When, Where, and What: Three Web-to-Database Applications for an Academic Environment

Kathryn F. Gates
The University of Mississippi
Oxford, MS

James F. Ball
The University of Mississippi
Oxford, MS

Carl E. Raffa
The University of Mississippi
Oxford, MS

This paper highlights three successful Web-to-database applications developed at The University of Mississippi as part of the WWW-based campus information service: (1) a campus calendar, (2) a campus map, and (3) a course schedule. Each application utilizes separate mechanisms for the storage and presentation of information in a manner consistent with the Dexter hypertext model. A Mini SQL database is used for content storage, and dynamically generated Hypertext Mark-up Language (HTML) is used for content presentation. Users benefit from a consistent, usable interface with a powerful search capability, and administrators see an improved environment for the ongoing maintenance and support of the information service. In the few months that these applications have been available, they have become indispensable resources to the UM campus community. They clearly illustrate the practical usefulness of Web-to-database technologies by providing efficient, universal access to a wide range of information.
When, Where, and What: Three Web-to-Database Applications for an Academic Environment

Introduction
The success of the World Wide Web (WWW) over the last few years has created new and exciting possibilities for offering universal, convenient, and immediate access to information. It is now common for institutions of all types, including universities, businesses, and government agencies, to offer information and services to their “surfer-clients” through the WWW. While this is revolutionary in itself, far more can be gained by combining WWW interfaces with well-known means for managing data through databases.

Common practice today is to use HTML text files for both the storage of information content and the definition of presentation specifications. This places significant limitations on the ways in which information can be accessed. The quantity of information available on the Internet demands more sophisticated methods for presenting and managing information. Not only do users need to search and access information in a variety of contexts, administrators need tools for managing information in a systematic and effective way. Real power and flexibility comes by separating the storage of information from the presentation of information. This paper presents three applications for an academic environment in which a separation was achieved by using a relational database for the storage of data and a WWW server for the delivery of data.

The individuals who developed these applications come from mostly scientific computing backgrounds and have significant experience with UNIX environments. Previous projects using Web-to-database technology include an online account application system (EASY) and a Web-accessible researcher database. EASY was implemented on a Free BSD UNIX system with a Mini SQL database and Perl CGI scripts. The researcher database was implemented on an SGI IRIX system with an Informix database and Perl CGI scripts.

Overview of UM Web Development
During the 1995 spring semester, University of Mississippi (UM) administrators appointed a committee to direct the development of the campus Web project. Until that point, the campus Web site (UM Web) had been managed solely by the Network Design and Engineering Support Group (NDES) in the Office of Computing and Information Systems (OCIS). The Web site did not adequately represent the many facets of the university; only a handful of departmental Web sites existed, and key units within the university such as Libraries, Publications and Public Relations had not been enlisted as contributors.

The original UM Web committee consisted of a diverse group of individuals, including students, faculty and staff representing academic departments, Libraries, Human Resources, Publications, Public Relations/Marketing, OCIS, Student Life, and Alumni Affairs. Soon after its formation, the committee was partitioned into two groups - one to direct the design of UM Web and one to address policy issues.

Design
The design committee began the task of organizing and molding UM Web into a more representative, attractive, and usable Web site. The committee struggled with how to present existing and planned content so that it could be accessed easily by the end user. One of the most significant decisions early on was to use a marketing approach in organizing the information content. The committee identified target audiences, and, with several brainstorming sessions, determined the information content that would be of interest to each group. The resulting groups were current students, potential students, faculty, alumni, and visitors. Each group was given a clear access point on the “front door” of the Web site (Figure 1).
UM Web as a Campus Information Service

Initially, the design committee focused on the role of UM Web as a recruiting/marketing tool in reaching potential students and faculty, alumni, peer institutions and funding agencies. As it evolved, UM Web became a critical campus information service. One of the first simple, but very successful, applications to be added was the faculty/staff directory. The completion of the campus network during the 1995-96 school year, which included the wiring of all residence halls, further facilitated the use of UM Web as a comprehensive campus information service.

