WebLogger: A Data Collection Tool for Web-use Studies

Robert W. Reeder, Peter Pirolli, Stuart K. Card
Xerox Palo Alto Research Center
3333 Coyote Hill Rd.
Palo Alto, CA 94304 USA
+1 650 812 4749
rreeder@parc.xerox.com, pirolli@parc.xerox.com, card@parc.xerox.com

ABSTRACT

Considering the amount of interest in studying Web-browsing behavior, there is a relative lack of tools for data collection in this area. Those tools that do exist have significant limitations on the data they are able to collect or on their suitability for efficient analysis. We present WebLogger, a tool which instruments Microsoft®'s Internet Explorer Web browser. We have found that WebLogger alleviates some of the problems associated with other approaches to browser-based data collection methods.

Keywords

WebLogger, Internet Explorer, browser, instrumentation, Web, WWW.

INTRODUCTION

Although there has been much interest in studying and characterizing Web-browsing behavior, there are few tools to aid in the collection of data toward this end. Some past studies have looked at data captured from servers and proxies, but detailed information about a user's experience can only be captured client-side (i.e. at the Web browser) [5]. Other past studies have looked at client-side data captured from an instrumented version of the now-obsolete XMosaic browser [2, 6], but few studies have looked at data collected from modern browsers. At least one recent study [1] used videotaped data, which is very time-consuming to analyze, and thus limits the number of subjects that can be observed.

Web site designers, user interface researchers, cognitive psychologists, and others interested in characterizing user Web-browsing behavior need a tool which instruments today's popular browsers. However, few attempts have been made to instrument them because these browsers do not offer open source code [5]. In response to the need for such a tool, we have developed WebLogger, which records user- and application-generated events during a user's interaction with one of the standard Web browsers, Microsoft®'s Internet Explorer (IE). This tool overcomes most of the difficulties in Web browser instrumentation that have plagued other researchers (such as how to instrument the Back button, see [7], or scrolling).

WEBLOGGER

Description

WebLogger is a Windows-based application written in Visual Basic that detects and records significant events during a session of use of IE. When WebLogger starts, it simultaneously launches an instance of IE. WebLogger produces a text file, called the *log file*, which documents user and application events at three conceptual levels, which may be called the *Input*, *Interface*, and *Application Levels*. The Input Level corresponds to user actions on the mouse or the keyboard, such as clicks or key presses. The Interface Level corresponds to user actions on the interface elements of IE, such as toolbar button presses, menu selections, and scrolling. The Application Level corresponds to the high-level actions of IE, such as retrieving a Uniform Resource Locator (URL). A complete listing of the events that WebLogger records and their parameters is beyond the scope of this paper.

WebLogger can also, if requested, save the actual Web content (i.e. the text, images, scripts, etc.) at which a user looked during a browsing session with IE. It does this by periodically copying the content stored in IE's local disk-based cache into a "safe" directory where the content cannot be overwritten. Because of the ever-changing nature of the Web, it is difficult to recall the exact content that a user saw during a browsing session, but WebLogger's content-saving feature makes this possible.

Event Detection Methods

WebLogger uses two different methods to detect significant events. The first method involves using Windows system hooks, a common technique for detecting low-level system events such as mouse clicks, keyboard actions, and operating system messages. A hook is a function which is associated with some system event or events. When the event or events associated with the hook fire, Windows calls the hook. This allows the hook to record the event.

The second method WebLogger uses to detect significant events involves Microsoft®'s Component Object Model (COM) technology. COM technology allows independent software objects (i.e. applications or sub-application widgets) to communicate through software interfaces. IE exports an interface called WebBrowser, which allows other applications to call some of IE's Web-browsing functions and to be notified when IE executes certain actions. WebLogger maintains a pointer to IE's WebBrowser interface. Through this pointer, WebLogger receives notification when IE starts to load a URL, when it finishes loading a URL, and when it is scrolling its main window. Documentation and further discussion on COM and the WebBrowser interface is available at [4].

