AUTOMATING INFORMATION OVERLOAD: LINKING DATABASES TO THE WEB

Leonard A. White
Technology Training and Resource Specialist
Information Technology Training
U.C.Davis
Davis, CA 95616
lawhite@ucdavis.edu
tel: (916)752-2906 fax: (916)754-8470

ABSTRACT

University departments create, receive, and process large amounts of information every week. A challenge in any office is storing this information in a way that allows easy access for users and quick, efficient retrieval of desired elements. More and more relevant information is entirely digital, needing to be shuffled into large, expanding databases. With the rapid development of the World Wide Web over the last two years into an accessible and versatile medium of high speed communication, new options have emerged for how we manage information. The Web has already become a significant medium for University departments, many relying on Web publishing as their primary avenue to communicate about courses, events, and other campus news. Departments are now becoming interested in having their Web pages collect information from their potentially huge user community, without requiring a large staff overhead. Being able to serve this emerging need may be an important role for campus technical training and support organizations. We can now create Web interfaces that directly access our already existing databases. Our gain is twofold: The cross-platform and distributed access power of the Internet strengthens our service capabilities and enables the interactivity and responsiveness our departments need.

This paper presents a practical guide for user service organizations to employing the Web as an interface for accessing important database information and gaining a higher level of Web site interactivity. It covers some of the ways this innovation can make our services more productive and efficient. Also, it reviews some of the technological innovations that now give non-programmers the ability to design and create Web-to-database interfaces.

Introduction

As the commercial world generates a frenzy of hype and activity around the World Wide Web, technologies are emerging to satisfy their needs that could have a profound impact on the way campus computing service organizations do business as well. As departments begin to struggle with the demand of publishing useful, up-to-date information for students and support organizations struggle with the need to show them how, technologies are becoming available that automate the process of dynamic Web publishing. In this paper I'll present a piece of the picture of how the web can be an access to information currently locked inside databases. I'll start by trying to provide a glimpse at how linking databases to the Web can make campus organizations more effective and productive. Next, I'll briefly summarize the basic workings of the exchange of information between users, the web, and databases. Then I'll provide an overview of some of the software that makes creating Web-to-Database interfaces accessible if not easy. Finally, I'll present a summary of my own experiences linking a class evaluation database to the Web using one of these software tools. A bibliography and URLography is available at the end for further reading.

From a service organization's perspective, why connect a database to the World Wide Web?

The answer to this question could be very different depending on the particular service in mind. A big common denominator, however, is that Web-to-Database interfaces can make our jobs easier and more efficient. Serving a campus community means serving

thousands of people, many asking the same questions or needing the same immediate responses. In the past, increasing efficiency might have meant setting up an automated telephone help desk, so customers could direct themselves to the person with the answers they're looking for. With the World Wide Web becoming the standard of distributed information and research worldwide, even more efficient, cost-effective, and in some cases easier to implement methods of increasing efficiency are being created. These new methods of increasing efficiency rely largely upon the ability to use the World Wide Web as a gateway to information, not simply a new publishing medium.

Take the corporate world as an example of a community keenly interested in efficiency. This is reasonable considering that efficiency can quickly impact profits, market share: all things holy to a corporation. Taking a ride on the Web today yields product registration forms, company club applications, verbose product info, technical support databases, and the online shopping cart. Almost all of these new roads to efficiency make use of Web-to-Database interfaces.

Where might Web-to-Database interfaces fit in a campus service organization's business?

Most simply, Web-to-Database interfacing provides a Web page with interactivity: the ability for the Web to go beyond a computer-based publications medium into the domain of feedback, response, assessment, and choice.

Consider the service dilemma when an entire campus is switching to a network-based administrative system in a short period of time. Huge numbers of users may have the same question about how to access a 342-56 form. One way to alleviate help desk overload could involve publishing a frequently asked questions page on the Web and trying to steer users to it. One main drawback of this approach is the challenge it may be to steer users to static pages of a Web site. Consider an alternative utilizing a Web-to-Database interface. The help site could involve an initial interactive 'user questionnaire' in which the user is prompted with multiple questions regarding his or her problem. Upon submission of this form, the server queries (requests information from) a database of frequently asked questions, indexed by multiple topics. The user's screen responds with a page of the matching questions, their possible solutions, and links to see related topics. Beyond the efficiency with which user and solution are brought together, this method has another major benefit: it centralizes the help desk information. Once the frequently asked questions have been entered as individual records in the help desk database, there is no need to spend the time creating a separate web page for publication. Information entered into a Web-connected database is published on the Web automatically for any user requesting it.