Policy

The policy committee began the task of modifying the Computer and Network Appropriate Use Policy\(^2\) to include rules for individual Web pages. The committee also developed a separate policy for “official” UM Web pages (e.g., departmental Web pages)\(^3\). The work of this committee resulted in guidelines on the role of the university in monitoring individual Web pages, specifications of required elements on official pages, and definitions of appropriate and inappropriate content in individual Web pages.

The Ongoing Role of the Committee

The UM Web committee continues to play an important role in the development of the campus WWW site. By drawing on the talents of a diverse group of people who represent the many facets of the university, the chances of achieving a well-organized and balanced information service are increased. Editors, designers, and domain experts help with organization and layout, while the technical staff determines the best information access strategies. The committee assists in placing priorities on competing projects. Requests for special treatment (e.g., “My department needs a link on the front door.”) are referred to the committee for consideration, allowing the technical staff to focus on implementation.
Vigorous debates take place in meetings and through the committee LISTSERV on reaching a balance between aesthetics and accessibility. Providing forums for interaction between the various players is an activity that cannot be replaced and is critical to reaching the best possible product.

A Technology Champion
An equally important factor in the success of UM Web has been the presence and support of a high-level administrator to serve as champion for the project - someone who is genuinely excited about the technology, who provides encouragement, and who builds necessary support.

Enabling Departments to Become Information Providers
In parallel with the work of the Web committees, an effort was begun to have every department on campus create a Web page. Hands-on workshops were arranged by the university administration and OCIS consultants for all academic and administrative department chairs. Many of the workshop participants had little experience with computers, WWW, or the Internet in general. During these workshops, the benefit of establishing a Web presence was illustrated by helping participants connect to outstanding Web sites at other universities. Department chairs were asked to name departmental webmasters who would be responsible for creating and maintaining the department’s Web site. Seminars on HTML development were then arranged for the webmasters. In most cases, webmasters enthusiastically volunteered for the task. The pool of webmasters now includes individuals at all levels, including department heads, faculty, staff, and students. Just over a year ago, only a handful of UM departments and no student organizations had WWW sites. Today, there are more than 90 active departments and over 50 organizations on UM Web.

Motivation and Requirements
During the process of determining what items would be of interest to various target audiences and to campus users, the need for three new Web-based applications became clear: a campus calendar, a campus map, and a course schedule. All three applications had existing, paper versions. The Physical Plant, together with Publications, supported a campus map that was distributed through the Police Station; Public Relations/Marketing distributed to the media a monthly calendar of campus events; and Student Life, OCIS, and Publications jointly produced the course schedule. By making these items available through UM Web, anyone with WWW access, on-campus or off-campus, could reach them at any time.

Campus Calendar
Several departments maintained calendars, and, in some cases, the calendars were available on-line from the departmental Web pages. Public Relations/Marketing maintained a comprehensive calendar that was distributed to the media each month. Because the information was scattered among several sources, some on-line and some not, it was difficult to answer the question, “What is happening on campus today? … this week? … this month?” Members of the design committee came to the conclusion that a comprehensive, Web-accessible calendar was needed. The calendar would serve as a consolidated source of information, including athletic events, concerts, art exhibits, brown bags, intramural activities, seminars, meetings, and so on. The online calendar would need to support searches by content, event type and date. Furthermore, maintenance of the calendar would be distributed among several departments and accomplished through a WWW interface.

Campus Map
Every university Web site, including UM Web, needs a good map for campus visitors, new students, existing students, faculty and staff. (Even long-time UM staff members reference the on-line campus map before heading off to meetings across campus!) Fortunately, existing materials - a layered campus map created in AutoCAD by the Physical Plant - could be re-used for the on-line version.

