 222, and Figure 141 on page 223, you can see the page returned by the CGI script when viewed using IBM WebExplorer Java Technology Demo, the Netscape Navigator Atlas Preview Release 2 for AIX, the Netscape Navigator Atlas Preview Gold Release 2 for Windows NT, and Charlotte for VM respectively.


IBM WebExplorer - win-nt.itso.ral.ibm.com:80 welcomes a stranger
File
Options
Configure
Navigate
QuickList
Help

http://win-nt.itso.ral.ibm.com/cgi-bin/welcome.bat

Welcome

All this server knows about you, because it is not too shy to ask...

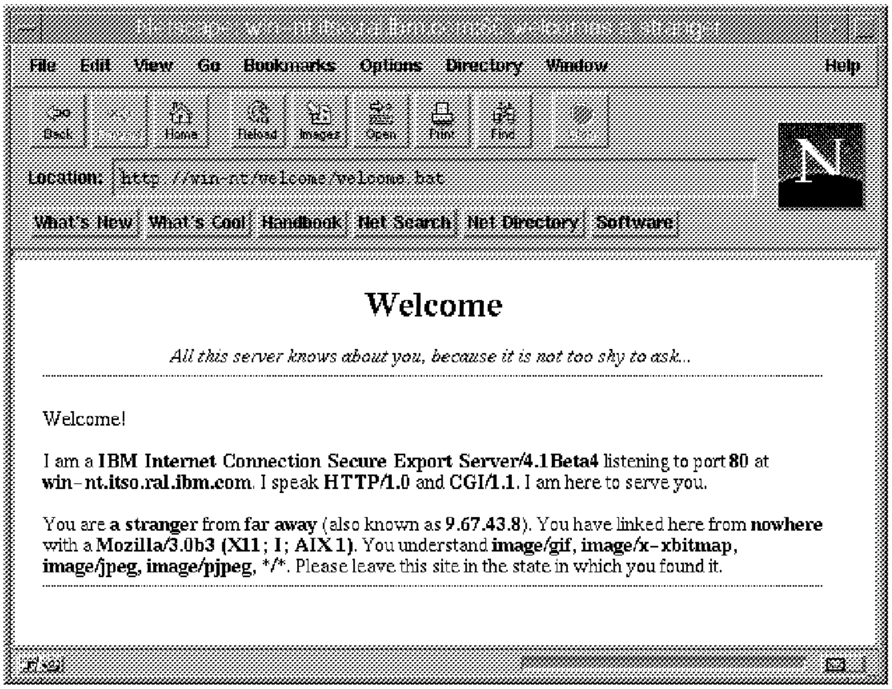
Welcome!

I am a **IBM Internet Connection Secure Export Server/4.1Beta4** listening to port **80** at **win-nt.itso.ral.ibm.com**. I speak **HTTP/1.0** and **CGI/1.1**. I am here to serve you.

You are **a stranger** from **far away** (also known as **9.24.104.244**). You have linked here from **nowhere** with a **IBM WebExplorer DLL /v960311 Beta**. You understand **text/html to say the least**. Please leave this site in the state in which you found it.

Load the home document (URL)...

Figure 138. The Welcome Page when Browsed with an IBM WebExplorer Java Technology Demo



Location: http://win-nt/welcome/welcome.bat

Welcome

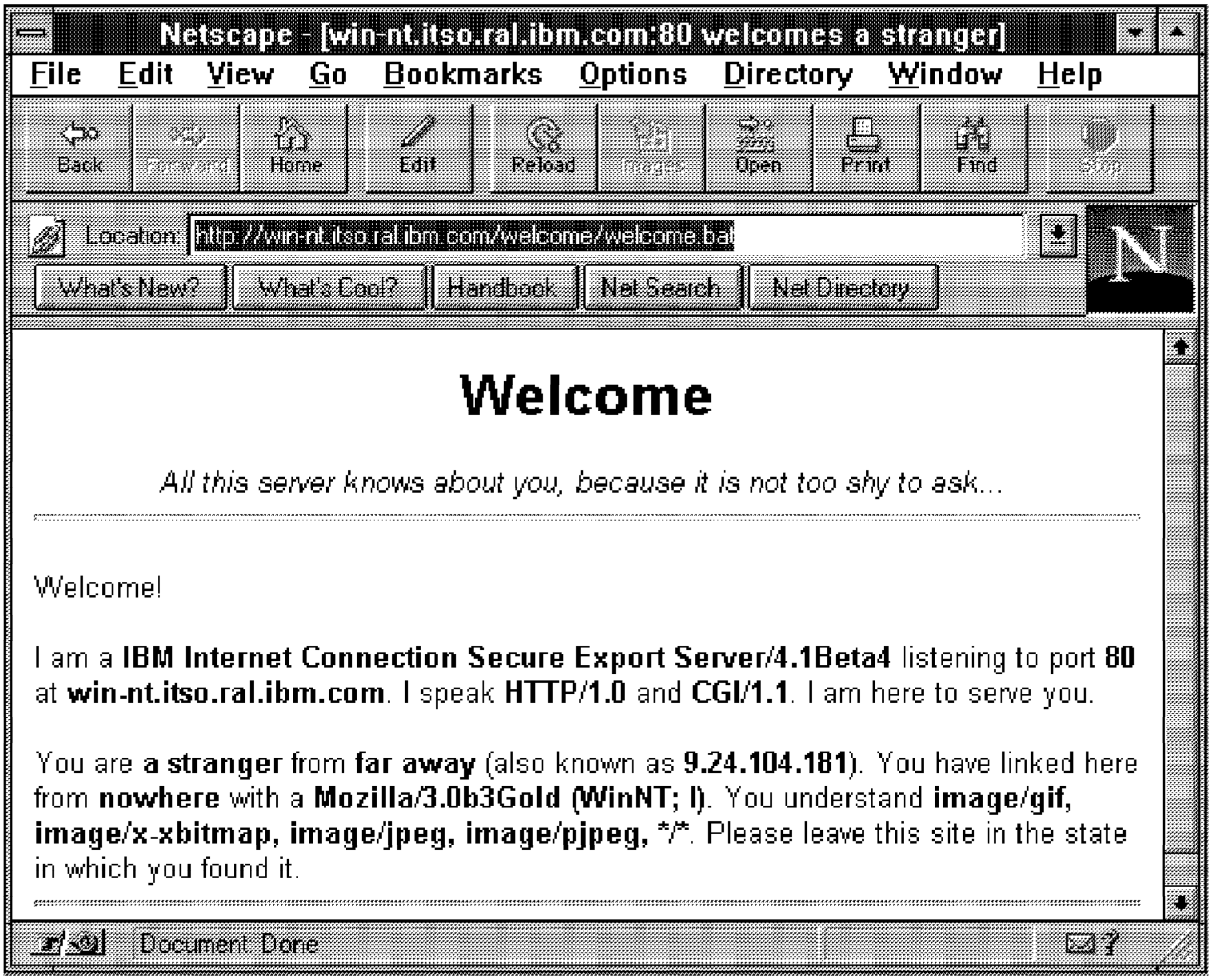
All this server knows about you, because it is not too shy to ask...

Welcome!

I am a **IBM Internet Connection Secure Export Server/4.1Beta4** listening to port **80** at **win-nt.itso.ral.ibm.com**. I speak **HTTP/1.0** and **CGI/1.1**. I am here to serve you.

You are **a stranger** from **far away** (also known as **9.67.43.8**). You have linked here from **nowhere** with a **Mozilla/3.0b3 (X11; I; AIX 1)**. You understand **image/gif, image/x-xbitmap, image/jpeg, image/pjpeg**, */*. Please leave this site in the state in which you found it.

Figure 139. The Welcome Page when Browsed with a Netscape Navigator Atlas Release 2 for AIX



Location: http://win-nt.itso.ral.ibm.com/welcome/welcome.bat

Welcome

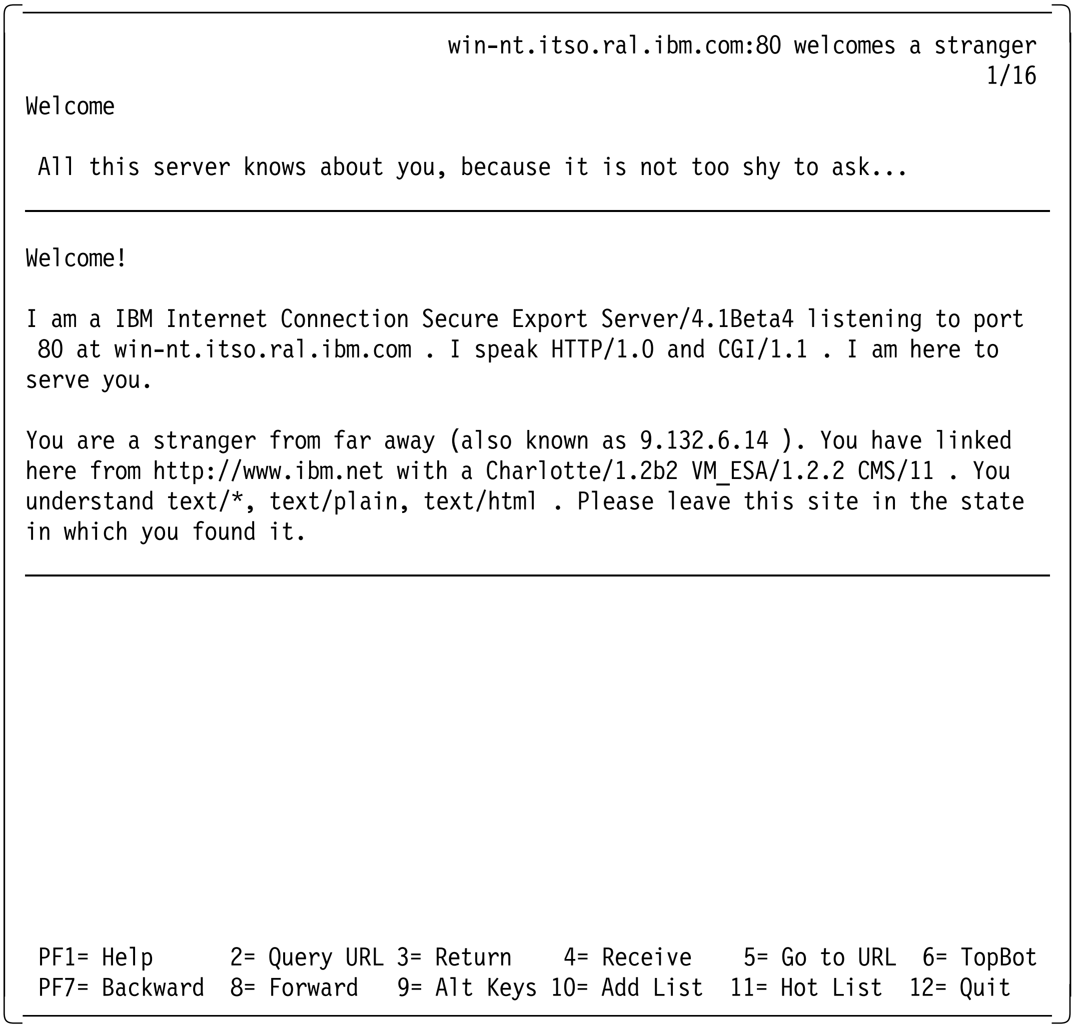
All this server knows about you, because it is not too shy to ask...

Welcome!

I am a **IBM Internet Connection Secure Export Server/4.1Beta4** listening to port **80** at **win-nt.itso.ral.ibm.com**. I speak **HTTP/1.0** and **CGI/1.1**. I am here to serve you.

You are **a stranger** from **far away** (also known as **9.24.104.181**). You have linked here from **nowhere** with a **Mozilla/3.0b3Gold (WinNT; I)**. You understand **image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */***. Please leave this site in the state in which you found it.

Figure 140. The Welcome Page when Browsed with a Netscape Atlas Preview Gold Release 2 for Windows NT



```
                                 win-nt.itso.ral.ibm.com:80 welcomes a stranger
                                                                          1/16
Welcome

 All this server knows about you, because it is not too shy to ask...
________________________________________________________________________________

Welcome!

I am a IBM Internet Connection Secure Export Server/4.1Beta4 listening to port
 80 at win-nt.itso.ral.ibm.com . I speak HTTP/1.0 and CGI/1.1 . I am here to
serve you.

You are a stranger from far away (also known as 9.132.6.14 ). You have linked
here from http://www.ibm.net with a Charlotte/1.2b2 VM_ESA/1.2.2 CMS/11 . You
understand text/*, text/plain, text/html . Please leave this site in the state
in which you found it.
________________________________________________________________________________










 PF1= Help      2= Query URL 3= Return    4= Receive    5= Go to URL  6= TopBot
 PF7= Backward  8= Forward   9= Alt Keys 10= Add List  11= Hot List  12= Quit
```

Figure 141. The Welcome Page when Browsed with Charlotte

The Welcome page is simply called by linking to the URL of the welcome.bat batch file, which during our project was:

```
http://win-nt.itso.ral.ibm.com/welcome/welcome.bat.
```

To be able to launch this batch program from a browser we added this line in our HTTPD configuration file:

```
Exec /welcome/* C:\WWW\Welcome\*
```

Note that if we had added this line in our HTTPD configuration file:

```
Exec /welcome/* C:\WWW\Welcome\\welcome.bat
```

the users would have been able to trigger the application by linking to the simpler http://win-nt.itso.ral.ibm.com/welcome URL.

The batch file itself is listed below.

```
@echo OFF

REM Read the environment variables

    SET GI=%GATEWAY_INTERFACE%
    SET HA=%HTTP_ACCEPT%
    SET HU=%HTTP_USER_AGENT%
    SET RA=%REMOTE_ADDR%
    SET RH=%REMOTE_HOST%
    SET RI=%REMOTE_IDENT%
    SET RU=%REFERER_URL%
    SET SN=%SERVER_NAME%
    SET SO=%SERVER_PORT%
    SET SR=%SERVER_PROTOCOL%
    SET SS=%SERVER_SOFTWARE%

REM Correct some values

    IF "%HA%"=="" SET HA=text/html to say the least
    IF "%HU%"=="" SET HU=unknown user agent
    IF "%RH%"=="" SET RH=far away
    IF "%RI%"=="" SET RI=a stranger
    IF "%RU%"=="" SET RU=nowhere

REM Send the document type

    cgiutils -ct text/html

REM Send the HTML

    echo ^<HTML^>
    echo ^<HEAD^>
    echo ^<TITLE^>%SN%:%SO% welcomes %RI%^</TITLE^>
    echo ^</HEAD^>
    echo ^<BODY BGCOLOR=FFFFFF^>
    echo ^<CENTER^>
    echo ^<H1^>Welcome^</H1^>
    echo ^<I^>All this server knows about you, because it is not too shy to ask...^</I^>
    echo ^<HR^>
    echo ^</CENTER^>
    echo ^<P^>Welcome!
    echo ^<P^>I am a ^<B^>%SS%^</B^> listening to port ^<B^>%SO%^</B^> at ^<B^>%SN%^</B^>.
    echo I speak ^<B^>%SR%^</B^> and ^<B^>%GI%^</B^>.
    echo I am here to serve you.
    echo ^<P^>You are ^<B^>%RI%^</B^> from ^<B^>%RH%^</B^> (also known as ^<B^>%RA%^</B^>).
    echo You have linked here from ^<B^>%RU%^</B^> with a ^<B^>%HU%^</B^>.
    echo You understand ^<B^>%HA%^</B^>.
    echo Please leave this site in the state in which you found it.
    echo ^<CENTER^>
    echo ^<HR^>
    echo ^</CENTER^>
    echo ^</BODY^>
    echo ^</HTML^>
```

Figure 142. The Welcome.bat Batch File

This simple batch file processes the server′s environment variables and sends an HTML page to the browser that summoned it.

This example is fairly self-explanatory, so we do not go any further with explaining DOS batch file programming.

The following line is quite important, however:

```
cgiutils -ct text/html
```

This creates a minimal HTTP 1.0 header containing the MIME content-type field. The cgiutils tool is a convenient way to create standard HTTP 1.0 headers. Note that the cgiutils tool must be located in the application path, and that this would normally be the case after the Internet Connection server installation. For more details about the HTTP header, please refer to Chapter 3, “Web Browsers” on page 47. For more details about the cgiutils tools please refer to the *InternetConnection Server Up and Running!* manual.

6.1.3.1 The Browser′s Host and Domain Name

Looking at the examples illustrating the Welcome page, it seems that the Remote_Host environment variable is never set. However these examples were produced while the DNS-Lookup directive of the HTTPD configuration file was set to Off, as we strongly recommend. The DNS-Lookup directive specifies whether the server looks up the names of the clients. If the DNS-Lookup directive had been set to On, then the Welcome page would have returned the browser′s full host name.

Note that the server always knows the browser′s IP address, and also note that this address may also be the one associated with the browser′s SOCKS or proxy server.

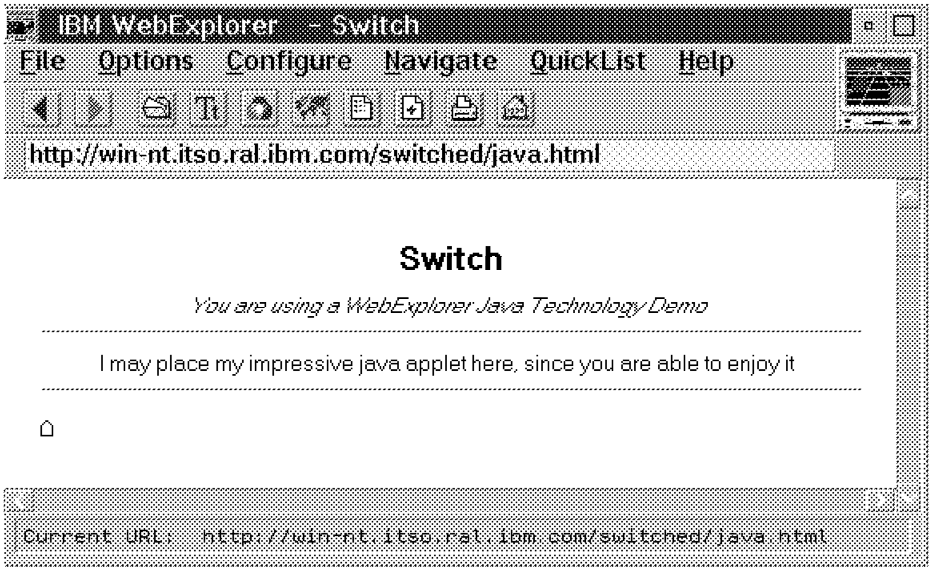
When the DNS-Lookup directive is set to Off, it is possible to compute the browser′s host name from its IP address by using the HOST command, shipped with every good TCP/IP stack.

6.1.4 Using Environment Variables - Switch

In the previous example we introduced the environment variables. In this example we show some further uses for them.

What we are trying to achieve in this example is a home page which is compatible with as many browsers as possible.

The user always connects to the same URL (http://win-nt.itso.ral.ibm.com), but the server sends a different page whether the browser is the IBM WebExplorer Java Technology Demo or any other browser, as shown in Figure 143 on page 226 and Figure 144 on page 226, respectively.



http://win-nt.itso.ral.ibm.com/switched/java.html

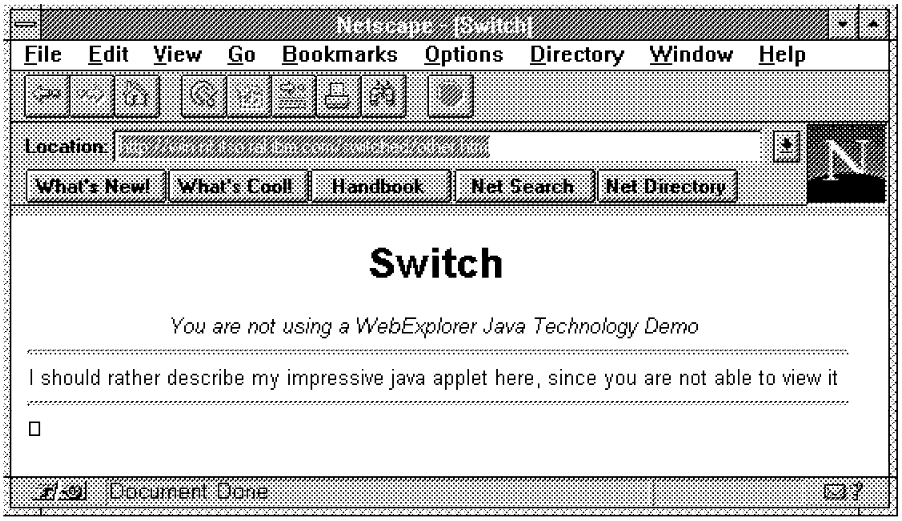
Switch

You are using a WebExplorer Java Technology Demo

I may place my impressive java applet here, since you are able to enjoy it

Current URL: http://win-nt.itso.ral.ibm.com/switched/java.html

Figure 143. Browsing with an IBM WebExplorer Java Technology Demo


Netscape - [Switch]
File Edit View Go Bookmarks Options Directory Window Help
Location:
What's New! What's Cool! Handbook Net Search Net Directory

Switch

You are not using a WebExplorer Java Technology Demo

I should rather describe my impressive java applet here, since you are not able to view it

Document Done

Figure 144. Browsing with Netscape Navigator for Windows 3.1

Let us first describe the program that returns the location to load depending on the browser′s type. As the programming involved is rather elementary (an IF THEN ELSE branch) we used the DOS batch file listed below.

```
@ECHO OFF

REM Determine the browser

    SET HU=%HTTP_USER_AGENT%

REM Determine the page corresponding to the browser

    SET HP=java.html
    IF NOT ″%HU%″==″IBM WebExplorer DLL /v960311 Beta″ SET HP=other.html

REM Tell the browser to load the relevant page

    ECHO Location: http://win-nt.itso.ral.ibm.com/switched/%HP%
    ECHO.

REM And don′t forget the mandatory empty line (ECHO.)
```

Figure 145. The switch.bat Batch File

We named this batch file switch.bat, and we located it in the C:\WWW\Switch directory.

In order to be able to run this batch file from a browser, we must add the following line to the HTTPD configuration file:

```
EXEC /switch/* C:-WWW-Switch-*
```

It is now possible to view the page corresponding to our browser by linking to http://win-nt.itso.ral.ibm.com/switch/switch.bat.

But we want this to happen upon linking to http://win-nt.itso.ral.ibm.com. To achieve this we simply add the following line prior to the above EXEC directive:

```
MAP /* /switch/switch.bat
```

This makes sure that any link to http://win-nt.itso.ral.icm.com/* will call the switch.bat CGI script. Thus, any link to a file or directory that is not stated in an EXEC, PASS or REDIRECT directive prior to the previous MAP directive will call the switch.bat program. For example, if a user links to http://win-nt.itso.ral.ibm.com/foo/foo.html and neither the foo directory nor the foo.html file are defined, then the user will launch the switch.bat script.

Note that instead of having:

```
MAP  /*       /switch/switch.bat
EXEC /switch/* C:\WWW\Switch\*
```

we could have specified:

```
MAP  /*       /switch/*
EXEC /switch/* C:\WWW\Switch\switch.bat
```

6.1.5 Linking to a CGI Script - Calendar

In this sample, we show you:

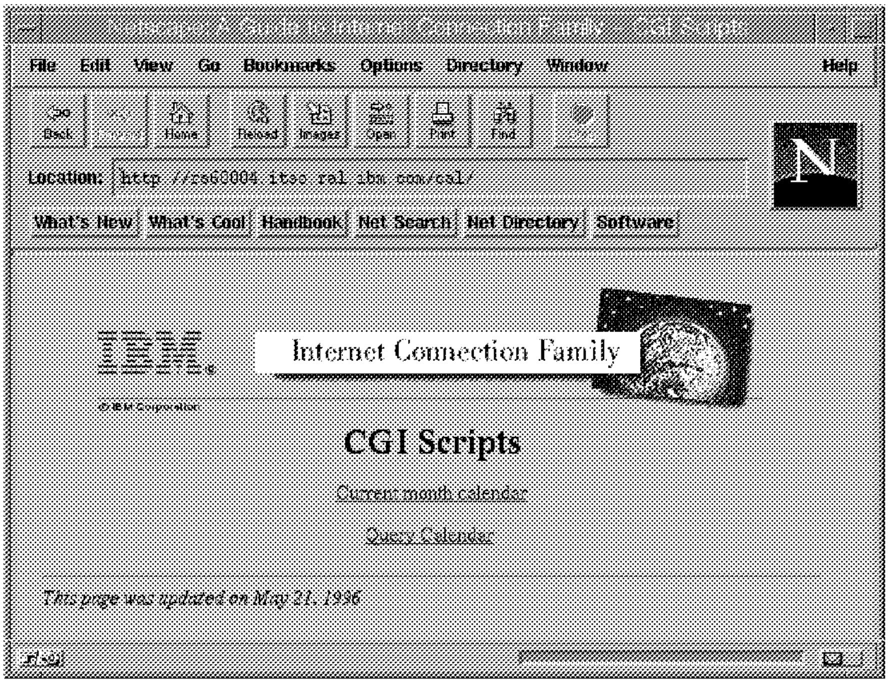
- How to invoke your script from your HTML page
- How to design your CGI script
- Where to store your files
- How to customize your Internet Connection server to serve the new files

Let′s start with a description of this sample.

