Automating Course Management

Ron Zucker, Jeanne Zucker
Computing Systems and Technology
Faculty of Business
Auckland University of Technology
Auckland, New Zealand
ronald.zucker@aut.ac.nz

via the Web

ABSTRACT

This paper is meant to demonstrate a prototype of a Course Management System that is used to produce web pages with a consistent look and feel. The overall goal is to produce a system capable of developing Lecturer's Home pages and Course Synopses web pages with little or no knowledge of HTML. The resulting system provides lecturers home pages and direct links to individual course synopsis based on a predetermined template and data structure. The template and individual data can be changed and new web pages can be generated with little effort. It is meant to stimulate further efforts to produce tools for producing consistent web pages in order for students and faculty alike to gain access to course materials via the worldwide web. We intend to make the software freely available to any interested parties. We look forward to shared collaboration in the future.

1. INTRODUCTION

Spending too much time on course and academic administration? Then this paper may be of interest to you. This paper is the result of an effort to reduce some of the course workload in maintaining web pages and home pages. The resulting Course Management System, using a relational database as the central core, provides a relatively simple way to create lecturer "business card" pages, which include hyperlinks to the synopsis of the courses offered by the lecturer. The synopses are a result of the same system. The Course Management System (CMS) can be a prototype for a much larger and more comprehensive Academic Information System that could eventually handle all aspects of course and curriculum management.

2. HISTORICAL BACKGROUND

AUT has been disseminating course information via network drives. Some drives are the domain of the faculty, while others are for student access. Lectures are typically stored on the drives available to the students so that the students may access them

after the lecture has been presented. Additionally, handouts are provided to the students during the lecture. Most teacher-student communication is through live lectures augmented by photocopied handouts. In some cases the handouts are sold as a booklet at the bookstore for a nominal cost, while others are photocopied prior to each lecture. The latter is quite costly in terms of time and photocopying expense. Some students miss the lectures and request the handout at a later session. This places an additional load on lecturers, requiring them to carry the semester's handouts for those who missed out.

AUT is now transitioning into web developed content. This move is not without problems or concerns. "One of the main problems with this approach is the sheer amount of time it takes to create a web page." Matt Melchert (2000). Sunil Hazari (1998) points out, " Not all faculty have embraced the idea of moving courses for online delivery. Faculty resistance has resulted from perceived notion of being intimidated about rapidly changing newer technologies, constantly keeping up with a need to update skills, losing control over the teaching process, dealing with copyright issues by accessing or making available materials for open access. " Other concerns include student's acceptance of the new delivery system, though anecdotal evidence has shown that the majority of students embrace the use of web based content.

3. BUILD VERSUS BUY

The question may arise, "Why not simply buy a tool such as WebCT or BlackBoard to develop web content?" The answer to this question may be found at the bottom of theTable 1 created by Catherine L. Poyner (1999). Campus Wide Information System/ Module and Integration with Campus systems is costly, difficult, or non-existent. Another concern is that these products are course centric rather than faculty, school, programme, or department centric. "With many course development products currently available and new ones being developed (both commercially and as university projects), there does not seem to be any standardization and interoperability between course development systems …which would make it possible to exchange materials between courses" (Hazari, 1998). The prototype CMS views a programme of study as a series of integrated modules with an integrated look and feel, rather than a collection of independent parts. Blackboard and WebCT concentrate on individual course content, student collaboration efforts, testing, marking, and recording, all of which are valuable tools. These products appear to be looking at the trees. The CMS approach is to look at the entire curriculum as a series of courses with aims, outcomes, and timetable information that can

4. CONSTRAINTS

The system is being developed with the following self-imposed constraints:

4.1 Software Components Must be Free:

The software components must be taken from open source and free (not shareware). Sun Microsystems Java (JDK1.3) is the language used for all of the programming aspects of the project. MySQL has been chosen as the final database engine. It should be noted that Microsoft Access is being used to develop the prototype, but will not be used in the final product.

