BBx® Goes Online

By Allen Miglore

The Internet and the World Wide Web have gained the attention of the media, computer and non-computer alike, in a way that astounds many of us in the software business. The Web has made cover stories for *Time* and *Newsweek*, has been the focus of countless television and newspaper stories, and has driven sales of computers to all-time records as people scrambled to Web and e-mail enable their homes and offices.

As awareness of the Web has grown, so has the keen interest of small businesses in this new medium. It is hard to find a small business that does not have e-mail, and is not at least considering creating a Web page. Some businesses are jumping fully onto the bandwagon, putting up complete Web sites, as opposed to a simple Web page, with online catalog and customer service applications. Even more businesses are finding that the Web technology-browser and server software-is useful on their internal networks. This simple client/server technology, with its mix of new (GUI) and old (block mode transmission) technology, is becoming a standard tool for many internal applications.

As a BBx developer, you may be wondering if there is a way to capitalize on this new opportunity. After all, Web technology is software technology, and you have worked hard to ensure your customers think of you when they think of software. In this article, I'll outline the various pieces of Web technology and then discuss different approaches you can use to take advantage of this new technology with your current BBx applications.

The Technology

The Web is a client/server technology. Most of us are familiar with this concept, especially those of us who have installed a BASIS Data Server. A standard communication protocol was devised a few years ago, called Hyper Text Transfer Protocol. In this technology, the browser is the client. Browsers can take many forms, but the most common form is that of a graphical workstation product, such as Netscape Navigator or Microsoft Internet Explorer. The server is a task that runs on a Web server in the background, and it is called a Hyper Text Transfer Protocol Daemon, or httpd.

The primary purpose of the browser is to display documents that are written in a language called Hypertext Markup Language (HTML). In addition, browsers can display plain text and graphical images, can act as anonymous FTP clients, and e-mail clients. The purpose of the server is simply to listen for requests, and send responses back to the client. A document request is always in the form of a Uniform Resource Locator (URL), which might look like this: "http://i99.com/sdsi/index.html". A URL consists of a protocol

"http://", a domain name "i99.com", a path "/sdsi", and a document "index.html."

HTML is a simple text markup language that consists of plain text and embedded tags that indicate to the browser how the text should be rendered. For example, the text "<h1> Document Title </h1>" will render the text "Document Title" as a level 1 heading, and center it. In addition to display characteristics, markup tags can specify a clickable link to another document (this is where the name "hypertext" comes from), an image, a table, an input form, and more. There are many HTML editors, all offering features similar to word processing packages, though manyauthors prefer to use standard text editors in order to have complete control over the document content.

CGI-Dynamic HTML

HTML documents allow for presentation of static information: the file that the server sends to the browser is simply a file on the server's disk. What happens, however, if the information to be presented is constantly changing, or if there are thousands of potential documents, such as items in a catalog, in which all of the items look alike except for specific content fields? The architects of the HTTP protocol anticipated a need for dynamically generated documents, and developed a standard called Common Gateway Interface (CGI). This standard defines how the Web server can launch an executable program, give that program some data, and get a document as a response from that program. With this standard, documents can be created on-demand, they exist only transiently, and they can contain information from any source that the program can access, including BBx data files, if a BBx program is part of a CGI process.

Here's how a standard CGI process works on a UNIX server (On Windows servers, things can be a little different.):

1. The server gets a request for a document that the server understands to be a CGI request. This is normally by naming convention or directory mapping. A program name is identified in that request.
2. The server establishes an environment for the program, including such things as the URL, the remote client address, data strings, and more. In some cases, additional information can be sent to the program's standard input channel. The server starts the program, and waits for a response to come back on the program's standard output channel.
3. The program runs, sends a document out its standard output channel, then quits.
4. The server sends the document to the browser.

Using BBx with CGI

BBx programs can function as CGI programs if they are given a proper startup. A BBx program is not a native executable program, and so it must be launched in a native execution layer, which can be a shell script, a batch file, or other mechanism that can start a BBx task and program and then wait for it to complete.

Once started, the BBx program can open and read files, call programs, perform business rule logic, and create an HTML result for the browser. In short, the BBx CGI program can do anything any other BBx program can do, except work directly with the user's keyboard and screen. As with any CGI task, User I/O goes by way of a proxy-the Web server.

The advantages of this approach are significant:

- Platform independence; the same programs will run on Unix and Windows
- Direct access to local or network BBx files, and the associated speed of reading data natively
- Leverage of your existing BBx expertise, with no need to learn a new language
- Re-use of existing BBx program code, so your business rules don't need to be rewritten

Using BB-Web

To help BBx developers with the native launching layer, the CGI I/O, and the development of HTML from BBx files and strings, Synergetic Data Systems Inc. developed a product called BB-Web. BASIS endorses BB-Web as part of their Internet strategy, and they, along with other developers, have used BB-Web to develop Web-enabled applications using BBx as the application's engine.

Using ODBC

Another option which is viable under many circumstances is to use the Basis ODBC Driver™ and a Windows-based server. Using various technologies from vendors such as Microsoft and Netscape, you can develop Web applications based on SQL access to BBx data files, embedding the results of queries, along with limited processing logic, into dynamic HTML documents.

The world is moving to the Web. BB-Web and the BASIS ODBC Driver will allow you to take advantage of this new medium by incorporating your BBx applications and data into HTML documents that can be viewed by a browser. BB-Web allows developers to create applications that run with BBx as the engine, while the BASIS ODBC Driver makes BBx data accessible to Web applications using SQL. Using these tools, you can stay competitive and provide your customers the innovative Web applications they want.