Enhancing User Models by Client-Side User-Monitoring

David Hauger
hauger@fim.uni-linz.ac.at

Institute for Information Processing and Microprocessor Technology,
Johannes Kepler University, Linz, Austria
http://www.fim.uni-linz.ac.at

Abstract. Interactivity of web applications has been steadily increasing over the years, especially since the emergence of Web 2.0 interaction paradigms. Nevertheless, "client-side" user interactions in the browser are hardly being monitored within traditional adaptive hypermedia systems. Moreover, despite semantic structure of content, server-side monitoring techniques treat documents as atomic elements. This lack of information corresponds to a level of uncertainty and ambiguity, which might be lowered by means of more granular information on client-side user behavior. This paper presents an approach to retrieve this information by means of Web 2.0 technologies and suggests ways to improve user modeling adaptivity resulting thereof.

Key words: user monitoring, client-side interaction, Web 2.0, user modeling, asynchronous adaptive hypermedia

1 Introduction

Web 2.0 technologies and Rich Internet Applications are establishing new paradigms of interaction on the web, characterized primarily by increased interactivity akin to what was previously only possible in desktop applications. Traditionally, in adaptive hypermedia systems (AHS) user monitoring is done on the server side and therefore interactions within the browser are only being monitored if they cause the browser to send a request to the server. On the server-side this has been typically used for user modeling and consequently adaptations.

Most client-side interactions, however, do not cause requests on the server and are thus not being monitored. Additionally, requesting a document might not be sufficient to assume that the contained content has been read (or interacted with). To retrieve more information, requests need to be further analyzed (e.g. to estimate the "time spent reading" [1]). Subsequent requests allow limited additional assumptions (e.g. interest in the section containing the referring link), but bring up another problem: Lack of further requests (e.g. when closing a browser window) results in no additional information available. As users are not equally interested in all parts of a page [2] a higher level of granularity concerning interactions within a browser is required to tell whether a user really spent time

on a piece of information, looked at all parts of a requested content, what sections of a hypermedia document are especially of interest in, etc. [3]. Moreover, the retrieval of information should not be bound to the request-response cycle.

Although devising more elaborate algorithms to extract information out of a sequence of requests, the approach put forward in this paper departs from that line of work, and seeks to enhance the monitoring process itself. The main goal is to provide adaptive systems with additional information on users by monitoring their client-side interactions, which should lead to enhancements in user modeling by increasing granularity and accuracy. Additional hardware and software shall not be required as this would limit the applicability of the approach.

2 State of the Art

In order to overcome the limitations of traditional modeling techniques, some approaches have already been developed to achieve improvements.

Calculating Time Differences of Requests One way to retrieve additional information without requiring additional adaptation authoring effort is calculating the time between subsequent requests, the "time spent reading" [1]. Although this value is an important indicator for interest, calculating it on the server-side might result in imprecisions, e.g. when opening links in different browser tabs and switching between them. To improve on this, calculating the "time spent reading" on the client-side instead should result in more accurate measurement.

Mapping Pages to Concepts As a document may cover several topics, the total time spent reading might not be sufficient to determine the interest in a single one. Mapping pages to concepts [4] does not change the data available for a single page, but the number of requests and the total time spent on all pages containing a certain concept allows assumptions on the user's interest in a certain concept.

However, if a user is interested in the topics A and B, opens a document containing A and C and a second one containing B and C, the system will assume the highest interest for concept C, which the user might never have read.

Calculating a Level of Activity In order to retrieve information on whether a user has actively been working with a page, another approach has been based on using a browser plugin to monitor mouse and scrolling events and add them up to a "level of activity" [5]. A similar approach described in [6] uses JavaScript for monitoring to increase the acceptance, sends all event information back to the server using hidden form fields and saves the it for future use. However, sending data along with subsequent requests results in losing it if a user leaves a page without following an internal link.

Eye Tracking One of the most advanced technologies to determine what parts of a page a user has actually read is eye tracking [7], which allows the identification of the locus of attention, as well as the level of attendance and tiredness. Unfortunately this approach requires additional hardware and software, which makes a wide-spread use difficult right now.

3 Client-Side Event-Monitoring

It has been found that some client-side user interactions like scrolling and spending time on a page are well suited to determine strong and specific interests, while others like single mouse moves are just indicative of general interest [8]. In order to retrieve the necessary information, corresponding monitoring technologies have to be provided. As additional hardware or software requirements might reduce the level of acceptance and applicability, the current approach uses JavaScript as the core technology for monitoring.

