TOWARD THE CREATION OF A *GREEN* CONTENT MANAGEMENT SYSTEM

Matteo Ciman, Ombretta Gaggi and Marco Sbrignadello
Department of Pure and Applied Mathematics, University of Padua, via Trieste, 63, 35121 Padua, Italy
mciman@studenti.math.unipd.it, gaggi@math.unipd.it, msbrigna@studenti.math.unipd.it

Keywords: Content management systems, Web engineering, Green web sites, Accessibility, Green computing.

Abstract: In this paper we discuss the problem of efficient use of Content Management Systems in the development of a web site, and we propose a new approach towards the definition of a *Green Content Management System* (GCMS). Our GCMS distinguishes between pages which contain dynamic data and pages which are not supposed to change frequently. This second kind of documents is generated, *off-line*, every time that a change occurs, thus avoiding waste of CPU time, and improving the response time of pages. For this reason, we call our CMS *"green"*. Moreover, our GCMS supports the creation of fully accessible web sites.

1 INTRODUCTION

A *Content Management System* (CMS) is a collection of procedures designed to support the publication of contents on the web by users without specific knowledge about web programming. Among others, it allows to control access to information based on the user role, to reduce data replication (both for input and output) and to improve the ease of report writing. The last feature is particularly important in the creation of a web site since all its pages have many parts in common, e. g., the header, the footer and the navigation bars. Only the content changes for each page. This means that a change in the content of one of the shared components without the help of a CMS, may require to modify all the pages of the web site, with waste of time and money. This is one of the main reasons for authors and designers to adopt a CMS.

On the other hand, the introduction of a CMS in the management of a web site can have some drawbacks from the user's point of view. Let us analyze the behaviour of a web server when a user that requests a web page through a browser.

In presence of a CMS, to return the content required by the user, the server cannot simply retrieve a file in its file system (see Figure 1), but the CMS engine must create the web page, which does not exist a priori. Therefore the server must receives an URL from the web browser and requests the corresponding page to the CMS, which requests the content of the page to a database, retrieves the content and build up the page. Then, the web server can return the HTML code to the browser (see Figure 2).

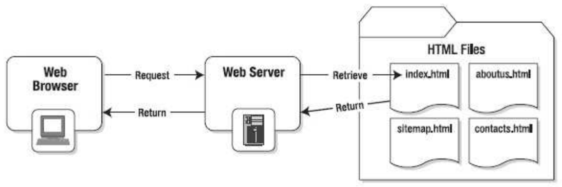

Figure 1: Web site without a Content Management System.

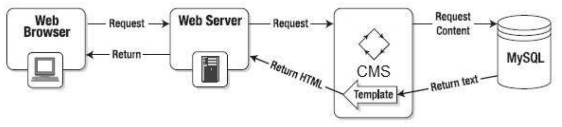

Figure 2: Web site with a Content Management System.

All these operations require time, therefore, the use of a CMS is not completely transparent to the user because it affects the response time of the web site, especially when the size of the site increases, in terms of number of requested pages and visitors. Moreover, the introduction of a CMS requires resources (power, network bandwidth, etc.) to build up each requested page, which is generated on the fly when a user asks for it. This approach is not cost-effective when the content of the page is essentially static, i. e., it does not change frequently in time. As an example, consider a web site for e-commerce, which can be divided in pages containing dynamic data like the presence of products in the stock for which a CMS can help, and static pages (e. g., information about seller) for which the use of a CMS wastes time and resources.

Another problem is that modern CMSs often do not generate accessible web pages. A web page is *accessible* if all users can use it, despite their operating systems, browsers, devices and user capabilities. *Web Accessibility* means that people with disabilities can perceive, understand, navigate, and interact with the page. Accessibility does not benefits only people with disabilities because search engines can be compared to users visually impaired, therefore accessible web pages usually obtain a better rank.

