Chapter 10
Using Ajax to Track Student Attention

Jan Newmarch
Box Hill Institute and Monash University, Australia

ABSTRACT

Tracking the behaviour of users of online learning systems is an important issue, but current techniques have not been able to give deep views on what users do with Web-based learning systems. This paper shows how the use of Ajax can provide a richer model of how users interact with Web systems. In this paper, the authors will discuss a case study used to better track behaviours of online learning systems and how Ajax improves this understanding of user interactions.

INTRODUCTION

Any producer of web-based material is interested in what users do with the pages they visit: what do they visit, how long do they spend there, and what do they do while there? This can be of high commercial value, as information about users can be used to revise pages in order to draw customers into a commercial site, and hopefully to spend money. In the educational domain, knowledge of a user's activities can help to build a better educational experience. The intent is to build up a model of the user and to customise the site to desirable users.

There have been three common techniques used to track user activity: web server logs (Schluting, n.d.), custom-built browsers (Edmonds, 2003; Velayathan & Yamada, 2006) or visual observation such as in a usability lab. They are all well-known to have significant drawbacks, as discussed in the next section. Recently a technique called Ajax (Holdener, 2008) (for Asynchronous JavaScript and XML) has come to the fore. Primarily this is used to give a more interactive experience with a web site, and has been used by companies such as Google (in Google maps).

DOI: 10.4018/978-1-4666-0336-3.ch010

HTML 4 compliant browsers support event tracking using languages like JavaScript, such as when a user enters and leaves a page. They also allow "focus" tracking, which can occur when a user switches to, say, an email program without leaving a web page. Combined with the asynchronous aspects of Ajax, we show in this paper how this can be used to give a clearer picture of what a user is doing. We demonstrate the use of this with a formal course for teaching Linux administration.

Consider the scenario. Johnny has been instructed to look at some courseware in his browser. He navigates to the page, but after 5 minutes he gets bored. He switches to another tab so he can read his Google mail for 20 minutes. Then he switches back to the courseware page. After another 5 minutes he decides to talk to a friend and starts up Skype. 10 minutes later he returns to the page and finally follows a link in that page to another page of the courseware.

Simple observation of server logs would suggest that Johnny spent 40 minutes on the first page, whereas a closer examination shows that he only spent 10 minutes. This paper shows how to perform some of this closer examination.

The structure of this paper is as follows: the next section discusses current techniques for tracking user behavour. The section after that looks at Ajax and how we use this to generate information. Following this are a number of sections discussing issues arising from this use of Ajax and how to analyse the information gained. Finally an examination of actual server logs is given to show how to give a more accurate picture of a user's browsing habits, and future work is discussed.

The principal contribution of this paper is that it shows how deeper analyses of student use of web-based courseware can be performed, and illustrates this with a case study. Similar techniques could be used in other situations.

TRACKING USERS

HTTP logs are collected by HTTP servers such as Apache. Generally these logs use the Common Logfile Format (WWW Consortium, n.d.a). These record which pages are accessed, the date, which IP address made the request, and optional other information such as referring page. These logs form a relatively simple way of measuring what users are accessing. However, they only give partial information. They show the requests that actually made it to the server: many organisations now use proxy caches, and if there is a "hit" on a cache, then the request will be handled by the proxy and not make it back to the source server. This can be alleviated by setting the Expires time for each document to zero, but breaks the value of caching.

If the user makes use of the Back button in the browser, then the document will be retrieved from the browser's own cache. This cannot be avoided except by disabling the Back button.

The principal problem is that the server logs can only show that a page is requested from a server. What is done with that page is unknown. A user may examine it for a long time or simply discard it. Further, it is not clear whether it is a human using a browser or some automated agent such as a spider.

After any one request, if another is made to the same site then another entry is made in that server's log. This provides an upper bound to the time spent on the requested page. But the user may have been somewhere else, or may just never come back. A second technique is to use a special-purpose browser which logs each user activity. Such browsers can record a great deal of information. There is a minor problem of getting the information back to the server. The major problem is persuading people to use such a browser. Typically this can only be done with a relatively cooperative group of people, as a research experiment.