Ezio Hafner
Dario Ogrizović
Faculty of Maritime Studies
Studentska 2, Rijeka
Croatia

Technical paper
UDK 004.42
004.451
004.738.5
Received: 5th July 2004
Accepted: 7th July 2004

WEB PORTAL DEVELOPMENT IN LAMP ENVIRONMENT

Dynamic and individualized Web portal development projects have become the centerpiece of IT development strategy for many institutions in order to create a strategic advantage, to provide services for increasingly demanding users and expand market opportunities.
Open Source LAMP development environment is used to create complex, dynamic and secure Web applications.
Open Source technologies will improve reliability, reduce costs and provide new revenue streams.

Key words: Web portals, LAMP, portal development, Web applications, Open Source

1. INTRODUCTION

When collection, management, and publishing process becomes too complex and when institutions want to provide single sign-on access to all web applications and services, continuous availability, remote access to the institution's applications, automated workflow capability and a personalized desktop the timing may be right for institutions to think more seriously about developing a Web portal rather than restructuring the static Web site.

A portal is a personalized and customized gateway designed for useful and comprehensive access to information, people, and processes. While portals have a rapidly evolving set of features and characteristics, they can be described as personalized and customized user interfaces providing access to both internal and external information. With some computer programming institutions can convert a static Web site into a Web portal, linking servers to particular databases and information systems. Depending on design sophistication, a Web portal could provide various

categorization and personalization levels to various groups and individual visitors.

The emergence of Free [1] and Open Source Software [2] and the WWW as a platform for distributed applications development has led to the availability of the LAMP web development environment which is robust enough to allow the development of complex on-line services and applications.

2. LAMP

LAMP is an Open Source Web development environment based on Linux as the operating system (or other Open Source operating system, such as FreeBSD, NetBSD, OpenBSD); Apache as the Web server (often along with Open Source Java app. servers such as Tomcat or JBoss); MySQL as a relational database management system (RDBMS) with add-on tools for Web-based administration (or other Open Source alternatives, such as PostgreSQL); and PHP is a popular object-oriented scripting language that encompasses the best features of many other programming languages to make it efficient for Web development, but alternatively Perl, Python and Ruby as well.

In addition to the LAMP Web development environment CVS is used to keep track of programming projects.

LINUX

Linux [3] is a freely distributable version of Unix, originally developed by Linus Torvalds, who began work on Linux in 1991 as a student at the University of Helsinki in Finland.

Linus released the initial version of Linux for free on the Internet, inadvertently spawning one of the largest software-development phenomena of all time.

Inspired by Andrew Tanenbaum's Minix operating system, Linux began as a class project in which Linus wanted to build a simple Unix system that could run on a 386-based PC.

On October 5, 1991, Linus announced the first "official" version of Linux, version 0.02. At this point, Linus was able to run bash (the GNU Bourne Again Shell) and gcc (the GNU C compiler), but not much else was working. The primary focus was kernel development, none of the issues of user support, documentation, distribution, and so on had even been addressed. Today, the situation is quite different, the real excitement in the Linux world deals with graphical user environments, easy-to-install distribution packages, and high-level applications such as graphics utilities and productivity suites.

After Version 0.03, Linus bumped the version number up to 0.10, as more people started to work on the system. After several further revisions, Linus increased the version number to 0.95, to reflect his expectation that the system was ready for an "official" release very soon (software is not assigned the version number 1.0 until it's theoretically complete or bug-free.) Version 1.0 appeared in March 1994.

Linux could not have come into being without the GNU tools created by the Free Software Foundation. Their gcc compiler gave life to Linus Torvalds's code. GNU tools have been intertwined with the development of Linux from the beginning. Because of the critical contributions of these tools, the Free Software Foundation even requests that distributions of Linux with accompanying utilities be called GNU/Linux.

Berkeley Unix (BSD) has also played an important role in Linux, not so much in its creation, but in providing the tools that make it popular. Most of the utilities that come with Linux distributions are ported from BSD. Networking daemons and utilities are particularly important.

Today, Linux is authored and maintained by a group of several thousand developers loosely collaborating across the Internet. Companies have sprung up to provide Linux support, to package it into easy-to-install distributions, and to sell workstations pre-installed with the Linux software. Almost all major free software packages have been ported to Linux, and commercial software is becoming available.