In this paper we discuss the problem of the efficient use of CMSs, and propose a new approach towards the creation of a *Green Content Management System* (GCMS) which preserves CPU resources and response time. Our GCMS allows to divide layout from content structure in order to allow the easily change of the shared components of a web site, but it distinguishes between pages which contain dynamic data and pages which are not supposed to change frequently. This second kind of documents is generated, off-line, every time that a change occurs and not every time a user ask for it, thus avoiding waste of CPU time, and improving the response time of pages with static data. For this reason, we call our CMS *"green"*.

The GCMS in this paper is intended for expert web designers and authors, therefore it does not provide a graphic interface to avoid writing XHTML code, since many solutions to address accessibility issues require the full control of the page code. Our aims is to help this kind of users to divide the shared parts of a web page from the real content of the page to avoid data replication, but preserving efficiency.

This approach has been tested in the web site of the first and second level degree in Computer Science of the University of Padua in Italy (University of Padua, 2010) with positive results.

2 RELATED WORK

Green computing, or Green IT, is a set of principles, procedures and policies to improve the efficiency of computing resources, in order to reduce the environmental impact of their utilization (Rajguru et al., 2010). Just as we need to plant a new tree for each cut tree, power consumption strategies must maximize the conservation of energy until renewable forms become more readily available and the same for other resources like network bandwidth, paper and so on.

Internet may help to reduce paper waste or people movement, thus reducing air pollution, but its power consumption for web sites is quickly increasing in such a way that it is not clear whether energy saving through ICT overbalance its energy consumption, or not (Coroama and Hilty, 2009). Therefore, it is an important issue to enhance the energy efficiency of Internet and in particular of the Web.

Many works address this problem. Dick et al define the *Green Web Engineering* as "... the art of developing, designing, maintaining, administrating, and using a website in such a manner, that direct and indirect energy consumption within the complete life cycle of a website is reduced". In (Dick et al., 2010) they defined a framework with 12 web engineering principles which suggest, the use of virtualization strategies, correct configuration of cache expiration, optimization of CSS and a correct use of format and quality of multimedia elements.

One of the principles of green computing is the so called *"Re-duce, Re-use, Re-cycle"* philosophy, where the word "reduce" aims at minimizing the set of needed resources. Abaza and Allemby (Abaza and Allenby, 2009) explore the use of virtualization to duplicate hardware with software. They combined the virtualization and green computing to develop a production model for virtual desktop environments.

Among CMSs let us analyze a set of open source solutions. Joomla! (joomla.org, 2010) is an open source CMS, based on Apache server. It provides a WYSIWYG interface to edit web content, an advanced caching strategy, and partial support of UTF-8 charset. Moreover many predefined layout are available to easily create a web site from scratch. Unfortunately, it neither produces a valid XHTML code, nor supports the production of accessible web pages.

A step towards the realization of a CMS producing XHTML-complaint web site is represented by Drupal (drupal.org, 2010). Drupal allows to realize valid web pages. The produced code tends to remain clean and light but it requires to install many plugins. Drupal is intended for expert developers, and not for all the possible users. Unfortunately, the tests done show that its support for accessibility is still limited.

TYPO3 (Typo3 Association, 2010) is an open source CMS that declares to provide a full support for the development of accessible web site. TYPO3 is a user-friendly, intuitive tool, allowing content editors to produce and maintain web pages, using sophisticated functions. TYPO3 allows a complete separation of design and content and does not limit the design options expected by professional website designers. Despite this CMS represents the first real tentative to automatically create accessible web pages, the tests done reported a set of bugs. As an example, TYPO3 does not allow to mark words in a language that is different from the one defined in the page header. More-

Table 1: Comparison of most important Content Management Systems: Joomla!, Drupal and TYPO3. FA = Free Add On; L = Limited support.