Web-to-Database interfaces can also increase productivity through cross-platform independence. The Web provides a gateway accessible by all modern systems. For example, even if a training department's class registration information is stored in a Microsoft Access database running on a Windows NT machine, any Macintosh user with a web browser can examine, change, and add to the database. Web-to-Database interfaces also enable remote database access with equal agility. Updating a calendar database while away from the office at a conference can be as easy as e-mail.

How does a Web site communicate with a database?

There are several methods now in use that allow Web-based input forms written in HTML to communicate with a database. There are two main approaches: Common Gateway Interface and Server Software.

The Common Gateway Interface (CGI) is the oldest and most proven method but may require a bit more complex coding to implement. It is a standardized exchange of information between four components: the user's Web browser (client), the server, a CGI program (script), and the database. Most Web server software is compatible with the CGI standard.

Consider the information flow when you are about to perform a search on the Web. You start by entering information into an HTML form. You enter text, click radio buttons, checkboxes, and menus. While you do this your browser is doing all the work, processing all your input until you press the submit (or "search") button. Once submit is pressed, your browser opens a connection to the server and sends all the information you've entered or selected in a stream. If the search form was set up to utilize a CGI program (designated in the ACTION attribute of the FORM tag), the server will send the stream to a separate CGI sub-routine running on the server. The CGI program's job is to separate (parse) the stream into usable chunks and send commands to the database to find records that match the search criteria. The database, after locating the needed records, sends this information back to the CGI

program. As its last responsibility, the CGI program constructs the HTML files that contain the search results for the user. The CGI program sends these files to the server which passes them along to the user's browser. The CGI functions as a gateway, interfacing the Web server software with the database, hence its name.

CGI programs are also known as scripts, primarily because they are often compiled just in time for execution rather than beforehand. One of the most popular languages for writing CGI scripts is Perl (Practical extraction and reporting language). Others include Tcl, Applescript, and C. Traditionally, it has required a fairly thorough understanding of one or more of these programming languages to build a Web-to-Database interface. For many people the task of learning and writing scripts in Perl will be a much longer learning curve than for HTML.

Server software is a more recent addition to the tools for linking the Web to databases. Unlike CGI programs that are entirely different programs running on the server computer, server software becomes an extension of the Web server software itself, adding functionality and the ability to interface with databases. Server software is sometimes called 'plug-ins'. Some Web-to-Database server software (such as Web.sql) allows for database query commands to be included in the HTML file as special tags. Others provide the same kind of functionality as CGI scripts. An important consideration with server software is that it is usually designed for a certain type of Web server software. That is, before buying server software plug-ins you must determine the type of server software your running (e.g., Netscape Enterprise Server, WebSTAR, Windows Information Server) and make sure the plug-in is compatible.

What tools exist to make linking databases to the Web easy?

Tango

Tango by EveryWare Software is one of the most fully developed tools for creating a Web-to-Database interface. There are three packages to consider: Tango for Filemaker, Tango for Access, and Tango Enterprise. Tango Enterprise is a comprehensive tool that allows Web-to-Database interface design for any of a large class of databases (ODBC compliant).

Tango works by installing a custom CGI program (the Tango Application Server) on your server computer to handle the kinds of database interactions you want available over the Web. You start with Tango Editor, a tool which allows you to create custom HTML forms based on the fields in your database. For example, if you have a frequently asked questions database with *topic*, *computer_type*, and *software_type* fields, Tango Editor allows you to drag these over onto a search form page. As each field is moved from a database window to a search form window, the fields become part of the HTML form that the user sees to start their search. Similarly, you define all the database fields in each record that you want the user to see when a result is returned. This way, only those fields that are relevant to the user's needs are displayed. Once all the search fields are in place, you can then add an additional level of detail by linking any search result field to another set of database fields. For example, suppose a user's search returns a record with a quick_solution field. That quick_solution text can be defined as a link to a set of other fields that give a more verbose description, if the user needs it. Search results can also be made updatable by the user. This might be useful in an internal registration database where you want your colleagues to be able to update student phone numbers or addresses via the Web.

Tango also provides easy creation of HTML forms that allow addition of new records to the database. Once Tango is configured to work with a certain database, HTML forms are created by dragging and dropping the desired fields into a construction window. HTML snippets can be added as headers and footers to each page to cutomize their look.

