Abstract: The Internet in the modern day provides the developer with a multitude of functionality in the delivery of information. The role of developer lies primarily in the presentation of content. In fulfilling this role it is not a requirement that the developer generate the content to be published. This is indeed the case in most business environments and certainly is the case in regards to the Department of Electrical Engineering. In such an environment the edit-review-publish lifecycle becomes a ‘Webmaster/mistress bottleneck’ when the amount of content or content types becomes in-feasible to maintain both technically and organisationally. In this scenario there exists a need for a Content Management System.

Project Goals

It is the primary goal of this project to develop a content management system to increase the effectiveness and efficiency of content elicitation and delivery for the Undergraduate web domain.

In fulfilling the primary goal the following sub-goals must be met:

- The system must allow staff to easily generate/update content without prior knowledge of complex HTML concepts.
- The system must provide a method to dynamically provide content for publishing.
- The system must provide a means to publish content.
- The system must provide a means for database maintenance.
- The system must cater for a majority of browser technologies.
- The system must be developed utilising the software development life cycle.

Introduction

In the early days of the Internet, web sites consisted of a small number of hand coded pages. The content was typically simple descriptions of a business and its product. Changes were made by the Webmaster/mistress, an individual with the technical skill to code the HTML pages and monitor the relationships between linked pages. This form of content management ultimately led to a ‘Webmaster/mistress bottleneck’. [1]

As content changed its method of delivery did not, and most web domains retained static pages for their means of delivery. In organisations such as the Department of Electrical and Information Engineering where there exists a large amount of content to publish and revise, it becomes both technically and organisationally infeasible to maintain web domains via static pages.
Within the system implemented the process of creating and adding value to the e-organization can be transferred from the programmers to domain experts within the organization. Because the content is served from the database the staff are now freed from the constraints of design context and can more effectively develop pages to serve specific content type and user groups.

Dynamic Content Publishing

“A page is said to be dynamic if some or all of its content is generated at run-time” – Raghavan et a: Crawling the Hidden Web[2]

The implemented system varies the contents of a page based on the context of the viewing. This covers a wide range of possibilities, this form of dynamic content is called context dynamism. Predominately sources of context is explicit information by the user, for example a request to view the course description of a specific course.

Issues considered when developing a dynamic content generation system include; efficiency, processing location, and language.

The most important issue to consider is efficiency. Due to the nature of dynamic content publishing (DCP) serving dynamic pages is ultimately more taxing on the processor as pages must be by definition be put together at run time. However this in-efficiency can be some what alleviated by carefully deciding the processing location. The DCP system implemented utilises DHTML. This enables the shifting of some processing requirements over to the client rather than the relying solely on the server for these tasks. The system utilises the server to fetch requested information from the database. Content is then passed to the client from where it is rendered for viewing through the use of the DHTML tools; CSS and JavaScript.

Dynamic content publishing is important as:

- **It provides an easy method for adding/editing content.** This is because these tasks can now be done independently of the formatting of content. Responsibility for formatting now resides with the DCP system.
- **Searchable and scheduleable content.** Since all the content is in a database, searching and results can now be dynamically generated. The DCP system can also now utilise content scheduling so that certain content shows only at certain times.
- **Content reusability.** Content and formatting code now proves reusable. The DCP system implemented stores the site template in the database, thus for instant site-wide changes modifications can be made to the template via a GUI linking the user to the database record.

Advantages of dynamic content include:

- **System resources.** As pages are generated as requests arrive one page can now serve many requests.
- **Interaction with client input.** Information can be provided specifically to the users request.

Disadvantages of dynamic content include:

- **Performance penalty.** The system now has to put the page together when a request is received before it can provide the requested content.
- **Need for centralised content.** As the DCP system retrieves content from a centralised database, if the database fails then usability is greatly reduced.

Software Design

The application provides the graphical user interface (GUI) for the user. The GUI is provided via the use of; CSS, JavaScript and HTML.

These two categories, determined by access privileges, allows data access and/or manipulation.

Issues method calls utilising SQL to manipulate or retrieve content. Appends formatting data based on the content type.

Centralised location where content is stored. Content stored includes faculty data as well as site template information.
Tools Utilised

Server Side Scripting
Web server scripting enables the necessary ‘backend’ to provide logic for web applications. HTML and HTTP do not themselves provide a way to access databases or carry information about users from page to page. The advantage of using server side scripting when developing a web application such as a CMS can be easily seen from the following example: server side scripting allows the use of templates for creating HTML documents on the fly. The content of a page can come from anywhere: database, plain text file, calculation, and be dynamically inserted into the template before it is sent to the user. Thus the code for the template does not need to reside in multiple locations but as a singular entity called upon when required. Thus server side scripting proves to be an important tool for separating content into presentation and data, making design and data management simpler and more uniform.

PHP Hypertext Pre-processor
PHP is an open source scripting language developed specifically for web use. PHP implements the following core features of a scripting engine:
- **Speed**: is advantageous over CGI as it does not carry the external program processing time inherent with CGI.
- **Stability**: uses its own resource management system, making it intrinsically a robust system.
- **Security**: customisable security levels.
- **Simplicity**: utilises many of the program and functional implementations of C and PERL.
- **Connectability**: Comes with native support for a variety of technologies including MySQL, Oracle, XML and PDF. Modular system of extensions, so interfacing with libraries is simple.
- **Compatibility**: Compatible with a variety of other key web technologies, including Java, COM, XML and Macromedia Flash.

