Proceedings of the IASTED International Conference
INTERNET AND MULTIMEDIA SYSTEMS AND APPLICATIONS
November 19-23, 2000, Las Vegas, Nevada, USA

VERSIONWEB: A TOOL FOR HELPING WEB PAGES VERSION CONTROL

MARINALVA DIAS SOARES
RENATA P. MATTOS FORTES[1]
DILVAN DE ABREU MOREIRA

{mdsoares, dilvan}@icmc.sc.usp.br
renata@cc.gatech.edu

University of São Paulo
Institute of Mathematics and Computing

Phone: (+5516) 2739671
Fax: (+5516) 2739702
SCE-ICMC-USP CP: 668
Rua Dr Carlos Botelho, 1465
13560-970 São Carlos - S.P. - Brazil

ABSTRACT- In the continually changing world of computing, the Web is an example of an environment where information evolves very rapidly. In addition to Web information that changes very much and very frequently, Webmasters are faced with hard work when many people are involved in the parallel development of a set of related Web pages. In the face of such problems, a software tool, *VersionWeb*, was developed. The idea behind this tool is to make Web page version control available during browsing to users. The main goal of *VersionWeb* is to provide the Webmasters with an easy way of controlling Web page versions, through the Web itself.

Keywords: Collaborative systems, version control, software engineering over the Internet, configuration management.

1. INTRODUCTION

The World Wide Web has been presented as an environment where information evolves very rapidly. Because the Web information changes so much and so frequently, browser users usually become surprised when previously available information is no longer available. In the face of such problem, a software tool named *VersionWeb* is presented. The idea behind the tool is to make Web page version control available during browsing by users. The main goal of *VersionWeb* is to provide Webmasters with an easy way of controlling Web page versions, through the Web itself.

The Web information is a new approach to Information Technology. Thus, the development of Web applications can be seen as a new kind of software development. Recently, many software companies are starting gradually to adopt the disciplines of Software Configuration Management (SCM) as usual routines for software development [Kilpi 1997]. The change process is a reality for large software systems. Changes are required throughout all life cycle of a system [Sommerville 1995]. During the development of large-scale software, uncontrolled modifications come up and can lead quickly to chaos. Thus, change management procedures must be carried out to ensure that costs and benefits could be analyzed and changes could be controlled. Developers of Web pages (authors or webmasters) have faced hard work when many people are involved in the parallel development of sets of related pages. Because the authors work independently in their individual copies, the integration of the copies in a final hyperdocument (merging two or more versions to generate another) becomes a known problematic issue [Sommerville et al. 1998]. In addition, the volume of the involved documents usually is high, modifications in the documents happen fast and the modifications are difficult to track. In fact, to provide an efficient mechanism the change control should combine human procedures with automatic tools [Pressman 1997].

VersionWeb was developed to provide Webmasters with an easy way of controlling the Web page versions, through the Web itself. Additionally, during Web browsing the user is able to recover previously available information from the Web page, to visualize the differences between different versions of the page and to

[1] Visiting professor at Georgia Institute of Technology.
Her research has been funded by FAPESP – São Paulo State Foundation for Research Support, Brazil – Grant 99/09829-9

be notified when new version of it became available. Another objective of the tool is to facilitate the cooperation among authors, allowing them to have simultaneous access to the files, to manage the different versions of a file, and to locate changes that have been made.

Currently, the existing tools for file version control and change management have helped software developers in change controlling. They save time and the physical space required for the several copies created through the development process. Tools for version control are examples of them and mainly consist of systems for file version management. Some good examples are: SCCS (*Source Code Control System*) [Rockhind 1975], RCS (*Revision Control System*) [Tichy 1985], CVS (*Concurrent Version System*) [Cederqvist 1993; CVS 1999], ClearCase [Leblang 1994], COV [Lie et al. 1989], and HOISTS [Zeller and Snelting 1995]. In authorship environments, specially targeting the Web, we could find tools such as WebRC, V-Web and AIDE.

