Unobtrusive Data Collection for Web-Based Social Navigation

Katja Hofmann[1], Catherine Reed[2], Hilary Holz[2]

California State University, East Bay
25800 Carlos Bee Boulevard
Hayward, CA 94542
[1]katja.hofmann@gmail.com
[2]{catherine.reed, hilary.holz}@csueastbay.edu

Abstract. In initial laboratory studies, *subsymbolic user behavior*[1] has shown promise as a source of information for social navigation. Scalable, unobtrusive methods are needed for acquiring data on subsymbolic user behavior in field studies or live systems. Current methods are not suitable for use outside the laboratory because they interfere with normal user behavior and environment. We present a method for unobtrusively collecting subsymbolic user behavior in web-based systems, and report results from a field study. Our method is unobtrusive in that it uses current web technologies, works on the vast majority of current browsers, requires minimal instrumentation of existing web-based systems, and requires no additional user effort. This unobtrusive data collection method paves the way for future research on using subsymbolic user behavior to improve social navigation.

Introduction

Recently, researchers have been exploring the use of *subsymbolic user behavior* for social navigation. This source of information can be used in situations where explicit feedback is difficult or impossible to acquire, or would interfere with normal user behavior. Claypool et al. [2] analyzed subsymbolic user behavior in recommender systems and found time spent reading (TSR) and time spent scrolling to be good indicators for user interest in web pages, comparable to ratings provided as explicit feedback. Farzan and Brusilovsky [4] show that time-spent reading can improve the quality of a social navigation mechanism in educational context.

Subsymbolic user behavior may contain valuable information that could complement currently used approaches. Hijikata [6] describes a method for extracting keywords that interest the user based on subsymbolic user behavior within a web page (i.e. following text with the mouse, highlighting text). Hijikata shows that this method can extract keywords with much higher accuracy than approaches based on the page as a whole. Subsymbolic user behavior could also be used to identify users that interact similarly with resources, opening up a way to personalize social navigation.

[1] We use the term *subsymbolic user behavior* to refer to low-level user behavior, that is, users' interactions through mouse and keyboard with a web-based system.

We expect subsymbolic user behavior to be especially useful in relatively poorly understood domains, such as in educational adaptive hypermedia.

Little research exists that systematically analyzes the use of subsymbolic user behavior in social navigation. A major barrier is the lack of scalable, unobtrusive methods for acquiring subsymbolic user behavior. Commonly, unobtrusive data collection is done using server log files [4][7] or “strong indicators” [8]. Server log files record user behavior on the level of click streams, *i.e.*, which web pages were visited at what time, and can be used to determine TSR. Strong indicators, such as buying a book or downloading a recipe, record activity that is part of normal user behavior. Strong indicators can be very accurate, but are domain specific and may be difficult to identify for domains that are not well understood.

Initial research on subsymbolic user behavior developed various methods of data collection that were employed in small-scale laboratory experiments. Goecks and Shavlik [5] use a browser plug-in that keeps track of user behaviors such as link pointing, scrolling, and links followed from a website. Claypool et al. [2] collect implicit and explicit feedback through a custom web browser. Hijikata [6] employs JavaScript, Java Applets and an embedded proxy server to record how users interact with textual information on a web page. However, these methods are not suitable for large-scale experiments, field studies, or deployment in a live system, as they require additional user effort, such as installation of software, or configuration changes, or do not submit collected data to a central server.

We present a methodology for unobtrusively collecting subsymbolic user behavior in web-based systems, suitable for large-scale field studies or live systems. Our method is unobtrusive in that it uses current web technologies and requires no additional user effort. We present our method and the results of using that method in a field study on a live system.

The Usertrack Data Collection Method

We developed our data collection method using iterative and participatory design. We began by specifying data and design requirements. Using those requirements we evaluated possible data collection technologies. The requirements, filtered through the possible technologies, determined what kinds of data could be collected and the ultimate experiment design.

