Abstract

Web Pages can be difficult to keep up to date. Often times the information contained in them becomes outdated while a company completely rethinks the design of their website. Hiring outside companies to update the information can be costly and take time. In order to make the update process easier we have developed a web page that can be dynamically updated without the pain of having to alter any code. This allows the web page to be edited by even the least computer inclined person.
1. Introduction

Designing and updating web pages can be a painful process for someone who has worked with HTML. The language is not intuitive and can take some time to learn. In a job where the staff is constantly changing continuity and having people to keep a page up to date can be a nightmare. This is the case with the Kansas State University Army ROTC web page. The Cadre change out every 2 to 3 years and with each one comes a new set of skills, and often web page design is not one of them. In order to ease the process of constant changes within the ROTC program we have created a page that can be dynamically updated, by using page templates and just filling in the important information.

1.1 The problem with the current page.

The current web page was designed in HTML and was very confusing. The page used java scripts and was created using Microsoft FrontPage, this caused the person responsible for keeping it up to date to have to sift through code that did not have proper line breaks and a lot of erroneous code. The state of the current page made it almost a must to completely re-write the entire site. The ROTC department usually hired an outside company to rebuild the page any time that the information needed to be updated. This would cause periods of time in which the page was out of date and cost the department hundreds of dollars per update.

1.2 Importance of our solution

The solution that we created allowed the user to use a simple template system to edit the content of the web site. This allows even the most computer illiterate people to easily modify the site. With the constant change in cadre there needs to be a simple way to update the page that can be easily taught to incoming cadre.

This ability saves the ROTC department money by allowing any of the cadre to update it. In the past, they have always hired an outside company to develop the site. Over the past several years they have spent several thousand dollars to keep the page updated. Our solution will eliminate the need to spend any more money on the upkeep of the web site.

2. Related Work

There are several different options that attempt to make creating content for web pages easier. These vary from Applications that allow you to point and click to create web pages to server side applications that hold a template for your site and you just input the personalized information for each page.
2.1 Applications

Applications for creating web pages come from many different sources. Microsoft FrontPage and Macromedia Dreamweaver are two examples of application packages that are designed to create and manage web sites. Though you can see exactly what you are creating every time, it takes a good amount of knowledge to be able to create, update, and upload web pages. Though some cadre members would be able to figure out these programs, they have the major drawback that they must be installed on every computer that is going to be used to create and manage content. These applications would cost the department money, in order to purchase licenses, and upgrade them as time goes on. This solution, although good for an individual’s informative website, would not be optimal for the business style site, that we will be creating.

2.2 Content Managers

According to wikipedia “A content management system (CMS) is a computer software system for organizing and facilitating collaborative creation of documents and other content.” Several of these, such as tiki-wiki, and drupal, are open source server side applications that require no special software to be installed on a client computer. We explored using drupal CMS for our site.

Server site CMS’s have many advantages such as being able to create a template for the site. We found that it was easy to install on top of the Gentoo Linux system that we had built. After doing some tinkering with it we found that even though it would be easy to integrate the additional tools that we would need for the site, it would require too much cadre knowledge of HTML or PHP code in order to be successfully maintained. Because we could not find a suitable solution we decided that the best way to go was going to be developing our own, small server side content management system with limited functionality.

3. Implementation

In creating the web site we had several requirements that had to be considered. These requirements left us with several decisions. The first of which was decided what platform to develop the site on. We also had to decide what language to write it in and how much control over the look and feel of the site that we would actually allow the cadre to have. One of our main concerns in making our decisions is to keep the site as cost effective as possible.

3.1 Setbacks
In creating the ROTC web page we encountered several setbacks that caused us to re-evaluate our plan. These only slowed the development of the web page rather than stopping it completely. Many of these setbacks could have cost the ROTC Program money, but we were able to find ways around the monetary costs.

The first setback that we came across was the fact that the server that the web page was hosted on did not allow for any scripting languages. The way we first conceived the project had us developing just a dynamically generated web page. Because of the limitations put upon us by the web server we quickly had to find a new way to host the site. It took a while to get CNS to approve of the ROTC program having its own server, with a K-State DNS pointing at it. After several weeks of e-mailed back and forth between CNS and us it was finally approved.

The next thing we had to come up with was a computer to host the site on. We looked into using one of the old computers that the department owned, but found that they were too slow and too old to be able to run an efficient server. This setback caused little or no development to be done throughout the summer. Once the semester started back up, we were able to acquire an old computer that CNS had available, and because we did not need a copy of Windows on it, we were able to get it for free. Because of this donation we kept up with our plan to have a cost free solution to the new Web Page.

3.2 Platforms

The first major decision that we made was what platform to develop the web site on. Initially the web site was hosted on the KSU UNIX server, which did not allow for any languages such as PHP and ASP to be used. Use of this web server severely reduced our ability to create a web page that could be dynamically updated. We needed to find a way to use these dynamic languages.