Once the interactive HTML forms are designed this way, Tango Editor must generate a final working copy of each document, called a 'Query Document'. Query documents are written in Tango's own proprietary interface language and can be later edited and modified. In order for query documents to become usable Web forms, they must reside on the server computer and Tango must install an accessory program to interpret them. This accessory program can be either a special CGI program (called an acgi program) if your Web server is CGI compatible or a Server Software application designed for a specific type of Web server. Which accessory program to use depends on the Web server in use and the system it's running on (e.g. Windows NT, Macintosh). Since Tango can make use of the CGI standard for its operation, it has the added advantage that query documents can be created on a Macintosh but implemented (run on) a Windows NT server. This would require the use of Tango Enterprise as opposed to Tango for Filemaker or Tango for Access. The following summarizes which Tango

packages work on which systems for the development phase and Web serving:

Tango for Filemaker

Development: Macintosh
Servers: WebSTAR
Approx. price: $349

Tango for Access

Development: Windows 95, NT 3.5.1 and higher
Servers: Microsoft Internet Information Server, Netscape FastTrack Server, Netscape Enterprise Server, Netscape Commerce Server, any WinNT Server supporting CGI
Approx. price: $199

Tango Enterprise

Development: Macintosh, Windows NT, Solaris SPARC, SGI IRIX, AIX
Servers: Any CGI compatible, as plug-in using ISAPI, NSAPI, WSAPI
Approx. price: $995 (5 users)

Another important question to ask is what databases can be linked to the Web using Tango. From the above, it clear that Filemaker Pro for Macintosh is directly supported. Oracle databases and Butler SQL for the Macintosh are also directly supported. Other databases (including Microsoft Access) are supported through what is called the Open Database Connectivity (ODBC) standard. Tango Enterprise supports any database that has an ODBC driver making it very flexible.

One of Tango's drawbacks is that it doesn't integrate with HTML editing tools very easily. Since each Web form is constructed via the database fields, it is not as easy to use existing Web forms (with perhaps more intricate layout) to generate the query documents. Also, Tango may not support Web access to some of the more sophisticated features of many database applications (e.g. calculations with Filemaker Pro).

In summary, Tango makes designing basic database search and update interfaces amazingly simple. Within an hour you can create Web-to-Database interfaces allowing multiple topic queries and record additions.

WebFM

WebFM from the Web Broadcasting Corporation is another powerful tool for building Web-to-Database interfaces but has a much more limited scope of use. WebFM, as its name implies, connects Filemaker Pro databases to the Web. Further, WebFM is a strictly Macintosh program and thus will only run on a Macintosh Web server (e.g. WebSTAR). However, if you are running a Macintosh server and need a Web interface to a FileMaker Pro database, WebFM provides more features than any other product on the market. WebFM functions as a Web server plug-in (as opposed to through CGI) to increase its response speed. WebFM allows for much more sophisticated operations than Tango, including use of calculation fields, repeating fields, and complex find operations. WebFM costs about $245.

A Quick Note About Java

I wanted to mention Java since it has received so much press lately and will, I believe, become an important part of Web-to-Database interfaces in the future. Java is a programming language designed for the Internet. It's programs are only half-compiled at the programmer's end, sent over the Internet to the user's machine where they are fully compiled and executed. Since Java is simply a programming language, its programs can theoretically do anything any other program can do. Its value in the world of database connectivity has to do with Java's ability to relieve some of the burden on the Web server. Whenever a user clicks the submit button on a database-connected HTML form, the processing is then put in the hands of the Web server, CGI program, server plug-ins, and database. All these components usually run on the server computer which can become quickly burdened if millions of users visit the site every week. Java programs can become a sort of middle-man in this scenario. By building a Java interface that does some of the initial calculation work on the user's input (e.g., adding a column of numbers, checking for data validity), the server is freed up to do the core database lookup. Some of the processing is put in the user's computer rather than the server.

At this point Java is in its early stage of development and easy-to-use tools for building Java interfaces are still in their infancy. As the technologies for connecting databases to the Web evolve, Java will most likely become more accessible to non-programmers.

Bibliography & Resources

Ashley, Ed and Epperson, Beth with DeRosier, Kerri, 1996. *Sybase SQL Server on the World Wide Web.* International Thomson Computer Press.

EveryWare Development Corporation
http://www.everyware.com
Phone: 905-819-1173
Fax: 905-819-1172
Email: support@everyware.com

Web Broadcasting Corporation
http://www.webfm.com
555 Bryant St. #386
Palo Alto, CA 94301
USA
Phone: 415-329-9676
Fax: 415-323-5418
Email: info@macweb.com

IBM DB2 Web Connectivity
http://www.software.ibm.com/data/db2/www/

Java Information
http://www.gamelan.com/index.shtml