Nevertheless, monitoring exclusively existing JavaScript events it is not sufficient. Hijikata [9] showed that, for instance, text tracing is a strong indicator of user interest, which is not a predefined browser event. However, as it can be regarded as a sequence of other events, it points to the necessity of supporting client-side event-preprocessing as well as event aggregation and generation. This is also important for events of a temporal basis (like spending time on a page) and may be used to tailor the level of detail of events to current needs.

3.1 Introducing Page Fragmentation

One main drawback of traditional modeling techniques is the lack of information on the locus of attention. User interactions need to be associated with spatial information, document elements or sections.

The first approach directly supported by JavaScript is to retrieve the exact position of events. However, resizing windows changes the semantic of positions. Mapping events to the HTML element where they took place might solve this problem, but raises new ones: depending on the structure of the document, an element might be a letter as well as the whole page. Moreover, hovering a single word is not necessarily related to interest in this word, but to this area. Mapping events to keywords extracted from active areas faces the same problems when determining the active area. Therefore, pages have to somehow be split up into (relatively stable) segments with which activities can be associated. This leads to the concept of page fragmentation.

The approach put forward in this paper splits the page vertically into k fragments (e.g. for $k = 5$, each fragment represents a vertical 20% of the page, resizing with the browser window). Although actual line breaks may change slightly, this should be a sufficient approximation as mouse positions are an approximation for the locus of attention as well. For instance, in case of $k = 2$, events mapped to the first fragment refer rather to the beginning of the page, whereas the second fragment rather represents the end of the page.

One benefit of splitting pages this way is the simplification of determining whether the whole page has been visible to the user (which is a prerequisite for reading a document online). Moreover, assuming there is an algorithm calculating whether a page fragment has been read (e.g. based on the time it has been visible and the events that took place) it will be possible to tell how much of a page has been read. Spending a lot of time at the top of the page will no longer be treated equally to slowly going through the whole page.

Further to the above, additional fragments may be added based on semantic information. This may be done manually (if an author wants to have more detailed information on sections of the page) or automatically (e.g. by defining all images as fragments to determine the time spent on them or to split pages by topographical information like headlines). For hypermedia documents assembled from multiple sources (e.g. due to conditional fragment inclusion) it would also be possible, during assembly, to denote and semantically characterize fragments on the basis of their source, their "function" in the assembly, etc.

3.2 Monitoring and Modeling Process

The prototype developed for the approach described herein includes a JavaScript library for monitoring client-side user interactions. Events are already being preprocessed to remove unnecessary data and add additional information on fragments. It also includes custom event triggers for page reading. All events are put on a local event bus. Client-side listeners may use these events to trigger new ones, e.g. if an event is defined by the occurrence of others (text selection), or if events are aggregated (mouse moves, scrolling).

As not all events should be sent to the server, the event dispatcher may be configured to listen to a specified set of events that are sent to a server-side event bus using AJaX. This allows for transmitting data even if there is no subsequent request and out of the request-response cycle. Server-side filtering and event processing is used to set up an interaction model of users and page fragments and to update existing user models.

The system is currently being integrated into a version of AHA [10] embedded in the Sakai e-learning platform [11]. Therefore the modeling server of AHA can be used to store updates on specific user model attributes, which can be readily used by authors.

4 Anticipated Benefits

Monitoring of client-side user-interactions combined with page fragmentation offers a number of additional benefits and possibilities for both structured and unstructured content.

Enhancing Existing Systems by Accurate Data One benefit for existing systems is the availability of more accurate information without requiring further authoring effort e.g. when asking for values for "having read a concept" or the percentage of what has been read. For pages mapped to concepts, the time spent reading may be calculated specifically for each concept instead of treating all concepts within a page equally.

Extending User Models In addition to improving existing user model attributes, models may also be extended, e.g. by identifying learning styles. Current approaches analyze page requests and map pages to content types [12]. As most

pages combine text and picture elements it is not possible to differentiate between graphical and textual ones, but only between interactive and non-interactive ones [13]. However, as it is now possible to identify the time and number of interactions users performed on images (excluding design elements) and to compare them to the amount of time spent on text passages, it should be possible to make more concrete assumptions on the learning style. Another possibility to gain additional information about the user lies in analyzing usage patterns. Perkowitz and Etzioni [14] have already done some work on analyzing access logs; an approach which can be extended by using the additional client-side interaction data. Maybe this will help to identify reasons why a user has skipped a page, as this might be because of an expert already knowing the content, or alternatively a novice not interested in the topic or regarding it as too difficult.

Social Navigation Support "Social navigation" has been defined as "moving 'towards' a cluster of other people, or selecting objects because others have been examining them"[15]. Both can be achieved by the current approach; users may be clustered by their behavior and preferences and it is possible to annotate pages and even fragments of pages that others have found most interesting. In AnnotatEd [16] it has been shown that users tend to select highlighted links; Knowledge Sea II [1] already used page visits and the time spent reading for social navigation. The current approach can on the one hand provide additional information for pages; on the other hand social navigation can now take place even within a page by highlighting popular fragments.