	WYSIWYG Editor	UTF-8	Content Reuse	Valid XHTML	WCAG Compliant
Joomla!	Yes	L	Yes	No	No
Drupal	FA	Yes	L	Yes	L
TYPO3	Yes	Yes	Yes	Yes	L FA

over, there are some errors also in the management of list, so that the requirement of generation of valid XHTML code sometimes is rejected.

Table 1 summarizes some features of the examined CMSs. Summing up, the tests made report that it does not exists a CMS which is able to produce, contemporarily, an accessible and XHTML-valid web site. The solutions that try to address these issues inevitably end up creating software that is not so user-friendly. Moreover, the produced code, even if the CMS uses some optimization techniques, is far away from the quality that can be obtained by a web developer, both in terms of efficiency (number of bits) and in terms of clearness and maintainability of the code.

3 A *Greeen* CMS

As discussed in the introduction, the use of a CMS simplifies the management of a web site but it can also affect its response time. Our solution has 3 goals: to improve the management of a web site supporting easily modification of common parts, to be *completely* transparent to the users, i.e., it should not increase time needed to receive a page and GCMS must support *accessibility*. The last two points are not currently fulfilled by any other CMS.

The solution adopted was to create an evolution of CMSs, which allows to minimize the work of the server and transmission of data, thus reducing time needed to process any request. Our GCMS satisfies this goal lowering the number of dynamic requests.

The idea is to classify content in two categories, on the basis of who can modify it and its update frequency. Data are originally stored in one, or more, databases (relational databases or XML files) or in XHTML pages. If a web pages is modified only by expert authors at predefined date, we ask to the authors to force the *off-line* generation of that page after any change. We apply the same strategy to data stored in databases which are seldom modified and with a predefined schedule. In this way, the web server stores and retrieves this content from static XHTML files, thus improving its performances. If the content of a web page is modified by non expert authors, e.g. the administrative staff of a company, or it is not possible to determine when and with what frequency it changes, data are collected into a database, and the page is generated *on-line* each time it is visited.

Our system was implemented in Perl and it is based on an Apache Web Server and MySql Database Server. The architecture of the system was studied to combine positive aspects of a pure web server with the improvements brought by CMSs: from a visitor perspective, the system behaves exactly as a web server which returns static web pages; from a web developer point of view instead, GCMS manages the presence of shared parts in a web page like a traditional CMS.

Our site generation system organizes data in folders where each folder represents a section of the site. Global information common to all HTML pages, such as top-level menus or the files containing header and footer, are collected into a folder, called `globalDetails`. This information must be available for all languages used in the web site. New pages or entire sections can be added at any time.

Each folder related to a specific section of the site contains a subfolder `source` with the source code of the web pages and two XML files: the list of pages that must be generated for that section and some particular information about them (e.g., metatags, title of the pages, etc.), and the second level menu within that section. This second XML file is optional, since, very simple sections can be composed of a single page. Each XML file, when applicable, must be present for each language for which web pages are available.

Each XHTML file inside folders `source` is a template which represents a page that will be created. This file contains only the content of the page and a set of XML tags that are placeholders for that parts of the web page which are in common with all the pages of the web site and will be composed during the off-line generation, replacing these tags with the appropriate HTML code. As an example, the tag `<firstLevelMenu/>` is replaced with the content of the XML file containing the first level menu.

In case of dynamic web pages, whose content is stored in a database, the off-line generation produces a template containing all the common parts and a placeholder for the content. This choice helps to reduce response time even in this case, since the GCMS must retrieve *on-fly* only the content.

In order to support accessibility, each page is validated, contains links to skip the menus, to access the content and to jump at the top of the page from its bottom. Moreover, the authors are required to mark text written in a language different from the one of the page. This information helps the screen readers.