With the combined efforts of companies as well as individuals, Linux has evolved into a modern operating system that incorporates protected memory, multitasking, fast TCP/IP networking, shared libraries and multi-user capabilities.

APACHE

Apache [4] is an HTTP Web server developed by a group of volunteers. The base of the Apache Web server was a public domain HTTP server developed by Rob McCool at the National Center for Supercomputing Applications (NCSA), University of Illinois, Urbana-Champaign. Soon after its development, many Web masters were creating their own extensions of this server. At that point, only a common distribution was required. In February 1994, Brian Behlendorf and Cliff Skolnick started inviting ideas about improvising Apache from people all over the world through a mailing list. Several volunteers collaborated and wrote the source code for Apache. The name of the Apache Web server originated from the phrase "A patchy" (the developers who wrote the source code for Apache made improvisations to the code in the form of patches). Later, the Apache Group was formed. This group consisted of eight core members of the original development team: Brian Behlendorf, Roy T. Fielding, Rob Hartill, David Robinson, Cliff Skolnick, Randy Terbush, Robert S. Thau, and Andrew Wilson. The outcome of this disciplined, collaborative effort was the first release of

the Apache Web server, 0.6.2, in April 1995.

Though this server was successful, the quest for perfection was enough incentive to drive the developers to redesign the Web server. After a lot of overhauling and the addition of many new features, Apache 1.0 was released in December 1995. Because it went through a series of beta tests before being officially released, it was a more stable than its predecessor.

In 1999, the members of the Apache Group formed the Apache Software Foundation. This foundation was created to provide legal and financial support for the development of the Apache Web server.

The Apache Web server attained its position because of the hard work put in by developers all over the world and freely available source code allowed developers to examine the source code and suggest valuable ideas to improve its design. The movement is still going on and is called the Apache HTTP Server Project. The Apache Server Project is an improvement initiative that allows people, irrespective of their location, to contribute code, ideas, and documentation.

Today, Apache is the most popular Web server, with a market share of about 65% according to the Netcraft Web server survey (see http://www.netcraft.com/Survey/). Apache is available for download, but also ships with most Linux distributions.

MySQL

MySQL [5] is a SQL (Structured Query Language) client/server relational database management system. MySQL includes a SQL server, client programs for accessing the server, administrative tools, and a programming interface.

MySQL's roots began in 1979, with the UNIREG database tool created by Michael Widenius for the Swedish company TcX. In 1994, TcX began looking around for a SQL server for use in developing Web applications. They tested some commercial servers, but found all too slow for TcX's large tables. They also took a look at mSQL, but it lacked certain features TcX required. Consequently, Michael Widenius began developing a new server. The programming interface was designed explicitly to be similar to the one used by mSQL because several free tools were available for mSQL, by using a similar interface, those same tools could be used for MySQL with a minimum of porting effort.

In 1995, David Axmark of Detron HB began to push for TcX to release MySQL on the Internet. Axmark also worked on the documentation and on getting MySQL to build with the GNU configure utility. MySQL 3.11.1 was unleashed on the world in 1996 in the form of binary distributions for Linux and Solaris. Today, MySQL works on many more platforms and is available in both binary and source form. The company MySQL AB has been formed to provide distributions of MySQL and to offer support and training services.

PHP

PHP [6] was conceived sometime in the fall of 1994 by Rasmus Lerdorf. Early non-released versions were used on his home page to keep track of who was looking at his online resume.

The first version used by others was available sometime in early 1995 and was known as the Personal Home Page Tools. The parser was rewritten in mid-1995 and named PHP/FI Version 2. The FI came from another package Rasmus had written which interpreted html form data. He combined the Personal Home Page tools scripts with the Form Interpreter and added mSQL support and PHP/FI was born. PHP/FI allowed developers to embed structured code inside HTML tags. PHP scripts could parse data submitted by HTML forms, communicate with databases, and make complex calculations on the fly. PHP/FI grew at an amazing pace and people started contributing code to it.