The live system we used for our field study is the Adaptive Collaborative UNIX Tutorial (ACUT). ACUT was conceived and initially developed to teach UNIX skills required for computer science (CS) studies, based on small group learning, an informal education metaphor [3]. The current version of ACUT is an open source system primarily intended as an experimental platform for research in adaptive hypermedia based on the small group learning metaphor [1].

As ACUT is based on informal education, the applicability of user modeling approaches traditionally used in Intelligent Tutoring Systems and Educational Adaptive Hypermedia is limited. In contrast to traditional online educational systems based on textbook or problem solving metaphors, ACUT does not prompt for additional information such as answers to questions or specification of problem-solving steps. Rather, ACUT seeks to models users without interfering with normal user behavior or normal user environment.

The need to collect data in the field mandates the following requirements for the data collection method: the method needs to be unobtrusive so as to not interfere with normal user behavior patterns; the method cannot require any additional user effort, such as installation of software or configuration changes; the method must be platform independent, i.e. able to run in any browser the ACUT tutorial can run in; collected data must be transmitted to our server; and the data collection method should require minimal instrumentation in the existing tutorial software.

The general problem of unobtrusive data collection in web-based systems requires a client-server mechanism. User interactions with the web page need to be recorded on the client side; then the recorded data needs to be submitted to a web server for persistent storage and further processing.

Although several client-side scripting languages are in use for recording subsymbolic user behavior, JavaScript is by far the most platform-independent and widely used [6]. The kinds of data that can be collected comprise all events that can be received by JavaScript event handlers. These include mouse movement, clicks, keystrokes, periods of inactivity, scrolling, resizing of windows, moving the mouse pointer over certain elements, etc.

Two possible technologies exist for sending the recorded data back to the server: cookies and hidden form fields. Cookies require little implementation effort as they are automatically submitted to the server and JavaScript has convenient methods to set and read cookies. However, this option suffers from serious size restrictions (commonly 20 cookies of 4096 bytes per domain), and is therefore only suitable for small amounts of data. In addition, cookies are intrusive, as they persist on a user's computer. For the second option, hidden form fields, all links within a web page need to be altered to call a JavaScript function instead of directly issuing an HTTP request. The JavaScript function adds the collected data to a hidden form field within the page. The HTTP request is sent as a POST request, allowing an unrestricted amount of collected data to be sent to the web server. In addition, hidden form fields conform to the stateless nature of the HTTP protocol without intruding onto the user's computer.

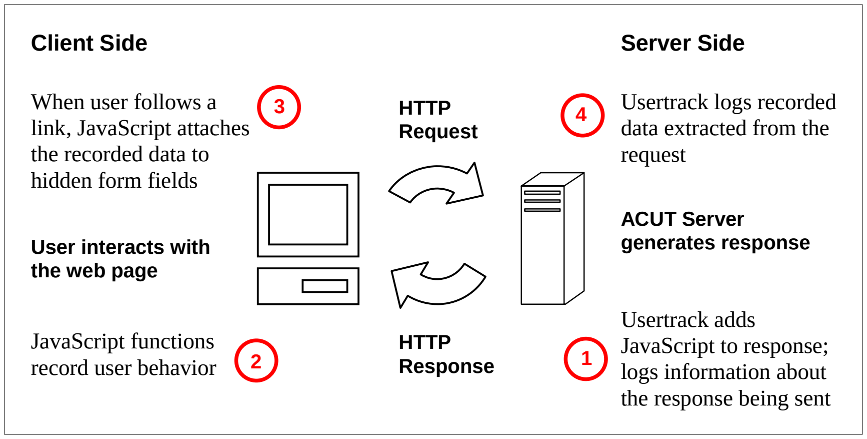
Collecting data through JavaScript and hidden form fields allows truly unobtrusive collection of implicit feedback. Sending data through hidden form fields is part of the HTTP protocol and works on any JavaScript enabled web browser. Thus, our initial analysis identified unobtrusive data collection using JavaScript and hidden form fields as the most promising approach.

