MIDRANGE COMPUTING®

JULY 1999 • VOLUME 17 • ISSUE 3

COMMENTARY & OPINION

SOLUTIONS SOURCE

DECISION SUPPORT

E-BUSINESS

GROUPWARE

NEW TECHNOLOGY

HTTP UNDERCOVER

Take a peek at the Web's underlying protocol.

by Teresa Pelkie

Most of us AS/400 professionals are already familiar with the acronym HTTP, although we seldom have to think about how Hypertext Transfer Protocol itself works. After all, connecting to the Internet is usually a matter of configuring TCP/IP on the workstation or the server. Once the connection is made, you can request and view Web pages from practically all of the Web servers available on the Internet.

However, TCP/IP provides only the connection between computers. When you enter a URL in the address bar of your browser or click on a link to a URL, your browser sends a request to a Web server, using HTTP. Along with the request, your browser can send additional information about the browser and your preferences. Using HTTP, the Web server can examine the request and the additional information and return the requested Web page and its associated files. If an error is encountered while processing your request, the Web server sends an error code and description, again using HTTP.

If you are familiar with HTML, you know that Web pages are described in terms of ASCII text and a set of tags that indicate to the browser how the page is to be rendered. It seems that it should be a simple matter to develop a protocol that can transmit ASCII text from a server to a browser, and indeed, the first version of HTTP was very simple, given that it provided only that functionality. However, with the wide availability of graphical browsers, HTTP evolved to support additional data types, such as the inclusion of binary graphics on a Web page. The current version, HTTP 1.1, provides additional support for Web-based communication with particular emphasis on performance issues.

In this article, I show how a request is transmitted from a browser to a Web server and how the response is sent using HTTP. I also describe some of HTTP's configuration options that control the HTTP response for the V4R3 and above versions of **IBM** HTTP Server for AS/400.

Start at the Browser

You can request a Web page at your browser by entering a URL in the format *http://host:port/path*. With that format, *http:* is used to identify the protocol, *host* is the name or IP address of the Web server host computer, *port* is the TCP/IP port (or the default port 80), and *path* is used to optionally specify the path to the resource you are requesting. At a minimum, you need to supply the host name. The recent versions of **Microsoft** Internet Explorer and **Netscape** Navigator assume the *http://* part of the URL if you do not enter it.

METHOD NAME	DESCRIPTION
CONNECT	Reserves method name for use with a proxy that can dynamically switch to being a tunnel.
GET	Used to retrieve information identified by the Request URI (Uniform Resource Identifier).
HEAD	Identical to GET, except that the server does not return a message body as a response. Used to request metainformation and headers. Usually used for testing links for validity, accessibility, and modification.
OPTIONS	Requests information about the communication options available. Can be used by the client to determine the options and requirements of a server without actually initiating a retrieval.
POST	Requests the server to accept the entity sent with the request. Usually used when submitting a Web form to the server.
TRACE	Used to invoke a loop-back of the request message. Allows the client to see what is received by the server.
PUT	Requests the server to store the entity sent with the request. (Supported at V4R4 on IBM HTTP Server for AS/400.)
DELETE	Requests the server to delete the resource identified by the Request URI. (Supported at V4R4 on IBM HTTP Server for AS/400.)

Figure 1: The first string of any HTTP request includes a request method.

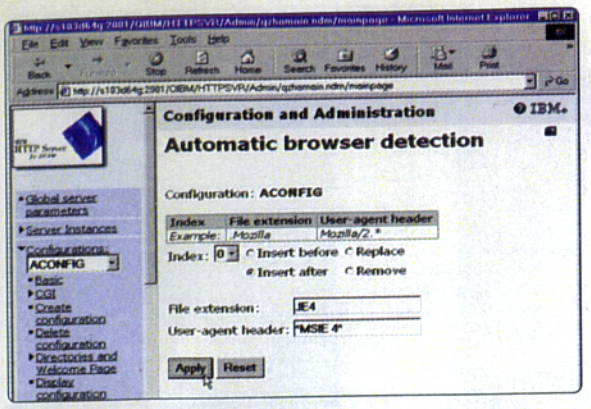

Figure 2: Use the browser-based configuration and administration program to configure IBM HTTP Server for AS/400.

After entering the URL, the browser formats a series of text strings that are sent to the host. To retrieve a Web page, the first string includes a *method*, which is used to indicate the type of request the browser is making to the Web server. Figure 1 lists the methods used with HTTP. The GET method is used to indicate to the Web server that a specific resource is being requested. The protocol and version are also sent along with the path and name of the resource so that the Web server will know what level of HTTP the browser supports.