The URL:

`http://rs60004.itso.ral.ibm.com/cal`

identifies the following page:



http://rs60004.itso.ral.ibm.com/cal/

CGI Scripts

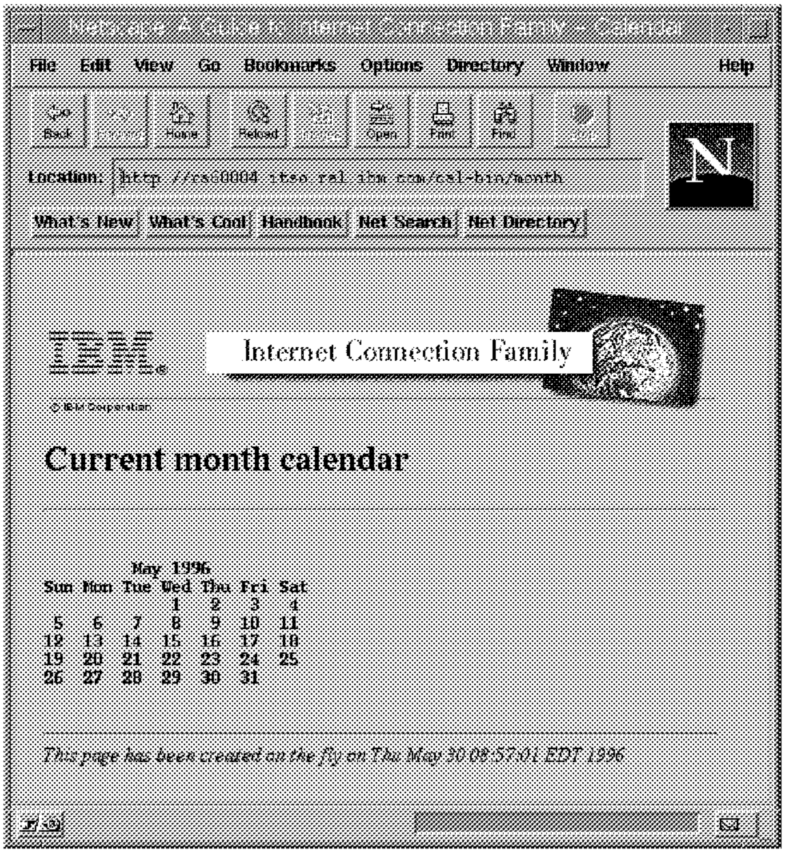
Current month calendar

Query Calendar

This page was updated on May 21, 1996

Figure 146. Calendar Home Page

By clicking on the **Current Month Calendar** link, the user is redirected to a new HTML page that contains a calendar of the current month:



Location: http://rs60004.itso.ral.ibm.com/cal-bin/month

Internet Connection Family

Current month calendar

```
          May 1996
Sun Mon Tue Wed Thu Fri Sat
              1   2   3   4
  5   6   7   8   9  10  11
 12  13  14  15  16  17  18
 19  20  21  22  23  24  25
 26  27  28  29  30  31
```

This page has been created on the fly on Thu May 30 08:57:01 EDT 1996

Figure 147. Built on the Fly Calendar HTML Page

6.1.5.1 Invoking a Script from a Link

Invoking a script from a link is as simple as linking a page to another page.

The statement to declare a link to another page would be:

```
<A HREF="/cal/anotherpage.html"Another Page</A>
```

Whereas for a call to the month CGI script it is:

```
<A HREF="/cal-bin/month">Current month calendar</A>
```

where /cal-bin/month is the request path that identifies the CGI script.

Here is the code being used on the home page shown in Figure 146 on page 228.

```
<HTML>
<HEAD>
<TITLE>A Guide to Internet Connection Family - CGI Scripts</TITLE>
</HEAD>
<BODY>
<CENTER>
<H1>
<IMG SRC=″/img/masthead.gif″ ALT=″A Guide to Internet Connection Family″>
<BR>CGI Scripts</H1>
<P>
<A HREF=″/cal-bin/month″>Current month calendar</A><P>
<A HREF=″/cal/qcal.html″>Query Calendar</A>
</CENTER>
<P>
<HR>
<I>This page was updated on May 21, 1996</I>
</BODY>
</HTML>
```

Figure 148. HTML Source

6.1.5.2 Designing the CGI Script

From the URL displayed in Figure 147 on page 229, you may have guessed that the script is named month.

Since month relies on the standard UNIX cal command and does not require high-level routines, a shell script seems to be an appropriate language to use.

Month′s task is to return to the client:

- A header describing the nature of the response
- The body of the response which is, in this case, the text of the HTML page

As defined in the Common Gateway Interface standard, the output of the script should go to the standard output. Thus, month should write to stdout some plain text such as that shown in Figure 149.

```
Content-type: text/html

<HTML><HEAD><TITLE>Page title</TITLE></HEAD>
<BODY>
Page body
</BODY>
</HTML>
```

Figure 149. HTML Source Pattern

The month script is now straightforward. Our version in shell script is listed in Figure 150 on page 231 but we are sure you will find it easy to rewrite it in your favorite language.

```
#!/bin/ksh

#
# month script
#
# This script is invoked without any argument
# It builds an HTML page including :
#       all required HTML tags (<HTML>, <HEAD>, <BODY>, etc )
#       a H1 header
#       the calendar of the month (as displayed by the Unix cal command)
#
# The output of this script will be sent as is to the client (browser)
# this means that it must first print the header
#

echo ”Content-type: text/html\n”
echo ”<HTML><HEAD><TITLE>”
echo ”A Guide to Internet Connection Family - Calendar”
echo ”</TITLE></HEAD><BODY>”
echo ”<IMG SRC=-”/img/masthead.gif-” ALT=\”A Guide to Internet Connection \”>”
echo ”<H1>Current month calendar</H1><HR>”
echo ”<PRE><B>”
cal
echo ”</B></PRE><HR>”
echo ”<I>This page has been created on the fly on date</I>”
echo ”</BODY></HTML>”
```

Figure 150. Month Shell Script

Note for UNIX developers

Make sure that the first statement in your script indicates the shell required to execute it otherwise you will get an internal error (execve() failed).

6.1.5.3 Storing Files

Deciding where to store the files does not look like a crucial job in this particular example; nevertheless it′s worth discussing this matter because the way you organize your files has an impact on maintenance and security.

Keeping the files related to your program in a specific directory is certainly a good habit to develop. It will help you with all administration tasks such as backup, access restrictions, program maintenance, etc. You should create a dedicated subdirectory in the root directory of your Web server. In this example, we have created the /usr/local/server_root/cal subdirectory on our AIX platform. This subdirectory contains the document files. It would have been:

- `D:\WWW\CAL`
- `/usr/local/ServerRoot/cal`

for OS/2 or Windows NT and MVS respectively.

You should remember to have a welcome page in all your document directories unless you have set the DirAccess directive to Off in your configuration file. This will help you to make sure that you don′t let the server list directories when you don′t mean to and furthermore it will help the user to find the key HTML document in your directory.

In our example, the Welcome Page (shown in Figure 146 on page 228) is named index.html.

The Internet Connection Server distinguishes requests between documents and CGI programs according to the URL. The mapping directives in the configuration file (Exec, Fail, Map, Pass and Redirect) control which requests your server accepts and maps to your actual files.

You see in 6.1.5.4, “Customizing the Internet Connection Server” which directives are required for this specific example.

In some situations, if you mix CGI programs and document files in the same directory, it may happen that CGI programs are viewed by the client as mere documents thus allowing the client to download them to its workstation. This could happen if:

- DirAccess is set to On and there is no welcome document in the directory.
- DirAccess is set to selective, there is no welcome document in the directory and the directory is browsable. (.www_browsable file exists in the directory.)

This is why we advise you not to locate your CGI files in the same directory as your document files.

You should create a subdirectory to keep CGI files. In our example:

- `/usr/local/server_root/cal/cgi`
- `D:\WWW\CAL|CGI`
- `/usr/local/ServerRoot/cal/cgi`

for AIX, OS/2 or Windows NT and MVS respectively.

6.1.5.4 Customizing the Internet Connection Server

At this point in the implementation process, the users still do not have access to the files. Additional directives are required in the configuration file in order to map requests to complete file paths for both CGI programs and document files.

Since we want our server to respond to requests starting with /cal with a document in /usr/local/server_root/cal, the following directive should be added to the configuration file:

```
Pass    /cal/*         /usr/local/server_root/cal/*
```

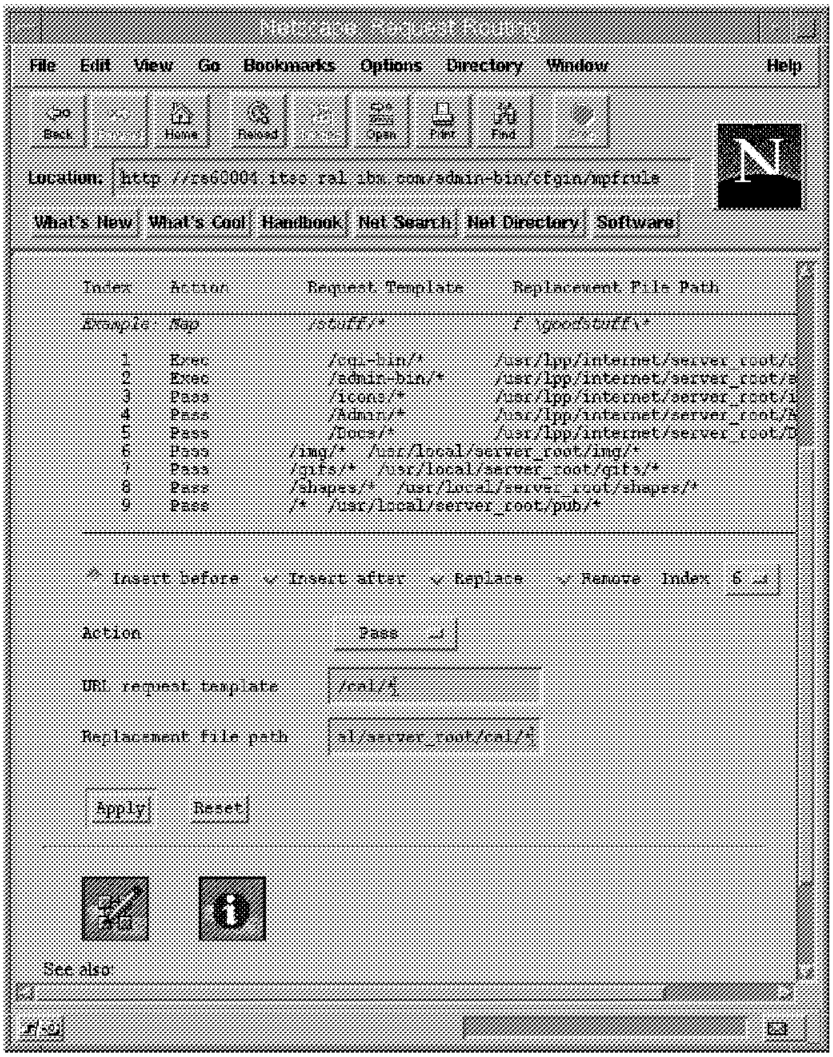
We also want our server to respond to requests starting with /cal/cgi-bin with the execution of CGI programs in /usr/local/server_root/cal/cgi; therefore, the following directive should be added to the configuration file:

```
Exec    /cal-bin/*    /usr/local/server_root/cal/cgi/*
```

These directives can be added by using the Configuration and Administration Forms. The corresponding entry in the menu is:

- Request Routing - Route URL requests to server files

The form is shown in Figure 151 on page 233.



Location: http://rs60004.itso.ral.ibm.com/admin-bin/cfgin/mpfrule

```
Index    Action        Request Template      Replacement File Path
Example: Map           /stuff/*              f:\goodstuff\*

    1    Exec           /cgi-bin/*      /usr/lpp/internet/server_root/c
    2    Exec           /admin-bin/*    /usr/lpp/internet/server_root/a
    3    Pass           /icons/*        /usr/lpp/internet/server_root/i
    4    Pass           /Admin/*        /usr/lpp/internet/server_root/A
    5    Pass           /Docs/*         /usr/lpp/internet/server_root/D
    6    Pass       /img/*  /usr/local/server_root/img/*
    7    Pass       /gifs/*  /usr/local/server_root/gifs/*
    8    Pass       /shapes/*  /usr/local/server_root/shapes/*
    9    Pass       /*  /usr/local/server_root/pub/*
```

● Insert before ○ Insert after ○ Replace ○ Remove Index 6

Action: Pass

URL request template: /cal/*

Replacement file path: al/server_root/cal/*

Apply Reset

Figure 151. Request Routing Form

For the first directive to be added:

- Action is Pass.
- URL Request Template is /cal/*.
- Replacement File Path is /usr/local/server_root/cal/*.

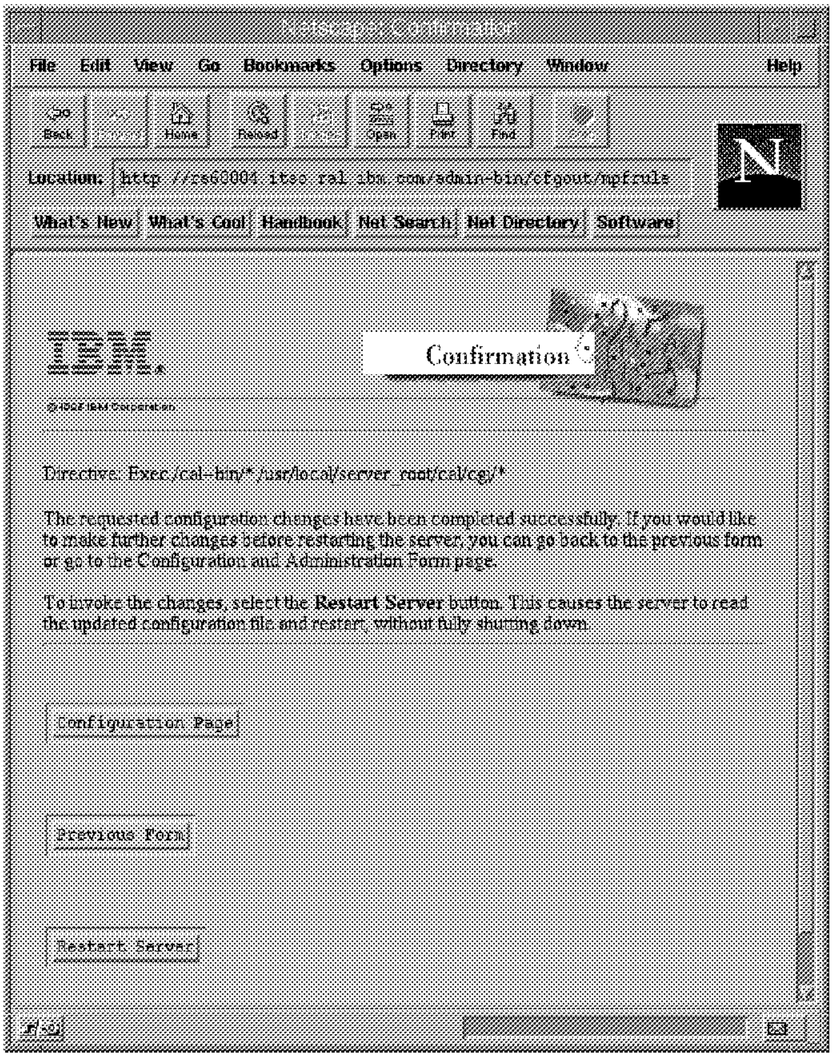
This directive should be inserted in any case before the last directive, that is:

```
Pass /* /usr/local/server_root/pub/*
```

which routes in last resort to the server root.
Click on the **Apply** button of the form to validate it.

After a few seconds, the Confirmation page will invite you either:

- To make further changes before restarting the server
- To invoke the changes


Netscape: Confirmation
File Edit View Go Bookmarks Options Directory Window Help
Back Forward Home Reload Images Open Print Find Stop
What's New What's Cool Handbook Net Search Net Directory Software
Confirmation
©1996 IBM Corporation

Location: http://rs60004.itso.ral.ibm.com/admin-bin/cfgout/mpfrule

Directive: Exec /cal-bin/* /usr/local/server_root/cal/cgi/*

The requested configuration changes have been completed successfully. If you would like to make further changes before restarting the server, you can go back to the previous form or go to the Configuration and Administration Form page.

To invoke the changes, select the **Restart Server** button. This causes the server to read the updated configuration file and restart, without fully shutting down.

Configuration Page

Previous Form

Restart Server

Figure 152. Request Routing Confirmation

When you are finished with the changes and the directives have been added to your configuration, restart the server in order to activate the configuration changes you have made. After a restart of the server, you can access the Calendar Home Page and from there launch your CGI script.

Note that, as for any CGI script, you could launch the month CGI script manually; in this case by requesting the URL:

```
http://rs60004.itso.ral.ibm.com/cal-bin/month
```

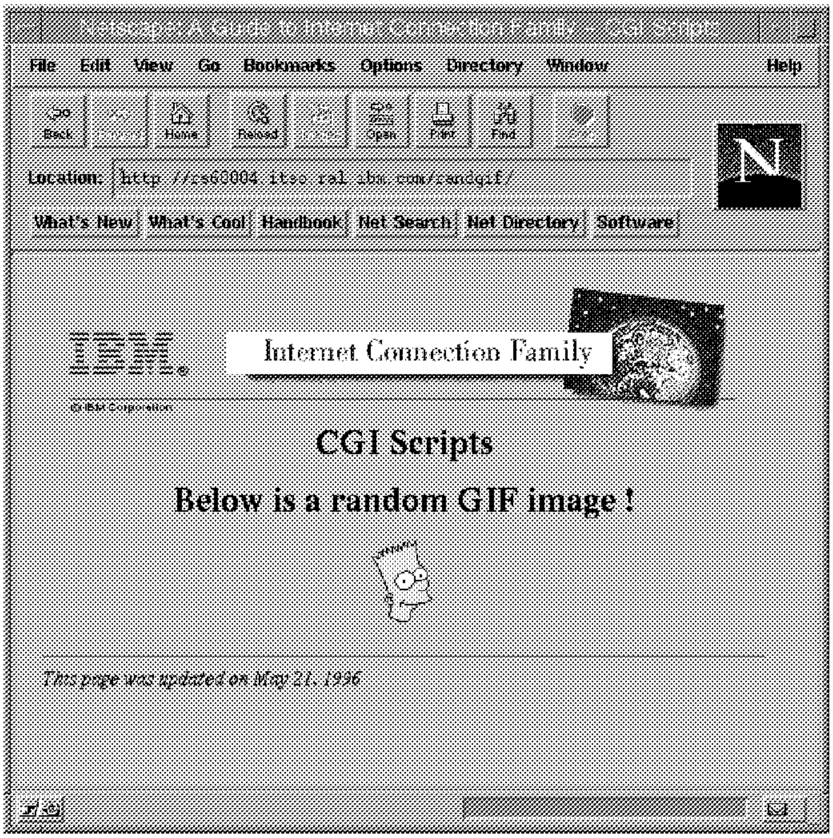
6.1.6 Building On-The-Fly Pages - The Simpsons

There are many occasions when you would like to build on-the-fly pages in your Web server. For example:

- Having a random image on your home page to animate your server
- Advertising for one of your business partners
- Displaying a contextual image based on criteria such as date, time, etc.

In order to illustrate this technique, we chose the following scenario: when accessing the URL httpd://rs60004.itso.ral.ibm.com/randgif an HTML page will be returned to the browser including a header, some text and a random image. This image will be randomly chosen from the portraits of the Simpsons characters. (The GIF files of the Simpsons family were found on the Web at http://www-pcd.stanford.edu/gifs.)

An example follows:



Netscape: A Guide to Internet Connection Family - CGI Scripts

Location: http://rs60004.itso.ral.ibm.com/randgif/

Internet Connection Family

CGI Scripts

Below is a random GIF image !

This page was updated on May 21, 1996

Figure 153. HTML Page with Random GIF Image

6.1.6.1 Integrating a Random Image in a Page

The technique would be the same if you wanted to integrate a page-hit counter or an advertisement or any other image in your HTML document.

The following HTML code:

```
<IMG SRC="/randgif-bin/randgif">
```

is included in the HTML source page. This means that when someone requests the document, the user's browser will also request an image file named /randgif-bin/randgif to be loaded. This is not a real image file, but it causes the Web server to run the program in order to build the image.

Therefore, the code for the HTML page is as follows:

```
<HTML>
<HEAD>
<TITLE>A Guide to Internet Connection Family - CGI Scripts</TITLE>
</HEAD>
<BODY>
<CENTER>
<H1><IMG SRC="/img/masthead.gif" ALT="A Guide to Internet Connection Family">
<BR>CGI Scripts</H1>
<P>
<H1>Below is a random GIF image !</H1>
<P>
<IMG SRC="/randgif-bin/randgif">
<P>
</CENTER>
<HR>
<I>This page was updated on May 21, 1996</I>
</BODY>
</HTML>
```

Figure 154. HTML Source

6.1.6.2 Designing the CGI Program

The CGI program, randgif, randomly chooses one of the images in the pool of Simpsons family images and writes it to the standard output. As in the example in 6.1.5, "Linking to a CGI Script - Calendar" on page 228, the CGI script must first send a header prior to the image itself. This means that the output should look something like the following.

```
Content-type: image/gif

<GIF file contents (binary)>
```

Figure 155. HTML Source Pattern

To develop this CGI script, we chose to write it in C language. In order to randomize the choice of the GIF image, we used the result of the integer division of the current time by the number of images in the pool. We relied on the UNIX time() routine which returns the current time expressed in seconds since 00:00:00 Coordinated Universal Time, January 1, 1970.

The C source is included in the figure below.

```
/*
 *      randgif.c
 *
 *      input parameters:
 *              none
 *
 *      output:
 *              This program send to the standard output:
 *                      - a header to give information on
 *                        the nature of the following data
 *                      - the binary data corresponding to a GIF image
 *                        randomly selected in a pool of images
 */

#include <stdio.h>
#include <errno.h>
#include <time.h>
#include <fcntl.h>

/* definitions */
/* directory where GIF files are stored */
#define GIFPATH ″/usr/local/server_root/randgif/gifs″
#define BUFSZ   256     /* temporary storage size */

/* GIF filenames array - as listed in GIF directory */
char    *gifs[] = {
        ″simp.baby.gif″,
        ″simp.bart.gif″,
        ″simp.boss.gif″,
        ″simp.clown.gif″,
        ″simp.granpa.gif″,
        ″simp.homer.gif″,
        ″simp.lisa.gif″,
        ″simp.moe.gif″,
        ″simp.nerd.gif″,
        ″simp.saunders.gif″
};

/* response header for GIF image */
char    header[] = ″Content-type: image/gif-n-n″;
```

Figure 156 (Part 1 of 2). Randgif.c Source Code

```
/*
 *      main
 */
{
        time_t  curtime;                /* current time */
        int     nb_of_gifs;             /* number of entries in array */
        int     fd;                     /* GIF file handler */
        int     count;                  /* I/O counter */
        char    gif_file[ 255 ];        /* GIF file name */
        char    buf[ BUFSZ ];           /* temporary storage */

        /* get current time */
        curtime = time( NULL );

        /*
         * choose filename ″randomly″ :
         *      remainder of the integer division serves as random number
         *      in the correct range
         */

        /* calculate the number of entries in the filename array */
        nb_of_gifs = sizeof( gifs ) / sizeof( char * );

        /* build the GIF filename, using the ″random″ number */
        sprintf( gif_file, ″%s/%s″, GIFPATH,
                  gifs[ curtime % nb_of_gifs ] );


        if ( ( fd = open( gif_file, O_RDONLY ) ) < 0 ) {
                perror( ″open″ );
                exit( 1 );
        }

        /* write the header to standard output */
        write( 1, header, strlen( header ) );

        /* copy GIF data file to standard output */
        while ( count = read( fd, buf, BUFSZ ) ) {
                write( 1, buf, count );
        }
        close( fd );
        exit( 0 );
```

Figure 156 (Part 2 of 2). Randgif.c Source Code

6.1.6.3 Customizing the Internet Connection Server

In this example, we organized our files in the following manner:

- We created a new directory:

  `/usr/local/server_root/randgif`

  in which we stored the Welcome HTML page (named index.html).

- We created a subdirectory:

  `/usr/local/server_root/randgif/gifs`

  in which we stored the Simpsons GIF files.

- We created a subdirectory:

  ```
  /usr/local/server_root/randgif/cgi
  ```

  in which we stored the CGI program (source and executable files).

We added the following directives to our configuration file:

```
Exec    /randgif-bin/* /usr/local/server_root/randgif/cgi/*
Pass    /randgif/* /usr/local/server_root/randgif/*
```

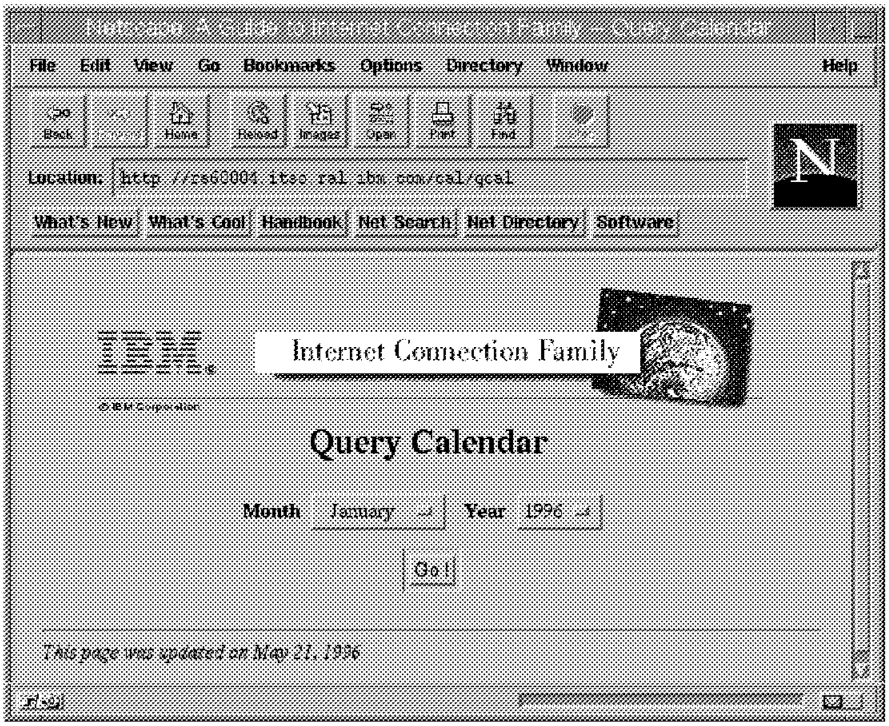
Note that there is no need to add any directive relative to the /usr/local/server_root/randgif/gifs directory since this directory is never directly accessed by the Internet Connection server.