4.2 WWW will be the Main Communications Network:

Using the internet instead of telnet allows users to access and update data from browsers rather than telnet sessions or file FTP.

4.3 Reduced Coding Requirements:

This constraint is meant to allow the web page developer to modify page content without programming, separating content from presentation.

4.4 Minimum Technologies/ Languages:

Web developers have access to a variety of tools and languages to assist them. Unfortunately all of these tools have a learning curve associated with them. Minimising the number of different languages and technologies helps to keep the system simple. Here

be combined to make a forest. Ultimately this can be used to detect missing and/or duplicated topics from the curriculum. Lecturers will maintain the CMS data in support of their coursework; thus the data will be an accurate reflection of the actual versus envisioned curriculum.

Features	Web Course In a Box	WebCT	Blackboard CourseInfo	Vista Compass
Cost: software	Free	$2000	$4,500	$6 per student
Cost: Technical Support from Company	$3,000.00 required	Good Free	$500	Good Free
Simple Interface	Good	Fair	Good	Good
Security	Fair	Fair	Fair	Good
Integration with other software	Poor	Good	Good	Good
Integration with Learning materials (ex. McGraw-Hill)	None	Good	None	None
Integration with 3rd party authoring tools	None	Good	None	None
Bulletin Board	None	Good	Good	Good
Threaded Discussion	Good	Good	Good	Good
Synchronous (Chat)	Good	Good	Good	Good
Testing	Fair	Fair	Fair	Fair

Features	Web Course In a Box	WebCT	Blackboard CourseInfo	Vista Compass
Gradebook	None	Fair	Good	Fair
Access Tracking	Fair	Good	Good	None
Multimedia	None	Good	Good	None
Flexible Page Design	Good	Good	Good	Good
Foreign Language Characters	Poor	Fair	None	Good
Document Searching	None	Good	Good	Good
Backup Tool	None	Good	Good	Good
CAMPUS WIDE FEATURES				
Campus Wide Information System/Module	$2000 more	Available Yet Complex	$150,000 or more	Good
Campus Technical Support	Good	Good	Good	Need Training
Integration with Campus systems	None	None	None	Good

Table 1: Web Course Management Tools Comparison Chart

http://www2.truman.edu/~cpoyner/gridcomparingproducts2.html

again Java was chosen. The reasons for choosing Java include:

- A single language that can be used on a Windows platform then ported seamlessly to the UNIX/Linux server environment
- Web friendly language
- Object Oriented language.
- Flexibility: Java supports applets, server pages, Servlets, XML etc. Applets are small programs that will run on the client computer, using client cycles instead of server cycles. Applets are downloaded when an HTML <applet> tag is encountered. Java Server Pages (JSP) are HTML documents with embedded HTML tags that allow access to Java Servlet code. JSP also supports XML data. The

embedded code is run on the server side, thus allowing access to otherwise prohibited areas such as database retrieval. Servlets are stand-alone server side Java programs that can provide middletiered applications that can be useful for accessing datastores. Servlets are triggered by web requests (such as CGI scripts).

- Only one compiler and language used throughout. It should be noted that Jigsaw and TomCat (webservers used to help develop and test the prototype) are written in Java.

It also should be noted that the author's knowledge of Java and lack of knowledge of Perl, CGI, PHP, or Python were significant factors in this decision.

5. OVERVIEW OF THE PROTOTYPE

By creating a Course Management System the workload necessary to develop web pages, and maintain a database containing course, workload, and lecturer information is eased. There are two major components to the Course Management System. First, and foremost, is the database for maintaining course content, lecturer information, schedules, etc. Second, a series of Java programs (appplications, servlets, Java Server Pages, and applets) are used to convert the raw data into web pages for dissemination.

