Building a Graphical Web History Using Tcl/Tk

Frederick J. Hirsch
The Open Group Research Institute
Cambridge, MA

Building a Graphical Web History Using Tcl/Tk

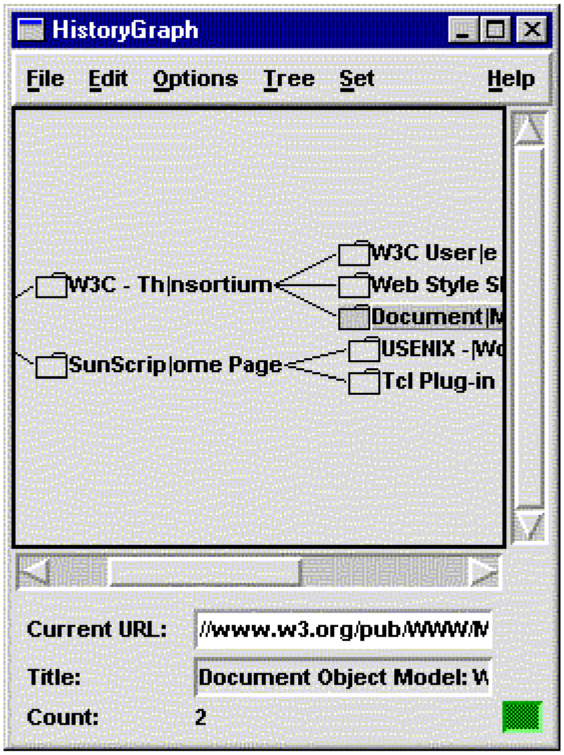
Frederick J. Hirsch (*f.hirsch@opengroup.org*)
The Open Group Research Institute
11 Cambridge Center, Cambridge MA 02142, USA

Abstract

This poster describes the design and implementation of a Web history tool that automatically tracks user browsing activities, presents a graphic visualization of this activity, and provides a mechanism for manipulation and use of the history. This tool, caled *HistoryGraph*, demonstrates the power of using Tcl and Tk, especially through the reuse of existing components to create a powerful application in a short time.

Our goal was to create a browser independent tool which automatically creates a browsing history, making it easy to record sites visited, easy to return to sites, and possible to create trails to share with others. Unlike bookmarks, this history is displayed graphically, is directlymanipulatable and may be integrated with other tools which work with URLs.

HistoryGraph automatically records the URLs and titles of sites visited, and builds a tree representation of the browsing history (See Figure 1).



Current URL: //www.w3.org/pub/WWW/M

Title: Document Object Model: W

Count: 2

Figure 1 - *HistoryGraph* User Interface

The user may reorganize this tree using drag and drop, and revisit sites by clicking on them in the tree. They may create sets of sites by selecting sites using pattern matches on the URLs or titles. Sets may be passed to other applications which may also add sites to the tree. The use of *HistoryGraph* is described in detail elsewhere [Hirsch97].

We decided to implement *HistoryGraph* using Tcl/Tk in order to take advantage of the Tk graphical user interface toolkit, the rapid prototyping environment offered by Tcl/Tk, and the ease of integration with other web applications we had written in Tcl. Our group had attempted previous implementations of *HistoryGraph* using C and user interface libraries but these approaches had taken too long and were too difficult to modify. Using Tcl/Tk we were able to get a basic prototype operating in two months. Tcl/Tk also allowed us to perform incremental development, such as first creating a basic browsing history, then adding the ability to create and manipulate sets, and finally integrating *HistoryGraph* with other applications.

Implementation of *HistoryGraph* required more than Tcl/Tk code. We had to build special code for the NT platform, incorporate a tree drawing widget, and implement browser interfaces. Although Tcl/Tk is a portable interpreted language, we had to build an interpreter for Windows NT, and used the tknt40r1 package to do this.

In order to build an intuitive user interface, showing a graphical representation of sites visited, we used an existing tree widget. This tree widget was built in C++ and required building a special *wish* to incorporate it.

To make a browser-independent tool, we needed two application programming interfaces from browsers:

- An interface to install a callback which notifies the tool when the page changes, passing URL, title, and content to it.
- An interface for the tool to request that the browser load and display a page.

Not all browsers support these interfaces, but those which fully support the Spyglass Software Development Interface [SDI] do. Netscape Navigator 3.02 supports this on Windows NT using the Dynamic Data Exchange (DDE) implementation [Netscape-DDE]; Mosaic supports it on Unix using the NCSA Common Client Interface (CCI).

An alternative to using a browser interface is to intercept the HTTP stream between the browser and server, and capture URL and document information from the stream. This approach may become confusing to the user because the browser cache and "back" button may prevent all information from being transmitted over the network. This approach also requires the user to proxy their browser, and may not work with encrypted streams. We decided that close integration is best.

We integrated the *HistoryGraph* with other applications written in Tcl/Tk by using "send", and by later implementing a notification facility [Meeks]. One such application is designed to determine which pages have or have not changed since a specified time (WhatsNew) and the other is used to return the pages which are linked to by the page (LinkTree). Using Tcl/Tk allowed us to integrate these standalone applications in less than a week, substantially increasing the power of *HistoryGraph* through the reuse of existing applications.

Installation was an issue which made it difficult for users to use the system. Using *HistoryGraph* requires installing a special version of Tcl/Tk on their machine, installing *HistoryGraph*, and remembering to run this application when browsing. This is too cumbersome for everyday use. We would like to find an alternative implementation which would simplify installation and use. We are considering using the Tcl/Tk plugin which could simplify installation while allowing us to use the latest version of Tcl/Tk. This approach would require the use of shared libraries for custom functionality such as DDE support.

Our primary concern is the lack of a portable browser application programming interface which will continue to be supported. The only interface which supports installation of a callback for notification of page changes is DDE on NT with Netscape Navigator, or CCI on Unix with Mosaic. We have not discovered a way to implement this functionality using OLE, plugins, JavaScript or with Java Applets. General support for the Spyglass Software Development Interface is lacking, and full support for the DDE implementation is limited. Without such an interface, the techniques we used for *HistoryGraph* will be impossible to implement. What is needed is an portable application interface supporting tight integration of applications with browsers. Such a portable and supported interface would open the door to many interesting applications involving browsers. We need full support for an interface such as the Spyglass Software Development Interface, ideally from plugins.

A demonstration version of the original *HistoryGraph* software is available at http://www.osf.org/www/waiba/webwarev2_1/.

Acknowledgments

This research was supported in part by the Defense Advanced Research Projects Agency (DARPA) under the contract number F19628-95-C-0042. The views and conclusion contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

My thanks go to Charles Brooks and Scott Meeks for their contributions to this project.

References

Brighton	A. Brighton, *Tree-4.0.1 - A Tree Widget for Tk4.0 based on C++ and [incr Tcl],* http://www.NeoSoft.com/tcl/ftparchive/alcatel/extensions/tree-4.0.1.tar.gz
Netscape-DDE	*Netscape's DDE Implementation*, March 22, 1995. http://www.netscape.com/newsref/std/ddeapi.html
Hirsch97	F. Hirsch, W. S. Meeks, C. Brooks, *Creating Custom Graphical Web Views Based on User Browsing History,* http://www.osf.org/www/waiba/papers/www6/hg.html
Meeks	W. S. Meeks, *Building a Notification Infrastructure Using Tcl, Zephyr, and Linda* In this proceeding.
SDI	*Software Development Interface*, Spyglass, Inc. http://www.spyglass.com:4040/newtechnology/integration/iapi.htm
Welch	B. Welch, *Practical Programming in Tcl and Tk*, Prentice Hall, 1995.