We had already had a computer donated to the department that we were going to use. After speaking with CNS and being allowed to create armyrotc.ksu.edu, I needed to decide what operating system I was going to use to create the web server. Our first option would have been to use Windows, but this would not solve our requirement to save money, because we would have to purchase a license for the computer. The next option was the one that CNS recommended, which was Red hat Linux through the K-state site license for a $50 per year fee. After talking with the cadre we decided that we would explore the free options first.

We had several meeting with the cadre and spoke with CNS several more times on the advantages and drawbacks of several of the different Linux operating systems that were available. Finally we settled on using Gentoo Linux with an apache web server. This would allow use all of the functionality that we required, plus we had the added bonus of complete configurability. The only drawback of using Gentoo was the fact that we had to manually keep any security holes up to date. This was worth the cost because we had so
much power over everything else on the site. Now that we had the Computer with the operating system we needed to decide what language to use.

3.3 Languages

Different languages are available for the Gentoo server that would allow the functionality that we needed. The current page was written in standard html, which did not allow for any dynamically generated data. The language that we needed to use must support connections to databases so that we could dynamically update and generate the information that we needed.

One of the options that we had was to use ASP on the Linux server. According to wikipedia “ASP.NET is a set of web development technologies marketed by Microsoft. Programmers can use it to build dynamic web sites, web applications and XML web services. It is part of Microsoft's .NET platform and is the successor to Microsoft's Active Server Pages (ASP) technology.”(Wikipedia) Though it is possible to have asp support on the Linux box it was not a very good solution. Much of the existing code for asp connects to a Microsoft Access database, which is not compatible with Linux. The software that is required in order to implement ASP on Linux comes from a company called Chillsoft, but it is not open source and is not free. (Chan)

The option that we decided to go with was PHP, which is an open source language. PHP is “a widely-used general-purpose scripting language that is especially suited for Web development and can be embedded into HTML”(php.net). It is a language that we have used before, and decided that it would meet all of the requirements that were placed on us. We are able to write pages that can be dynamically updated, and stored in a MYSQL database, which is another free open source piece of software. The fact that it can be embedded into standard HTML makes it even better because we were able to pull some of the data from the existing ROTC web page.

3.4 User Control

Our last major decision was how much control to give the cadre. They need to be able to update the information easily as it changes. The major question that we had to research was whether they should be able to make changes to the look and feel of individual pages, or even the look and feel of the entire site.

We thought about the idea of allowing Wiki tags in order to allow simple integration of graphics and html objects. After discussing this with the cadre, it was found that this was not necessarily a good idea, because they wanted the page to have a uniform look. Because of this, in order to achieve the uniform look we created several functions that would allow the parsing of the data that was input, and convert it to the format that we needed. One place where we implemented this was in the cadre bio information, where
the user just inputs standard text with line breaks, and our function converts it into a bulleted unordered list.

In order to further prevent look and feel changes, we set up every page so that the template for inputting the data resembled the user page where the data would actually be seen. This would eliminate the want for cadre members to insert their own HTML code, in order to make the page look the way they want.

4. Evaluation

In evaluation our web page we tested the syntax of our code, and used our research into the efficiency and speed of the components and applications which we chose to use.

4.1 Testing

The web site was tested by running the HTML markup validation service, found at http://validator.w3.org/. It is designed to evaluate the syntax of the pages on the site. In order to make sure that it worked for every page we had to input each page’s address individually. After testing and repairing the pages several times, we were able to clean up all of the errors, which were found.

4.2 Performance Analysis

The code that we have written is all $O_n$ in complexity therefore is fairly optimized. Each one of the pictures has been reduced in size to provide minimal loading time for the site. The man limitation of the web site is the fact that it is running on a Pentium III with only 256 MB of ram. If the server were to receive a large amount of traffic it would considerably increase the load times for the site. There are applications available to test the performance of each PHP file, but in order to keep the project completely free we decided not to use any of them. We decided that the way in which we developed the web site did not need any special performance evaluation, because the code, which we had developed, was fairly linier.

5. Conclusions

Updating web pages can require a vast knowledge of HTML. With businesses that change employees often, and whose employees are not trained in web page design, this can be a major issue. Often times these companies resort to hiring outside sources in order keep their web sites current. Over time this can cost a company thousands of dollars, and cause out of date information to remain on the site while they are waiting for the updates to finish.
5.1 Our Solution

We have now created a web page that allows minimal knowledge of web design and coding procedures in order to update the information. This will save the K-State army ROTC department several thousand dollars over the next 10 years. All of the data can be dynamically updated using simple template pages that closely resemble how the user will see the data.

Through careful analysis we were able to decide on a solution that ended up costing the ROTC department nothing monetarily. It is very easy to understand and is compliant with all of the W3C standards.
Works Cited

Chan, Mike “ASP on Linux” http://linux.omnipotent.net/article.php?article_id=6864

“What is PHP?” www.php.net