A major consideration for the on-line campus map was image size. Many UM students use modems to reach the Internet and UM Web; because of this, large images can lead to much frustration and make access difficult or impossible. Before determining the requirements for the UM campus map, the developers and design committee members explored other university maps. In some cases, detailed campus map images
were found to exceed 100 kilobytes in size. Developers set the following guidelines for the UM campus map images:

- Use no more than 16 colors. For the UM campus map, 16 colors were more than sufficient, and when compared with 256 colors (8 bit), the savings in size was significant.

- Use vector-drawn graphics for source images. Bitmap graphics lose information about fonts and lines. This is most apparent when an image is resized, a necessary requirement for the UM campus map.

- Provide a "zoom" feature. From a top-level view most campus maps look cluttered, but many users like to see the entire campus on one screen. A zoom capability provides necessary detail and removes the need for large images.

An additional requirement was that developers had to work from existing map images. Some wanted three-dimensional maps, and others wanted more attractive images. A practical consideration was that the maintenance of the map images stay with the Physical Plant and that a method for easily converting AutoCAD images to legible GIF images be found.

A campus map requires support for various types of searches such as by department and by building. The map must be easy to maintain, and data must be presented in a consistent manner from screen to screen.

Course Schedule

Placing the course schedule on-line offered many benefits. To keep costs to a minimum, the paper version of the course schedule is printed on thin newsprint with a small font size, making it difficult to read. Frequent changes occur after the initial printing. During pre-registration and registration, almost one hundred copies of the active schedule, containing section limits and current enrollment, are printed each morning and distributed to various offices for advising students. An on-line version was needed which would support alphabetical listings of courses, searches by course number, instructor, or location, and which would contain near-current enrollment data.

The Model: Separating Content and Presentation

Over the last decade, the hypertext community has developed research on the design and implementation of hypertext systems, much of which can be applied to WWW environments. The Dexter Hypertext Reference Model was initiated in 1988 with the goals of providing a basis for comparing systems and establishing a common terminology for the hypertext field. Three layers of a hypertext system are defined in the Dexter Model: storage, run-time (or presentation), and within-component. The storage layer handles the storage and retrieval of components and links. The run-time layer handles the display of documents and interactions with the user. The Dexter Model uses presentation specifications to define the interaction between the storage and run-time layers.

The developers selected a model loosely based on The Dexter Hypertext Model. A Mini SQL (mSQL) database is used for the storage of data. A httpd daemon, with mSQL-enabled Common Gateway Interface (CGI) programs, functions as the presentation layer, delivering dynamically-generated HTML text to the end-user (Figure 2). Presentation specifications are defined as part of the CGI scripts.

![Figure 2. A Model for a Web to Mini SQL Application](image-url)
Benefits to the User
This model offers many advantages to the end user. Database items can be presented to the user in multiple contexts and from multiple avenues. For example, a user can view information for a particular department in the context of the campus map, the departmental directory, or as a link from the faculty/staff directory. Information is stored once in the database and is then presented differently depending on how it was requested. Presentation specifications for a given item are localized to a CGI program; as a result, all analogous items (e.g., events) are presented in a consistent format. Advanced searches are possible through the built-in search capabilities in the database.

Benefits to the Maintainer
The model also offers many advantages to the individuals who will support the information service. The task of providing accurate and current information can be accomplished more easily by applying the database principle: one fact in one place. In systems where HTML text files are used for both storage and presentation, information content is often duplicated for each context in which it occurs, resulting in difficulties with ongoing maintainance. With a Web-to-database model, re-use of information content in multiple contexts is possible. Information content is managed through the tools available in the selected database.

Implementation Decisions
The same tools were used for each application: a Mini SQL database was used for the storage of information content, the Apache WWW server was used for the delivery of information, and CGI programs were used for communication between the httpd daemon and the mSQL daemon. CGI programs for the map were implemented in the C programming language, while CGI programs for the calendar and course schedule were implemented in Perl. By using software in the public domain, costs were kept to a minimum and the critical parts were readily available to the developers and the users.