Implementation

The devised mechanism for collecting data with JavaScript and hidden form fields was implemented as a mod_perl module named Usertrack, which was then layered onto the existing ACUT server.

Figure 1 shows how Usertrack interacts with ACUT. Step 1: all web pages sent are modified so that links and form buttons within the page do not send an HTTP request, but rather call a JavaScript function. In addition, JavaScript functionality for recording user behavior is added to the web page. Step 2: the processed web page is sent to the client browser where the JavaScript functions keep track of the user's interaction with the web page. Step 3: when the user follows a link the link calls a JavaScript function that adds the recorded data to the request, thus submitting the data to the server as part of the HTTP request. Step 4: the collected data is retrieved from the request and stored in a database.

Fig. 1. Overview of the architecture of the Usertrack data collection mechanism


Client Side
Server Side
When user follows a link, JavaScript attaches the recorded data to hidden form fields
3
HTTP Request
4
Usertrack logs recorded data extracted from the request
User interacts with the web page
ACUT Server generates response
JavaScript functions record user behavior
2
HTTP Response
1
Usertrack adds JavaScript to response; logs information about the response being sent

The mechanics of JavaScript and the fact that many current browsers do not comply with standards raised several implementation issues. As the data collection mechanism is intended to be as platform independent as possible, browser incompatibilities limit some of the data Usertrack can collect.

In order to use a client-server mechanism for recording online user behavior, event timestamps need to be normalized between client and server. To resolve this issue, the client-side JavaScript only records relative timestamps. When the recorded events are sent to the server, the server calculates the actual event timestamps by adding the recorded offset to the time when the corresponding request was served to the client. This method neglects the delay introduced by transmitting a webpage over a network. However, this delay is in the order of milliseconds and will likely not interfere with applications of this data collection mechanism.

Building on Farzan & Brusilovsky's use of "time spent reading" as implicit feedback, we decided to collect data on periods of inactivity. We define periods of inactivity as time during which the user does not interact with the web page via mouse or keyboard. To record such periods of inactivity, a JavaScript function sets a variable to the current timestamp on every "MouseMove" and "KeyDown" event. If the time since the last update exceeds a certain threshold we record inactivity and duration.

In our implementation we collected the following data: mouse-over events (timestamp, html element on which the event occurred); periods of inactivity (timestamp, duration); mouse clicks (timestamp, html element on which the event occurred); scrolling (timestamp); resizing the browser window (timestamp).

Evaluation

We employed Usertrack in a field study to collect usage data for ACUT. The mechanism performed well and we were able to collect the data as planned. In this first field study using Usertrack, 30 users visited more than 600 pages and generated more than 5000 events.

A limitation of the current Usertrack implementation is that it can only send collected data back to the server when the user follows a link within our website. For this reason, the collected data will not be sent to the server when users navigate using the browser's "Back" and "Forward" buttons, type a new address in the browser's address bar, or close the browser window. In our field study we achieved coverage of about 70%, which corresponds with the results on server-side data collection described in [2]. One way to increase coverage would be to exploit asynchronous requests for web-based applications, which allow the client to send and retrieve additional data to and from the web server without issuing a new HTTP request. However, asynchronous requests are only supported by the latest generation of web browsers, and are currently not standardized. Thus asynchronous requests could be added to the Usertrack approach as an additional layer, with browsers not supporting this technology falling back to transmitting data in hidden form fields.

Additionally, web pages on which we want to collect usage data have to be retrieved through our web server. However, technologies such as proxy servers could be used to retrieve any web resource and prepare it for data collection with Usertrack. In ACUT, third party web pages are retrieved via Light-Weight Processes (LWP) and displayed within the tutorial pages.