WebRC (*Configuration Management for a Cooperation Tool*) is the first cooperation-based tool designed on the Web with full support to configuration management. The tool helps the cooperation of groups on the Web by means of the concept of cooperation workspaces [Fröhlich and Nejdl 1997]. In WebRC, the users work on separated workspaces, and the workspace access by another user can be done through either locking the files or downloading the files to his own workspace. Instead of WebRC, *VersionWeb* does not lock the files, since CVS does the concurrency controlling. *VersionWeb* allows concurrent access to a file but each one of the users works on the file inside his own directory, i.e., they share the same file but not the same workspace. Additionally, *VersionWeb* provides the option of file downloading to help users who prefer to work on their own computer.

V-Web is a project that allows a set of instances (or versions) of a page to be visualized separately. Further more, it allows the user to add a version to the version set, to visualize the version's author, the date and time of the version creation, and other functions [Sommerville et al. 1998]. The system has been developed regarding as its main goal the collaborative support to Web authoring. However, V-Web does not provide the facility of showing the differences between two versions yet. *VersionWeb* has two advantages in relation to V-Web: showing the differences between two versions by means of highlighting them with colored fonts, and allowing files modification through the Web.

AT&T Internet Difference Engine (AIDE) is a system that looks for and shows Web page changes. The system consists of some components, including a Web-crawler, that detects file changes, a file containing Web page versions, a tool called HtmlDiff, that highlights the changes between two versions (or instances) of a page, and a graphical interface to visualize the relationship between the pages [Douglis et al.1998]. AIDE can find page changes, show the pages difference and find different page versions. However AIDE does not allow file manipulation through the Web as *VersionWeb* does.

These tools can help cooperative work in version control using the Web, making easy for authors to manipulate the several versions that usually are created during the development of a Web page or Web site. Even though the tools emphasize some ways of supporting the cooperative work for controlling Web page versions, *VersionWeb* is more concerned with providing Webmasters with an easy way for controlling version through the Web itself.

The paper is organized as follows. The Section 2 presents the architectural approach adopted for the *VersionWeb* tool. Section 3 describes the *VersionWeb* functions and presents its mechanisms of interaction, discussing a few aspects of the tool validation. Finally, in Section 4 we present our conclusions and perspective for continuing the development of the work presented here.

2. ARCHITECTURE OF THE *VERSIONWEB*

VersionWeb is part of the project "QualWeb" (under development) that aims at ensuring quality in a Web hyperdocument by providing mechanisms, such as version control of Web pages. It is a fact that, in spite of the increase in the use of Web resources and the facilities they make available, many of the hyperdocuments and hypermedia systems currently being produced can be intuitively qualified as "lacking quality", i.e., according to [Pressman 1997] lack of "conformance to explicitly stated functional and performance requirements, explicitly documented standards, and implicit characteristics that are expected of all professionally developed software".

VersionWeb helps the cooperative work done by Web page's authors and webmasters that can be situated in different and remote locations. *VersionWeb* also allows the user to find changes in a page, recover previous versions and to view differences between them. Its main feature is to facilitate the combination of a good version control system with a wide accessible interface (Web-browsers).

To provide its main functional feature, *VersionWeb* encapsulates CVS (Concurrent Version System), that is an efficient file version control system. CVS allows simultaneous access to a file. The access can be made through the network and it is possible to visualize the differences between two file versions (except binary files) as well as it can run under several operating systems [Cederqvist 1993].

Our work complements CVS by providing a Web-based interface for a friendlier interaction, since CVS has only a command-oriented interface. To include CVS utilities in a Web-based system an architectural approach was adopted. For CVS control files, they must be located in a CVS repository. To make a CVS repository available through the Web, it was decided that it had to be located in the same server as the Web server for security reasons.