In 1997, a pair of Israeli students named Andi Gutmans and Zeev Suraski attempted to use it for building an online shopping cart, considered cutting-edge enough to be a university project. Shortly after they started, they stumbled upon various bugs in PHP that made them look under the hood at the source code. To their surprise, they noticed that PHP's implementation broke most of the principles of language design, which made it prone to unexpected behavior and bugs. Always looking for good excuses not to study for exams, they started creating a new implementation. In part, the task was a test of their programming abilities, in part a recreation. A few months later, they had rewritten PHP from scratch, making it a real, consistent, and robust language for the first time. Having spent so much time on the project, they asked the course teacher, Dr. Michael Rodeh, for academic credit in an attempt to avoid unnecessary exams. Being the manager of the IBM Research Lab in Haifa and well aware of the overwhelming number of different languages to choose from, he agreed—with the stipulation that they cooperate with the existing developers of PHP/FI instead of starting their own language.

When Andi and Zeev e-mailed Rasmus with the news about their rewrite, they wondered if he would accept this new work, as it essentially meant discarding his implementation. Rasmus did accept it, and a new body was formed—the PHP Core Team, known today as the PHP Group. Along with Andi, Rasmus, and Zeev, three other developers—Stig Bakken, Shane Caraveo, and Jim Winstead—were accepted to the Core Team. A community of developers started growing around PHP.

After seven months of development, alpha and beta testing, PHP version 3.0 was officially released on June 6, 1998, and started bending the curve of PHP's growth to unprecedented angles. PHP's functionality was growing on a daily basis, and PHP applications were popping up everywhere. Following the release, Open Source projects written in PHP flourished. Projects like Phorum tackled long-time Internet

tasks such as hosting online discussion. The PHPLib project provided a framework for handling user sessions that inspired new code in PHP.

A few months later, on January 4, 1999, Zeev and Andi announced a new framework that promised to increase dramatically the performance of PHP scripts. They dubbed the new framework the Zend Engine. Early tests showed script execution times dropping by a factor of 100. In addition, new features for compiling scripts into binary, debugging, optimization, and profiling were planned. This announcement officially ended the PHP 3.1 project, which was supposed to bring better Windows support to PHP 3 but failed to gain momentum, and officially started the planning of PHP 4.

Work on the Zend Engine and PHP 4 continued in parallel with bug fixes and enhancements to PHP 3. During 1999, eight incremental versions were released, and on December 29, 1999, PHP version 3.0.13 was announced. A PHP beta based on the Zend Engine became publicly available in July 19, 1999, and was followed by an intense development period of various components, some of which were brand new, such as built-in session handling, output buffering, and a Web server abstraction layer. The release of PHP 4 on May 22, 2000 attracted many object-oriented programmers but the limited implementation left them desiring more. PHP 5 addresses these needs with a strong, rebuilt object system. The number of people working on various levels of PHP has grown immensely, and new projects, most notably PEAR, gained momentum and started pushing PHP to new heights of popularity.

Today PHP is widely used in both personal and corporate worlds as an efficient Web application platform. In most cases, PHP is introduced in a corporation because of its speed, absence of license fees and fast development cycle.

CVS

The creation process of software becomes more complex with growing size. An increasing number of developers have to collaborate closely to achieve effective development in larger systems. Furthermore, the introduction of latent bugs in the early stages of development can pose a real problem in large programs, since tracing the problem to its origin can become a tedious task.

Versioning control systems are the solution to this challenge. All versions of the source code are archived and maintained in a central repository (the "master" copy), usually at a separate server. Developers are granted access to this repository only indirectly through formal check-in and checkout of code. The idea is that all developers work with their own local copy of source code, and only check their code in once it compiles and runs. Versioning control systems keep track of who is

currently working on which files, and therefore prevent developers from accidentally deleting or overwriting one another's code. In addition to that, the repository keeps a complete history of all versions of the source code, so any version of a file may be retrieved later on, or the differences of any two versions can be analyzed.

The dominant versioning control system today is an Open Source Unix/Linux command line tool called Concurrent Versions System, or short CVS. It is widely used in both the Commercial and the Open Source software development world. CVS is based on RCS (Revision Control System).