Following the method request, the browser usually sends one or more *request headers*. The headers are used to convey additional information about browser capabilities and user preferences to the server. For example, the User-Agent request header indicates the browser name and version number. An example of an optional request header is the Accept-Language header. You can configure your browser to request Web pages in different national languages (for example, Spanish and, if that is not available, English). The Web server can examine the request headers and select the most appropriate Web page to return to the browser when it has a choice of pages to return. The following lists some of the request headers that can be sent from the browser to the server:

- *Accept*. Specifies media types that are acceptable for a response (for example, Accept: text/html).

You don't need to know the details of HTTP to use the protocol.

- *Accept-Charset*. Indicates character sets that are acceptable for a response.
- *Accept-Encoding*. Indicates content codings that are acceptable in the response (for example, Accept-encoding: compress, gzip).
- *Accept-Language*. Indicates the set national languages preferred in a response (for example, Accept-Language: es, en).
- *From*. Provides the email address of the requester to the server.
- *Host*. Specifies the port number and host address.
- *User-Agent*. Contains client information (for example, browser identification and version).

Server Processing

After the request and headers are received at the server, the server can start processing to prepare a response. If only one Web page can be returned, the server simply sends that page to the browser. However, you can configure your Web server to work with the request headers sent from the browser to select the most appropriate Web page to return.

For example, you might create Web pages that take advantage of certain browser capabilities, such as VBScript support in Explorer. The Web server can determine the browser that sent the request by examining the content of the User-Agent header. In your Web server configuration, you can associate a specific file extension with data in the User-Agent header, as shown in Figure 2. When processing the request, the Web server will select a Web page to return if it includes the specified file extension, preferring that over a file that does not include the extension. The file extension can be specified at any point in the file name; some examples are Page1.html.ie4 and Page1.ie4.html.

STATUS CODE	DESCRIPTION
1xx — Informational	
100 Continue	The initial part of a request has been accepted, and the client should continue.
2xx — Successful	
200 OK	The request succeeded.
202 Accepted	The request was accepted for processing, but the processing is not yet completed.
204 No Content	The server fulfilled the request but does not need to return an entity body.
206 Partial Content	The server fulfilled a partial GET request for the resource.
3xx — Redirection	
301 Moved Permanently	The requested resource has a new URI.
302 Found	The requested resource is temporarily at a different URI.
4xx — Client Error	
400 Bad Request	The request could not be processed because of malformed syntax.
401 Unauthorized	The request requires user authentication.
403 Forbidden	The server understood the request but refused to fulfill it.
404 Not Found	The server could not locate the resource specified in the Request URI.
405 Method Not Allowed	The method is not allowed.
407 Proxy Authentication Required	The client must authenticate itself to the proxy.
410 Gone	The requested resource is not available on the server, and there is no known forwarding address.
5xx —Server Error	
500 Internal Server Error	The server encountered an unexpected condition and cannot fulfill the request.
501 Not Implemented	The server does not support the functionality required to fulfill the request.
503 Service Unavailable	The server is unable to handle the request because of a temporary condition.

Figure 3: A Web host may set up custom HTTP status messages, but more often, they simply send the standard HTTP status codes to browsers.

The Web server configuration options for IBM HTTP Server for AS/400 can be used to specify supported request methods, languages, and encoding as well as the port number and values for persistent HTTP connections. You can use the browser-based configuration and administration forms (as shown in Figure 2) to configure the server or directly edit the server configuration file and its directives using the Work with HTTP Configuration (WRKHTTPCFG) command.

Back to the Browser

After locating the file or files to send to the browser, the Web server starts sending the files, preceded by one or more *response* and *entity headers*. Response headers indicate the status of the request, and the browser uses entity headers to determine how to render the response entity, which is the data from the file. Some of the more interesting entity headers that can be sent from the Web server to the browser include Content-Encoding, Content-Language, Content-Length, Content-Location, Content-Type, Expires, and Last-Modified.

Status Codes

It may happen that a Web page you request is not available on the Web server or that you are not authorized to view the page. In that case, the Web server returns a status message to the browser instead of the file. Figure 3 lists some of the status codes that can be sent from the Web server to the browser. In some cases, the browser can automatically respond to the status code and attempt the operation again. Other status codes simply appear in your browser (for example, the famous 404 Not Found status code).

An Evolving Protocol

Additional needs and requirements for HTTP become apparent as more applications are hosted on the Internet. For example, HTTP 1.1 includes many features that relate directly to performance when compared with HTTP 1.0. You can find more information about HTTP 1.1 (the current version) and discussion of future enhancements to HTTP at *www.ietf.org/ids.by.wg/http.html*.

As you know from your experience using the Web and possibly configuring a Web server, you don't need to know much about the details of HTTP itself to successfully use the protocol. However, the more you know about how HTTP works "under the covers," the more options you have for configuring your browser or Web server and creating Web pages that are intended for specific audiences.

Teresa Pelkie is the owner of **WWWSights** in the San Diego, California, area. She is an adjunct faculty member at Palomar College, where she teaches Introduction to the Internet and HTML. She can be reached at *teresa@wwwsights.com*.