In the prototype, the database is maintained via a set of Java applications and via direct updating using SQL. Using SQL to update the database is being eliminated as more programs come on line. These programs allow the lecturer to enter/modify timetable information, aims, and assessments in a fairly straightforward manner, without any knowledge or concern for HTML or the final format. By utilising the database, this information can easily be retrieved in a number of ways. The initial fields used in the database have been defined to meet the immediate needs of home page and synopsis creation. For example, suppose there was a request for the aims of each course in the curricula. The data can easily be created using a simple SQL statement. Course content can also be obtained in a similar fashion. This information can be invaluable for curriculum development and assures that the course content is as planned. Once the timetable is laid out, MS PowerPoint can be used to develop lectures. The final presentation can also be saved as an HTML document. Adding a hypertext reference to it from the course page or timetable completes the connection. Thus the student can access the faculty business card page, hyperlink to the course page, which contains links to the synopsis, lectures, and assignments. As the system becomes more widely used, additional fields relating to programme origination, module descriptions, Committee on Academic Programmes (CUAP), Programme Approval and Review Committee (PARC), and data supporting the IMS Learning Resource Meta-data Information Model can be added.

6. THE DATABASE

The prototype database tables are primarily based on:

- Lecturer data
- Course Information
- Lecturer schedules

Additional tables include:

- Academic calendar data
- Assessment data (but not grade reporting)

One table was created that contains templates for the web pages themselves. The template table contains the HTML documents necessary for creating the web pages. Once a layout has been agreed upon, the HTML document is saved in a table. References are made to various sections of the document with sequence numbers providing the proper sequencing. This provides quasi format independence. A Java program merges the template data with the Academic Information tables to provide web pages with a consistent look and feel. The disadvantage to this solution is the need to modify Java source code if and when the web pages undergo significant changes. This requires staff with technical skills in Java. JSP and XML are being investigated as replacements to this solution.

7. WEB PAGES CREATED BY THE

SYSTEM

The web pages created by this system are meant to be presented in a standardised format. Obviously there is flexibility in the design, allowing lecturers to meet specialised circumstances, but the benefits of the system are lost if each lecturer chooses to create his/her own individual pages. Initially the formats were stored in a database by section. A Java application (later a Servlet) was created to construct the web page using the sections interspersed with selections from the database for the individual page. This method had the advantage of creating static pages that could then be retrieved rather than building the pages "on the fly". The main disadvantage of this approach has been that the data and presentation are bound together by an application. If simple modifications are made to data or format then no programming changes are required and the modification is straightforward. The difficulty comes when the changes require modifying the sequence of sections or data retrieval with a different presentation style. If this occurs, either a new Servlet must be written or the existing Servlet modified, re-compiled, and tested. This requires more than a passing knowledge of Java. JSP allows the developer to create both presentation and data retrieval in one document providing greater freedom in form and content. The JSP developer is still required to have knowledge of Java but not quite to the extent of the Servlet approach.

7.1 The Business Card Page

The business card web page shown in Figure 1 is simply a home page that conforms to some sort of standard or format as defined by the institution or group. It provides a standard format that can be readily and easily understood and used by readers. This is the "front door" to the faculty member. Any personal creativity the faculty member decides to demonstrate may be located on his/her home page, which may be linked from the business card. Since the card has a consistent look and feel, it can be represented by a template, with data supplied via the central database. Examples of possible data appearing on a lecturer's card are:

- name
- title
- locations
- phone

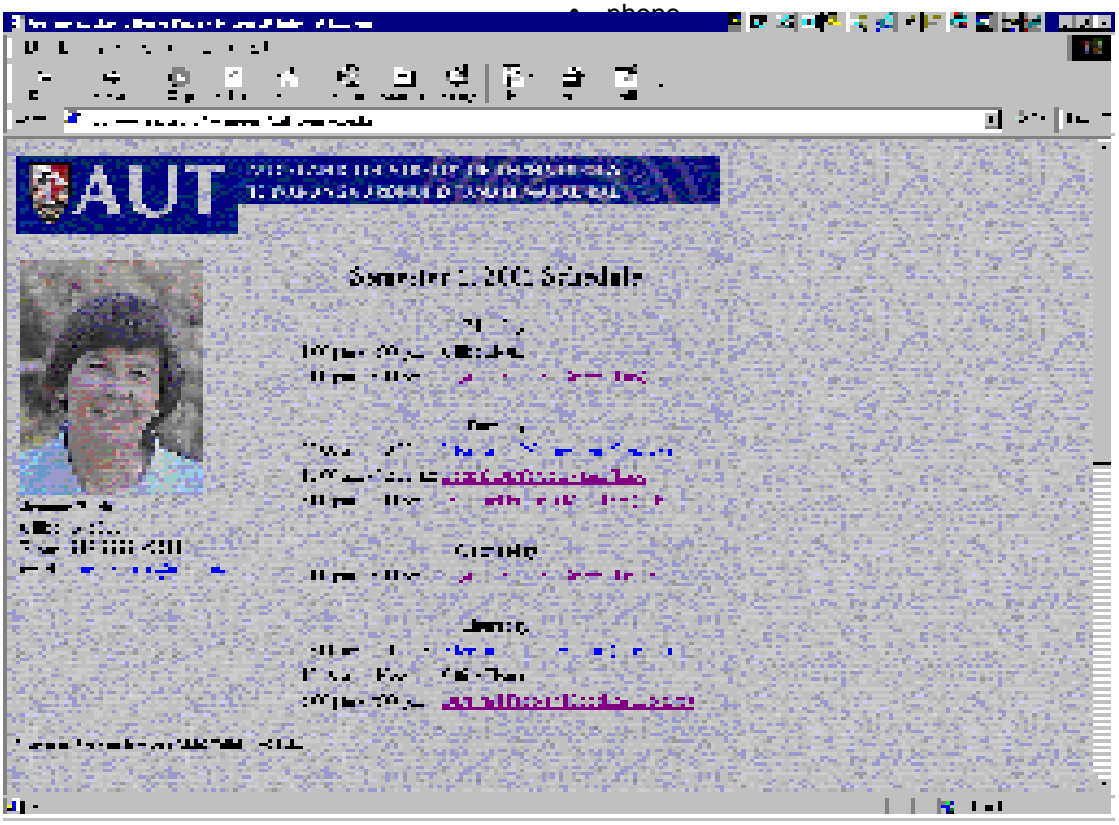

Figure 1
Business Card Webpage

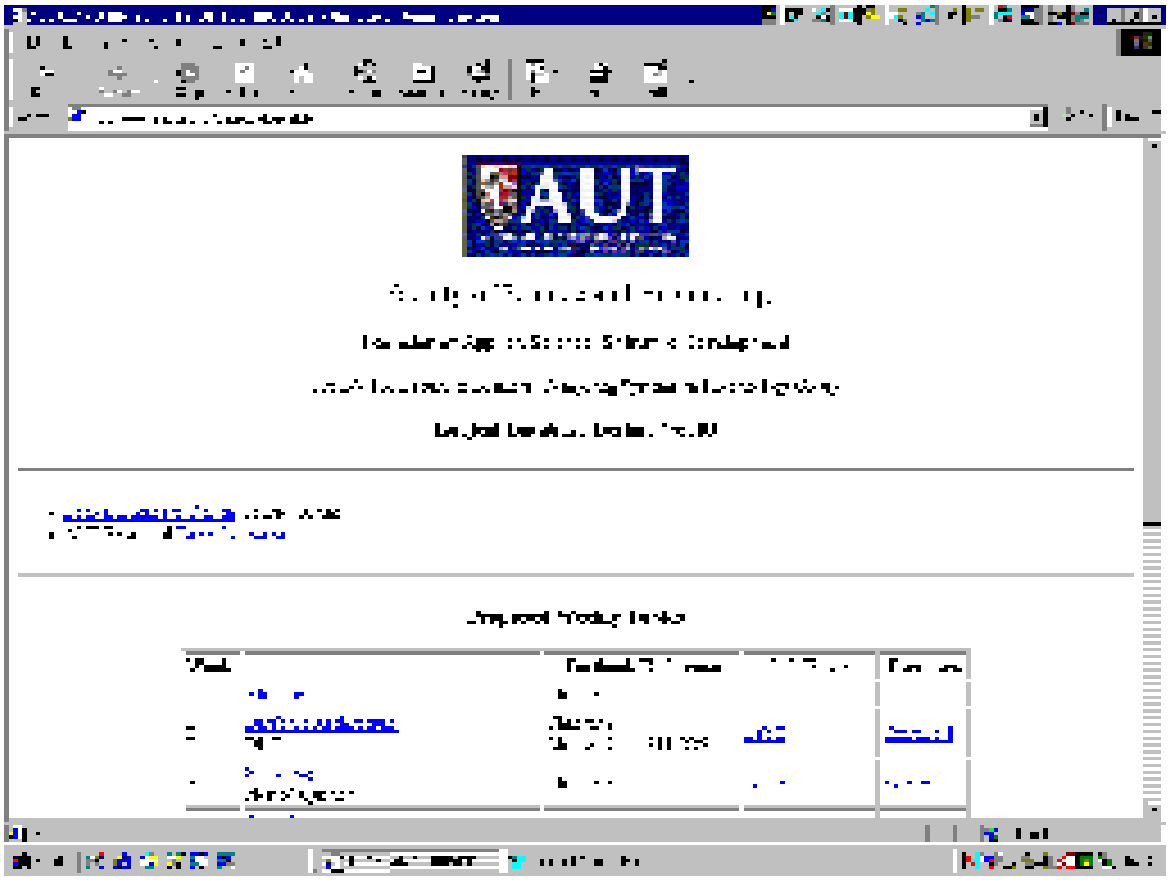

Figure 2
Course Page

- Fax
- e-mail address.

In addition, the lecturer's schedule for the current semester, including hyperlinks to course information are included. The whole idea behind the business card is to provide a concise, professional view of the lecturer. Additional information should be added sparingly.

7.2 Course Page

The Course Page, shown in Figure 2, is a working document that contains day to day postings including links to the lectures, assessments and any "handouts" or reference sites as they become relevant. Additionally, there is a link to the course synopsis. This page could be automatically updated as dates and weeks pass.

7.3 Synopsis

Most faculty members tend to create their own format (or borrow from others) and reuse the format over several papers. Few lecturers begin from scratch each time they develop a synopsis. Again the pattern is clear, a template is created, then data that is pertinent is supplied to that particular paper. Synopsis data is often bullet oriented: aims, outcomes, timetable data, assessments, etc. This makes the data conducive to relational database storage and retrieval. (See Figure 3, CMS Relational Tables.)

8. UPDATING THE DATABASE

Without question the most difficult technical aspect of this system is updating the database. For simple updates Servlets triggered by HTML forms were used. Lecturer information falls into this category because it requires (at this time) only name, phone,