Our GCMS lacks of a WYSIWYG editor. This choice can be better explained if we consider the target. Our system is designed for expert web developers, i. e., people that deal with the realization of complex web sites, therefore people who know very well web standards. For this kind of users, the knowledge of some notion of XHTML, CSS and web accessibility is not an hard requirement. Many CMSs are available, but none of them allows the generation of clear, maintainable code, which is also WCAG complaint. Therefore, in order to obtain an accessible web page, web developers must often modify the source code produced by the CMS.

CMSs are usually not able to generalize. Let as consider an author who highlights some cells of a table with different colors. Using a WYSIWYG editor of a CMS, she/he chooses a color from a palette and assigns it to a (set of) cell(s). Then she/he chooses another color and repeats the same operation. The CMS usually adds the corresponding CSS code to each single cell, while the correct approach is to define a class with that particular color, and assign it to the cells. This solution allows to change the color to all the cells using it with a single operation, the CMS approach no. Moreover, the solution adopted by CMS does not separate content from layout and produces much more verbose and weighty code.

Many other examples are available reporting similar problems. For this reason, the current state of art of the editors does not allow an expert web author to create a page using only the WYSIWYG interface, but she/he often need to modify the source code of a web page, in order to improve its quality and accessibility. Some CMS like TYPO3 makes a step further, but still limited, in this direction, and we must also consider that, as soon as the set of features offered by a CMS increases, also the time needed to learn how it works increases in such a way to overbalance sometimes the efficiency of their use.

This problem does not affect the *occasional web authors* for whom traditional CMS are sufficient since they do not usually consider accessibility and efficiency of web pages as their goals. For this reason, our GCMS is not intended for non expert users.

We must note here that our approach does not preclude the use of a conventional HTML editor, it simply leaves this choice to the author, who can use a WYSIWYG interface or not according to her/his goals and preferences. In this sense, the absence of a visual editor can be positive, because it does not impose to the author to change her /his preferred editor: we simply force authors to split the page in different files, shared parts are stored only once for the entire site, and single pages contains only their content without header, footer and menus.

We call the proposed CMS *"green"* because its off-line generation system allows to *reduce* energy consumption and *reuse* resources since web pages are created once, and visited many times. GCMS *reduces* the response time of the web server minimizing the number of dynamic pages that must be composed before visualization thus *reducing* transmission of data between the CMS and the database (see Figure 2).

4 CASE STUDY

GCMS was tested during the development of the web site of the first and second level degree in Computer Science of the University of Padua in Italy (University of Padua, 2010) in order to verify what type of result can be reached with the adoption of our GCMS. The web site is installed on a virtual machine on a Fujitsu Siemens RX3000 Server with 4 CPU cores Intel(R) Xeon(R) CPU E5504 @ 2.00GHz with 4GB of RAM, 384MB of which were assigned to the virtual machine running the web site. The same hardware runs also other 3 virtual machines with web servers and other services and this choice helps to reduce power consumption and needed resources (Dick et al., 2010).

The web site was published on July 22th, 2010, and was accessed by 44,103 visitors in 6 months, for a total of 223,866 pages views. The web site response time was measured with Google Webmaster Tools Labs, which reports that, on average, the loading of pages takes 0.8 seconds, and that the web site was faster than 94% of sites accessed by Google[1]. This results can be reached thanks to the features of our GCMS.

The site underwent two different trials. We check its accessibility with the help of a group of expert users and the W3C validators and Total Validator(Total Validator, 2010). The results were positive. All the pages of the site are complaint to web standards and WCAG 2.0 AAA recommendation. Morevore, a totally blind user deeply inspected the web site and reported very positive comments: he did not run into any difficulties during navigation and he judges sufficient the navigation aids offered by all the web pages. Moreover, since accessibility also helps search engines, the web site results to be the first result returned into the Google responses page for a set of keywords containing the words "Univer-

[1]We must note here that this tool provided by Google is an experimental function of Google Labs.

sity", "Computer Science" and the name of the city hosting the courses.