6.1.7 Processing Forms - Calendar

In this sample, we show you how to process forms with a CGI script. Forms can be used for many purposes such as:

- E-mail gateway
- Feedback form
- Order form
- Bulletin board systems

The scenario described in this section returns to our earlier theme of calendars (see 6.1.5, “Linking to a CGI Script - Calendar” on page 228). This time the users will be able to choose, via a form, the month and the year for which they want the calendar to be displayed. The end result of our efforts is shown in Figure 157.



Netscape: A Guide to Internet Connection Family - Query Calendar

Location: http://rs60004.itso.ral.ibm.com/cal/qcal

Query Calendar

Month January Year 1996

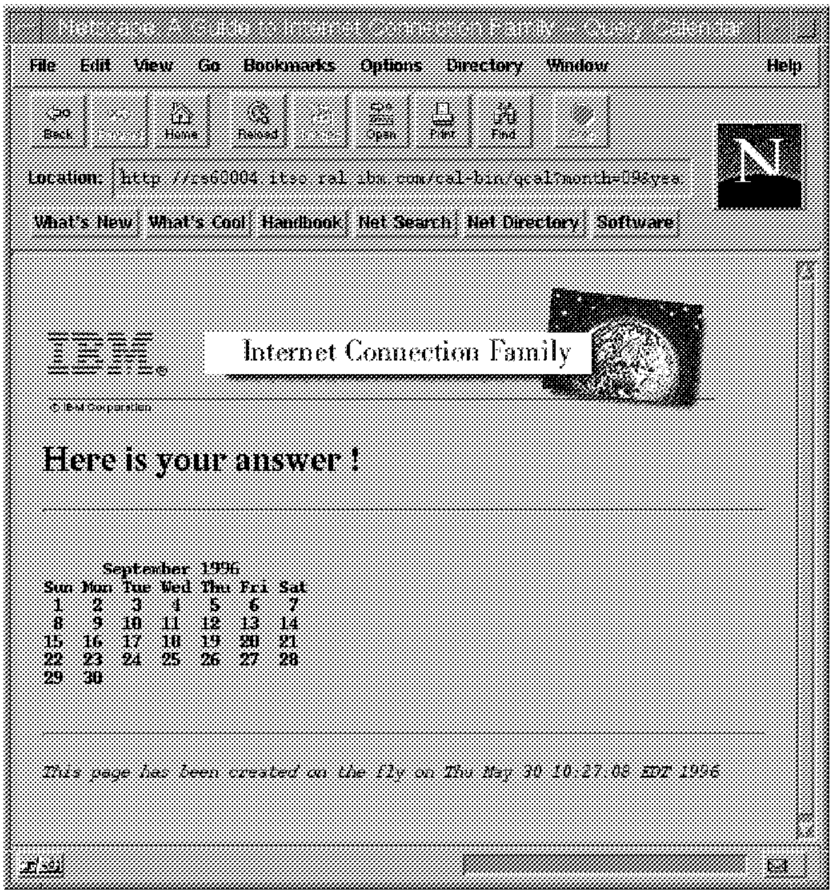
Go!

This page was updated on May 21, 1996

Figure 157. Calendar Form

As you can see, users have a simple, easy-to-understand interface. They just have to select the month and the year from the select-lists, then click on the **Go!** button.

The page returned to the browser by the script returns the calendar corresponding to the request:



Location: http://rs60004.itso.ral.ibm.com/cgi-bin/qcal?month=09&yea

Here is your answer !

```
        September 1996
Sun Mon Tue Wed Thu Fri Sat
  1   2   3   4   5   6   7
  8   9  10  11  12  13  14
 15  16  17  18  19  20  21
 22  23  24  25  26  27  28
 29  30
```

This page has been created on the fly on Thu May 30 10:27:08 EDT 1996

Figure 158. Answer to Calendar Query

The HTML source code listed in Figure 159 on page 241 shows you that the key to building such pages is the <FORM> tag.

```
<HTML>
<HEAD>
<TITLE>A Guide to Internet Connection Family - Query Calendar</TITLE>
</HEAD>
<BODY>
<CENTER>
<H1>
<IMG SRC=”/img/masthead.gif” ALT=”A Guide to Internet Connection Family”>
<BR>Query Calendar
</H1>
<P>
<FORM ACTION=”/cal-bin/qcal” METHOD=”GET”>
<B>Month </B>
<SELECT NAME=month><OPTION SELECTED VALUE=01>January
<OPTION VALUE=02>February
<OPTION VALUE=03>March
<OPTION VALUE=04>April
<OPTION VALUE=05>May
<OPTION VALUE=06>June
<OPTION VALUE=07>July
<OPTION VALUE=08>August
<OPTION VALUE=09>September
<OPTION VALUE=10>October
<OPTION VALUE=11>November
<OPTION VALUE=12>December
</SELECT>
<B>Year </B>
<SELECT NAME=year><OPTION SELECTED VALUE=1996>1996
<OPTION VALUE=97>1997
<OPTION VALUE=98>1998
<OPTION VALUE=99>1999
</SELECT>
<BR>
<INPUT TYPE=submit NAME=”submit” VALUE=”Go !”></FORM>
</CENTER>
<P>
<HR>
<I>This page was updated on May 21, 1996</I>
</BODY>
</HTML>
```

Figure 159. HTML Source

The <FORM> tag is quite complicated in that it requires many statements spread on several lines in our sample.

The first line:

```
<FORM ACTION=”/cal-bin/qcal” METHOD=”GET”>
```

contains the beginning tag and tells the server what to do with the information that the user will fill in. When the user validates the form, the information will be sent to the server and the server will execute the CGI program whose URL is /cal-bin/qcal, using the GET method.

Note that the METHOD element could be omitted in this tag since the default method is the GET method.

The lines:

```
<SELECT NAME=month><OPTION SELECTED VALUE=01>January
```

```
<SELECT NAME=year><OPTION SELECTED VALUE=1996>1996
```

tell the server which information the user will have to give. In this case, the user will have to choose the month and the year using selection lists.

The line:

```
<INPUT TYPE=submit NAME="submit" VALUE="Go !">
```

tells the server to execute the CGI program when the Go ! button is clicked and ends the <FORM> tag.

Before invoking the CGI program, the server will set the environment according to the CGI standard specification. The environment variables are described in the Writing Common Gateway Interface Programs chapter of the Up and Running! manual.

In the GET method, the information is passed to the CGI program in the QUERY_STRING variable. In our sample, the information takes the following form:

```
month=09&year=1996&submit=Go+%21
```

Refer to 6.1.7.2, "Choosing between POST or GET Methods" on page 244 for a discussion on which method to use in your forms and an explanation regarding the format of the information passed to the scripts.

6.1.7.1 Designing the CGI Script using Perl

This sample uses qcal, a simple Perl script that takes the information from the user and uses it to create a personalized HTML page. The qcal script is simple and is quite easy to customize to give you the results you want. The structure is basic and straightforward:

- Initialization
- Processing
- Output
- Termination

In the initialization step, the script parses the argument. In this specific case, the cgiparse and cgiutils tools shipped with the Internet Connection server were used. The arguments could also have been retrieved from the environment variables.

In the processing step, the variable elements of the outgoing HTML page are evaluated.

In the next step, the header and the HTML page code are sent to the standard output.

No specific termination code was developed in this simple sample (no lock to release, no temporary file to erase, etc.).

The Perl script is documented in Figure 160 on page 243.

```
#!/usr/local/bin/perl

#
# qcal script
#
# This Perl script is invoked with two arguments :
#       - month: decimal number (01-12) - month of year
#       - year:  decimal number - year with century
# It builds an HTML page including :
#       all required HTML tags (<HTML>, <HEAD>, <BODY>, etc )
#       a H1 header
#       the calendar of the required month (as displayed by the Unix cal
#       command)
#
# The output of this script will be sent as is to the client (browser)
# this means that it must first print the header
#

# define variables
$parser=′/usr/lpp/internet/server_root/cgi-bin/cgiparse′;
$utils=′/usr/lpp/internet/server_root/cgi-bin/cgiutils′;

# parse arguments
$month = $parser -value month; chop( $month );
$year = $parser -value year; chop( $year );

$cal = cal $month $year;
$date = date;
$header = $utils -ct text/html;

print ″$header″;
print ″<HTML><HEAD><TITLE>″;
print ″A Guide to Internet Connection Family - Query Calendar″;
print ″</TITLE></HEAD><BODY>\n″;
print ″<IMG SRC=\″/img/masthead.gif\″″;
print ″ ALT=\″A Guide to Internet Connection Family\″>\n″;
print ″<H1>Here is your answer !</H1><HR>\n″;
print ″<PRE><B>\n″;
print ″$cal\n″;
print ″</B><HR>\n″;
print ″<I>This page has been created on the fly on $date</I>\n″;
print ″</PRE></BODY></HTML>\n″;
```

Figure 160. Perl Source Code

If you are looking for Perl samples, take a look at the following URL:

```
http://www.yahoo.com/Computers_and_Internet/World_Wide_Web/Programmi
ng/Perl_Scripts/
```

In addition, there are many public domain Perl libraries on the Web which will help you to develop your scripts. These include standard routines for processing forms, parsing arguments, etc. Take a look, for example, at:

```
http://www.bio.cam.ac.uk/cgi-lib/
http://www.ics.uci.edu/WebSoft/libwww-perl/
```

6.1.7.2 Choosing between POST or GET Methods

There are actually two methods to submit a form to a Web server: POST and GET.

In both methods, the data passed to the CGI application has the same form: ordered pairs of information with an equal sign tying together two elements of a pair and an ampersand tying pairs together; a plus sign takes place of spaces and special characters are encoded in hexadecimal. This is easier to see than to describe, for example:

```
sel1=value1&sel2=value21+value22&submit=Apply+%21
```

When the GET method is used, the data is passed by the server in the QUERY_STRING environment variable.

When the POST method is used, the CONTENT_LENGTH environment variable is set to the length of the data and the data is sent by the server to the standard input of the CGI program.

When choosing the method (GET or POST) for your application, consider the amount of data that will be passed. Because the GET method relies on an environment variable, the amount of data is limited. POST will consequently be preferred for large amounts of data.

The trend is to have the CGI program accepting both methods, relying on the REQUEST_METHOD environment variable to figure out which method was used to invoke the CGI.

Note that if you are using in an HTML page a link to a CGI script such as:

```
<A HREF=/cgi/yourscript?data>
```

your script is invoked using the GET method.

6.1.7.3 CGI Security Considerations

At this point of the discussion, we must warn you against the risks you have to face when you develop and install CGI scripts on your Web server.

The Web is like the real world, with its fair share of malevolent people who will take advantage of the first security hole in your server to break it down. No doubt; a single security mistake in a CGI can give a hacker complete access to your machine.

Let′s have another look at the Query Calendar Form and the qcal Perl script. The qcal script could be invoked by hand using the URL:

```
http://rs60004.itso.ral.ibm.com/cal-bin/qcal?month=09&year=1996
```

The HTML Form may be considered as an aid to users for building a URL based upon their own criteria. The problem is there is no way to prevent a user from submitting to the server a handmade URL.

As you can see in the listing of the qcal script in Figure 156 on page 237, no data validation is performed for the year and month parameters. This means that a malicious hacker looking for some hole in your CGI script could guess that your script is a Perl script running on a UNIX system and is simply passing the arguments to an existing UNIX command.

Any UNIX user knows that the ″;″ character is a command separator allowing two commands to be executed in sequence. Our hacker might therefore try to run the script file with the URL:

```
http://rs60004.itso.ral.ibm.com/cal-bin/qcal?month=02&year=1996;cat+/etc/passwd
```

The parsing of the year calendar by the:

```
$year = $parser -value year; chop( $year );
```

command would supply to the year variable the value:

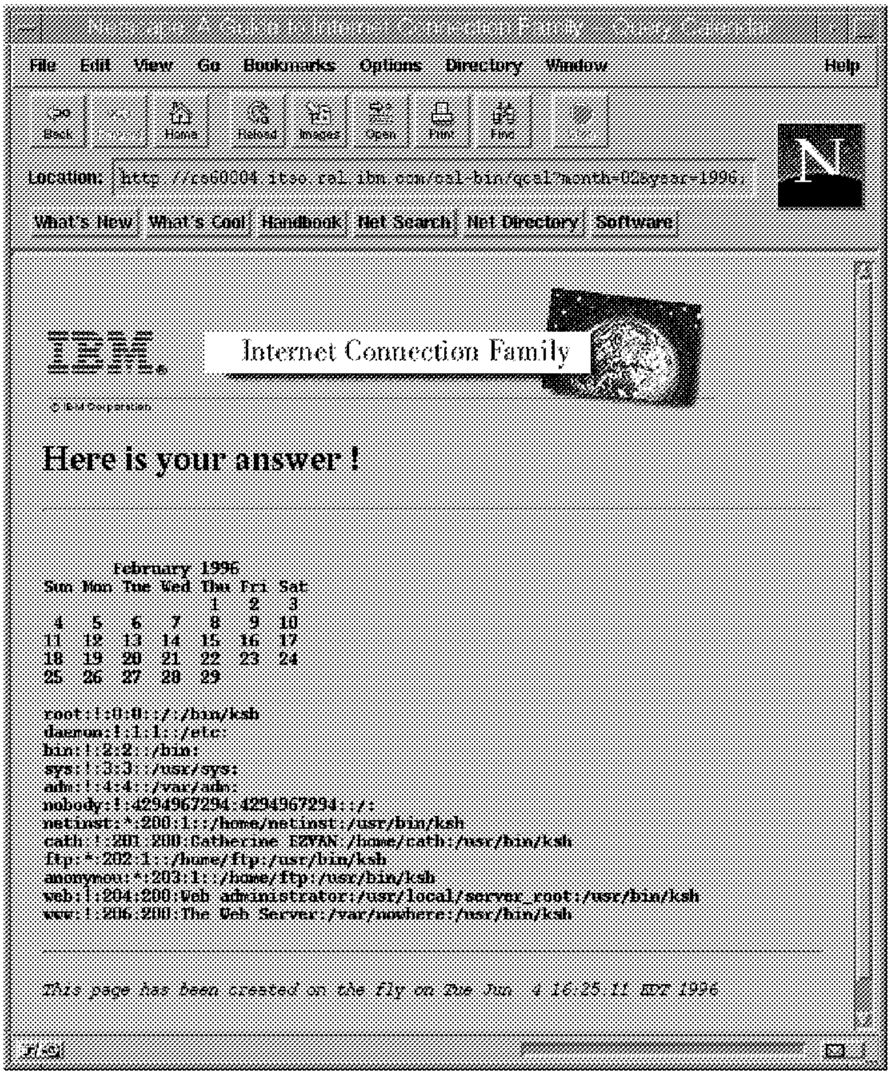
```
1996;cat /etc/passwd
```

Furthermore, cal could be abused as a result of the following command:

```
cal 02 1996;cat /etc/passwd
```

That is, the calendar for February 1996 immediately followed by a request for the contents of the /etc/passwd file. For people not familiar with UNIX, the cat command is equivalent to the DOS type command and writes to the standard output the files passed as arguments.

The resulting page would be as depicted in Figure 161.



http://rs60004.itso.ral.ibm.com/cal-bin/qcal?month=02&year=1996;

Here is your answer !

```
      February 1996
Sun Mon Tue Wed Thu Fri Sat
                  1   2   3
  4   5   6   7   8   9  10
 11  12  13  14  15  16  17
 18  19  20  21  22  23  24
 25  26  27  28  29

root:!:0:0::/:/bin/ksh
daemon:!:1:1::/etc:
bin:!:2:2::/bin:
sys:!:3:3::/usr/sys:
adm:!:4:4::/var/adm:
nobody:!:4294967294:4294967294::/:
netinst:*:200:1::/home/netinst:/usr/bin/ksh
cath:!:201:200:Catherine EZVAN:/home/cath:/usr/bin/ksh
ftp:*:202:1::/home/ftp:/usr/bin/ksh
anonymou:*:203:1::/home/ftp:/usr/bin/ksh
web:!:204:200:Web administrator:/usr/local/server_root:/usr/bin/ksh
www:!:206:200:The Web Server:/var/nowhere:/usr/bin/ksh
```

This page has been created on the fly on Tue Jun 4 16:25:11 EDT 1996

Figure 161. Misusage of CGI Script

Sending /etc/passwd is not as serious in AIX as it sounds since it does not contain the user passwords. But this shows you the kinds of doors that can opened by poor CGI scripts.

Clearly, you must perform data validation in your CGI scripts.

A secured Perl script for our example could be coded as shown in Figure 162 on page 247.

```
#!/usr/local/bin/perl

#
# qcal script
#
# This perl script is invoked with two arguments :
#       - month: decimal number (01-12) - month of year
#       - year:  decimal number - year with century
# It builds an HTML page including :
#       all required HTML tags (<HTML>, <HEAD>, <BODY>, etc
#       a H1 header
#       the calendar of the required month (as displayed by the Unix cal
#       command)
#
# The output of this script will be sent as is to the client (browser)
# this means that it must first print the header
#

# define variables
$parser=′/usr/lpp/internet/server_root/cgi-bin/cgiparse′;
$utils=′/usr/lpp/internet/server_root/cgi-bin/cgiutils′;

# parse arguments
$month = $parser -value month; chop( $month );
$year = $parser -value year; chop( $year );

$header = $utils -ct text/html;

# test arguments validity
$_ = $month;
if (  ! /[\d\d$/ ) {
        &outout;
}
$_ = $year;
if (  ! /[\d\d\d\d$/ ) {
        &outout;
}

$cal = cal $month $year;
$date = date;

print ″$header″;
print ″<HTML><HEAD><TITLE>″;
print ″A Guide to Internet Connection Family - Query Calendar″;
print ″</TITLE></HEAD><BODY>\n″;
print ″<IMG SRC=\″/img/masthead.gif\″″;
print ″ ALT=\″A Guide to Internet Connection Family\″>\n″;
print ″<H1>Here is your answer !</H1><HR>\n″;
print ″<PRE><B>\n″
print ″$cal\n″;
print ″</B><HR>\n″;
print ″<I>This page has been created on the fly on $date</I>\n″;
print ″</PRE></BODY></HTML>\n″;
```

```
# routine
sub outout {

        print ”$header”;
        print ”<HTML><HEAD><TITLE>Out out !!</TITLE></HEAD>”;
        print ”<BODY><H1><IMG SRC=\”/gifs/shock.gif-”> OUT OUT !!</H1>-n”;
        print ”</BODY></HTML>\n”;
        exit;
}
```

Figure 162 (Part 2 of 2). Secured Perl Script

This script verifies the validity of the arguments: month must be a two digit number and year a four digit number.

If invalid arguments are passed to the CGI script, the returned HTML page will send a special HTML page informing the client that it noticed the misusage of the script (refer to outout routine in Figure 162 on page 247).

Another piece of advice is, when possible, to avoid giving to your scripts names that will inform the user of the nature of the script. Avoid naming a Perl script myscript.pl or a shell script myscript.sh, for example.

We leave you to imagine what a hacker would have been able to do if he or she managed to break into the Web server using the security exposure described earlier. We cannot stress enough that the server should be run with as low a level of privileges as possible; only what is needed to read the documents and run the scripts, not more.

6.1.7.4 Customizing the Internet Connection Server

In this example, we organized our files in the following manner:

- We stored the HTML page, qcal.html in:

  `/usr/local/server_root/cal`

- We stored the Perl script in:

  `/usr/local/server_root/cal/cgi`

As a consequence, the following directives were required in our configuration file:

```
Exec    /cal-bin/* /usr/local/server_root/cal/cgi/*
Pass    /cal/* /usr/local/server_root/cal/*
```

6.1.8 Configuration and Administration Forms

The Internet Connection server package for most platforms (OS/2 Warp, Windows NT, AIX, etc.) includes several CGI programs (executable binaries). These are:

- Htimage which handles clickable images
- Cfgin and cfgout which are used by the Configuration and Administration Forms for general administration tasks
- Larin and larout which are used for Logging And Reporting Forms
- Sctyin and sctyout which are used to set up security for the server (available with Internet Connection Secure Servers)

- Db2in and db2out which are used by the administration procedures for the DB2 Gateway (only for platforms which support the DB2 Gateway).

Htimage is discussed in 5.9.1, “Clickable Image Maps - Shapes” on page 196. Note that in Internet Connection Server V4.1 htimage is shipped as a program which relies on the IBM ICAPI.

In this section, we discuss how administration and configuration procedures are implemented using CGI programs in the Internet Connection server. See 4.6, “Configuration” on page 96 to read more about configuring your server using Administration and Configuration tasks.

6.1.8.1 Administration and Configuration Interface

Immediately after installing Internet Connection Server on your system, the home page of your brand new Web server gives you a link to the Configuration and Administration Forms main page.

Because configuration and administration tasks are not supposed to be done by any client, you certainly should not keep this link in any of your HMTL pages.

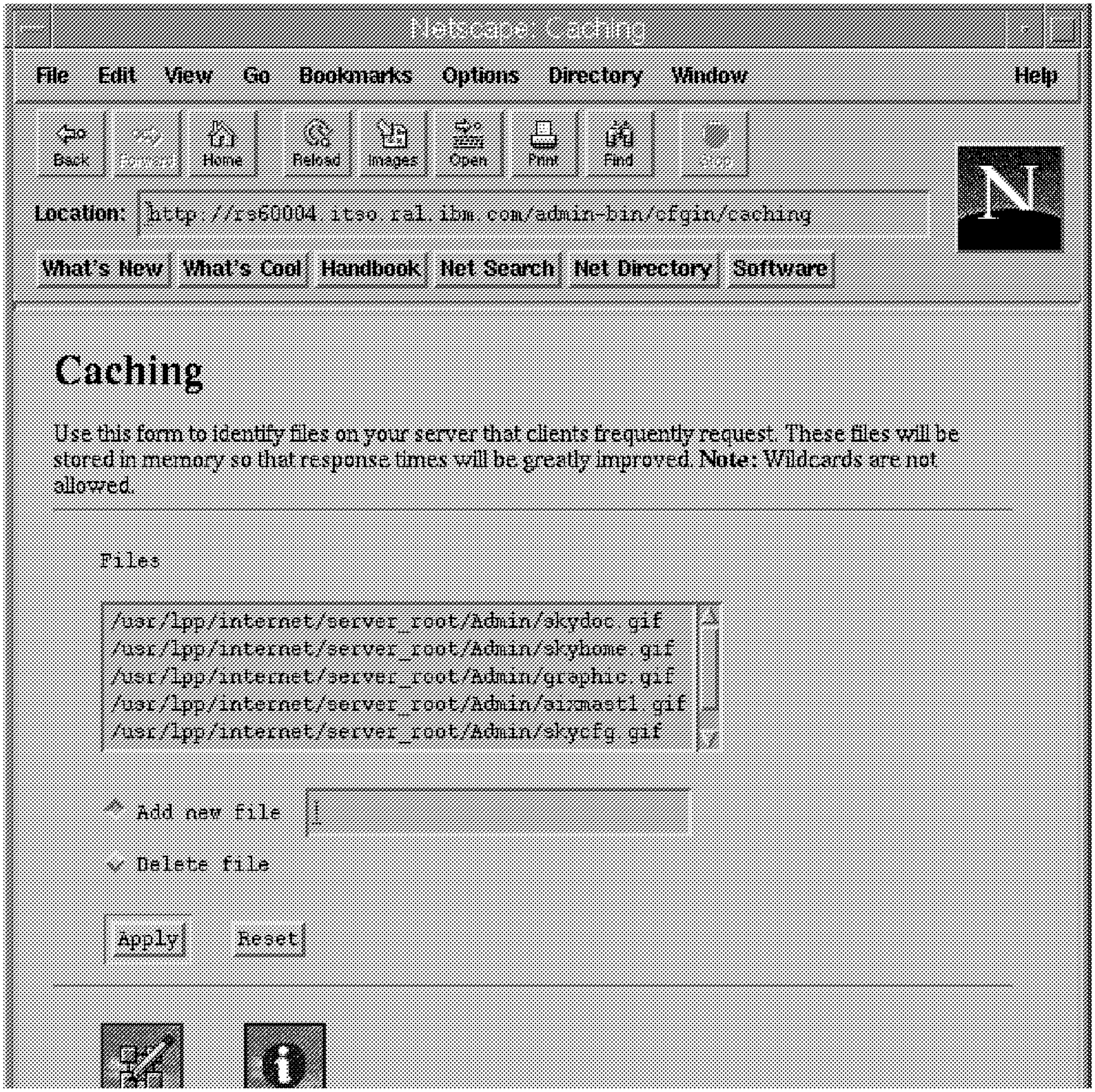
However, except if you changed or deleted the directives related to administration in your configuration file, you are still able to access the Configuration and Administration Forms using the URL:

```
http://<your_system_name>/admin-bin/cfgin/initial
```

at least from any authorized hosts.

We see in section 6.1.8.2, “Administration and Configuration Forms Implementation” on page 252 which directives are required in the configuration file for the Configuration and Administration Forms to work properly.

From the main page of the Configuration and Administration Forms, you can link to each of the input forms. A click on **Caching**, for example, takes you to the caching form, as shown in the following figure.


Netscape: Caching
File Edit View Go Bookmarks Options Directory Window Help
Back Forward Home Reload Images Open Print Find Stop
Location:
What's New What's Cool Handbook Net Search Net Directory Software

http://rs60004.itso.ral.ibm.com/admin-bin/cfgin/caching

Caching

Use this form to identify files on your server that clients frequently request. These files will be stored in memory so that response times will be greatly improved. **Note:** Wildcards are not allowed.