Commercial advantages
- **Price**: Free
- **Multi-platform compatibility**, including: Windows(95,98,NT,2000), Macs, most UNIX’s.
- **HTTP server interfaces. PHP currently will load into Apache, IIS, AOLServer, Roxen and THTTPD. Alternatively it can be run as a CGI module.
- **Zend Middleware Products**.

MySQL
MySQL is a database that is powerful and flexible while at the same time lightweight and efficient. While the MySQL database engine is no where as powerful as commercial databases, it does provide a large enough set of features for it to be a powerful asset in developing data driven applications requiring a mid-range database management tool.

Database Design

The database used for the CMS is called Ugrad. Below is a diagram outlining the structure of the database:

```
       UGRAD
              |
            UOS_Internal
              |
            UOS_External

       |
      |
    Coordinators

    |
    |
  Year_Advisors

    |
    |
Announcements

    |
    |
Template
```

The Content Management System

The Undergraduate web domain can be broken up into six separate sections of content; Degree Information, Enrolments, Units of Study, General Information, Scholarships and Announcements. At the crux of all these content categories there lies a need to develop tools to enable a differentiation between content management and content publishing.

Degree Information
This section of content highlighted the effectiveness of content reuse. Scripts were written to facilitate the creation of tables outlining core and recommended units of study as outlined in the resolutions for single degrees. Retrieval of this information was simple, as it made up fields of records already stored in the unit of study database table.

This section of content also necessitated the development of a GUI for entering external units of study information. This system mirrored that used for internal units of study. It became necessary as records of these units did not previously exist.

Enrolments
This section of content once again utilised content re-use. Once again content was generated from content that was already available. Functionality included listing core subjects and a calculation of electives available per semester.

This functionality highlighted the system’s effectiveness. A staff member, for example, could now make one change through the unit of study GUI to remove a subjects core requirements and this change would instantly be reflected throughout the site.
Units of Study

This section of content highlights the true effectiveness of dynamic content. This section includes content in relation to courses offered by the department and also outlines for these units. In order to facilitate these functions the following scripts needed to be developed:

- GUI for addition/deletion/modification of courses and/or course material.
- Dynamic publishing scripts to list all available subjects and course descriptions.

The development of the user interface for staff members was done so that users would need no prior knowledge of HTML. This was achieved by providing text fields into which raw text could be entered. The system validated all material entered so as to ensure data consistency. Users were given an option to preview the entered material before it was recorded. When presenting the user with this material the script applied the set presentation rules, so the user could gauge the correctness of the page under a publishing environment.

The script for course list and description was done so by querying the database. The script for course description replaced the need for static pages for every course, thus a one-to-many functionality.

General Information

Content stored in this section included staff content details, year advisors, course coordinators and student representatives.

An important development point would be that of the storage of staff details. By centrally storing this content, all pages referencing this content would be dynamically updated when a change to staff details was made. The previous system would have required that this change be made on several pages whereas staff now are able to make one change with other pages intelligently adapting to this modification.

Announcements

This section of content proved to be the most diverse. Content in this section, by its nature, could be added/edited or removed at anytime. Thus by providing a portal to the content, staff members could easily manipulate the content stored. All data was entered through the use of text fields with presentation data appended upon a request by the user.

Other Tools

Other tools implemented included an editor for the site template. By storing the template in the database and providing a GUI for editing, a user could make a central change resulting in a site-wide change.

Future Development

Future development of this system lies primarily in the need to distribute content to a variety of user devices not solely via HTML. Such a system could utilise XML. By storing XML in the database instead of plain content, content could be rendered in a method specific to the device it is intended for. This could be done utilising XSL.

XSL provides a means to render XML documents in a specific format. This includes formats for the web, mobile phones and PDA’s. Such a system not only provides a complete content management system but also an extraordinary ability to adapt managed content to a variety of user needs.

Conclusion

Web site production and delivery has evolved in the last few years from producing flat HTML files into a more complex database-driven and structured approach to production, design and automation. As content-driven web sites grow in complexity, so does the need to provide data reusability as well as effective collaboration between the organisations staff. In such an environment the need for a dynamic content management system becomes an organisations edge over its competitors cutting the time required to revise and publish its content.

Although the system developed specifically targets the need of the Department of Electrical and Information Engineering the concept of a content management system is not lost on many businesses. When the existing edit-review-publish lifecycle proves infeasible such a system is certainly a viable option.

The technology utilised in development also highlights that such a system is quite practical for many small-medium enterprises. Costs involved in developing such a system are relative small in comparison to technologies such as Perl CGI. PHP is multi-platform and thus scripts developed will be as effective if there comes a need to migrate operating systems. Such an advantage is not available with other server side scripting technologies including; ASP and ColdFusion. Such a system, however, is not completely focused on SME’s. PHP provides many extensions and is compatible with a large range of industrial databases including; MS SQL, Oracle, Informix, PostgreSQL and many others.