We also ask to visitors to answer a questionnaire, judging their experience with the site. Fifty students filled the questionnaire. Even in this case, the site reports positive results. In particular, only for the 21% of visitors, the experience was not positive since they did not find the wanted information. For what relates the response time of the web site, 69% of visitors gave a positive score[2] 12% judged it sufficiently and 20% of visitors gave a negative response (i. e., visitors that did not find the information they look for).

5 CONCLUSIONS

In this paper we present a *Green* Content Management System which preserves CPU process time and the use of network bandwidth. It aims at merging some positive aspects of traditional CMS (the centralized management of the common parts of a web page like header, footer, menus, etc. in order to facilitate their change) with some concepts of the *"Re-duce, Re-use, Re-cycle"* philosophy. For this reason, differently from other CMS, it generates *off-line* all the pages that are not subject to frequent changes, even if they originally retrieve from a database, so that web pages, whenever possible, are created once and visited many times.

To better evaluate our work, we have tested the GCMS in situation of very few resources. We installed the GCMS over plug computer, in particular we chose a SheevaPlug, based on a powerful 1.2GHz Sheeva, ARM-compatible, processor(Marvell, 2010) with 512 MB of RAM. The SheevaPlug stores data on a 2GB Secure Digital card and declares a DC Consumption of 15W while traditional computers consume, on average, 250-300W.

The tests reveal that the time need for the off-line generation of the web site increased of about 40%, but this performance degradation affects only the authors and not the visitors of a web site, therefore it is sustainable if we consider the resources used.

Then we organized a second test to verify if the use of a GCMS is transparent to the visitors even in case of resources rationing. If this assumption is true, our approach is really *green*. We asked to a set of 7 users to navigate through web pages, some of which resided on the SheevaPlug and some not. Then, we explained to users the situation and we asked them to try to identify the sets. The result was very positive since all the users reports that there are no substantial differences in the web servers response times, therefore it was not possible to understand where the pages are placed on. Therefore we can argue that our GCMS is transparent to the visitors of a web page. Moreover, it helps to reduce the power consumption and, more in general, the resources needed by a content management system.

Our future work will be devoted to integrate our GCMS into a traditional CMS in order to solve the problem of the absence of a WYSIWYG editor for accessible web pages. Moreover, we want to better investigates the possibility to produce optimized and clear code and to automatically identify what sections of a page are in common with the whole web site.

[2]We consider positive only the values *excellent* (40% of the feedbacks) and *good* (29% of the feedbacks).

REFERENCES

Abaza, M. and Allenby, D. (2009). Virtual Green Computing - Assessment of the potential for virtual computing environments to supply a responsive and environmentally responsible alternative client computing environment. *ICGST International Journal on Computer Network and Internet Research, CNIR*, 9(2):1–9.

Coroama, V. and Hilty, L. M. (2009). Energy consumed vs. energy saved by ict a closer look. In V Wohlgemuth, B Page, K. V., editor, *Environmental Informatics and Industrial Environmental Protection: Concepts, Methods and Tools, 23rd International Conference on Informatics for Environmental Protection*, pages 353–361.

Dick, M., Nauman, S., and Hell, A. (2010). Green web engineering. a set of principles to support the development and operation of "green" websites and their utilization during a websites life cycle. In *WEBIST'10, Proceedings of the 6th International Conference on Web Information Systems and Technologies*, pages 48–55, Valencia, Spain.

drupal.org (2010). Drupal, http://drupal.org/.

joomla.org (2010). Joomla!, http://www.joomla.org/.

Marvell (2010). SheevaPlug, http://www.plugcomputer.org/.

Rajguru, P., Nayak, S., and More, D. (2010). Solution for Green Computing. *(IJCNS) International Journal of Computer and Network Security*, 2(6):51–54.

Total Validator (2010). http://www.totalvalidator.com/.

Typo3 Association (2010). Typo 3, http://typo3.org/.

University of Padua (2010). First and Second Level Degree in Computer Science Web Site.