Files

```
/usr/lpp/internet/server_root/Admin/skydoc.gif
/usr/lpp/internet/server_root/Admin/skyhome.gif
/usr/lpp/internet/server_root/Admin/graphic.gif
/usr/lpp/internet/server_root/Admin/aixmast1.gif
/usr/lpp/internet/server_root/Admin/skycfg.gif
```

● Add new file

○ Delete file

Apply Reset

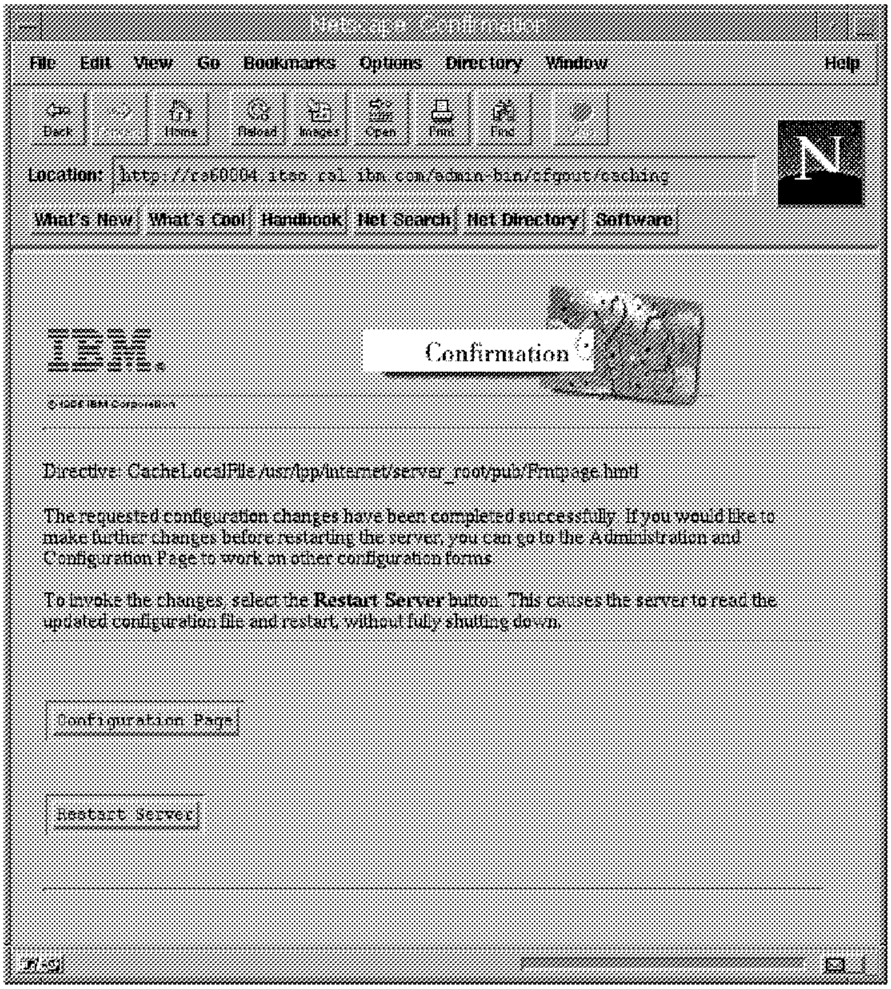
Figure 163. Caching Setup Form

The new URL is:

```
http://<your_system_name>/admin-bin/cfgin/caching
```

If you fill in the form and click on **Apply**, the server runs a CGI program and shows you a message contained in a confirmation page indicating that your input was accepted.

The confirmation page is shown below.


Netscape: Confirmation
File Edit View Go Bookmarks Options Directory Window Help
Back Forward Home Reload Images Open Print Find Stop
Location: http://rs60004.itso.ral.ibm.com/admin-bin/cfgout/caching
What's New What's Cool Handbook Net Search Net Directory Software
IBM
Confirmation
Configuration Page
Restart Server

Directive: CacheLocalFile /usr/lpp/internet/server_root/pub/Frntpage.html

The requested configuration changes have been completed successfully. If you would like to make further changes before restarting the server, you can go to the Administration and Configuration Page to work on other configuration forms.

To invoke the changes, select the **Restart Server** button. This causes the server to read the updated configuration file and restart, without fully shutting down.

Figure 164. Caching Setup Confirmation Page

You are now located on:

```
http://<your_system_name>/admin-bin/cfgout/caching
```

If you take a close look at all of these URLs, you will find out that they all rely on cfgin, cfgout, larin, larout, sctyin and sctyout which are the administration CGI programs.

The way the CGI programs are invoked differs here from the ways we have seen before. In addition to QUERY_STRING (for GET method) and stdin (for POST method), arguments may be passed to the CGI program by an additional relative path information in the URL. This additional information comes after the script name but before any query data. For example, in:

```
http://rs60004.itso.ral.ibm.com/admin-bin/cfgin/basic
```

the additional relative path information is:

```
/basic
```

This additional information is passed to your CGI program in the PATH_INFO environment variable.

Therefore, any Internet Connection server′s Administration Form can be reached directly by passing the right parameter in the URL. Furthermore, if you do not specify any subcommand by means of this parameter, the CGI program will return to you an HTML page with the following error message: `Environment variable PATH_INFO not set`.

6.1.8.2 Administration and Configuration Forms Implementation

In this section we discuss where the files required by the Administration and Configuration Forms are stored and which directives are required in your configuration file to have the tool working properly.

The location of the CGI programs and of the documents they rely on (HTML pages and GIF files) differ between the different platforms.

AIX Platform: After the installation of the Internet Connection server on an AIX platform, the CGI programs and the files related to administration and configuration tasks are located according to the following scheme:

- /usr/lpp/internet/server_root/cgi-bin contains htimage (htimage.so in Version 4.1).
- /usr/lpp/internet/server_root/admin-bin contains:
  - cfgin, cfgout,
  - larin, larout (in Version 4.1)
  - sctyin, sctyout (in secure versions)
  - db2in, db2out
- /usr/lpp/internet/server_root/Admin contains the HTML pages and the GIF files related to the administration.

Consequently, the following directives are required in the configuration file:

```
Exec /cgi-bin/* /usr/lpp/internet/server_root/cgi-bin/*
Exec /admin-bin/* /usr/lpp/internet/server_root/admin-bin/*
Pass /Admin/* /usr/lpp/internet/server_root/Admin/*
```

In the case of Version 4.1 of the product, the following statements are also required in the configuration file:

```
Service /cgi-bin/htimage* /usr/lpp/internet/server_root/cgi-bin/htimage.so:HTImage*
Service /cgi-bin/imagemap* /usr/lpp/internet/server_root/cgi-bin/htimage.so:HTImage*
```

This is due to the fact that in this version, the htimage is no longer a CGI program but is in fact an IBM ICAPI program (see 6.2, “API Applications” on page 274).

OS/2 and Windows NT Platforms: After the installation of the Internet Connection server on an OS/2 or a Windows NT platform, the CGI programs and the files related to administration and configuration tasks are located according to the following scheme:

- $WWWDIR\CGI-BIN contains HTIMAGE.EXE (HTIMAGE.DLL in Version 4.1).
- $WWWDIR\ADMIN contains:
  - Cfgin, cfgout,
  - Larin, larout (in Version 4.1)
  - Sctyin, sctyout (in secure versions)
  - Db2in, db2out
  - The HTML pages and GIF files related to the administration

where $WWWDIR is the directory you chose for installing the product.

Consequently, if the product was installed in D:\WWW directory, the following directives are required in the configuration file:

```
Exec /cgi-bin/* D:\WWW\CGI-BIN\*
Exec /admin-bin/* D:\WWW\ADMIN\*
Pass /Admin/* D:\WWW\ADMIN\*
```

In the case of Version 4.1 of the product, the following statements are also required in the configuration file:

```
Service /cgi-bin/htimage* D:\WWW\CGI-BIN-htimage:HTImage*
Service /cgi-bin/imagemap* D:\WWW\CGI-BIN-htimage:HTImage*
```

This is due to the fact that in this version of the product, the htimage is no longer a CGI program but is in fact an IBM ICAPI program (see 6.2, "API Applications" on page 274).

6.1.9 OS/400 CGI - ITSO Company

In this section, we look at how the CGI is implemented on the OS/400 platform. We provide an example based upon an imaginary ITSO company but, prior to that, we look at some OS/400 CGI specifics. Many of the points discussed are also relevant to CGI programming on other platforms.

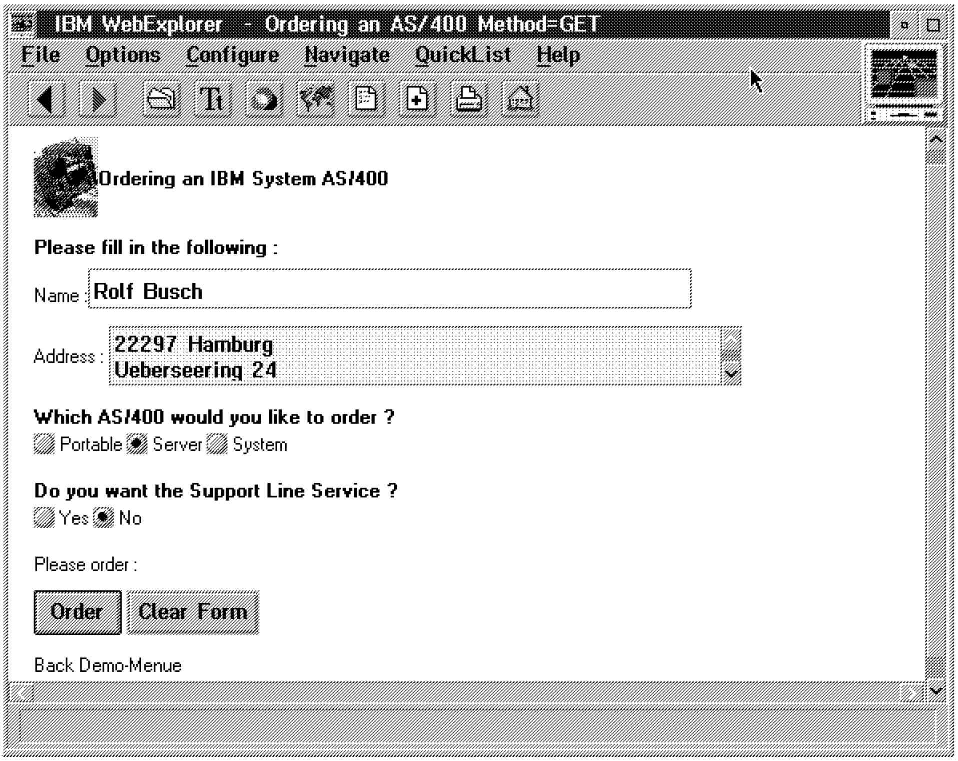
6.1.9.1 CGI Programming

For accessing a CGI program, you have to use the FORM tag in your HTML document. Figure 165 on page 254 shows an example on how to write an HTML document with the FORM tag.

```
<html>
<head>
<title>Ordering an AS/400 Method=GET</title>
</head>
<body bgcolor=”#F8F8FF”>
<img align=middle src=”as400.gif”>
<b>    Ordering an IBM System AS/400 </b>
<p>
<b>Please fill in the following :</b>
<form method=”GET” action=”/BonusCGI/ORDAS400G.PGM”>
<INPUT type=”hidden” name=”MBR” value=”ORDAS400E ” size=10>
Name :
<input type=text name=”NAME” size=”30” maxlength=”40” clear=”all”>
<p>
Address :
<textarea name=”ADDRESS” cols=30 rows=2 VALUE=” ”>
</textarea>
<p>
<b>Which AS/400 would you like to order ?</b>
<br>
<input name=”TYPE” value=”P1” type=radio checked>Portable
<input name=”TYPE” value=”P2” type=radio>Server
<input name=”TYPE” value=”P3” type=radio>System
<p>
<b>Do you want the Support Line Service ?</b>
<br>
<input name=”SERV” value=”T” type=radio checked>Yes
<input name=”SERV” value=”F” type=radio>No
<p>
Please order :
<p>
<input type=submit VALUE=”Order”>
<input type=reset VALUE=”Clear Form”>
</form>
</P>
<A HREF=”/web/DEMO2E.MBR”>Back</A> Demo-Menue
```

Figure 165. The HTML Source Using the FORM Tag

When the Web browser displays the HTML document, it is similar to the following figure.



Ordering an IBM System AS/400

Please fill in the following :

Name : Rolf Busch

Address : 22297 Hamburg
Ueberseering 24

Which AS/400 would you like to order ?
○ Portable ● Server ○ System

Do you want the Support Line Service ?
○ Yes ● No

Please order :

Back Demo-Menue

Figure 166. The HTML Document Using the FORM Tag Displayed by the Browser

A Form basically consist of three elements:

1. The Form method:

   ```
   <FORM METHOD="GET" ACTION="/BonusCGI/ORDAS400G.PGM">
   ```

   or

   ```
   <FORM METHOD="POST" ACTION="/BonusCGI/ORDAS400P.PGM">
   ```

2. The Input tags:

   ```
   text, textarea, checkbox, radiobutton, options...
   ```

3. Submit button

All three elements are used in Figure 165 on page 254. Note that the action keyword contains the URL address to send the input data to when the Submit button is selected. In this case, action="/BonusCGI/ORDAS400x.PGM".

A match with an EXEC directive in our HTTP configuration enables the HTTP server to execute a CGI program on behalf of the client.

In our case, action="/BonusCGI/ORDAS400x.PGM" is mapped with:

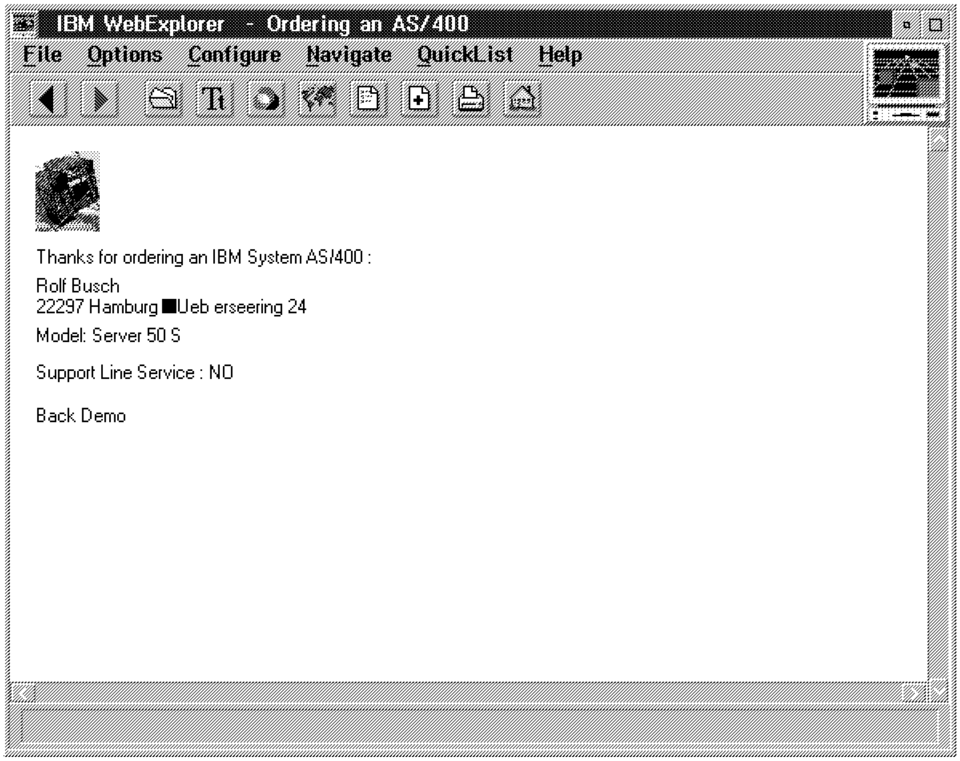
```
EXEC /BonusCGI/*  /QSYS.LIB/ITSCOIC400.LIB/*
```

Note

Please note that only programs in the QSYS.LIB library may be the target for the input data.

This is because only programs in the QSYS.LIB File System may be executed on the AS/400 system.

The following figure shows the HTML output after the execution of the CGI-BIN program.


IBM WebExplorer - Ordering an AS/400
File Options Configure Navigate QuickList Help

Thanks for ordering an IBM System AS/400 :

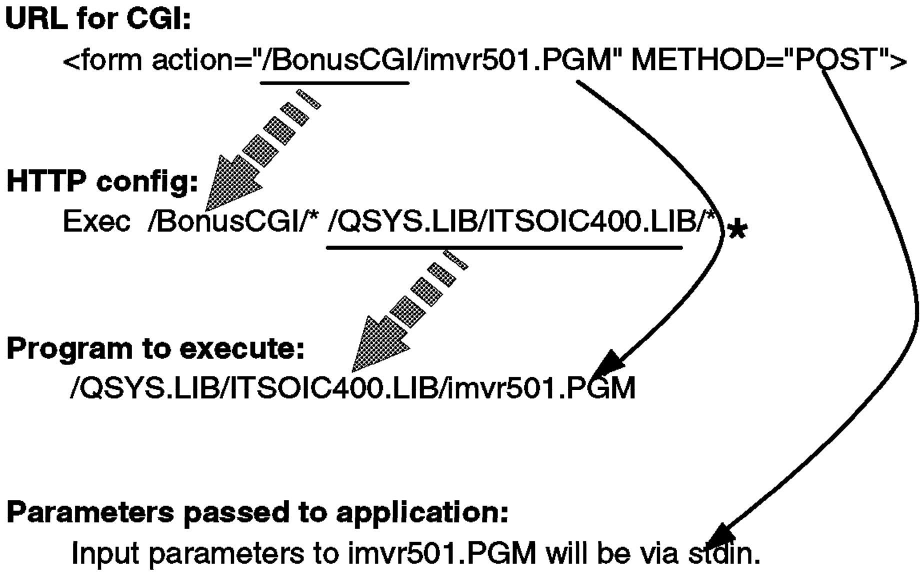
Rolf Busch
22297 Hamburg ■Ueb erseering 24

Model: Server 50 S

Support Line Service : NO

Back Demo

Figure 167. The HTML Output Information Back on the Browser



URL for CGI:
<form action="/BonusCGI/imvr501.PGM" METHOD="POST">

HTTP config:
Exec /BonusCGI/* /QSYS.LIB/ITSOIC400.LIB/* *

Program to execute:
/QSYS.LIB/ITSOIC400.LIB/imvr501.PGM

Parameters passed to application:
Input parameters to imvr501.PGM will be via stdin.

Figure 168. Mapping of CGI URL to HTTP Configuration

There are two methods that can be used to access your forms. Depending on which method you use, you receive the encoded results of the form in a different way. The two methods are:

Method	**Description**
GET	If your form in the HTML document has METHOD=″GET″ in its FORM tag, your CGI program receives the encoded form input in the environment variable QUERY_STRING.
POST	If your form in the HTML document has METHOD=″POST″ in its FORM tag, your CGI program receives the encoded form input on stdin. The server does *not* send you an End of File (EOF) on the end of the data. Instead, you must use the environment variable CONTENT_LENGTH to determine how much data you should read from stdin.

Note

POST is often the preferred method of operating with CGI-BIN programming.

The reason why POST is preferred is that GET uses the environment variable QUERY_STRING.

Some server platforms are limited in the length of environment variables.

A form with a lot of data could cause a loss of data.

In the POST method, there are no limits. (All data is passed through STANDARD INPUT, a way of transferring information in a ″stream″ without limits.)

As you can see, the form method determines the way of passing the encoded input data from the browser to the CGI program through the server. The server and the CGI program communicates in four major ways, all supported by the AS/400 system:

Environment variables	Used for the GET method.
The command line	Used in the case of an ISINDEX query.
Standard input	Used for the POST method.
Standard output	Used to produce output data.

The AS/400 HTTP server provides the means whereby called CGI programs can access environment variables and standard input stream (stdin). The server also provides the means for a CGI program to return data in the standard output stream.

Request for programs in an HTML document can be made using either the GET or the POST method. See Figure 169 on page 258 and Figure 170 on page 260.

Note This!

The AS/400 system runs CGI programs under the QTMHHTP1 user profile. The QTMHHTP1 user profile *must* have authority to access all the objects accessed by the CGI programs.

GET

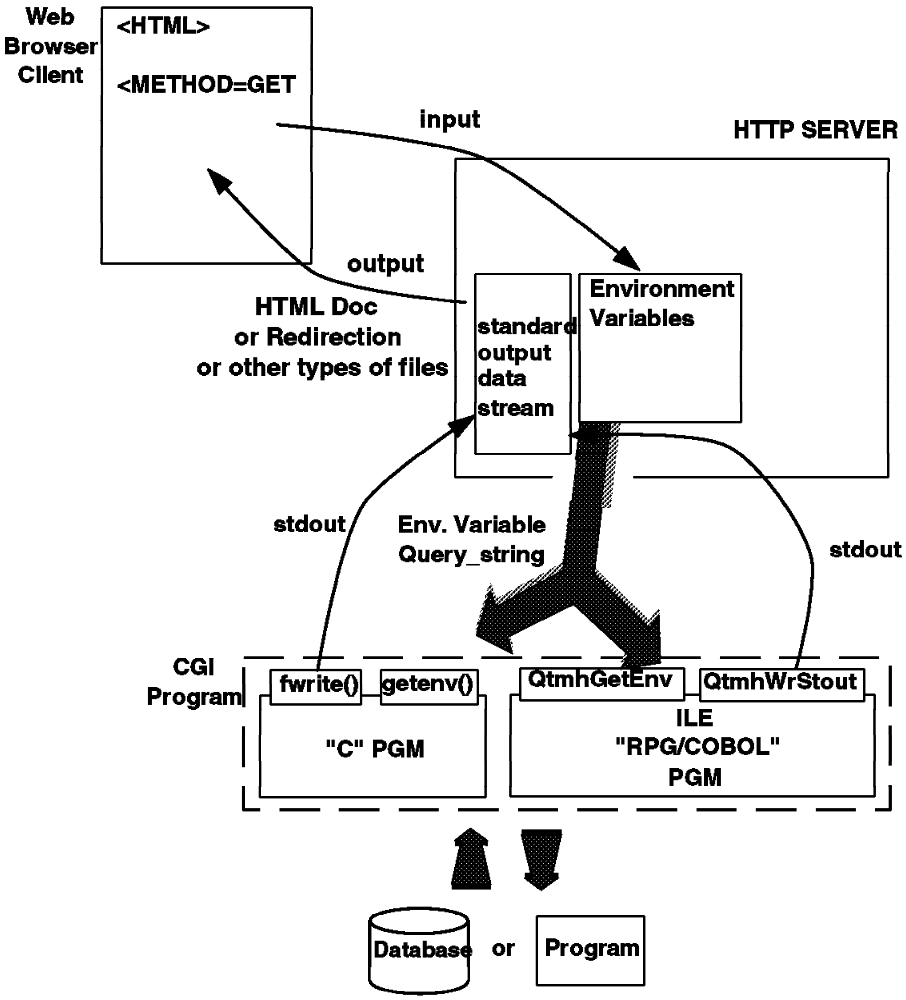


Figure 169. AS/400 CGI Using the GET Method

The HTML document displayed by the browser uses <FORM METHOD=″GET″.

When the GET method is used, the input parameters are passed to the called CGI program in one of two ways:

1. In command line parameters (program call parameters):

   Parameters are passed to the CGI program using command line arguments (program call parameters) only when the data in the request URL beyond the first question mark (?) contains *no* equal signs (=). This is called an ISINDEX. In the URL for an ISINDEX request, the CGI parameters are positional (being separated by plus (+) signs). The server parses the parameters and puts each parameter, according to its position in the string, into corresponding parameters of the CGI program call.

   An example of a URL is:

   ```
   http://sysnm003.sysnmaa.ibm.com/CGILIB/INDEX.PGM?value1+value2+value3
   ```

2. In the environment variable QUERY_STRING:

If the parameter string of the URL beyond the first question mark (?) contains an equal sign (=), the entire parameter string is put into the QUERY_STRING environment variable. This is the most normal case.

An example for the URL is:

```
http://sysnm003.sysnmaa.ibm.com/CGILIB/FORM.PGM?kwd1=value1
       &kwd2=value2&kwd3=value3
```

Note

The assumption is made that CGILIB has been mapped to a library in QSYS.LIB containing CGI programs with the MAP and Exec directives for the AS/400 HTTP server.

The mapping directives allow the Web Administrator on the AS/400 system to define a virtual hierarchy of Web resources that is presented by the AS/400 HTTP server. This virtual hierarchy is independent of the physical file system (or systems) on which the resources are actually stored. This is very useful since:

1. It allows resources to be physically relocated without changing the apparent hierarchy and its implied associations.
2. The details of underlying physical file systems can be hidden from client applications that are accessing the AS/400 HTTP server.

The second reason is especially important for the AS/400 HTTP server because of the differences between some of the AS/400 file systems and the tree structured hierarchical UNIX file systems upon which the HTTP protocol and the URL scheme are based.

The mapping directives, MAP and Exec, are used to specify a set of rules that define a process for translating and processing the path (and file) information contained in a URL that is received in a client request URL.

Each of the mapping directives may be specified multiple times and in any combination, and may be placed anywhere in the AS/400 HTTP server configuration. When the AS/400 HTTP server configuration is read during initialization, the mapping directives are stored in order of appearance in a mapping rule table.

The incoming URL is compared against each of the specified mapping rules in turn. If a directive applies, the specified translation is carried out. Depending on the rule that was matched, rule processing is either suspended or continues with the next rule using the newly translated URL.

The mapping and the Exec definitions are created through the WRKHTTPCFG command.

POST: When the POST method is being used, the QUERY_STRING environment variable contains a null string. The null string makes the CGI program look for its input in the standard input stream. See Figure 170 on page 260. The CGI program can determine the method, GET or POST, from the REQUEST_METHOD environment variable.