Log File Format

All events in WebLogger are recorded in the same format. WebLogger's event format was designed for both humans and analysis software to read easily. This format stores, in order, an event name, a list of event-specific parameters, the cumulative time elapsed since WebLogger was started (with millisecond precision), the differential time elapsed since the previous recorded event (with millisecond precision), the current Windows system time in "ticks" (a machine-readable format, with one-second precision), and the current Windows system time in human-readable format (with one-second precision). See Figure 1.

```
(NAVIGATE-COMPLETE(http://www.excite.com/)0.431s
0.271s 944777388 14:09:48)

(ACTIVE-SCROLL(0  8  736  2471)  10.475s  0.010s
944777398 14:09:58)

(MOUSE-BUTTON(DOWN   LEFT   941   287)   10.415s
2.944s  944777398 14:09:58)

(TOOLBAR(BACK)    12.788s    0.000s    944777401
```

Figure 1. Four examples of WebLogger event output from a log file. Each top-level parenthetical expression represents one event.

RELATED WORK

Researchers at the University of Toronto have developed WebTracker, a Windows-based application which, like WebLogger, records URL calls and browser menu requests. WebTracker does not capture scrolling events or save content, as WebLogger does, but WebTracker does have the advantage that it works with Netscape's Navigator browser as well as IE. [3]

As mentioned above, some past studies captured client-side browsing events using instrumented versions of NCSA's XMosaic. XMosaic was distributed as an open source application, and thus was easier to instrument than IE or Netscape's Navigator. Catledge and Pitkow in 1994 were the first to study user interface events using an instrumented version of XMosaic [2]. They were followed in 1995 by Tauscher and Greenberg [6].

In a more recent study, Byrne, John, Wehrle, and Crow used videotape to capture subjects browsing the Web, but only collected and analyzed five hours of usable data from eight subjects. [1]

CONCLUSION

WebLogger is a working system for collecting data during a Web-browsing session with Internet Explorer. WebLogger represents an advance over past tools for collecting such data, notably in its abilities to detect scrolling and to save content. Though WebLogger is limited to collecting data in a specific segment of the browser market (those using IE and Windows), this segment currently represents a large set of Web users. Future studies will further demonstrate WebLogger's usefulness as a data-collection tool.

ACKNOWLEDGMENTS

This research was funded in part by grant number N00014-96-C-0097 from the Office of Naval Research. We would also like to thank Don Turnbull of the University of Toronto for sharing his group's WebTracker with us.

REFERENCES

1. Byrne, M.D., John, B.E., Wehrle, N.S., and Crow, D.C. The Tangled Web We Wove: A Taskonomy of WWW

Use. In *Proceedings of CHI '99* (Pittsburgh PA, May 1999), ACM Press, 544-551. 1999.

2. Catledge, L.D. and Pitkow, J.E. Characterizing browsing strategies in the World Wide Web. In *Computer Networks and ISDN Systems* 27: 1065-1073. 1995.
3. Choo, C.W., Detlor, B., and Turnbull, D. Working The Web: An Empirical Model of Web Use. Paper accepted for HICSS 33 (Hawaii International Conference on Systems Science). Available at http://choo.fis.utoronto.ca/FIS/ResPub/HICSS/. 1999.
4. MSDN Online. Microsoft Developer's Network. See http://msdn.microsoft.com.
5. Pitkow, J.E. Summary of WWW Characterizations. In *Proceedings of the Seventh International WWW Conference*, Brisbane, Australia. Also available at http://www7.scu.edu.au/programme/fullpapers/1877/com1877.htm. 1998.
6. Tauscher, L., and Greenberg, S. Revisitation patterns in World Wide Web navigation in *International Journal of Human Computer Studies*, 47(1). 1996.
7. Wexelblat, Alan. Footprints: Interaction History for Digital Objects. Unpublished Ph.D. Thesis, MIT Media Lab. 1999.