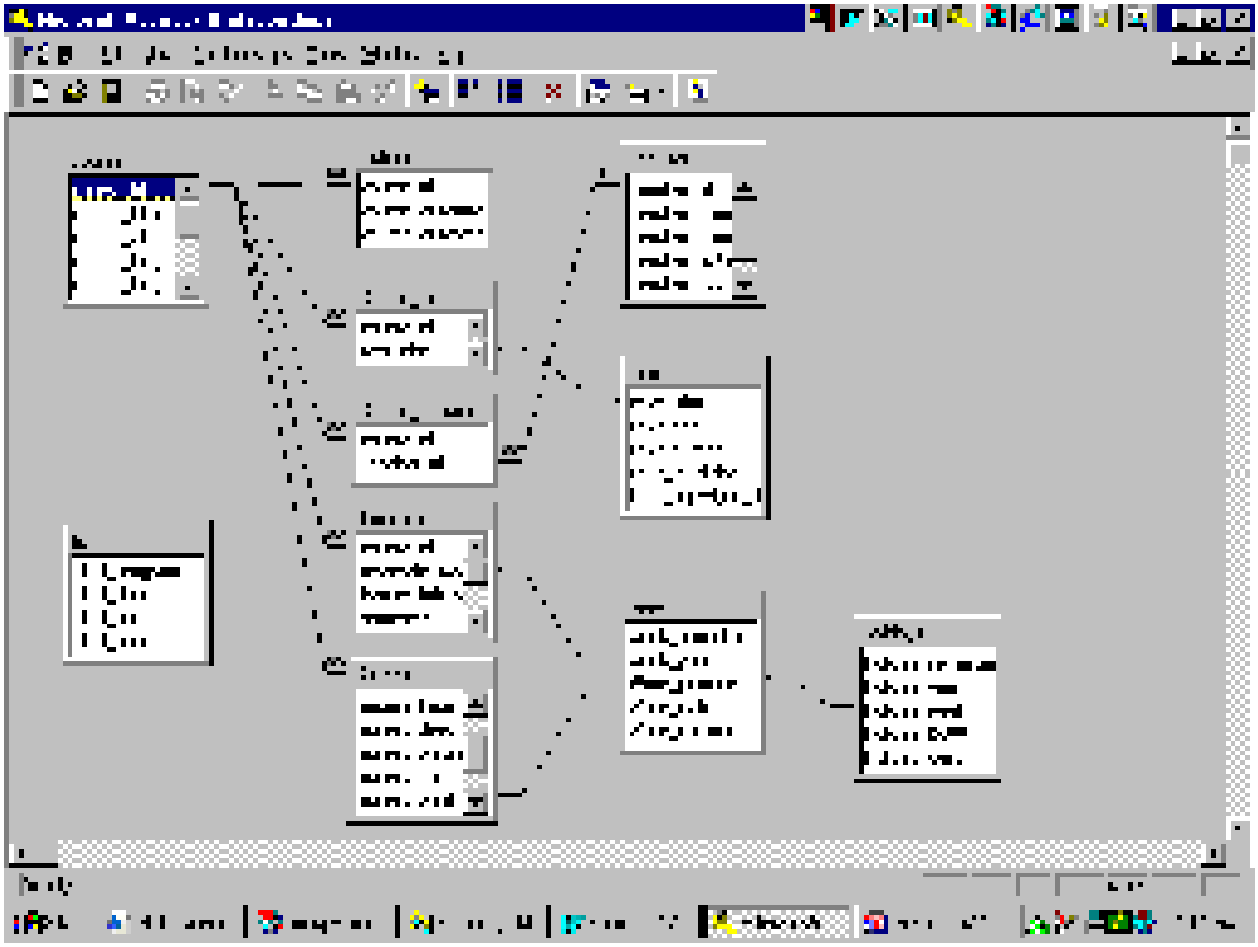

Figure 3: CMS Relational Tables

e-mail address, location and similar information. This data can be simply entered into an HTML form (see Figure 4 Simple Lecturer Entry Form). As the data becomes more voluminous and repetitious as in course outcomes and timetable data entry where tables of data must be added, entries deleted, moved, or copied, the simple HTML form can be cumbersome. Applets can provide some rather elegant solutions. The world is a much less friendly place then it was many years ago. Hackers invading your sites and changing content, viruses threatening your site and indeed, your entire network are some of the perils of computing on the web. Java's applet technology provides a much more user-friendly GUI base environment for entering and editing data (see Figure 5, Java Applet for creating Academic Calendar). The problem is that Java does not want to be responsible for the dangers described earlier. Thus updating a database from an applet requires a less than trivial solution. Sun has provided some interesting solutions. Remote Method Invocation (RMI) and trusted applets are options but these were rejected. RMI requires a dedicated application to be continuously running on the server and trusted applets require high overhead in key creation and distribution. Fields (1999) has provided some Java coding tips for posting data from an Applet and receiving data from a server. These tips are being explored as a solution in order to use Applet to Servlet technology for a more robust and user friendly interface. While allowing data to flow between the client to server and back to the client makes updating the database much more user friendly, it also makes the database vulnerable to unauthorised corruption. Clearly a userid/password combination is in order but the authors are not experts in web security and are not sure if a simple userid/password combination is adequate protection from attack.