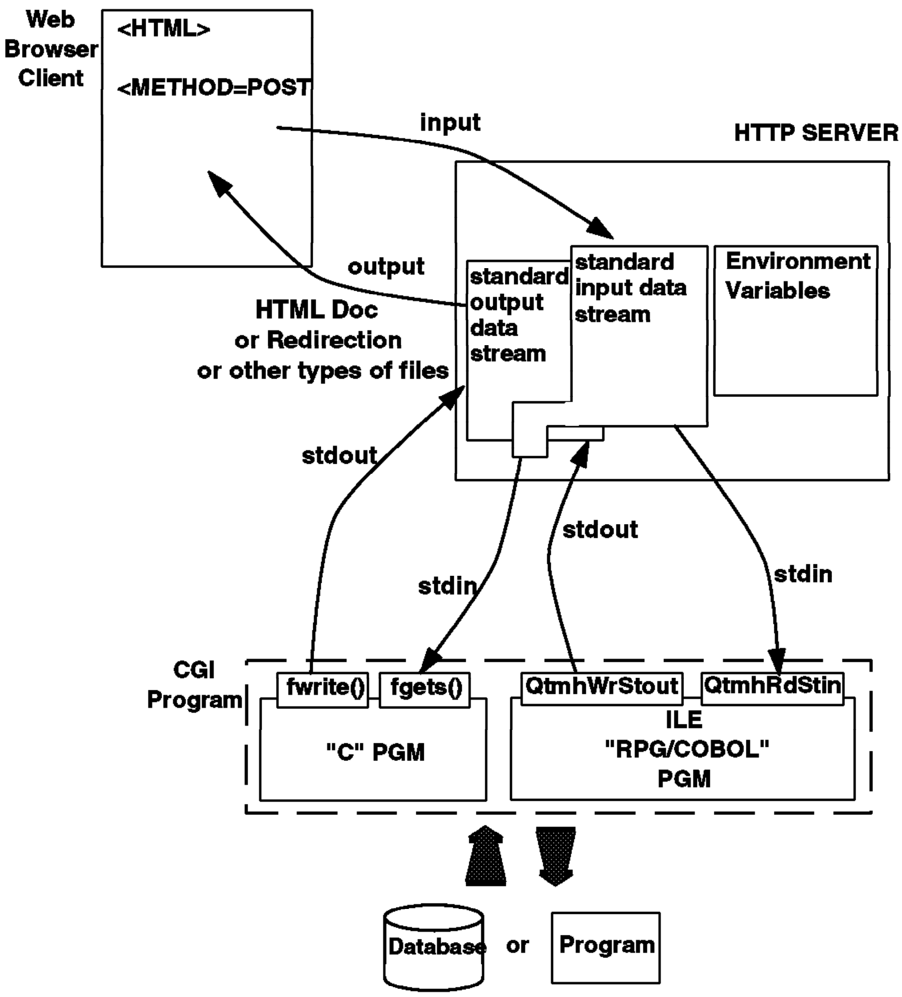


Figure 170. AS/400 CGI Using the POST Method

When the HTML document uses the <FORM METHOD=″POST″, the form is submitted with the standard input data stream, stdin. The CGI program inspects the request method to determine if post data is available by reading the REQUEST_METHOD environment variable. The CGI program can obtain other environment variables set by the HTTP server regardless of the request method.

In order to support environment variables, the AS/400 HTTP server creates an environment variable user index that contains the name and value for each environment variable set by the server. The index is named QTMHENVI in the QTEMP library.

The reason why it is created in the QTEMP library is that it has to be unique for each child job that defines it. So each child job creates the index that is accessible to the CGI programs that the server child job calls.

Each time a CGI program is called, a new index is created and new environment variables are set (preventing one CGI program from using the environment variables intended for another CGI program). Creation of the index is not dependent on the use of either method, GET or POST. The index is always created.

The QTMHENVI index contains the following environmental variables:

Table 11. CGI Environment Variables in the QTMHENVI Index File

Environment Variable	**Contain**
QUERY_STRING	Information that follows the first question mark (?) in the URL referencing the CGI program.
SERVER_SOFTWARE	The name and version of the information server software answering the request (and running the gateway).
SERVER_NAME	Host name, DNS alias, or IP address of the server.
GATEWAY_INTERFACE	The version of the CGI specification to which the server complies.
SERVER_PROTOCOL	The name and revision of the information protocol this request came in with.
SERVER_PORT	The port number to which the request was sent.
REQUEST_METHOD	The method with which the request was made. For HTTP, this is GET or POST.
PATH_INFO	The extra path information following the path information required to identify the CGI program name.
PATH_TRANSLATED	The server provides a translated version of PATH_INFO, which takes the path and does any virtual-to-physical mapping to it.
SCRIPT_NAME	A virtual path to the program being executed.
REMOTE_HOST	The host name making the request.
REMOTE_ADDR	The IP address of the host name making the request.
REMOTE_USER	This is the user name making the request.
CONTENT_TYPE	For queries that have attached information, such as HTTP POST, this is the content type of the data.
CONTENT_LENGTH	The length of data in the attached HTTP POST from the client.
IBM_CCSID_VALUE	The CCSID under which the current server job is running.

When the CGI program has finished executing, it is expected to return an HTML document or a redirection to an HTML document.

The CGI program passes output from other programs or databases plus any additional information back to the server. To do that, the CGI program uses the standard output data stream, stdout.

There is support for CGI programs written in:

- C language
- ILE RPG
- ILE COBOL

Necessary APIs are available to support the environment variables, the standard input stream, and the standard output stream.

The C language supports the environment variables through the getenv() API. The standard input stream and the standard output stream are supported through many APIs. Examples are fgets() and fwrite().

For the C programs, the APIs are found in the QTMHCGI *SVCPGM. A C language header file member named QTMHCGI is provided in H file in library QSYSINC.

The APIs to support ILE RPG and ILE COBOL are provided in source files. The source files are:

- QCBLLESRC for ILE COBOL
- QRPGLESRC for ILE RPG

Four APIs are provided as shown in Table 12.

Table 12. The APIs Provided for ILE RPG, ILE COBOL, and C Programs

Type of API	ILE RPG	ILE COBOL	C Language
Get Environment Variable	QtmhGetEnv	QtmhGetEnv	getenv()
Read from Stdin	QtmhRdStin	QtmhRdStin	Many, for example fgets()
Write to Stdout	QtmhWrStout	QtmhWrStout	Many, for example fwrite()
Parse CGI input	QtmhCvtDB	QtmhCvtDB	#pragma mapinc

Note This for RPG

For the CRTPGM, you *must* always specify the BNDSRVPGM QTCP/QTMHCGI to ensure they satisfy any module import request for QtmhGetEnv, QtmhRdStin, QtmhWrStout, and QtmhCvtDB.

The APIs are provided to simplify parsing form data from either stdin or from the environment variable QUERY_STRING. The CGI program is expected to provide the name of the DDS file specification that identifies the anticipated form variables and their attributes.

The server times the operation CGI programs and terminates child server jobs exceeding the time limit. There are three timeout keywords for input timeout, output timeout, and script timeout.

Also, the server protects itself from CGI program exceptions by providing exception handling that reduces any escape messages resulting from CGI program failures to diagnostic messages.

6.1.9.2 Examples for Environment Variables

On the following pages are some inquiry examples for environment variables.

Environment Variables with GET

Figure 172 on page 264 shows the inquiry results with the Method=GET.

The environment variable QUERY_STRING contains the name/value pairs.

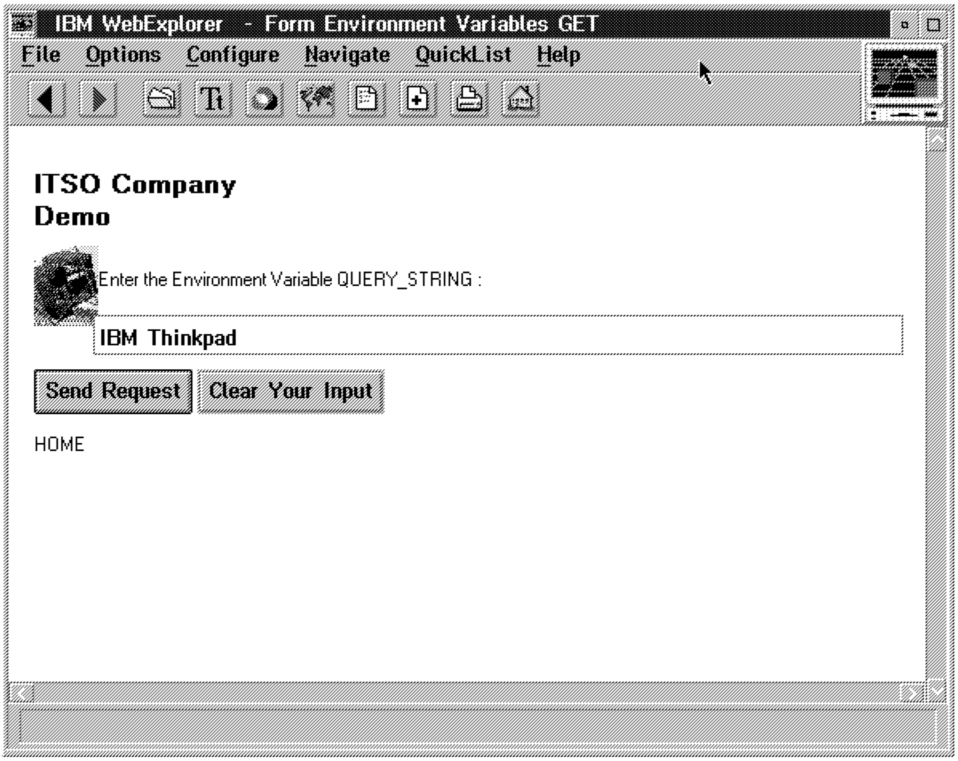
Environment Variables with POST

Figure 174 on page 265 shows the inquiry results with the Method=POST.

The environment variable QUERY_STRING is empty.

POST always reads from STDIN.

The following figure shows the form for program CGIENVGET and environment variables with METHOD=GET.


IBM WebExplorer - Form Environment Variables GET
File Options Configure Navigate QuickList Help

ITSO Company
Demo

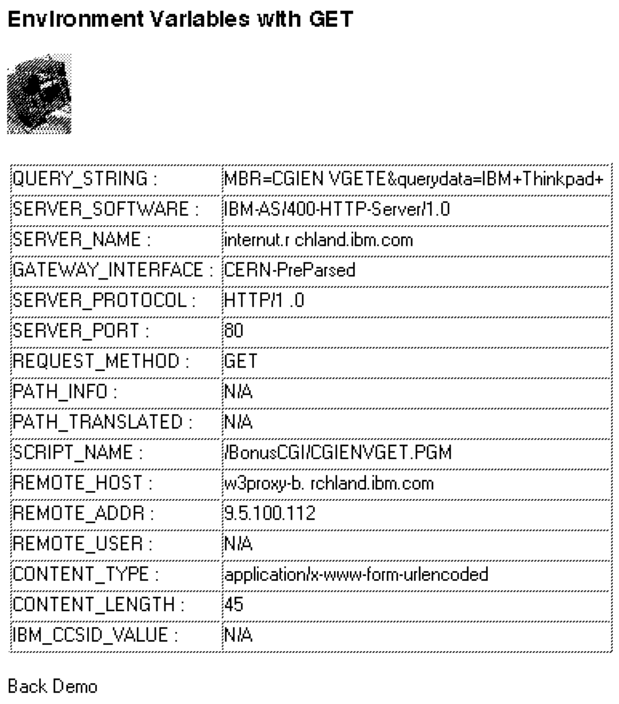
Enter the Environment Variable QUERY_STRING :

IBM Thinkpad

Send Request
Clear Your Input
HOME

Figure 171. The HTML Document Using the FORM Tag with Method=GET

The following figure shows the HTML output from program CGIENVGET.



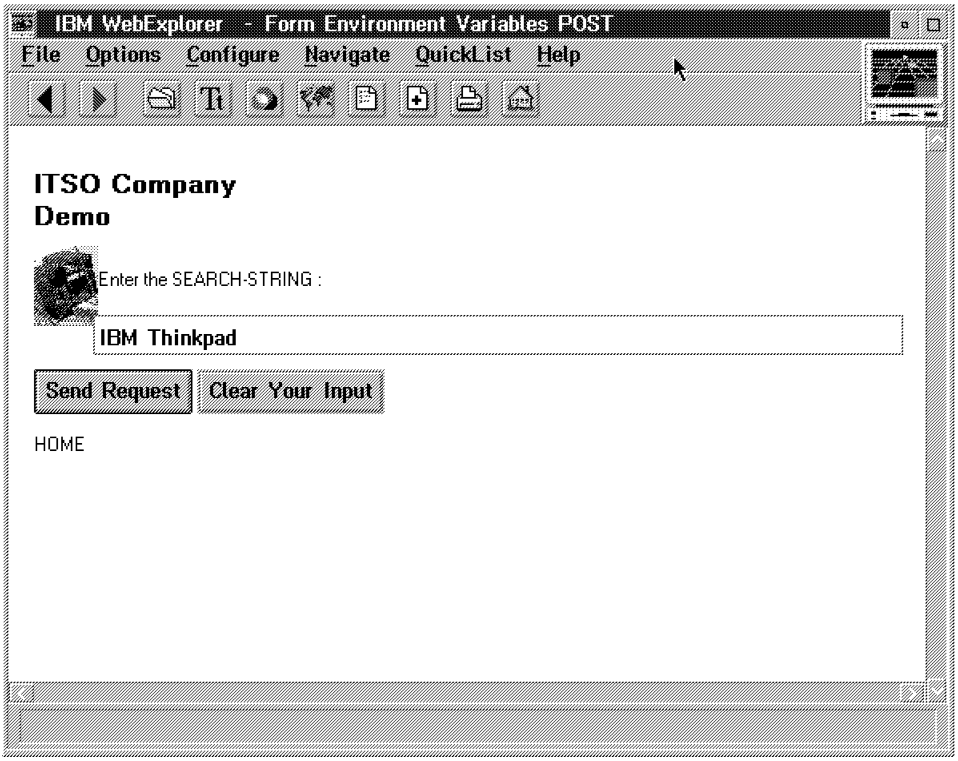
Environment Variables with GET

QUERY_STRING :	MBR=CGIEN VGETE&querydata=IBM+Thinkpad+
SERVER_SOFTWARE :	IBM-AS/400-HTTP-Server/1.0
SERVER_NAME :	internut.r chland.ibm.com
GATEWAY_INTERFACE :	CERN-PreParsed
SERVER_PROTOCOL :	HTTP/1 .0
SERVER_PORT :	80
REQUEST_METHOD :	GET
PATH_INFO :	N/A
PATH_TRANSLATED :	N/A
SCRIPT_NAME :	/BonusCGI/CGIENVGET.PGM
REMOTE_HOST :	w3proxy-b. rchland.ibm.com
REMOTE_ADDR :	9.5.100.112
REMOTE_USER :	N/A
CONTENT_TYPE :	application/x-www-form-urlencoded
CONTENT_LENGTH :	45
IBM_CCSID_VALUE :	N/A

Back Demo

Figure 172. Document Returned from the Program CGIENVGET

The following figure shows the form for program CGIENVPOST and environment variables with METHOD=POST.



ITSO Company Demo

Enter the SEARCH-STRING :

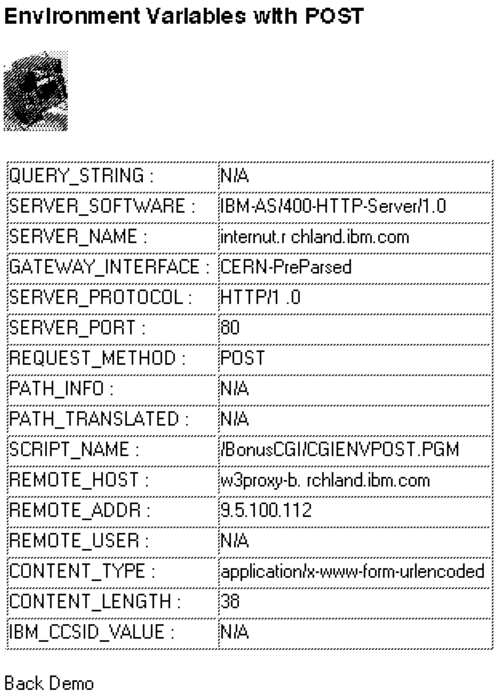
IBM Thinkpad

Send Request | Clear Your Input

HOME

Figure 173. The HTML Document Using the FORM Tag with Method=POST

The following figure shows the HTML output from program CGIENVPOST.



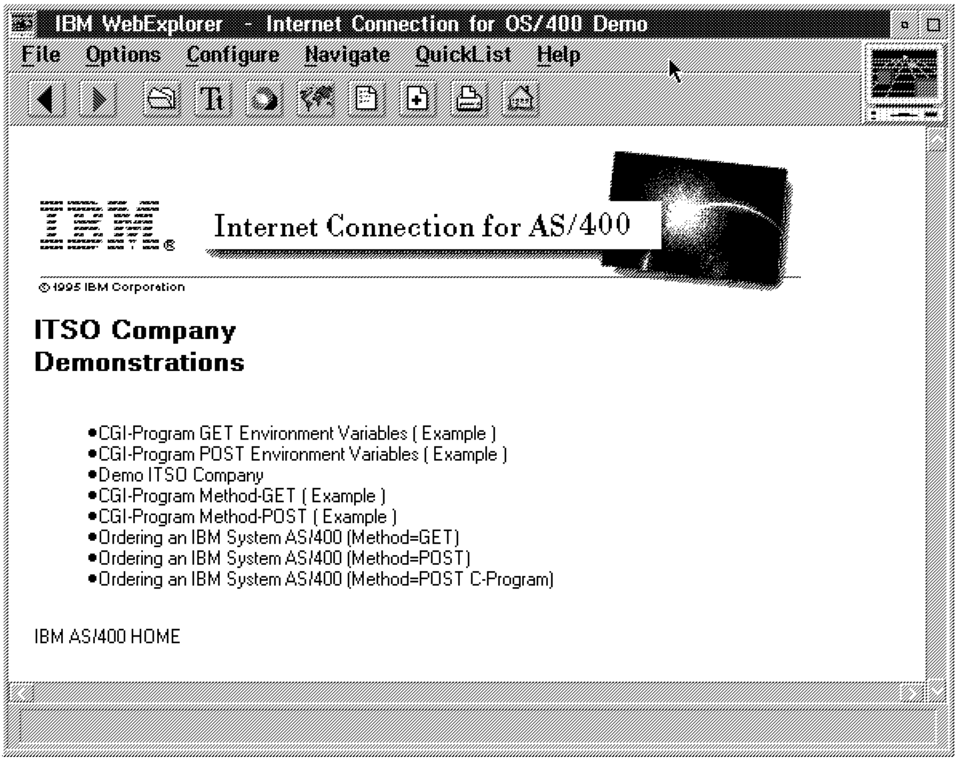
Environment Variables with POST

QUERY_STRING :	N/A
SERVER_SOFTWARE :	IBM-AS/400-HTTP-Server/1.0
SERVER_NAME :	internut.r chland.ibm.com
GATEWAY_INTERFACE :	CERN-PreParsed
SERVER_PROTOCOL :	HTTP/1 .0
SERVER_PORT :	80
REQUEST_METHOD :	POST
PATH_INFO :	N/A
PATH_TRANSLATED :	N/A
SCRIPT_NAME :	/BonusCGI/CGIENVPOST.PGM
REMOTE_HOST :	w3proxy-b. rchland.ibm.com
REMOTE_ADDR :	9.5.100.112
REMOTE_USER :	N/A
CONTENT_TYPE :	application/x-www-form-urlencoded
CONTENT_LENGTH :	38
IBM_CCSID_VALUE :	N/A

Back Demo

Figure 174. Document Returned from the Program CGIENVPOST

6.1.9.3 ITSO Company Demonstrations



IBM Internet Connection for AS/400

© 1995 IBM Corporation

ITSO Company Demonstrations

- CGI-Program GET Environment Variables (Example)
- CGI-Program POST Environment Variables (Example)
- Demo ITSO Company
- CGI-Program Method-GET (Example)
- CGI-Program Method-POST (Example)
- Ordering an IBM System AS/400 (Method=GET)
- Ordering an IBM System AS/400 (Method=POST)
- Ordering an IBM System AS/400 (Method=POST C-Program)

IBM AS/400 HOME

Figure 175. Demonstrations Welcome to the ITSO Company

All programs, files, HTMLs and source files are located in the library ITSOIC400. The images (GIFs) are in the directory ITSOIC.400.

Table 13. Overview Demo - Programs HTMLFILE=INPUT

Program	Function	Files	HTML-Mbr	HTMLO-Mbr	Language
CGIENVGET	Environment Vars GET	ENVFILE ENVGETDS HTMLO	FORMGETE	CGIENVGETE	ILE RPG
CGIENVPOST	Environment Vars POST	ENVFILE ENVPOSTDS HTMLO	FORMPOSE	CGIENVPOSE	ILE RPG
CGIGET	Search Online Catalog	INVFILE QUERYDS	L04P040		ILE RPG
CGIPOST	Feedback Talk To Us	TALKTOUS TALKDS	L04P060		ILE RPG
ORDAS400G	Ordering an AS/400 GET	ORDAS00F ORDASDS HTMLO	AS4FORMGE	ORDAS400E	ILE RPG
ORDAS400P	Ordering an AS/400 POST	ORDAS00F ORDASDS HTMLO	AS4FORMPE	ORDAS400E	ILE RPG
PARSECGIP	Ordering an AS/400 POST		RESIFORM		ILE C

Source-Code RPG Program ORDAS400G: ILE RPG program example for the Method=GET.

The information from the URL (value/name pairs) is assigned to the environment variable QUERY_STRING.

This gateway program picks out the name/value pairs of the variables from this QUERY_STRING through the API for ILE RPG.

The following example shows the GET environment variable QtmhGetEnv (subroutine GETENV).

.

```
 *************************************************************************
 *  Simple ILE RPG program ORDAS400G to test CGI Method=GET
 *
 *
 * 1. Compile this source member as module ORDAS400G ( PDM Option=15 )
 *
 * 2. Create program ORDAS400G from module ORDAS400G ( PDM Option=26 )
 *    with PROMPT(PF4) and BNDSRVPGM(QTCP/QTMHCGI)
 *
 * Define your files here
 *************************************************************************
 * Order Database file
 *************************************************************************
FORDAS00F  O  A E             DISK
 *************************************************************************
 * HTMLFILE Input file   ( read FORM input )
 *************************************************************************
FHTMLFILE  IF   E             DISK
 *************************************************************************
 * HTMLO   Output file   ( prepared HTML Output in SRC-PF HTMLO  )
 *                       ( MBR = hidden field in HTML Input Form )
 *************************************************************************
FHTMLO     IF   E             DISK    USROPN  RENAME(HTMLO:HTMLOUT)
 *************************************************************************
A
 *Variables for the CGI interface APIs
 *These are for the APIEnVar
DENBuff           S          2048A   INZ
DENBuffLn         S             9B 0 INZ(2048)
DENActLn          S             9B 0 INZ
DENVarName        S            64A   INZ
DENVarLn          S             9b 0 INZ
A
 *These are for the APICvtDB  Datastructure INPUT fields
 *************************************************************************
DDBFileName       S            20A   INZ(’ORDASDS   *LIBL     ’)
 *************************************************************************
DDBBuff           S          2048A   INZ
DDBBuffLn         S             9B 0 INZ(2048)
DDBDSLn           S             9B 0 INZ
DDBActLn          S             9B 0 INZ
DDBRespCd         S             9B 0 INZ
 *These are used for APIStdOut
DOutBuff          S          2048A   INZ
DOutBuffLn        S             9B 0 INZ(2048)
 *************************************************************************
 *Externally described data structure.  Used for Parsing
 *Need a different one in each CGI-BIN you write
DORDASDS        E DS
 *************************************************************************
```

Figure 176 (Part 1 of 5). ILE RPG Program ORDAS400G Method=GET

```
 **************************************************************************
 * Data structure for error reporting.  Copied from QSYSINC/QRPGLESRC(QUSEC
DQUSEC            DS
D*                                             Qus EC
D QUSBPRV                 1      4B 0 INZ(16)
D*                                             Bytes Provided
D QUSBAVL                 5      8B 0
D*                                             Bytes Available
D QUSEI                   9     15
D*                                             Exception Id
D QUSERVED               16     16
D*                                             Reserved
D*QUSED01                17    116
D*
D*                                      Varying length
 **************************************************************************
 *Constants for names of CGI APIs
DAPIStdIn         C                   'QtmhRdStin'
DAPIStdOut        C                   'QtmhWrStout'
DAPICvtDB         C                   'QtmhCvtDb'
DAPIEnVar         C                   'QtmhGetEnv'
 *Compile-time array for OVRDBF
DOVRDBF           S             80    DIM(2) PERRCD(1) CTDATA
DOVRARR           S              1    DIM(80)
D**************************************************************************
D* Define NewLine
DNewLine          C                   x'15'
D**************************************************************************
D* Define break
DBreak            C                   '<BR>'
 ****************************************************************
```

Figure 176 (Part 2 of 5). ILE RPG Program ORDAS400G Method=GET

```
 ****************************************************************
B
 * Get the Environment Variable called QUERY_STRING
 * Set the ENVarName to QUERY_STRING
 *You must count the length of the Var Name!
 *Set the ENVarLn to this length (12 In This Case)
C                   MOVEL     *BLANKS       ENBuff
C                   MOVEL     'QUERY_STRING'ENVarName
*****************************************************************
C                   Z-ADD     12            ENVarLn
*****************************************************************
C                   EXSR      GETENV
 *  Upon return, your Query_String data is in ENBuff with the len
 *  of the data returned in ENActLn
 *  Move this data to the DBCvt parms
 **************************************************************************
C                   Z-ADD     103           DBDSLn
 **************************************************************************
C                   MOVEL     ENBuff        DBBuff
C                   Z-ADD     ENActLn       DBBuffLn
B
 **************************************************************************
 * Circumvention for HIDDEN fields  ( find out MEMBER-NAME for HTMLO )
 **************************************************************************
C                   MOVEL     ENBuff        M14              14
C                   MOVE      M14           M10              10
C     '+':' '       XLATE     M10           MBR              10
 *******************************************************************
 * END Circumvention
 ********************************************************************
 * Parse using the CvtDB API
C                   EXSR      PARSE
 * The field names in your Ext DS now
 * contain  the Values passed in the POST data
 * Move them to the DB file fields
 **************************************************************************
 * OVR HTMLOUT with MEMBER ORDAS400 and open file
C                   MOVEA     OVRDBF(1)     OVRARR
C                   MOVEA     MBR           OVRARR(24)
C                   MOVEA     OVRARR        CMD              80
C                   Z-ADD     80            L                15 5
C                   CALL      'QCMDEXC'
C                   PARM                    CMD
C                   PARM                    L
C                   open      HTMLO
 *********************************************************************
 * Move FORM Input to Database fields
 **************************************************************************
C                   MOVEL     NAME          NAME_X
C                   MOVEL     ADDRESS       ADDRESS_X
C                   MOVEL     TYPE          TYPE_X
C                   MOVEL     SERV          SERV_X
 *********************************************************************
 * Write Database record file ORDAS00F
 **************************************************************************
C                   WRITE     ORDASR
 **************************************************************************
```