Additional Feedback for Authors The information on the usage of certain fragments can not only be used for social navigation support, but also as feedback for authors. It will help them to identify which parts have been attractive to users and which parts users tended to skip and possibly need to be improved.

5 Future Work

Thanks to initiatives like "The Curious Browser" [8] there are already user studies identifying interaction types best suited to determine user interests. Based on this information appropriate weights for summing up events to an interaction level have to be determined.

Moreover, an appropriate value for the number of fragments within a page has to be established. As within adaptive systems the size of a page may change (depending on what adaptive parts are included), it is important to have a fixed number of fragments (k). However, a certain percentage of a page might represent a very different amount of text. For large pages maybe a higher k would be required than for smaller ones. Two experimental approaches will be used to determine an optimal number of fragments. The first one will start with a high amount of fragments. Values for neighboring fragments will be compared in order to find out to how many differing sections this large amount of segments could be reduced to. The second approach will allow users to adjust the value for k, so that they get the subjectively optimal results in their user model.

Acknowledgements

The work reported in this paper is funded by the "Adaptive Support for Collaborative E-Learning" (ASCOLLA) project, supported by the Austrian Science Fund (FWF; project number P20260-N15).

References

1. Farzan, R., Brusilovsky, P.: Social Navigation Support in E-Learning: What are the Real Footprints? In: Third Workshop on Intelligent Techniques for Web Personalization (ITWP '05). At 19th Int. Joint Conf. on Artificial Intelligence. (2005)
2. Brusilovsky, P.: Methods and techniques of adaptive hypermedia. User Modeling and User-Adapted Interaction **6**(2-3) (1996) 87–129
3. Hauger, D.: Fine-grained user models by means of asynchronous web technologies. In Hartmann, M., Krause, D., Nauerz, A., eds.: ABIS 2008 - Adaptivity and User Modeling in Interactive Systems, Würzburg, Germany (2008) 17–19
4. De Bra, P., Calvi, L.: Aha: a generic adaptive hypermedia system. In: Proceedings of the 9th ACM Conference on Hypertext and Hypermedia. (1998) 5–12
5. Goecks, J., Shavlik, J.W.: Learning Users' Interests by Unobtrusively Observing Their Normal Behavior. In: Int. Conf. on Intelligent User Interfaces - Proceedings of the 5th Int. Conf. on Intelligent User Interfaces. (2000) 129–132
6. Hofmann, K., Reed, C., Holz, H.: Unobtrusive Data Collection for Web-Based Social Navigation. In: Workshop on the Social Navigation and Community based Adaptation Technologies. (2006)
7. García-Barrios, V.M., Gütl, C., Preis, A.M., Andrews, K., Pivec, M., Mödritscher, F., Trummer, C.: Adele: A framework for adaptive e-learning through eye tracking. In: Proceedings of IKNOW 2004. (2004) 609–616
8. Claypool, M., Le, P., Wased, M., Brown, D.: Implicit interest indicators. In: Intelligent User Interfaces. (2001) 33–40
9. Hijikata, Y.: Implicit user profiling for on demand relevance feedback. In: IUI '04: Proc. of the 9th int. conference on Intelligent user interfaces. (2004) 198–205
10. De Bra, P., Ruiter, J.P.: AHA! Adaptive Hypermedia for All. In: Proceedings of the WebNet Conference. (2001) 262–268
11. Sakai: collaboration and learning for educators, by educators, free and open source. http://sakaiproject.org (2009)
12. Papanikolaou, K., Grigoriadou, M.: Accommodating learning style characteristics in adaptive educational hypermedia systems. In: Individual Differences in Adaptive Hypermedia Workshop at the Third Int. Conf. on Adaptive Hypermedia and Adaptive Web-based systems, AH'04, Eindhoven, Netherlands (2004)
13. Graf, S.: Adaptivity in Learning Management Systems focussing on Learning Styles. PhD thesis, Vienna University of Technology (December 2007)
14. Perkowitz, M., Etzioni, O.: Adaptive web sites: Concept and case study. Artificial Intelligence **118**(1) (2001)
15. Dourish, P., Chalmers, M.: Running out of space: Models of information navigation. Proceedings of HCI'94 (1994)
16. Farzan, R., Brusilovsky, P.: AnnotatEd: A social navigation and annotation service for web-based educational resources. New Review in Hypermedia and Multimedia **14**(1) (January 2008) 3–32