The Mini SQL Database
The UNIX-based Mini SQL (or mSQL) database is a lightweight relational database engine. The developers chose Mini SQL for its low cost, simplicity, speed, and robustness. Another benefit of using Mini SQL with the Apache WWW server is that an authentication module is available, which allows one to store user id and password in a Mini SQL database. Several types of authentication are supported and can be specified through modifications to the Apache access.conf configuration file.

The calendar required an events table, an event types table, and a calendar administrator table (for authentication). The campus map required a departments table, buildings table, and a table containing building-department relationships. The course schedule was implemented with a separate table for each semester of course data.

CGI Programming with Perl
The calendar and course schedule CGI programs were written in the Perl programming language by Larry Wall. The powerful and elegant Perl syntax made it possible to develop this application quickly and easily. While Perl programs may not run as efficiently as programs written in other languages such as C, Perl performance was more than adequate for this application. The MsqlPerl Adapter by Andreas Koenig was used to make Perl 5.0 scripts capable of connecting to and communicating with the mSQL daemon.

CGI Programming with C/C++
The campus map was expected to be a popular and frequently-visited application. Because the WWW server hardware resources were already overloaded, the map CGI programs needed to retrieve and modify an image quickly after the Mini SQL database was accessed. While either C or Perl could handle the implementation of these map programs, the developers chose C because compiled C code normally executes faster than interpreted Perl. The Mini SQL package includes a library of C routines for accessing and modifying a mSQL database and, like the MsqlPerl Adapter, the C routines are high-level, which makes programming easier.
The gd GIF C Library

A requirement for the campus map was that locations - buildings and departments - be indicated with an arrow on the campus map image. While perusing the WWW FAQ file, developers found a reference to an image manipulation library, the gd GIF C library, developed by Thomas Boutell. The gd GIF C library contains routines for creating and manipulating GIF images, including polygon drawing routines. Developers used the polygon drawing routines to place an arrow on an otherwise unmarked campus map image. Coordinates for determining how the arrow should be placed were stored in the database along with other building data.

Production Details

To create the master campus map image, the appropriate layers - buildings, roads, and sidewalks - were selected from the Physical Plant AutoCAD map and exported to an encapsulated postscript file. The master map was edited to create zoomed sections. Two tools, ghostview and ImageMagick, were used to create four-bit images (sixteen colors). The xpaint tool was used to touch up the images, add color, and determine x and y coordinates for each building. Student workers built the department lists, including phone numbers and addresses, from the campus phone book and undergraduate catalog. Staff members and students shared the load of taking photographs of campus buildings. Image map files were created for main map and six zoomed maps, and CGI programs were developed to access the database and dynamically generate the appropriate Web pages.

The course schedule required two major activities: (1) populating the course schedule database and (2) building the CGI programs to support communication between the WWW daemon and the mSQL daemon. The raw course schedule data was obtained from the UM administrative mainframe. A student worker wrote a Perl program to parse the course schedule data file and populate a Mini SQL database. Staff members wrote Perl scripts to process user queries, construct the appropriate select statements, retrieve data from the Mini SQL database, and then display the retrieved data using HTML. In production mode, the raw course schedule data file is automatically transferred to the WWW server, and UNIX cron jobs on the WWW server periodically refresh the course schedule database by parsing the new course schedule data file, dropping the old course schedule tables, and building new course schedule tables.

Building the campus calendar required defining the set of possible event types, designing the appropriate database tables, and developing the CGI scripts to support the necessary queries. Modules were added to allow certain individuals to add, edit, and remove events. Authentication was handled by htaccess password protection using the Mini SQL/Apache authentication module.

Performance

Overall, the three applications have been very successful. The online course schedule was used for the first time during the 1996 Spring semester. During the weeks before fall pre-registration began for the 1997 Spring semester, students probed the Provost’s office for information on when the WWW-based course schedule would be available. The on-line course schedule was available more than a week before the paper version came out, and it received hundreds of “hits” only days after becoming available. The on-line course schedule continues to be a convenient and helpful service to students, faculty, and staff.