Figure 176 (Part 3 of 5). ILE RPG Program ORDAS400G Method=GET

```
 **************************************************************************
 * Create the HTML Output
 * Write HTML Required control records
 * ADD NewLine append after 80 to 120 characters to OutBuff
 **************************************************************************
C                   DO        9             I                 5 0
C                   READ      HTMLOUT                                98
C     OutBuff       cat       SRCDTA:0      OutBuff
C     OutBuff       CAT       NewLine:0     OutBuff
 * Prepare variable OUTPUT
C     I             ifeq      9
C     OutBuff       CAT       Break:0       OutBuff
C     OutBuff       CAT       Break:0       OutBuff
C     OutBuff       cat       NAME:0        OutBuff
C     OutBuff       CAT       Break:0       OutBuff
C     OutBuff       cat       ADDRESS:0     OutBuff
C     OutBuff       CAT       NewLine:0     OutBuff
C     TYPE          ifeq      ’P1’
C     OutBuff       CAT       Break:0       OutBuff
C     OutBuff       CAT       Break:0       OutBuff
C     10            CHAIN     HTMLOUT                            98
C     OutBuff       CAT       SRCDTA:0      OutBuff
C     OutBuff       CAT       NewLine:0     OutBuff
C                   endif
C     TYPE          ifeq      ’P2’
C     OutBuff       CAT       Break:0       OutBuff
C     OutBuff       CAT       Break:0       OutBuff
C     11            CHAIN     HTMLOUT                            98
C     OutBuff       CAT       SRCDTA:0      OutBuff
C     OutBuff       CAT       NewLine:0     OutBuff
C                   endif
C     TYPE          ifeq      ’P3’
C     OutBuff       CAT       Break:0       OutBuff
C     OutBuff       CAT       Break:0       OutBuff
C     12            CHAIN     HTMLOUT                            98
C     OutBuff       CAT       SRCDTA:0      OutBuff
C     OutBuff       CAT       NewLine:0     OutBuff
C                   endif
C     OutBuff       CAT       Break:0       OutBuff
C     OutBuff       CAT       Break:0       OutBuff
C     SERV          ifeq      ’T’
C     13            CHAIN     HTMLOUT                            98
C     OutBuff       CAT       SRCDTA:0      OutBuff
C     OutBuff       CAT       NewLine:0     OutBuff
C                   else
C     14            CHAIN     HTMLOUT                            98
C     OutBuff       CAT       SRCDTA:0      OutBuff
C     OutBuff       CAT       NewLine:0     OutBuff
C                   endif
C                   endif
C                   ENDDO
C                   MOVE      *OFF          *IN98
C     15            setll     htmlout
C     *IN98         DOWEQ     *OFF
C                   read      htmlout                                98
C     *IN98         CABEQ     *ON           EndHTM
C     OutBuff       cat       SRCDTA:0      OutBuff
C     OutBuff       CAT       NewLine:0     OutBuff
C     EndHTM        TAG
 * End variable OUTPUT
C                   ENDDO
```

Figure 176 (Part 4 of 5). ILE RPG Program ORDAS400G Method=GET

```
      ***************************************************************************
      * Read HTMLFILE until EOF
     C                   MOVE      *OFF          *IN99
     C     *IN99         DOWEQ     *OFF
     C                   READ      HTMLREC                                 99
     C                   ENDDO
      * Send OutBuff to standard output
     C     OutBuff       CAT       NewLine:0     OutBuff
     C                   EXSR      STDOUT
      * End program
     C                   CLOSE     HTMLO
     C                   MOVEA     OVRDBF(2)     OVRARR
     C                   MOVEA     OVRARR        CMD              80
     C                   CALL      'QCMDEXC'
     C                   PARM                    CMD
     C                   PARM                    L
     C                   MOVE      *ON           *INLR
      * These are the APIs used in subroutines to keep the main processing
      * simple.  They do not need to be SUBRs!
C
  |   * Subroutine to Get Environment Variable
  |  C     GETENV        BEGSR
  |  C                   CALLB     APIEnVar
  |  C                   parm                    ENBuff
  |  C                   parm                    ENBuffLn
  |  C                   parm                    ENActLn
  |  C                   parm                    ENVarName
  |  C                   parm                    ENVarLn
  |  C                   parm                    QUSEC
  |  C                   ENDSR
C
     C*  Parse subroutine
     C     PARSE         BEGSR
     C                   CALLB     APICvtDB
     C                   parm                    DBFileName
     C                   parm                    DBBuff
     C                   parm                    DBBuffLn
     ** Remember to code your External DS name.  The API returns your data
     ** in this structure. The field names are in the structure
      ***************************************************************************
     C                   parm                    ORDASDS
      ***************************************************************************
     C                   parm                    DBDSLn
     C                   parm                    DBActLn
     C                   parm                    DBRespCd
     C                   parm                    QUSEC
     C                   ENDSR
      * This is the STD OUT SUBR
     C     STDOUT        BEGSR
     C                   callb     APIStdOut
     C                   parm                    OUTBuff
     C                   parm                    OUTBuffLn
     C                   parm                    QUSEC
     C                   ENDSR
**CTDATA OVRDBF
OVRDBF FILE(HTMLO) MBR(1234567890) LVLCHK(*NO) OVRSCOPE(*JOB)
DLTOVR FILE(HTMLO) LVL(*JOB)
```

Figure 176 (Part 5 of 5). ILE RPG Program ORDAS400G Method=GET

Source-Code RPG Program ORDAS400P: ILE RPG program example for the Method=POST.

The logic of this program is identical with the program in Figure 176 on page 267.

Only three parts are replaced:

A by 1,
B by 2 and
C by 3

This CGI program receives the name/value pairs from the form as encoded form input from STDIN.

The environment variable CONTENT_LENGTH determines how much data *must* be read from STDIN.

The following example shows the read from STDIN QtmhRdStin (subroutine STDIN).

1

```
 ***************************************************************************
 *Variables for the CGI interface APIs
 *These are used for APIStdIn
DInData           S           2048A   INZ
DInDataLn         S              9B 0 INZ(2048)
DInActLn          S              9B 0
 ***************************************************************************
```

2

```
 ***************************************************************************
 * Get the Input parameters from the POST from STDIN
C                   MOVE      *BLANKS       OutBuff
C                   EXSR      STDIN
 *  Upon return, your POST data is in INData and its length is in
 *  INActLn  It is in the FLD=VAR format at this time
 *  Move this data to the DBCvt parms
 * Set up the parameters before CALLB
 *  This includes the length of your Ext DS (103 is correct)
 ***************************************************************************
C                   Z-ADD     103           DBDSLn
 ***************************************************************************
C                   MOVEL     INData        DBBuff
C                   Z-ADD     INActLn       DBBuffLn
 ***************************************************************************
```

3

```
 ***************************************************************************
 * Subroutine to read STD IN
C     STDIN         BEGSR
C                   CALLB     APIStdIn
C                   parm                    INData
C                   parm                    INDataLn
C                   parm                    INActLn
C                   parm                    QUSEC
C                   ENDSR
 ***************************************************************************
```

Figure 177. ILE RPG Program ORDAS400P Method=POST

6.1.9.4 Source-Code C Program PARSECGIP

ILE C program example for the Method=POST: This is the program for Ordering an AS/400 system in the C-Language.

C Programs #include qp0z1170.h

The qp0z1170.h must included to get the getenv() and putenv() functions which are not part of the stdio.h at this time.

The qp0z1170.h is a member in the file H in the library QSYSINC.

```
/************************************************************************
/*  Simple ILE C program PARSECGIP to test CGI Method=POST              *
/*                                                                      *
/*                                                                      *
/* 1. Compile this source member as module PARSECGIP ( PDM Option=15 )  *
/*                                                                      *
/* 2. Create program PARSECGIP from module PARSECGIP ( PDM Option=26 )  *
/*    with PROMPT(PF4) and BNDSRVPGM(ITSOIC400/UTIL)                    *
/*                                                                      *
/************************************************************************
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <qp0z1170.h>

#define MAX_ENTRIES 10000

typedef struct {
    char *name;
    char *val;
} entry;

char *makeword(char *line, char stop);
char *fmakeword(FILE *f, char stop, int *len);
char x2c(char *what);
void unescape_url(char *url);
void plustospace(char *str);


main(int argc, char *argv[]) {
    entry entries[MAX_ENTRIES];
    register int x,m=0;
    int cl;

    printf(″Content-type: text/html\n\n″);

    if(strcmp(getenv(″REQUEST_METHOD″),″POST″)) {
        printf(″This script should be referenced with a METHOD of POST.\n″);
        exit(1);
    }
    if(strcmp(getenv(″CONTENT_TYPE″),″application/x-www-form-urlencoded″)) {
        printf(″This script can only be used to decode form results. \n″);
        exit(1);
    }
    cl = atoi(getenv(″CONTENT_LENGTH″));
    for(x=0;cl && (!feof(stdin));x++) {
        m=x;
        entries[x].val = fmakeword(stdin,’&’,&cl);
        plustospace(entries[x].val);
        unescape_url(entries[x].val);
        entries[x].name = makeword(entries[x].val,’=’);
    }

    printf(″<html>\n″);
    printf(″<body>\n″);
    printf(″<H1>Query Results</H1>″);
    printf(″You submitted the following name/value pairs:<p>″);
    printf(″<ul>″);

    for(x=0; x <= m; x++)
        printf(″<li> <code>%s = %s</code>″,entries[x].name,
               entries[x].val);
    printf(″</ul>″);
    printf(″</body>\n″);
    printf(″</html>\n″);
}
```

Figure 178. ILE C Program PARSECGIP Method=POST

6.2 API Applications

The Internet Connection Server V4.1 includes two APIs:

- IBM ICAPI
- Netscape API

The IBM ICAPI allows you to extend the server′s base functions. You can write extensions to do customized processing, such as:

- Publish customized pages
- Enhance the basic authentication or replace it with a site-specific process
- Add error handling routines to track problems or alert for serious conditions
- Detect and track information that comes in from the requesting client.

The Internet Connection Server includes an NSAPI compatibility module that supports the Netscape API. Using this module, you can easily port your NSAPI programs to run on the Internet Connection Server without any loss of function. This compatibility module gives you the ability to use your NSAPI programs on a wider variety of operating systems: Windows NT, Solaris, HP-UX, AIX, OS/2, etc.

In this section, we focus on two ICAPI samples and we discuss the benefits and drawbacks of the API programs versus CGI programs.

6.2.1 API Development Rules and Principles

When you are developing and testing an ICAPI program, you should always keep in mind the following principles:

- The Internet Connection server runs in a multi-threaded environment; therefore your application functions must be re-entrant.
- Eliminate global variables or protect them with a mutual exclusion semaphore.
- Always check your return codes and provide conditional processing when necessary.
- Always free any allocated resources.
- Make sure to play only with your own data. If you mistakenly write into the server′s private data area, it will likely cause a fatal crash of your server.
- Keep the actions in your application within the scope of a thread. Do not perform any actions such as exit, change uid, etc.
- Do not forget to send complete responses to the HTTP requests including the Content-Type header.

The ultimate rule is to test rigorously your API application before installing it on a production server.

The ICAPI is platform independent: you will be able to port your APIs between Windows NT, OS/2 and UNIX platforms.

The ICAPI is language neutral: your APIs can be written in any of the programming languages supported by these platforms.

If you want your application to be portable and since the provided interface (include files, etc) and documentation are C oriented, you will probably find that the C language is the preferred language to develop your extensions.

The Internet Connection Server product provides you with:

- The API library containing the pre-defined functions:

  - /usr/lib/httpdapi.o
  - \WWW\DLL\HTTPDAPI.dll
  - \WWW\BIN\HTTPDAPI.dll

  for AIX, OS/2 and Windows NT respectively,

- An include file with pre-defined functions prototypes and return codes definitions:

  - /usr/samples/internet_server/API/HTAPI.h
  - \WWW\HTML\EXAMPLES\HTAPI.H

  for AIX and Windows or OS/2 respectively,

- An export file or a library file that identify the external symbols made available by the API library:

  - /usr/samples/internet_server/API/libhttpdapi.exp
  - \WWW\HTML\EXAMPLES\HTTPDAPI.LIB

  for AIX and Windows NT or OS/2 respectively.

You will find in the Web Programming Guide the instructions to build the Dynamically Linked Libraries (DLLs) or Shared Objects (SO) as required for your operating system.

When you are testing your brand new API application, you will certainly face the problem of having a previous release of your library (or shared object) stuck in memory. Unloading the current version of the library will be required before installing the code of the new version in the right place (right directory) and loading the new library. This means that you will have to:

- Stop your Web server
- Install the code of the library
- Restart your Web server

6.2.2 ICAPI Samples

In this section we introduce two ICAPI samples. These samples are basic and do not require an in-depth knowledge of the C language; they were developed only for tutorial purposes and would be unlikely to win any competitions. These samples have been developed and tested on an AIX platform but should run equally well on Windows NT and OS/2.

The first sample in 6.2.2.1, “ICAPI Log Sample” on page 276 is a transaction logging program. It logs information related to queries sent to the server by the GET method.

The second sample in 6.2.2.2, “ICAPI Service Sample” on page 280 is a request processor. It builds a page containing some of the variables passed to the service routines.

6.2.2.1 ICAPI Log Sample

This sample demonstrates how you can plug your own transaction logging function into the server′s request process.

In this sample, the information logged is:

- The IP address of the requester, that is the REMOTE_ADDR variable
- The searchpart of the URL submitted with the request, that is the QUERY_STRING variable
- The additional information passed in the URI (the portion that follows the string that identifies the CGI or API program), that is the PATH_INFO variable

This information is only logged when the QUERY_STRING variable is not null. That way, all the GET requests with arguments will be logged.

Each record in the log file is a set of data separated by the vertical bar character (|). For example:

```
9.67.43.8|bidule|/truc
9.67.43.8|month=08&year=1996&submit=Go+%21|
```

This example does not make much sense, except if you think of it as a means of detecting hackers who are trying to break into your system. Refer to 6.1.7.3, “CGI Security Considerations” on page 244 for a related discussion. The entry for the request of the malevolent user would have been:

```
9.9.9.9:month=02&year=1996;cat+/etc/passwd:
```

Note that the IP address 9.9.9.9 is purely fictitious and does not refer to any known hacker.

Figure 179 on page 277 shows the source code for this log application.

```
/*
 *      mylog.c
 *
 *      ICAPI log sample
 *
 *      This sample contains a log routine: MyLog()
 *      As stated in the documentation, this routine is called after the
 *      socket has been closed, during a request process, regardless of the
 *      success or failure of the request.
 *
 *      MyLog logs in a dedicated log file:
 *              - the searchpart of the URL submitted with the request (QUERY_STRING)
 *              - the additional information passed in the URI (PATH_INFO)
 *
 *      when QUERY_STRING is not NULL.
 *
 *      This sample was tested on an AIX platform, but should run as well
 *      on an OS/2 or Windows NT platforms
 */

#include <stdarg.h>
#include <stdio.h>
#include <string.h>

#include ″HTAPI.h″

#define ALLOCMEMORY     malloc
#define DUPLICATE(x,y)  x = strdup(y)
#define FREEMEMORY(x)   if (x) free(x)

/*
 * getvar routine
 *      extracts the value of a variable from the environment
 *      allocates memory for the string
 */

#define BUF_LEN 2

unsigned char *getvar( unsigned char *name )
{
        long            rc;
        unsigned char   buf[ BUF_LEN ],
                        *value = NULL;
        unsigned long   name_len,
                        value_len = sizeof(buf) - 1;

        name_len = (unsigned long)strlen( (char *)name );
        /* extract the value of the variable */
        HTTPD_extract( NULL, name, &name_len, buf, &value_len, &rc );

        if ( rc == HTTPD_BUFFER_TOO_SMALL ) {
                /* buf is too small - need to allocate a bigger buffer */
                value = (unsigned char *)ALLOCMEMORY( (size_t)value_len + 1 );
```

Figure 179 (Part 1 of 2). Mylog.c Source Code

```
                /* retry to extract the value of the variable */
                HTTPD_extract( NULL, name, &name_len, value, &value_len, &rc );
                if ( rc == HTTPD_SUCCESS )
                        value[ value_len ] = ′\0′;
                else
                        value = NULL;
        }
        else if ( rc == HTTPD_SUCCESS ) {
                /* allocate memory for string and duplicate */
                buf[ value_len ] = ′\0′;
                DUPLICATE( value, buf );
        } /* endif */

        return( value );
} /* getvar */

/*
 * Log entry point
 */

#define MYLOG   ″/tmp/mylog″

void HTTPD_LINKAGE MyLog( unsigned char *handle, long *return_code )
{
        unsigned char   *remote_addr, *query_string, *path_info;
        FILE            *mylog;

        /* open log file */
        mylog = fopen( MYLOG, ″a″ );

        if ( mylog ) {

                /* get the variables from the environment */
                remote_addr = getvar( ″REMOTE_ADDR″ );
                query_string = getvar( ″QUERY_STRING″ );
                path_info = getvar( ″PATH_INFO″ );

                if ( strlen( query_string ) )
                        fprintf( mylog, ″%s|%s|%s\n″,
                                remote_addr, query_string, path_info );

                if ( mylog )
                        fclose( mylog );

                /* free allocated memory */
                FREEMEMORY( remote_addr );
                FREEMEMORY( query_string );
                FREEMEMORY( path_info );
        }

        /* fill in return code */
        *return_code = HTTP_NOACTION;
        return;
} /* MyLog */
```

Figure 179 (Part 2 of 2). Mylog.c Source Code

Some comments on this sample source:

- The variables are extracted by a call to the pre-defined function HTTPD_extract()
- The variables are local (not global)
- All the required precautions have been taken in the getvar() routine to make sure that no stack or memory overflow could occur, and all the return codes are tested
- The returned code of MyLog() is HTTP_NOACTION. That way, the server will also perform its default action for the log step.

Note that if you have multiple applications (or extensions) in the same library and if those applications use a common set of basic routines like getvar() in this sample, the code for this set of routines will be loaded in memory only once for all the applications.

To associate this log function with the appropriate step of the request process, a directive should be added in the configuration file.

On our AIX test platform, the shared object mylog.so was stored as:

```
/usr/local/server_root/api/mylog.so
```

So that the required directive was:

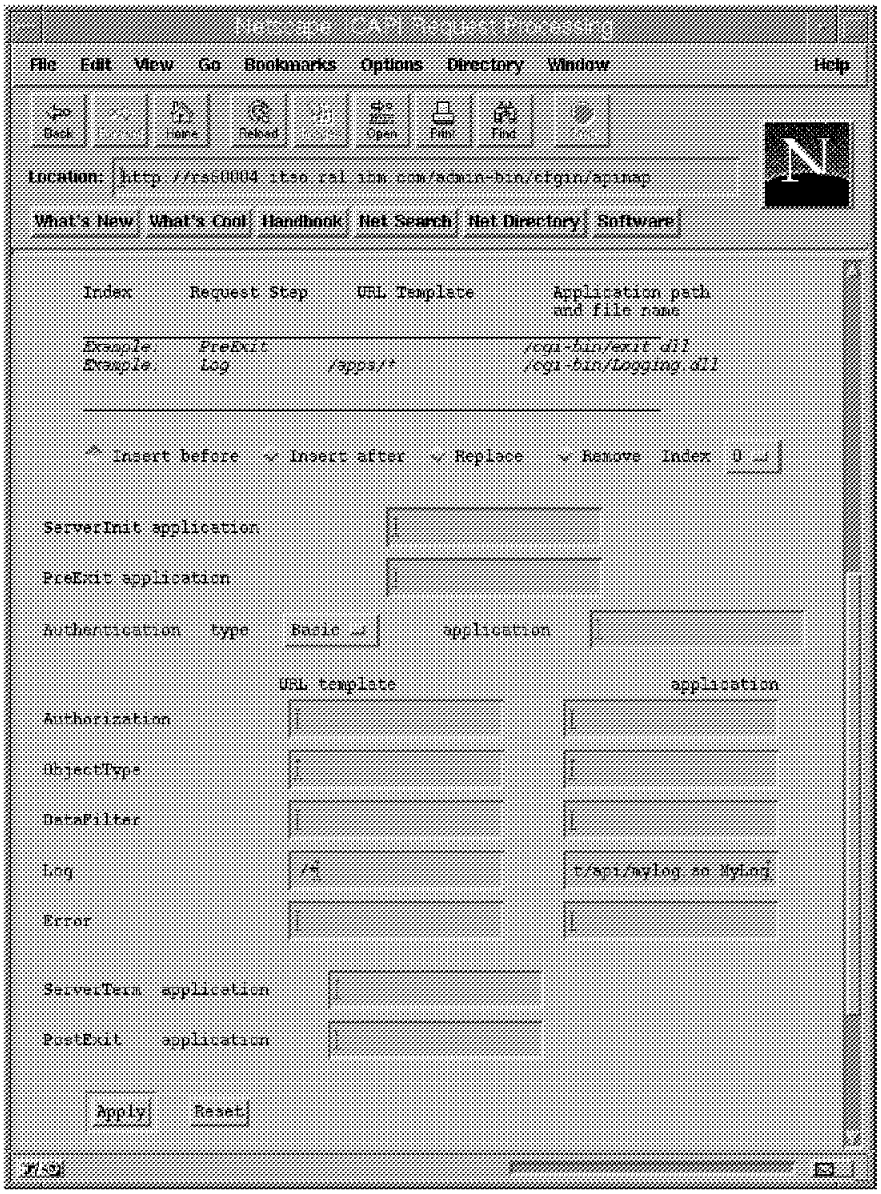
```
Log /* /usr/local/server_root/api/mylog.so:MyLog
```

The directive can be added manually by editing the configuration file or with the help of an administration form.

The link in the main menu of Administration and Configuration Forms is:

- ICAPI Application Processing - Specify which ICAPI Applications to call during request processing.

The ICAPI Application Processing Form is shown in Figure 180 on page 280.


Netscape: ICAPI Request Processing
File Edit View Go Bookmarks Options Directory Window Help
Back Forward Home Reload Images Open Print Find Stop
What's New What's Cool Handbook Net Search Net Directory Software

Location: http://rs60004.itso.ral.ibm.com/admin-bin/cfgin/apimap

Index	Request Step	URL Template	Application path and file name
Example:	*PreExit*		*/cgi-bin/exit.dll*
Example:	*Log*	*/apps/**	*/cgi-bin/Logging.dll*

○ Insert before ○ Insert after ○ Replace ○ Remove Index 0

ServerInit application

PreExit application

Authentication type Basic application

	URL template	application
Authorization		
ObjectType		
DataFilter		
Log	/*	t/api/mylog.so:MyLog
Error		

ServerTerm application

PostExit application

Apply Reset

Figure 180. ICAPI Application Processing Form

Note that the directive would have been:

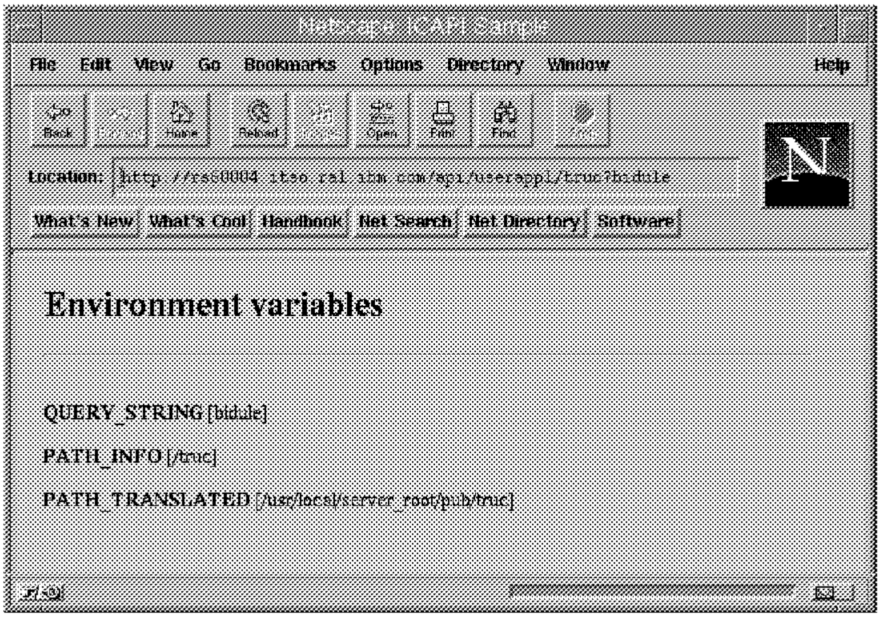
```
Log /cal-bin/* /usr/local/server_root/api/mylog.so:MyLog
```

to log only accesses to URLs matching the /cal-bin/* template.

6.2.2.2 ICAPI Service Sample

This sample demonstrates how you can plug your service application into the server′s request process.