According to the architectural approach of *VersionWeb* shown in Figure 1, its functional components (Application Interface and CGI Script*)* work as follows:

- The users, who are Web browsing from client machines located anywhere in the world, interact with CVS through the *VersionWeb* Application Interface and submit requests to the Web server (in general, encapsulating CVS commands).
- A CGI script (physically located on the server) receives and identifies the requests from the users.
- The CGI script, that received the request, makes a call to the CVS (physically located in the server) to pass to it the command to be run.
- The CVS runs the command and returns the results to the CGI script.
- The CGI script sends the results to the user by creating a HTML page using the results of the request.

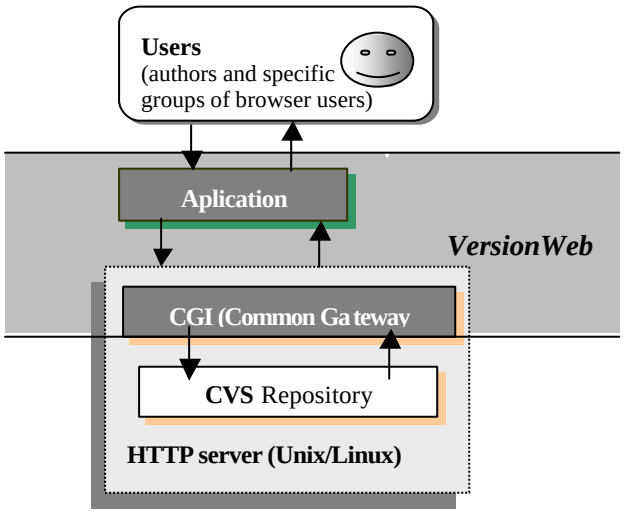

Figure 1 - Basic architecture of the *VersionWeb*

As it can be seen in Figure 1, the CGI Scripts and the CVS repository are physically located in same computer that hosts the Web server. We have chosen to use CGI scripts because they require simplified programming and do not overload the server when running the CVS commands. Furthermore, the Web user, running from a client browser, o not have to install the CVS software in his local machine to run *VersionWeb.*

The tasks are run in the server and not the client browser. Because CVS itself controls the concurrent access to the files, it also forbids information missing or information redundancy. To run *VersionWeb* the user only needs a browser as Netscape Navigator or Internet Explorer.

3. THE *VERSIONWEB*

The Web-based interface of *VersionWeb* was developed mainly to facilitate the user interaction with CVS. Because CVS has only a command-oriented interface, many developers have avoided using it. *VersionWeb* aims to promote a friendly usage of CVS.

VersionWeb has three main interfaces to support the interaction with different roles of its users. We have considered the likely users as general users (specific groups of browsing users), administrators and authors.

The first type of interface (Figure 2), for general users, is accessible through the page where the user is browsing in.

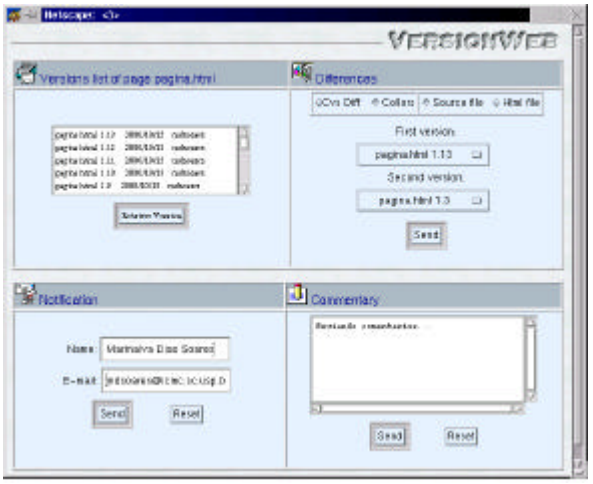

Figure 2 - Interface for general users in *VersionWeb*

This page has a link to some modules of the tool, which are responsible for showing links to all the page versions and the following options:

- To see the differences between page versions.
- To notify new available versions.
- To send criticisms or suggestions to authors related to the page.