Most Linux distributions ship with CVS. The versioning control system tools may also be downloaded from Web sites (e.g. http://www.cvshome.com/) as source code, or as binary version for all common operating systems.

3. WEB PORTALS

Web portals go beyond static web pages and require a sign on which then links to some knowledge the institution has collected about the visitor. Web portals also remove the need for multiple logins to various applications, let users perform individualized or self-service processes that previously only dedicated staff could handle, and let organizations target users for individualized services and information.

Different types of portals can be classified as public, vertical and enterprise portals.

Public portal offer wide range of network services such as email, chat rooms and services of common interest like the weather or stock market. Users can tailor their homepage by selecting from a set of preset components and they can add their own links or change the appearance of the portal (e.g. Yahoo).

Vertical portals focus on a specific industry, and the components offered are industry specific. For example, an education portal will have components that provide educational information and services from many resources.

Enterprise portals provide components for a single organization such as a University. The components offer links to information or services that are mainly hosted at the organization. Figure 1 illustrates an overview of an enterprise portal.

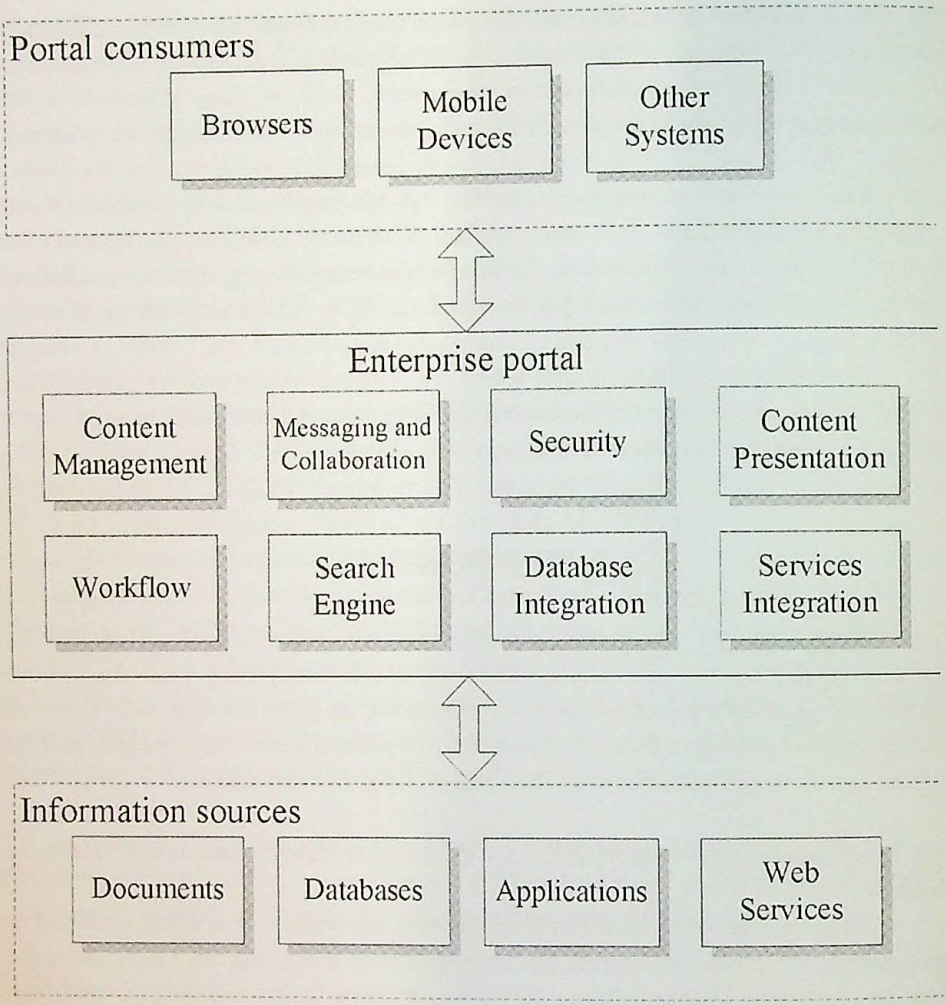

Figure 1. Enterprise portal

A portal is not a single technology, but it brings together a wide range of technologies and enables them to work together. To present users with information and services that are appropriate at any particular time, portals need to be modular and dynamic.