Table 1 summarizes major advantages of Usertrack as compared to related mechanisms described above. Usertrack does not require any additional effort on the side of the user. Users were able to use ACUT from their preferred web-browser without any additional configuration or installation. The collected data was transmitted to the web server, where it was stored in a database for further evaluation.

Table 1. User effort and location of data storage for each of the described methods for data collection

	Goecks & Shavlik [5]	**The curious browser [2]**	**Hijikata [6]**	**Usertrack**
User effort	Install browser plug-in	Install custom browser software	Install Java Set configuration to use embedded proxy server	none
Data storage	Stored on client system, retrieved manually	Stored on client system, retrieved manually	Data stored on central embedded proxy server	Stored in a data base on web server

The more general approach to collect subsymbolic user behavior with JavaScript and hidden form fields can be implemented in almost any web-based system. The only requirement for the client side is that JavaScript is enabled. On the server side, some form of processing request parameters is sufficient to collect and store recorded usage data (for example CGI, any programming language).

Conclusions and Future Work

Using existing web technologies, we have developed Usertrack, a truly unobtrusive method for collecting data on subsymbolic user behavior. This data collection method increases the amount of available data without increasing user effort or interfering with normal user behavior or environment. Collecting such detailed behavioral data unobtrusively will allow social navigation systems to build finer grained user and interaction models, especially when combining subsymbolic user behavior with current approaches, such as explicit feedback.

Although the coverage of this method is lower than that of the client-side data collection mechanisms described in [2] and [5], on the web pages covered, Usertrack achieves the same high data fidelity, while being truly unobtrusive and scalable. Thus, Usertrack closes a gap between small-scale laboratory experiments on subsymbolic user behavior, and larger scale field studies, as well as supporting the application of the results of these experiments to social navigation in real-world systems. In addition, Usertrack can yield high validity of the collected data as this method minimizes interference with normal user behavior.

In current research, we are investigating the data collected through unobtrusive data collection. Interesting applications include: finding groups of similar users based on patterns in subsymbolic user behavior; or, in web-based educational systems, recognizing when a student is not interacting with a resource effectively. Collecting subsymbolic usage data unobtrusively provides new possibilities for social navigation mechanisms, allowing adaptation based on how users interact with information within a web page.

References

1. Adaptive Collaborative UNIX Tutorial (ACUT) (2006). Retrieved March 01, 2006 from http://acc.csueastbay.edu/~acut/.
2. Claypool, M., Le, P., Wased, M., & Brown, D. (2001). Implicit interest indicators. *Proc 6th Intl Conf Intelligent User Interfaces (IUI '01),* Santa Fe, NM, Jan 2001, 33-40.
3. Farzan, R. (2003). Adaptive Collaborative Online UNIX Tutorial for Computer Science Students. CSU Hayward, CA. Retrieved February 9, 2004, from http://acc.csuhayward.edu/.
4. Farzan, R. & Brusilovsky, P. (2005). Social Navigation Support in E-Learning: What are the Real Footprints? *Proc 3rd Wkshop Int Tech for Web Pers,* Edinburgh, UK, Aug 2005, 49-56.
5. Goecks, J., & Shavlik, J. (2000). Learning Users' Interests by Unobtrusively Observing Their Normal Behavior. *IUI '00,* New Orleans, LA, Jan 2000, 129-132.
6. Hijikata, Y. (2004). Implicit user profiling for on demand relevance feedback. *IUI '04* Funchal, Madeira, Portugal, Jan 2004, 198-205.
7. Kurhila, J., Miettinen, M., Nokelainen, P., & Tirri, H. (2002). Enhancing the Sense of Other Learners in Student-Centred Web-Based Education. *Proc Intl Conf Computers in Education,* Auckland, NZ, Dec 2002, 318-322.
8. Svensson, M., Laaksolahti, J., Höök, K., & Waern, A. (2000). A receipe based on-line food store. *IUI '00,* New Orleans, LA, Jan 2000, 260-263.