There were several unexpected benefits from each application. For example, an administrator needed to know which classrooms were used for what purposes. By searching on location within the on-line course schedule, the question could easily be answered. An effort is now underway to display calendar items through the campus cable television service.

Responsiveness/Access Time

When the campus map was first made available, the UM WWW server was an SGI Indigo system which also served as the campus POP mail server, news server and gopher server. The response time on map queries was unacceptable; as a result, developers temporarily moved the map to a separate workstation. After the WWW server hardware was upgraded to an SGI Challenge L system, the map was moved back. Since then response time has been more than adequate.
An important benefit of Mini SQL is its speed. To retrieve all 3000 records from a typical course schedule table using the mSQL interface requires less than 1.5 minutes of combined user and system CPU time on the UM SGI Challenge L system. To select one record from the 3000-element course schedule given a 4-character code requires .07 CPU seconds. To search more than 450 events for a given string in the title field requires .1 CPU seconds. A significant concern when implementing Web-to-database system is the cost of database accesses. With Mini SQL, those costs are practically negligible.

**Usability**

Students were enlisted to help with the design of the user interfaces for the map, calendar and course schedule. Without a doubt, the students had the best ideas for making the applications more usable, and the students were not shy about expressing their opinions. The campus calendar uses HTML frames for its interface, making it easy to toggle between events for a day, week or month. A future enhancement to the calendar will be to display events in traditional calendar format. The Web-to-database design used in the calendar facilitates enhancements to its user interface.

Because the campus map was designed to support dynamic queries on buildings, other applications, such as the UM Just the FAQs service, can easily link to the map when buildings are referenced, resulting in a more usable interface. From Just the FAQs, the user is given building location in text as well as a hyperlink to the location of the building on the campus map.

When the campus map was first demonstrated to UM administrators, one individual sat back, not touching his mouse or keyboard, mumbling, “I don’t have time for computers.” After a few minutes of watching the other administrators having great fun with the map, he joined in and was one of the last to leave the demonstration. The developers interpreted this as an affirmation of the map’s usefulness!

**Outlook**

Since the implementation of the campus calendar, the campus map, and the course schedule, several other Web-to-database applications have been added. One of the most recent applications, Just the FAQs, provides the information that students and parents will need for a successful tenure at the university. Academic and administrative departments submit the most frequently asked questions (FAQs) from their areas; individuals can then access FAQs by topic or department or search FAQs through UM Web. The automated campus slide show and on-line faculty/staff, departmental and individual Web page directories are implemented with Mini SQL databases and Perl CGI programs. A project which allows distance learning students to submit homework assignments through WWW forms is in progress. Also in progress is an Internet Directory of Faculty Experts, which will encourage media relations with UM faculty members who can offer professional commentary on current issues and events.

Several older sections of UM Web need to be re-worked to access the databases. For example, the campus tour can be improved by accessing the campus map and departmental directory. An effort is underway to allow students to register for courses through a WWW interface, in addition to by telephone. This would require “live” access to the administrative systems and is being pursued by the OCIS administrative services staff.

While MsqlPerl works well for implementing Web-to-database interfaces, it would be useful to move to a three-tier approach in which multiple databases are supported. This arrangement would allow the developer to choose among several databases, such as Mini SQL, Oracle and Informix, with only minimal changes to CGI programs. Another exciting prospect for enhancements to the three applications is the Java-to-Mini SQL interface. Java classes are readily available for accessing Mini SQL database items. Many more alternatives are available for enhancing the application user interface with Java.

**Conclusion**

In the few months that the calendar, map and course schedule have been available, they have become indispensable resources to the UM campus community. These applications clearly illustrate the practical usefulness of Web-to-database technologies by providing efficient, universal access to a wide range of information.

*Special thanks to Maggie Klepzig (http://www.olemiss.edu/~ccmaggie) for her assistance with graphic images.*