Every portal provides a list of various features such as aggregation of content, personalized content, unified security model, collaboration features, localization, web services access, etc.

In addition to having a set of conceptual features, a portal framework also contains a set of commonly found components such as content management, search

Engine, web-based mail, etc. The portal market has provided a wide spread of features and components in each product.

Despite the overall depression in IT spending, Gartner [7] estimates that portal software spending grew 59 percent to $709 million dollars in 2001. Delphi Group [8] estimates that the Portal Market in 2003 reached $957 million worldwide, and will reach $1.14 billion in 2004. IDC [9] estimates the market for enterprise portal platforms will reach more than 2.6 billion dollars by 2006. Nearly all Web development efforts are being geared toward integration Web portals.

4. PLANNING AND DEVELOPMENT

Enterprise portals present several development challenges. On the technological side, a single application must bring together and make available vast amounts of organizational data. The administration must redefine business practices to let individual departments and business units update and maintain their information within the new environment. And, if the final product is to be a success, users must ultimately adopt the portal.

Colin White and Clive Finkelstein [10] have identified seven key tasks involved in planning and developing an enterprise portal project. These seven key tasks include:

1. Developing the portal business case and project plan
2. Conducting a business content and services requirements study
3. Selecting, installing, and integrating portal technologies and products
4. Designing the portal user interface and building or customizing any required portal content adapters
5. Developing and implementing the portal services required by portal users
6. Designing and implementing the portal security service
7. Implementing and deploying the portal

When planning and developing a portal, many concerns need to be addressed, but the two most important modeling determinations are the management of Web content data and the user communities that will be exposed to that data. Accommodations must be made to ensure that portal content is not rendered improperly to the disparate user base, which could compromise an individual's privacy. User profiles should be established so that content can be shown in proper user communities. To retain and increase their user bases portals need to maintain lucid navigation flows so that end-users know where they are and where they are going to go next. If a design does not provide a proper navigation model, users may become confused and lose interest.

5. CONCLUSION

Using Web portals every institution can streamline communication from on-line project management, status and reporting, to extending on-line content and document authoring for multiple branch offices while using centralized data management.

To create Web portal systems that meet both organizational and user requirements, institutions must first identify enterprise portal characteristics that contribute to users' satisfaction and potentially to their ultimate adoption of the system.

Institutions can realize an immediate return on their investment by streamlining and automating workflow, collecting information on-line, reporting on data from multiple independent systems and managing content and documents efficiently.

Using Open Source Software institutions choose to invest toward improving their enterprise software directly, with the entire investment going to directly improve mission support, because there are many advantages of using Open Source Software such as reduced license costs, reduced license compliance requirements, reduced administration, ability to lend/give software without fear of license infringement, benefit from improved product quality, simplicity, performance and resource-miserliness, possibility of local/own fixes and enhancements and many more.

REFERENCES

[1] http://www.fsf.org
[2] http://www.opensource.org
[3] http://www.linux.org/
[4] http://www.apache.org
[5] Paul DuBois, "MySQL", Sams, 2003
[6] PHP Documentation Group, "PHP Manual", CEST, 2000
[7] http://www.gartner.com/
[8] http://www.delphigroup.com/
[9] http://www.idc.com/
[10] Colin White, Clive Finkelstein, "The 7 Steps of Portal Development", DCI Corporate and E-Business Conference

Sažetak

RAZVOJ WEB PORTALA U LAMP OKRUŽJU

Dinamički i individualizirani projekti razvoja web portala su postali središnja IT razvojna strategija za mnoge ustanove kako bi stvorile strategijsku prednost, omogućile usluge za sve zahtjevnije korisnike i proširile tržišne prilike.

Open Source LAMP razvojno okružje primjenjuje se za stvaranje složenih, dinamičkih i sigurnih web aplikacija.

Open Source tehnologije poboljšati će pouzdanost, smanjiti troškove i omogućiti nove prihode.

Ključne riječi: web portali, LAMP, razvoj portala, web aplikacije, Open Source

Pomorski fakultet u Rijeci
Rijeka, Studentska 2
Hrvatska