The service given by this application is really basic; it returns to the requester the value of three variables that are accessible by the service routine itself: QUERY_STRING, PATH_INFO and PATH_TRANSLATED as shown in Figure 181 on page 281.



http://rs60004.itso.ral.ibm.com/api/userappl/truc?bidule

Environment variables

QUERY_STRING [bidule]

PATH_INFO [/truc]

PATH_TRANSLATED [/usr/local/server_root/pub/truc]

Figure 181. HTML Page Returned by Userappl ICAPI Application

The source code for this service application is listed in Figure 182 on page 282.

```
/*
 *      userappl.c
 *
 *      ICAPI sample
 *
 *      This sample includes one application function : UserAppl
 *      This function build a customized page displaying environment
 *      variables ( QUERY_STRING, PATH_INFO and PATH_TRANSLATED )
 *
 *      This sample was tested on an AIX platform, but should run as well
 *      on an OS/2 or Windows NT platform
 */

#include <stdarg.h>
#include <stdio.h>
#include <string.h>

#include ″HTAPI.h″

#define ALLOCMEMORY     malloc
#define DUPLICATE(x,y)  x = strdup(y)
#define FREEMEMORY(x)   if (x) free(x)

/*
 * getvar routine
 *      extracts the value of a variable from the environment
 *      allocates memory for the string
 */

#define BUF_LEN 2

unsigned char *getvar( unsigned char *name )
{
        long            rc;
        unsigned char   buf[ BUF_LEN ],
                        *value = NULL;
        unsigned long   name_len,
                        value_len = sizeof(buf) - 1;

        name_len = (unsigned long)strlen( (char *)name );

        /* extract the value of the variable */
        HTTPD_extract( NULL, name, &name_len, buf, &value_len, &rc );

        if ( rc == HTTPD_BUFFER_TOO_SMALL ) {
                /* buf is too small - need to allocate a bigger buffer */
                value = (unsigned char *)ALLOCMEMORY( (size_t)value_len + 1 );

                /* retry to extract the value of the variable */
                HTTPD_extract( NULL, name, &name_len, value, &value_len, &rc );
                if ( rc == HTTPD_SUCCESS )
                        value[ value_len ] = ’\0’;
                else
                        value = NULL;
        }
```

Figure 182 (Part 1 of 5). Userappl.c Source Code

```
        else if ( rc == HTTPD_SUCCESS ) {
                /* allocate memory for string and duplicate */
                buf[ value_len ] = ’-0’;
                DUPLICATE( value, buf );
        } /* endif */

        return( value );
} /* getvar */


/*
 * va_list_length routine
 *      estimate the amount of space needed to malloc before
 *      calling vsprintf()
 */

int va_list_length( const char *fmt, va_list marker )
{
        char    *q = (char *)fmt;
        int     len = strlen(fmt);
        va_list temp = marker;

        while ( q = strchr( q, ’%’ ) ) {
                /* search for the format type - might not be q[ 1 ]...
 */
                if ( q[ 1 ] != ’%’ ) {
                        int     i;
                        int     done;
                        char    *p;

                        for ( i = 1, done = 0; |done; i++ ) {
                                switch( q[ i ] ) {
                                case ’c’:
                                        len += 5;
                                        va_arg( temp, char );
                                        done = 1;
                                        break;
                                case ’s’:
                                        if ( p = va_arg( temp, char * ) )
                                                len += strlen(p);
                                        done = 1;
                                        break;
                                case ’d’:
                                case ’i’:
                                        len += 20;
                                        va_arg( temp, int );
                                        done = 1;
                                        break;
                                case ’h’:
                                case ’H’:
                                        len += 20;
                                        va_arg( temp, short );
                                        done = 1;
                                        break;
```

Figure 182 (Part 2 of 5). Userappl.c Source Code

```
                                case ’l’:
                                case ’L’:
                                        len += 20;
                                        va_arg( temp, long);
                                        done = 1;
                                        break;
                                case ’o’:
                                case ’u’:
                                case ’x’:
                                case ’X’:
                                        len += 20;
                                        va_arg( temp, unsigned );
                                        done = 1;
                                        break;
                                case ’e’:
                                case ’E’:
                                case ’f’:
                                case ’g’:
                                case ’G’:
                                        len += 20;
                                        va_arg( temp, double );
                                        done = 1;
                                        break;
                                case ’p’:
                                        len += 10;
                                        va_arg( temp, void * );
                                        done = 1;
                                        break;
                                case ’n’:
                                        len += 20;
                                        va_arg( temp, int * );
                                        done = 1;
                                        break;
                                default:
                                        break;
                                }
                        }
                } /* endif */
                q++;
        }
        return len;
} /* va_list_length */
```

Figure 182 (Part 3 of 5). Userappl.c Source Code

```
/*
 * vHprintf routine
 *      ICAPI vprintf
 */

int vHprintf( const char *fmt, va_list marker )
{
        long            rc;
        unsigned long   len = va_list_length( fmt, marker );
        unsigned char   *q = (unsigned char *)ALLOCMEMORY( ( 2 * len ) + 1 );

        if ( q ) {
                if ( ( len = vsprintf( (char *)q, fmt, marker ) ) > 0 ) é
                        HTTPD_write( NULL, q, &len, &rc );
                        if ( rc != HTTPD_SUCCESS )
                                len = 0;
                }
                else {
                        /* vsprintf returned a negative number, which is an erro
r */
                        len = 0;
                }
                FREEMEMORY(q);
        }
        else
                len = 0;

        return( (int)len );
} /* vHprintf */


/*
 * Hprintf routine
 *      ICAPI printf
 */

int Hprintf( const char *fmt, ... )
{
        va_list marker;
        int     rc;

        va_start( marker, fmt );
        rc = vHprintf( fmt, marker );
        va_end( marker );
        return( rc );
} /* Hprintf */
```

Figure 182 (Part 4 of 5). Userappl.c Source Code

```
/*
 * User Application entry point
 */

void HTTPD_LINKAGE UserAppl( unsigned char *handle, long *return_code )
{
        unsigned char   *path_info, *path_translated, *query_string;


        /* get the variables from the environment */
        path_info = getvar( ″PATH_INFO″ );
        path_translated = getvar( ″PATH_TRANSLATED″ );
        query_string = getvar( ″QUERY_STRING″ );


        /* write the customized HTML page */
        Hprintf( ″HTTP/1.0 200\n″ );
        Hprintf( ″Content-Type: text/html\n\n″ );

        Hprintf( ″<HTML><HEAD><TITLE>ICAPI Sample</TITLE></HEAD><BODY    >″ );
        Hprintf( ″<H1>Environment variables</H1><HR>″ );
        Hprintf( ″<P><B>QUERY_STRING</B> [%s]″, query_string ? query_s
tring : ″NULL″ );
        Hprintf( ″<P><B>PATH_INFO</B> [%s]″, path_info ? path_info : ″
NULL″ );
        Hprintf( ″<P><B>PATH_TRANSLATED</B> [%s]″, path_translated ? p
ath_translated : ″NULL″ );
        Hprintf( ″</BODY></HTML>″ );


        /* free allocated memory */
        FREEMEMORY( query_string );
        FREEMEMORY( path_info );
        FREEMEMORY( path_translated );

        /* fill in return code */
        *return_code = HTTP_OK;
        return;
} /* UserAppl */
```

Figure 182 (Part 5 of 5). Userappl.c Source Code

Some comments on this sample source:

1. The response is sent back to the client by calling HTTPD_write().
2. The response includes the required Content-Type header.
3. Special care has been taken in the vHprintf() routine to allocate enough memory to handle the string before calling vsprintf(). You shouldn′t take the risk of a buffer overflow in any of your API applications.
4. The allocated resources, in this case only memory, have been carefully released.
5. The return code of UserAppl() is HTTP_OK which means that the request has been handled by this application function. That way, the server knows that it should not call any other application function and should proceed with the next step which is data filter.

In order to associate this service function with the appropriate step of the request process, a directive should be added in the configuration file.

On our AIX platform, the shared object userappl.so was stored as:

```
/usr/local/server_root/api/userappl.so
```

The following directive was added to the configuration file:

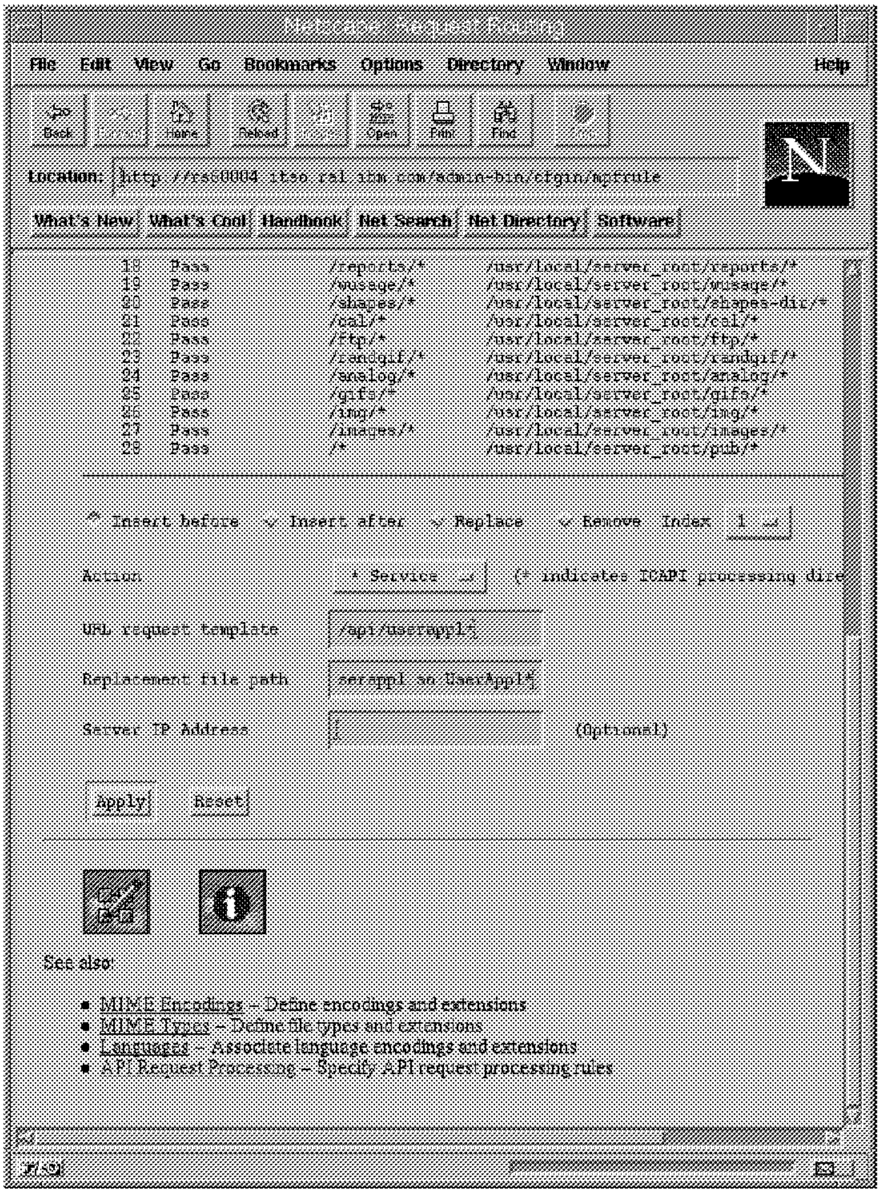
```
service /api/userappl* /usr/local/server_root/api/userappl.so:UserAppl*
```

The directive can be added manually by editing the configuration file or with the help of an administration form.

The link in the main menu of Administration and Configuration Forms is:

- Request Routing - Route URL requests to server files

The Request Routing Form is shown in Figure 183.



Location: http://rs60004.itso.ral.ibm.com/admin-bin/cfgin/mpfrule

```
18   Pass          /reports/*      /usr/local/server_root/reports/*
19   Pass          /wusage/*       /usr/local/server_root/wusage/*
20   Pass          /shapes/*       /usr/local/server_root/shapes-dir/*
21   Pass          /cal/*          /usr/local/server_root/cal/*
22   Pass          /ftp/*          /usr/local/server_root/ftp/*
23   Pass          /randgif/*      /usr/local/server_root/randgif/*
24   Pass          /analog/*       /usr/local/server_root/analog/*
25   Pass          /gifs/*         /usr/local/server_root/gifs/*
26   Pass          /img/*          /usr/local/server_root/img/*
27   Pass          /images/*       /usr/local/server_root/images/*
28   Pass          /*              /usr/local/server_root/pub/*
```

● Insert before ○ Insert after ○ Replace ○ Remove Index 1

Action: * Service (* indicates ICAPI processing dire

URL request template: /api/userappl*

Replacement file path: serappl.so:UserAppl*

Server IP Address: (Optional)

See also:

- MIME Encodings – Define encodings and extensions
- MIME Types – Define file types and extensions
- Languages – Associate language encodings and extensions
- API Request Processing – Specify API request processing rules

Figure 183. Request Routing Form

As a result, the URL for the application program is:

```
http://rs60004.itso.ral.ibm.com/api/userappl
```

Because this API application displays the QUERY_STRING, PATH_INFO and PATH_TRANSLATED variables, the returned HTML page will show null pointers unless you have a search string in your URL such as:

```
http://rs60004.itso.ral.ibm.com/api/userapp/truc?bidule
```

6.2.3 Using APIs versus Using CGIs

API programs have supporters but also detractors.

Some people look at CGI as an inefficient cross-platform interface that can break down a server′s performance.

Other people consider the API as a major risk to the reliability and stability of a server.

But all agree that both CGI and API extend the capabilities of a Web server. There is no doubt that CGIs and APIs are required to set up an active presence on the Internet.

Before trying to construct a list of benefits and drawbacks of APIs versus CGIs, let′s start with some reminders on their principles.

6.2.3.1 CGI Principles

CGI scripts are external programs that the Web server can invoke in a process environment.

When the URL passed to the Web server points to a CGI program, the server prepares a number of environment variables representing its current state and the request.

Then the server launches the CGI script with these variables set up which are valid only for that particular request. Launching a script means starting (forking) a new process.

A busy server often has many requests at once for the same URL; this leads to a situation where the same script is running simultaneously multiple times with different environments and on behalf of multiple processes.

6.2.3.2 API Principles

API applications are stored in the following libraries:

- Dynamically Linked Libraries (DLLs) on Windows platforms
- Shared Objects (SOs) on UNIX platforms

At startup, the server reads from the configuration file the list of libraries containing the applications and loads them into memory.

When the URL passed to the server points to an application function (service) into an API library, the server sets up memory variables in its memory space and then calls the routine.

When the application routine is invoked, its code is already in memory and has access to the server′s predefined functions. These predefined functions provide access to the variables associated with the request.

6.2.3.3 API Benefits and Drawbacks

From the principles of the two technologies and from their practical implementation, we can address the following list of benefits and drawbacks of API applications compared to CGI scripts.

API Benefits

1. APIs require less memory than CGIs because there is only one occurrence of the application code loaded in memory at once, even if multiple requests are processed simultaneously.
2. APIs have less CPU overhead as the libraries are loaded at startup time and stay resident. Furthermore, the applications run in the context of the server itself (threads).
3. APIs are not limited to requests servicing. APIs allow you to plug in to the server′s request process your own program functions such as authentication, authorization, logging, error handling, data filtering, etc.

API Drawbacks

1. API applications require specialized developers with qualifications in techniques such as multithreading, concurrency synchronization, etc.
2. Failure of API applications can affect directly the server and can cause it to crash. The applications should be intensively tested before being installed on a production server.

The existence in the market of different types of API and the fact that there was no standard for this interface used to be seen as a drawback of this technology. This is no longer true as you can observe with the Internet Connection servers: a compatibility module allows you to port your existing NSAPI programs from Netscape servers to the IBM Internet Connection servers.

Furthermore, we can expect to see in the near future an intermediate solution where API extensions will provide support for interpreted or pseudo-compiled languages such as Perl and Java. This would be a good compromise between performance and the ease with which programming extensions can be developed.

Chapter 7. Enhancing Your Web Site with Java Applications

Java is a programming language, developed by Sun Microsystems, which is object-oriented, distributed, interpreted, architecture neutral, and portable.

Java has definitely become the in-vogue Web programming language in a short space of time. While the scope of our project does not allow us to delve deeply into the subject of Java we feel that we should at least provide an example of how Java can be used to good effect. Accordingly, in the next section we provide such an example.

For a summary of Java support on the Internet Connection server platforms, please refer to Chapter 1, "The Internet Connection Family" on page 1.

7.1 Java - Choosing a Presidential Candidate

The scenario we consider in this section is how to access legacy data from your clients and how to cooperate your legacy systems and internet applications using Java. The main objective is to provide information from an MVS platform to your clients using Web technology.

We selected the 1996 Republican Presidential Race as a basis for developing a Java application. The data processing associated with the election can be handled by a typical online transaction system. Many transactions such as data input and query jobs are occurring at the same time. If clients want to know the current status of the election in real time from everywhere, the transaction system must be connected to a global network such as the Internet. If the clients are not familiar with 3270 screens, any GUI client software such as WebExplorer or Netscape Navigator is required.

- Business Requirements:
  - Serve the dynamic needs of their clients by improving customer service
  - Leverage investments in their existing systems
  - Exploit the Internet as another way to increase opportunities
  - Extend current access to their systems over the Internet quickly
- Solution
  - Implement a flexible, open, and distributed computing environment
- Characteristics:
  - Uses Java technology
  - Uses Web technology
- Benefits:
  - Reduces the development costs and time
  - Integrates new technology with existing systems leveraging hardware and software investments

This example is simplified for ease of understanding. The election data is updated by using an ISPF editor. If you develop real OLTP applications, you may use some subsystems like CICS and IMS, and you may need gateways between the subsystems and your Web server such as the CICS Internet Gateway.

7.1.1 Configuration

The software configuration used for this example is as follows:

- Web Server
  - MVS/ESA 5.2.2 OpenEdition
  - DFSMS, SMP/E, LE/370, RACF, ISPF, and TCP/IP
  - Internet Connection Server for MVS/ESA R1.0
  - Java applets and htmls
- Web Clients
  - OS/2 Warp Connect
  - WebExplorer Java Technology Demo

7.1.2 Considerations

There are some areas to consider which arise from developing a Java application in the environment described.

7.1.2.1 Portability

The Java environment is a new approach to distributed computing. Developing your applications using Java results in software that is portable across multiple machine architectures, operating systems, and graphical user interfaces. With Java, your job as a software developer is much easier.

7.1.2.2 Legacy Systems

The Internet related technologies such as Web technology will give your legacy applications a boost, via a more familiar graphical user interface, and they will require a minimal amount of redevelopment investment.

7.1.2.3 Application Model

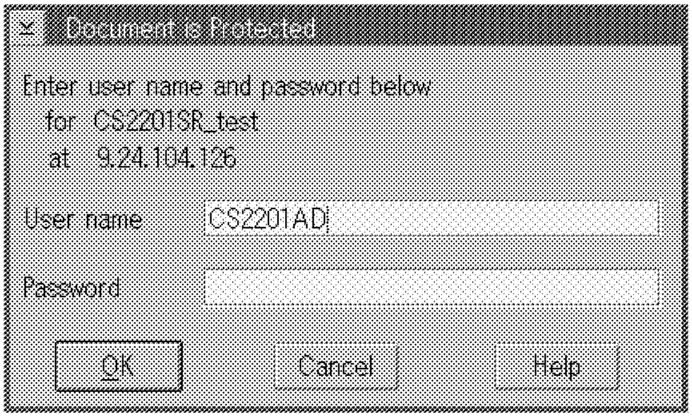
An application model for this example is the Enterprise Distributed Web Model. Enterprise Distributed Web applications are true open client/server applications. The client is coded in downloadable Java, which runs in any Java-enabled browser like Netscape Navigator or HotJava. A page containing the Java applet is downloaded by a Web server to a browser. While the applet runs in the browser, it opens its own communications session to the server. The applet also communicates with a server to provide access to the data in the legacy system.

7.1.2.4 Access Control

The server and applet should only deliver restricted documents to authorized users. ICS/MVS has extensive access control support including RACF validation of user IDs and passwords and access control through SURROGAT user ID support. Users are identified by an OMVS user ID (UID), kept in the RACF user profile, and an OMVS group ID (GID), kept in the RACF group profile. Although this scenario uses RACF for user IDs and passwords authentication, we do not recommend to use RACF over the Internet because there is the risk of intercepting the user ID and password.

7.1.3 Using the Application

When you access the Web server that runs this example for the first time, the sign-on panel is displayed by the access control of the Web server (Figure 184). If you enter the user ID and password correctly, you will be successfully signed on to the server.



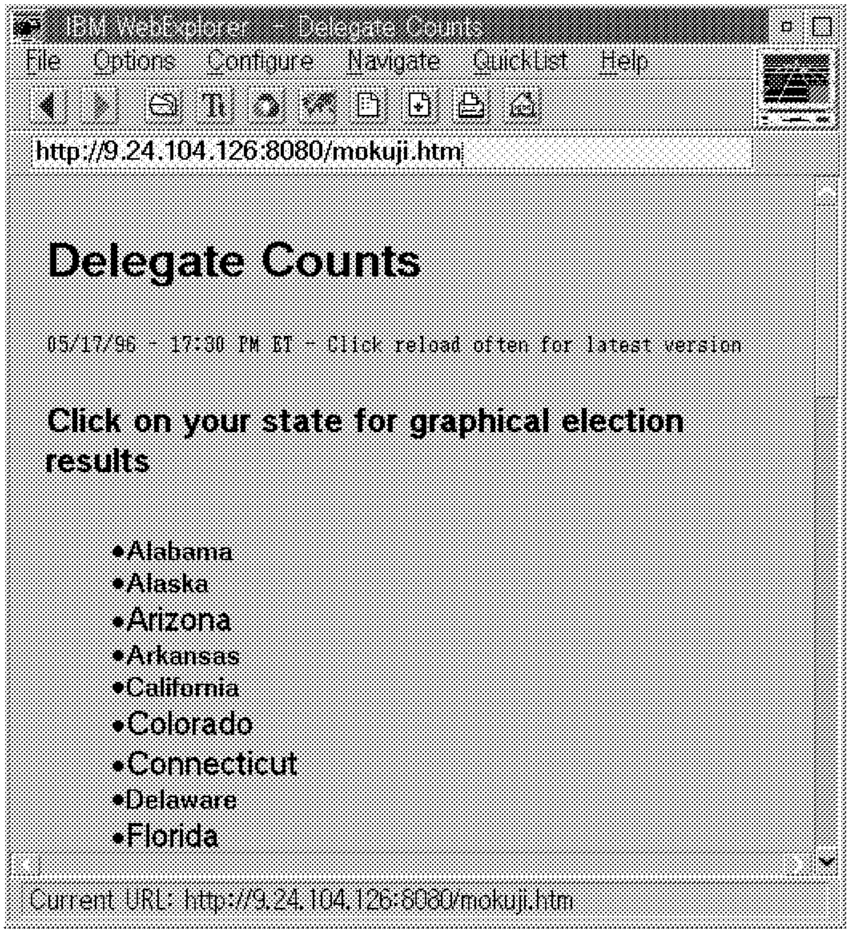
Enter user name and password below
for CS2201SR_test
at 9.24.104.126

User name CS2201AD

Password

Figure 184. Document is Protected

Figure 185 on page 294 is the first menu page of this example. If you click on any state name that you want to see, the browser starts to load the HTML file and the related applet. After loading, the applet starts to run on the browser.



http://9.24.104.126:8080/mokuji.htm

Delegate Counts

05/17/96 - 17:30 PM ET - Click reload often for latest version

Click on your state for graphical election results

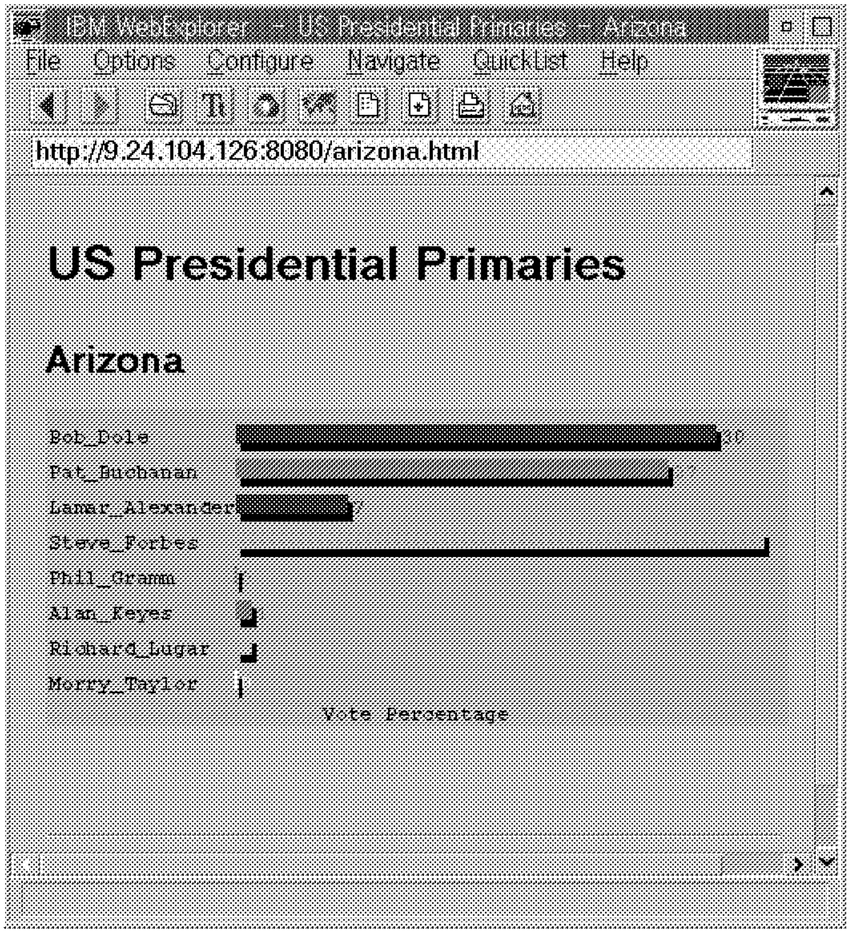
- Alabama
- Alaska
- Arizona
- Arkansas
- California
- Colorado
- Connecticut
- Delaware
- Florida

Current URL: http://9.24.104.126:8080/mokuji.htm

Figure 185. Delegate Counts - Selection Menu

When the applet starts running, it gets the election result data of the state from the Web server at specified intervals (value=10000 means about 10 second intervals; for more details refer to Figure 189 on page 297). It then displays a graph of the newest data on the browser (see Figure 186 on page 295). If the election data (the arizona.txt file) is updated, the graph is also updated within the specified intervals.