9. FUTURE DEVELOPMENTS

1. Connection to existing scheduling system for automated lecturer "business card" creation. If the system can get access to the same database that is a result of scheduling papers, it is not difficult to imagine a day when lecturers may go to their own business card to see what their schedule will be.
2. Increasing database content to include PARC and CUAP data for "cradle to grave" course management. Much of the data used on the form for proposing a new programme can be cut and pasted into the module description and later on into the synopsis. Such data includes aims, outcomes, delivery methods, assessments, prerequisites, corequisites etc.
3. XML, XLL, XSL, DTD, DOM for data handling and formatting. This technology is quickly maturing and may become the information storage and transfer vehicle of the future. When the XML database is constructed (not necessarily a trivial effort) the web pages could be formatted using stylesheets (XSL) or transformations.
4. Possible expansion of Applet, Servlet, and JSP applications, including on-line quiz and assessment capability, student collaboration activities (similar to WebCT, Blackboard and others)
5. The database has many uses beyond providing web page data. Often, faculty is required to organise curriculum, course aims, course outcomes, content, etc. Conversations with colleagues have resulted in a series of reports and enhancements. Some of the future enhancements include: Tickler reports indicating important assignment due dates, moderation efforts and end of year activity reporting indicating the papers and tutorials taught. This means locating data, cutting and pasting and reassembling existing data in many ways. Having the data contained on a 4GL database allows for ad hoc reporting in a fairly painless way. Harnessing the e-business concepts of XML, XSL, Java Servlets, JSP, Applets can increase the dynamic capability of web pages.

10. CONCLUDING REMARKS

1. The system has great potential for development/ expansion as discussed in future enhancements (above).
2. This system was used as a student project which introduced students to:
   - e-business concepts
   - emerging web technologies.
3. Students have found this to be a relevant assignment that provides challenges in web development, database design and use, communications, and security. The students use the technology so they are users as well as developers of the system, which aids them in the analysis and design phases.
4. The system provides a useful test bed for emerging technologies.

E-Business, e-commerce, e-nough already? In many cases industry is now emerging as the leader in research. Vendors are now developing new technologies and rushing them to market as never before. Keeping up with these technologies is a daunting job for industry and academia alike. This system is not just a theoretical blue-sky application but a practical application that uses the technology and provides a useful product for students, faculty, and administration alike! In developing this particular system, we undertook many technologies and are still exploring other implementations while maintaining the original constraints.

REFERENCES

AUT Academic Policies and Procedures(2001), AUT Manual for the Approval of New Programmes and Major Changes to Programmes.

Fields, Duane K. (1999) , Applet-to-Servlet Communication for Enterprise Applications. Retrieved [5 May, 2000] from the World Wide Web: http://developer.netscape.com/viewsource/fields_servlet/fields_servlet.html

Hazari, S. I. (1998). Evaluation and Selection of Web Course Management Tools. Retrieved [23,April, 2000] from the World Wide Web: http://sunil.umd.edu/webct

2000 IMS Global Learning Consortium INC (2000), IMS Reource Meta-data Information Model Version 1.1 - Final Specification.

Melchert (2000), Course Web: A Report on Converting Course Notes to Web Page, Proceeding of the NAACQ,2000

Poyner(1999), Retrieved [19 April, 2001] from the World Wide Web: http://www2.truman.edu/~cpoyner/gridcomparingproducts2.html

Shacor, Gal (1999), Tomcat - A Minimalistic User's Guide, Retrieved [29 March, 2001] from the World Wide Web: http://jakarta.apache.org/tomcat/jakarta-tomcat/src/doc/uguide/tomcat_ug.html