The management of the *VersionWeb* users is also made through the Web by the second type of interface, the administrators' interface (Figure 3). In this interface the administrator can manage three basic types of users: administrators (users authorized to manage others), authors (users authorized to manipulate the CVS repository that contains the pages under version control) and groups (users authorized to visualize the page versions and the differences between them).

All the modules implemented in the interfaces correspond to CGI Scripts. The scripts act as a friendly interface between the users and the CVS commands. They issue the proper CVS commands to be executed in accordance with the user requests.

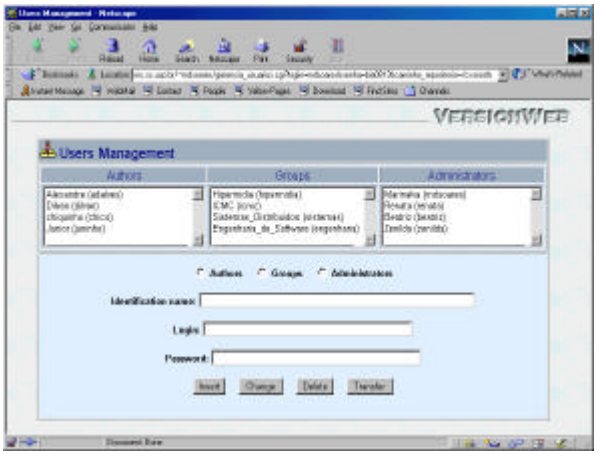

Figure 3 - Users Management in *VersionWeb*

The file management of the CVS repository is made through the authors' interfaces, the third type of interface (Figure 4). This interface offers options to manipulate files and directories. The manipulating options are: inserting files and directories in the CVS repository, removing files and directories from the CVS repository, listing the directory contents of the CVS repository. In addition, the author can make remote checkouts of text files for small changes and can commit them (to generate a new file version) soon after the change. It is also possible to commit later after a local checkout (if the file was modified).

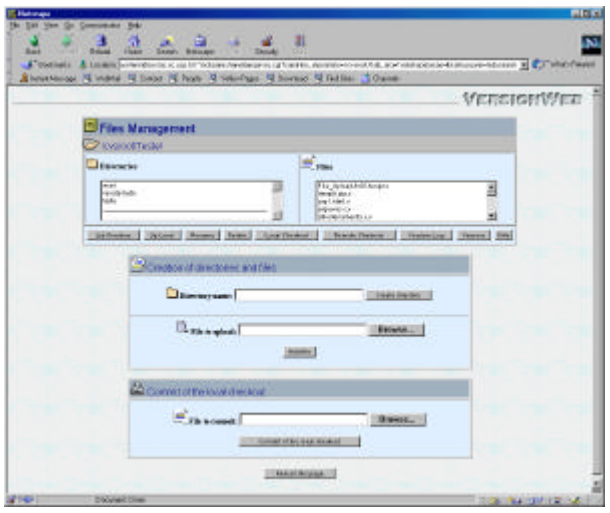

Figure 4 - Files Management in *VersionWeb*

VersionWeb also offers other useful options:

- Seeing a file of version log, i.e., a file containing all the versions of a selected file. The version author, dates, hour and reason of the change are also presented
- Seeing a versions list of a selected file, allowing the author to make checkout of a specific version, so to generate branches from the version, and to merge two different selected versions.
- Uploading a local file to the CVS repository, that is located in the Web server. Thus, the uploaded file can be under version control.

The authors' interface is still being modified and working is done to create operations that can help to generate Web pages releases.

In fact, for generic users and administrators, the interaction mechanisms of *VersionWeb* have already been completely implemented. The authors' interaction via *VersionWeb* is just being completed, but most operations are already ready.