The applet that we use in this example is BarChart.class which is based on the sample class Chart.class included in Sun′s Java Development Toolkit.



http://9.24.104.126:8080/arizona.html

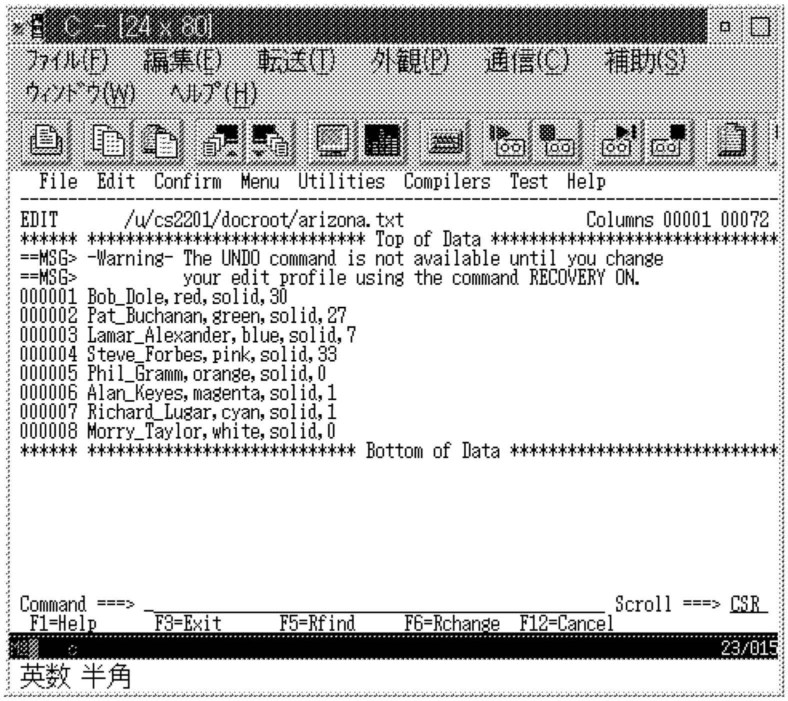
US Presidential Primaries

Arizona

Figure 186. US Presidential Primaries Result - State of Arizona.

Figure 187 on page 296 is a screen of the ISPF panel used for editing the result data of the state of Arizona (the arizona.txt file). For example, three candidates′ values are updated:

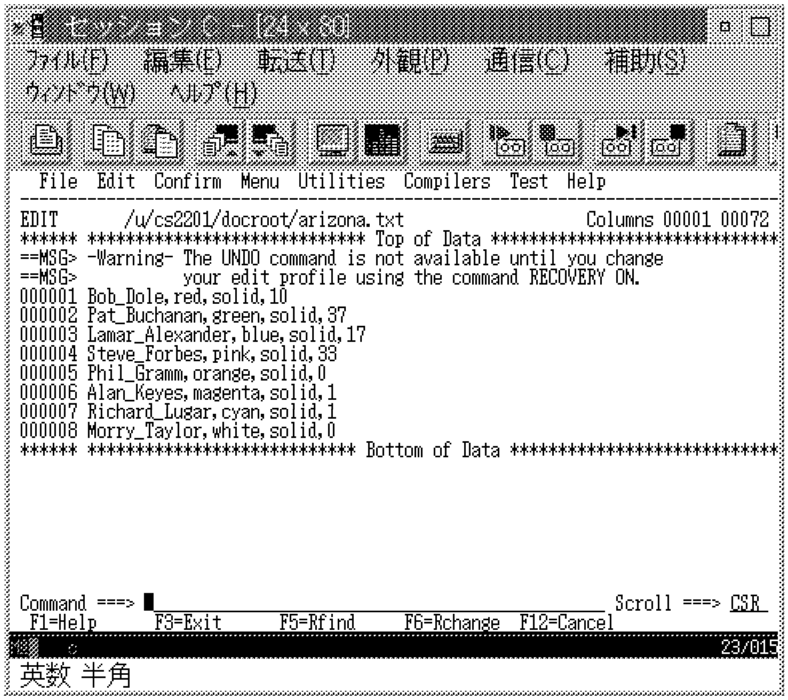
- Bob Dole 30% --> 10%
- Pat Buchanan 27% --> 37%
- Lamar Alexander 7% --> 17%



```
  File  Edit  Confirm  Menu  Utilities  Compilers  Test  Help
-------------------------------------------------------------------------------
EDIT       /u/cs2201/docroot/arizona.txt                   Columns 00001 00072
****** ***************************** Top of Data ******************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
000001 Bob_Dole,red,solid,30
000002 Pat_Buchanan,green,solid,27
000003 Lamar_Alexander,blue,solid,7
000004 Steve_Forbes,pink,solid,33
000005 Phil_Gramm,orange,solid,0
000006 Alan_Keyes,magenta,solid,1
000007 Richard_Lugar,cyan,solid,1
000008 Morry_Taylor,white,solid,0
****** **************************** Bottom of Data ****************************







Command ===> _                                                Scroll ===> CSR
 F1=Help      F3=Exit      F5=Rfind      F6=Rchange  F12=Cancel
```

Figure 187. ISPF Panel (before Updating)

Figure 188 shows the ISPF panel after updating.

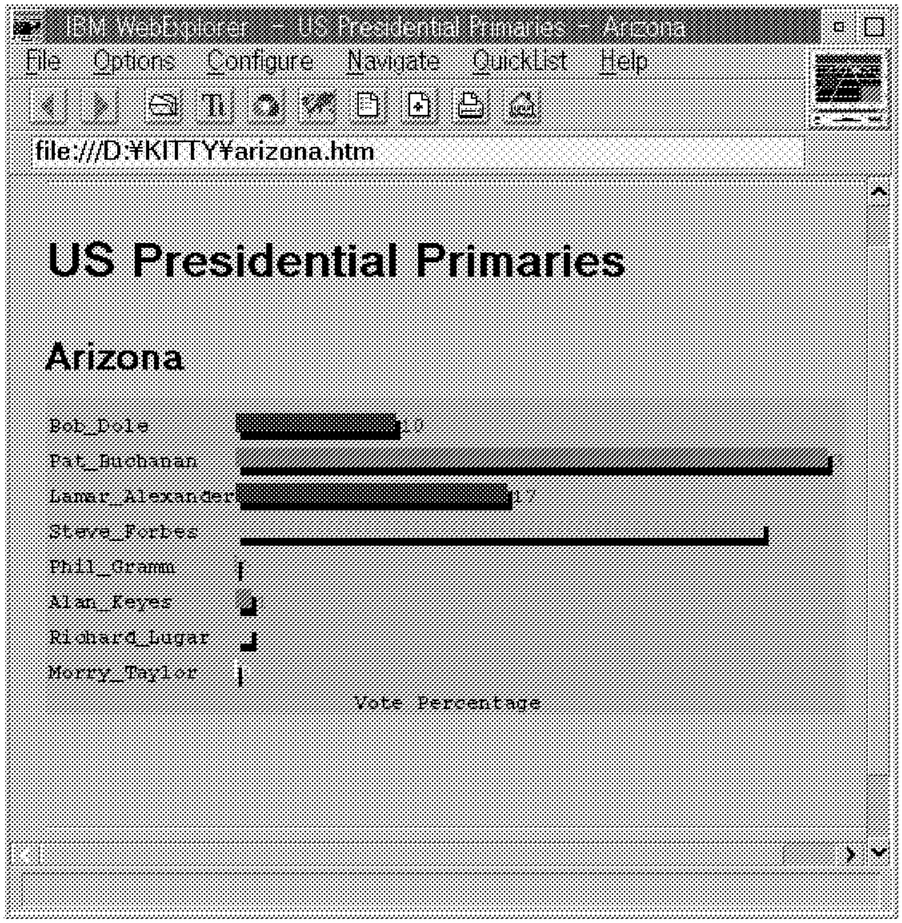


```
  File  Edit  Confirm  Menu  Utilities  Compilers  Test  Help
-------------------------------------------------------------------------------
EDIT       /u/cs2201/docroot/arizona.txt                   Columns 00001 00072
****** ***************************** Top of Data ******************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
000001 Bob_Dole,red,solid,10
000002 Pat_Buchanan,green,solid,37
000003 Lamar_Alexander,blue,solid,17
000004 Steve_Forbes,pink,solid,33
000005 Phil_Gramm,orange,solid,0
000006 Alan_Keyes,magenta,solid,1
000007 Richard_Lugar,cyan,solid,1
000008 Morry_Taylor,white,solid,0
****** **************************** Bottom of Data ****************************







Command ===> _                                                Scroll ===> CSR
 F1=Help      F3=Exit      F5=Rfind      F6=Rchange  F12=Cancel
```

Figure 188. ISPF Panel (after Updating)

After the save command is entered, the client can see the updated graph on the Web browser within the specified intervals (Figure 189 on page 297).

It is really a live graph on the browser.



IBM WebExplorer - US Presidential Primaries - Arizona

file:///D:¥KITTY¥arizona.htm

US Presidential Primaries

Arizona

Figure 189. US Presidential Primaries Result - State of Arizona (Updated)

The HTML source of arizona.html is as follows:

HTML source - arizona.html

```
<HTML>
<HEAD>
<title>US Presidential Primaries - Arizona</title>
</HEAD>
<BODY>
<hr>
<H1>US Presidential Primaries</H1>
<H2>Arizona</H2>
<applet code="BarChart.class" width=500 height=250>
<param name=title value="Vote Percentage">
<param name=orientation value="horizontal">
<param name=infile value="arizona.txt">    // Graph Data File
<param name=interval value="10000">        // refresh interval time
<param name=scale value="5">               // Data Display Scale
<param name=barwidth value="3">            // Bar width
<param name=interspace value="5">          // Space between Bars
</applet>
<hr>
</BODY>
</HTML>
```

It is very simple and easy to understand. The Java applet is included in the Web page with the <applet> tag. A Java-enabled browser displays any alternate HTML that appears between <applet> and </applet>. The <param> tag is to specify parameter values for the applet. In this example, the BarChart.class applet is loaded and seven parameter values are passed to the applet. The following table is the file of the result data for the state of Arizona. This file is read from the applet which runs on a Java-enabled browser, and it can be updated from an ISPF editor on the legacy host system. The file name is specified at the <parm name=infile value> in the HTML file, and is passed to the applet.

result data - arizona.txt

```
Bob_Dole,        red,     solid, 30
Pat_Buchanan,    green,   solid, 27
Lamar_Alexander, blue,    solid,  7
Steve_Forbes,    pink,    solid, 33
Phil_Gramm,      orange,  solid,  0
Alan_Keyes,      magenta, solid,  1
Richard_Lugar,   cyan,    solid,  1
Morry_Taylor,    white,   solid,  0
```

Appendix A. Two Simple Browsers and One Simple Server

An important part of this project was to study the elementary behavior of the servers and browsers we described. Basically, we needed to make visible the HTTP request of any browser and the HTTP response of any server. We therefore developed the tools presented in this appendix.

These tools are designed to be run on an OS/2 platform. But they are programmed in REXX and Java so porting them to any other platform should not be a problem.

Finally, they are quick and dirty: do not expect fancy interfaces or subtle error recovery procedures.

A.1 A Basic REXX Browser

We wrote the basic REXX browser in order to be able to send any request to a server, and to display the corresponding response from the server. As we used it, it is an HTTP browser. But you may use it to experiment with any other connectionless protocol based on TCP/IP. The protocol must be connectionless, because the server closes the connection after it gets a response.

The basic browser is composed of two parts: the agent and the user interface.

A.1.1 The Basic Agent

The agent is listed in Figure 190 on page 300. It simply does the following:

- Opens a socket
- Sends a request
- Receives a response and places it in a queue
- Closes the socket

```
/* Basic Agent */

/* Constants */
MAX_LEN = 512

/* Parse the parameters */
PARSE ARG addr,port,request

/* Initialize the Rexx Socket API */
rc = RxFuncAdd(″SockLoadFuncs″,″rxSock″,″SockLoadFuncs″)
rc = SockLoadFuncs()

/* Create a socket */
socket = SockSocket(″AF_INET″,″SOCK_STREAM″,″IPPROTO_TCP″)

/* Open the socket */
address.!addr   = addr
address.!port   = port
address.!family = ″AF_INET″
rc = SockConnect(socket,″address.!″)

/* Send the request */
rc = SockSend(socket,request)

/* Create a queue */
qnew = RXQUEUE(″Create″)
qold = RXQUEUE(″Set″,qnew)

/* Receive the response and place it in the queue */
DO WHILE SockRecv(socket,response,MAX_LEN) > 0
  QUEUE response
  DROP response
END

/* Reset the queue to the one used previously */
CALL RXQUEUE ″Set″,qold

/* Close the socket */
rc = SockClose(socket)

/* Return the name of the response queue */
RETURN qnew
```

Figure 190. The Basic Agent (AGENT.CMD)

A.1.2 The Basic Browser

The basic browser is listed in Figure 191 on page 301. It is a loop that first asks you for a socket in the format ″ip_address:port″, as 9.24.104.181:80 for example, then for a request, that can span over several lines. Enter an exclamation point (!) to stop the loop, or to signify the end of the request to the browser.

```
/* Basic Browser */

/* Constant */
CRLF = d2c(13)″″d2c(10)

/* Defaults */
addr = ″9.24.104.181″
port = ″80″

/* Loop */
DO FOREVER

  /* Read and parse the socket */
  SAY ″Enter the server′s socket (Enter=″addr″:″port″, !=quit)″
  PARSE PULL input
  SELECT
    WHEN input = ″″ THEN NOP
    WHEN input = ″!″ THEN LEAVE
    OTHERWISE PARSE VALUE input WITH addr″:″port
  END

  /* Read the request */
  SAY ″Enter the browser′s request (!=end)″
  request = ″″
  DO FOREVER
    PARSE PULL line
    IF line = ″!″ THEN LEAVE
    request = request″″line″″CRLF
  END

  /* Call the agent with the socket and the address */
  CALL AGENT addr,port,request

  /* Print the response */
  qnew = RESULT
  qold = RXQUEUE(″Set″,qnew)
  DO WHILE QUEUED() > 0
    PARSE PULL buf
    SAY buf
  END
  CALL RXQUEUE ″Set″,qold

END

EXIT
```

Figure 191. The Basic Browser (BROWSER.CMD)

To launch the basic browser simply enter BROWSER at an OS/2 command prompt.

A.2 A Basic Java Browser

The term browser is a little pompous for the little Java applet described here, because it does not return the server′s response. It only returns the date at which the data pointed by the URL passed as argument was last modified.

The source of the applet is listed in Figure 192 on page 302 and, as with every tiny applet, it includes a large overhead.

```
/* Import */
import java.applet.*;
import java.io.*;
import java.net.*;
import java.util.*;
/* The LastModified class */
public class LastModified extends Applet{
  /* main */
  public static void main(String[] args) {
    /* Variables */
    URL MyURL;
    Date MyDate;
    URLConnection MyURLConnection;
    /* Try to parse the URL */
    try {
      /* Parse the URL */
      MyURL = new URL(args[0]);
      /* Try to open the connection */
      try {
        /* Open the connection */
        MyURLConnection = MyURL.openConnection();
        /* Read the date the data was last modified */
        MyDate = new Date(MyURLConnection.getLastModified());
        /* Print this date */
        System.out.println(MyDate.toString());
      } catch (IOException e) {};
    } catch (MalformedURLException e) {};
  }

}
```

Figure 192. A Basic Java Browser (LastModified.java)

To launch the basic java browser on an OS/2 platform, enter `JAVA LastModified [URL]` at an OS/2 command prompt.

A.3 A Basic REXX Server

The purpose of the basic REXX server is to display the HTTPD request of any browser. As a basic REXX browser, you may use it to study any connectionless TCP/IP-based protocol.

The basic REXX server is listed in Figure 193 on page 303.

```
/* Basic Server */

/* Constants */
CRLF = d2c(13)″″d2c(10)

/* Load the Rexx Socket Support */
rc = RxFuncAdd(″SockLoadFuncs″,″rxSock″,″SockLoadFuncs″)
rc = SockLoadFuncs()

/* Create a socket */
domain = ″AF_INET″
type = ″SOCK_STREAM″
protocol = ″IPPROTO_TCP″
socket0 = SockSocket(domain,type,protocol)

/* I don′t know but I was told... */
rc = SockSetSockOpt(socket0,″SOL_SOCKET″,″SO_REUSEADDR″,0)
rc = SockIoctl(socket0,″FIONBIO″,1)

/* Bind */
dum = CHAROUT(,″Enter the server′s socket : ″)
socket = LINEIN()
PARSE VALUE socket WITH addr″:″port
address.!addr   = addr
address.!port   = port
address.!family = ″AF_INET″
rc = SockBind(socket0,″address.!″)

/* Listen */
rc = SockListen(socket0,32)

r.0 = 1
r.1 = socket0

rc = SockSelect(″r.″,″″,″″)

/* Accept */
socket1 = SockAccept(socket0)

/* Receive the first part of the request only */
rc = SockRecv(socket1,request,1024)

/* Send an error response */
rc = SockSend(socket1,″HTTP/1.0 401 Quo Vadis?″CRLF″″CRLF

/* Close */
rc = SockClose(socket1)

/* Close */
rc = SockClose(socket0)
```

Figure 193. A Basic REXX Server (SERVER.CMD).

To launch the basic server, simply enter SERVER at an OS/2 command prompt.

Appendix B. Special Notices

This publication is intended to help Web site administrators and Internet specialists to evaluate and implement the Internet Connection Servers. The information in this publication is not intended as the specification of any programming interfaces that are provided by the IBM Internet Connection Servers. See the PUBLICATIONS section of the IBM Programming Announcement for the IBM Internet Connection Servers for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM′s product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM′s intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, NY 10594 USA.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The information about non-IBM (″vendor″) products in this manual has been supplied by the vendor and IBM assumes no responsibility for its accuracy or completeness. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer′s ability to evaluate and integrate them into the customer′s operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any performance data contained in this document was determined in a controlled environment, and therefore, the results that may be obtained in other operating environments may vary significantly. Users of this document should verify the applicable data for their specific environment.

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

ACF/VTAM
AIX
AnyNet
AS/400
AT
C/400
CICS
DataJoiner
DB2
DB2/2
DFSMS
IBM
IMS
Language Environment
MQSeries
MVS/ESA
OpenEdition
OS/2
OS/390
OS/400
Power Series
RACF
RISC System/6000
SP
Ultimedia
VTAM
WebExplorer
400

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc.

PC Direct is a trademark of Ziff Communications Company and is used by IBM Corporation under license.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Microsoft, Windows, and the Windows 95 logo
are trademarks or registered trademarks of Microsoft Corporation.

Java and HotJava are trademarks of Sun Microsystems Inc.

ActiveX	Microsoft Corporation
AltaVista	Digital Equipment Corporation
Animator	Micro Focus Limited
Apollo	Apollo Computer, Incorporated
C + +	American Telephone and Telegraph Company, Incorporated
DCE	The Open Software Foundation
DDS	Sony Corporation
Digital	Digital Equipment Corporation
Hercules	Hercules Computer Technology
Lycos	Carnegie Mellon University
Magellan	Lotus Development Corporation
MS-DOS	Microsoft Corporation
NDIS	3Com Corporation and Microsoft Corporation
NET	NCR Corporation
Netscape	Netscape Communications Corporation
Network File System	Sun Microsystems, Incorporated
NFS	Sun Microsystems Incorporated
ORACLE	Oracle Corporation
PC Magazine	Ziff Communications Company
POSIX	Institute of Electrical and Electronic Engineers

Sidekick	Borland International, Incorporated
Sierra	Trimble Navigation Limited
Solaris	Sun Microsystems, Incorporated
Sun	Sun Microsystems, Incorporated
Sun Microsystems	Sun Microsystems, Incorporated
Sybase	Sybase Corporation
Visual Basic	Microsoft Corporation
Windows NT	Microsoft Corporation
X Window System	Massachusetts Institute of Technology

Other trademarks are trademarks of their respective companies.

Appendix C. Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

- *TCP/IP Illustrated, Volume 1*, SR28-5586 (ISBN 0-201-63346-9)
- *TCP/IP Illustrated, Volume 2*, SR28-5630 (ISBN 1-201-63354-X)
- *Spinning the Web*, SR28-5646 (ISBN 1-850-32141-8)
- *Surfing the Internet with Netscape*, SR28-5788 (ISBN 7821-1740-6)
- *Marketing on the Internet*, S246-0109 (ISBN 1-885068-01-8)
- *HTML Sourcebook*, ISBN 0-471-14242-5
- *JAVA Sourcebook*, ISBN 0-471-14859-8
- *The Web Page Design Cookbook*, ISBN 0-471-13039-7
- *Creating Your Own Netscape Web Pages*, ISBN 0-7897-0734-9

C.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see "How To Get ITSO Redbooks" on page 311.

- *Using the Information Super Highway*, GG24-2499
- *TCP/IP Tutorial and Technical Overview*, GG24-3376
- *Building a Firewall with the IBM Internet Connection Secured Network Gateway*, SG24-2577
- *Accessing the Internet*, SG24-2597
- *IBM TCP/IP V3R1 for MVS for MVS Implementation Guide*, SG24-3687
- *Practical TCP/IP for AIX V3.2/V4.1 Users*, SG24-4381
- *Accessing CICS Business Applications from the World Wide Web*, SG24-4547
- *Safe Surfing: How to Build a Secure WWW Connection*, SG24-4564
- *World Wide Web Access to DB2*, SG24-4716
- *Accessing OpenEdition from the Internet*, SG24-4721
- *TCP/IP Implementation in an OS/2 Warp Environment*, SG24-4730
- *How to Secure the Internet Connection Server for MVS/ESA*, SG24-4803
- *Cool Title about the AS/400 and Internet*, SG24-4815

C.2 Redbooks on CD-ROMs

Redbooks are also available on CD-ROMs. **Order a subscription** and receive updates 2-4 times a year at significant savings.

CD-ROM Title	Subscription Number	Collection Kit Number
System/390 Redbooks Collection	SBOF-7201	SK2T-2177
Networking and Systems Management Redbooks Collection	SBOF-7370	SK2T-6022
Transaction Processing and Data Management Redbook	SBOF-7240	SK2T-8038

CD-ROM Title	Subscription Number	Collection Kit Number
AS/400 Redbooks Collection	SBOF-7270	SK2T-2849
RISC System/6000 Redbooks Collection (HTML, BkMgr)	SBOF-7230	SK2T-8040
RISC System/6000 Redbooks Collection (PostScript)	SBOF-7205	SK2T-8041
Application Development Redbooks Collection	SBOF-7290	SK2T-8037
Personal Systems Redbooks Collection (available soon)	SBOF-7250	SK2T-8042

How To Get ITSO Redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, CD-ROMs, workshops, and residencies. A form for ordering books and CD-ROMs is also provided.

This information was current at the time of publication, but is continually subject to change. The latest information may be found at URL http://www.redbooks.ibm.com.

How IBM Employees Can Get ITSO Redbooks

Employees may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **PUBORDER** — to order hardcopies in United States
- **GOPHER link to the Internet** - type GOPHER.WTSCPOK.ITSO.IBM.COM
- **Tools disks**

  To get LIST3820s of redbooks, type one of the following commands:

  ```
  TOOLS SENDTO EHONE4 TOOLS2 REDPRINT GET SG24xxxx PACKAGE
  TOOLS SENDTO CANVM2 TOOLS REDPRINT GET SG24xxxx PACKAGE (Canadian users only)
  ```

  To get lists of redbooks:

  ```
  TOOLS SENDTO WTSCPOK TOOLS REDBOOKS GET REDBOOKS CATALOG
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET ITSOCAT TXT
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET LISTSERV PACKAGE
  ```

  To register for information on workshops, residencies, and redbooks:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ITSOREGI 1996
  ```

  For a list of product area specialists in the ITSO:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ORGCARD PACKAGE
  ```

- **Redbooks Home Page on the World Wide Web**

  http://w3.itso.ibm.com/redbooks

- **IBM Direct Publications Catalog on the World Wide Web**

  http://www.elink.ibmlink.ibm.com/pbl/pbl

  IBM employees may obtain LIST3820s of redbooks from this page.

- **ITSO4USA category on INEWS**
- **Online** — send orders to: USIB6FPL at IBMMAIL or DKIBMBSH at IBMMAIL
- **Internet Listserver**

  With an Internet E-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an E-mail note to announce@webster.ibmlink.ibm.com with the keyword subscribe in the body of the note (leave the subject line blank). A category form and detailed instructions will be sent to you.

How Customers Can Get ITSO Redbooks

Customers may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **Online Orders** (Do not send credit card information over the Internet) — send orders to:

	IBMMAIL	**Internet**
In United States:	usib6fpl at ibmmail	usib6fpl@ibmmail.com
In Canada:	caibmbkz at ibmmail	lmannix@vnet.ibm.com
Outside North America:	dkibmbsh at ibmmail	bookshop@dk.ibm.com

- **Telephone orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	(long distance charges apply)
(+45) 4810-1320 - Danish	(+45) 4810-1020 - German
(+45) 4810-1420 - Dutch	(+45) 4810-1620 - Italian
(+45) 4810-1540 - English	(+45) 4810-1270 - Norwegian
(+45) 4810-1670 - Finnish	(+45) 4810-1120 - Spanish
(+45) 4810-1220 - French	(+45) 4810-1170 - Swedish

- **Mail Orders** — send orders to:

IBM Publications
Publications Customer Support
P.O. Box 29570
Raleigh, NC 27626-0570
USA

IBM Publications
144-4th Avenue, S.W.
Calgary, Alberta T2P 3N5
Canada

IBM Direct Services
Sortemosevej 21
DK-3450 Allerød
Denmark

- **Fax** — send orders to:

United States (toll free)	1-800-445-9269	
Canada	1-403-267-4455	
Outside North America	(+45) 48 14 2207	(long distance charge)

- **1-800-IBM-4FAX (United States)** or **(+1) 415 855 43 29 (Outside USA)** — ask for:

  Index # 4421 Abstracts of new redbooks
  Index # 4422 IBM redbooks
  Index # 4420 Redbooks for last six months

- **Direct Services** - send note to softwareshop@vnet.ibm.com

- **On the World Wide Web**

Redbooks Home Page	http://www.redbooks.ibm.com
IBM Direct Publications Catalog	http://www.elink.ibmlink.ibm.com/pbl/pbl

- **Internet Listserver**

  With an Internet E-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an E-mail note to announce@webster.ibmlink.ibm.com with the keyword subscribe in the body of the note (leave the subject line blank).

IBM Redbook Order Form

Please send me the following:

Title	Order Number	Quantity

- **Please put me on the mailing list for updated versions of the IBM Redbook Catalog.**

First name Last name

Company

Address

City Postal code Country

Telephone number Telefax number VAT number

- Invoice to customer number

- Credit card number

Credit card expiration date Card issued to Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

DO NOT SEND CREDIT CARD INFORMATION OVER THE INTERNET.

Index

Printed in U.S.A.

SG24-4805-00