Although *VersionWeb* has only been used by some users, the opinions are that it helps the relevant procedures of version control in Web environment. When users are browsing the pages, they are able to perceive the changes that have been made easily. Therefore, *VersionWeb* reflects the fast and natural evolution of Web information. Additionally, *VersionWeb* favors the collaborative work among developers that can be located in different places, since all of them can access the copies of each other. The shared access to the Web page versions via CVS prevents conflicts among page contents and helps to integrate the page versions in a final copy.

4. CONCLUSIONS

Currently the Web has been presented as a challenging environment because its information evolves continuously. In addition, Web information changes very fast. Webmasters face a big problem when many users are developing pages in parallel. In the face of such problems, a software tool named *VersionWeb* was presented.

VersionWeb is part of a greater project (QualWeb) to help Webmasters ensure quality during development of Web hyperdocuments. *VersionWeb* helps the cooperative work of Web page's authors who can be situated in different and remote locations. In addition, *VersionWeb* allows the user to find changes in a page, recover previous versions of that page and easily see the differences between them.

To provide its main functional feature, *VersionWeb* encapsulates CVS (Concurrent Version System) that is an efficient file version control system. Our work complements CVS by providing a Web-based interface with a friendlier interaction, since CVS has only a command line oriented interface.

The benefits of *VersionWeb* (mainly for the authors and Webmasters) are: to find files changes and its authors easily and to reconstruct previous file versions (both done through the Web page itself). The visualization of differences between versions is shown using colors and therefore helps to position what and where the page version has changed. As future research, we expect to do more tests to evaluate the *VersionWeb* interface as well as its performance using actual data.

REFERENCES

[Cederqvist 1993] P. Cederqvist, Version Management with CVS, *Availabe on-line* *ftp://java.icmc.sc.usp.br/library/books/cvs.pdf*. Visited in August 1999.

[CVS 1999] Concurrent Versions Systems. Availabe on-line http://www.cyclic.com, http://www.loria.fr/~molli/cvs-index.html. Visited in August 1999.

[Douglis *et al* 1998] F. Douglis, T. Ball, Y.-F. Chen, E. Koutsofios, The AT&T Internet Difference Engine: Tracking and viewing changes on the web, *World Wide Web*, Vol. 1, Nº 1, 27-44, 1998.

[Kilpi 1997] T. Kilpi, Product Management Requeriments for SCM Discipline, *Lecture Notes in Computer Science*, 1235, Springer, 186-200, 1997.

[Leblang 1994] D. Leblang, The CM Challenge: Configuration management that works, *In Configuration Management*, W. F. Tichy, Ed., Vol. 2 of Trends in Software, Wiley, New York, 1-38, 1994.

[Lie *et al*. 1989] A. Lie, R. Conradi, T. Didriksen, E. Karlsson, S. O. Hallsteinsen, P. Holager, Change oriented versioning, *Lecture Notes in Computer Science*, 387, Springer, 101-117, 1989.

[Pressman 1997] R. S. Pressman, *Software Engineering*, MacGraw Hill, 4ª edition, 1997.

[Rochkind 1975] M. J. Rochkind, The source code control system. *IEEE Transaction Software Engineering*, Vol.1, Nº 4, 364-370, 1975.

[Sommerville 1995] Sommerville, I. *Software Engineering*, 5ª edição, Addison-Wesley, 1995.

[Sommerville *et al*. 1998] I. Sommerville, T. Rodden, P. Rayson, A. Kirby, A. Dix, Supporting information evolution on the WWW, *World Wide Web*, Vol. 1, Nº 1, 45-54, 1998.

[Tichy 1985] W. F. Tichy, RCS – A system for version control, *Software-Practice and Experience*, Vol. 15, Nº 7, 637-654, 1985.

[Zeller e Snelting 1995] A. Zeller; G. Snelting, Handling version sets through feature logic, *Lecture Notes in Computer Science*, 989, ESEC'95, 191-204, 1995.