Measuring children’s search behaviour on a large scale

Emiel Hollander, Theo Huibers, Hanna Jochmann-Mannak, Paul van der Vet

Human Media Interaction,
University of Twente
PO Box 217, 7500 AE Enschede
{e.s.hollander,t.w.c.huibers,h.e.mannak,p.e.vandervet}@ewi.utwente.nl

ABSTRACT
Children often experience problems during information-seeking using traditional search interfaces and search technologies, that are designed for adults. This is because children engage with the world in fundamentally different ways than adults. To design search technologies that support children in effective and enjoyable information-seeking, more research is needed to examine children’s specific skills and needs concerning information-seeking. Therefore, we developed an application that can monitor children’s search behaviour on a large scale. In this paper, we present the steps taken to develop this application. The basis of the application is UsaProxy, an existing system that is used to monitor the user’s usage of websites. We have increased the accuracy of UsaProxy and have developed an application that is able to extract useful information from UsaProxy’s log files.

Categories and Subject Descriptors
H.3.3 [Information Storage and retrieval]: Information Search and Retrieval – Query formulation, Search process, Selection process

General Terms
Measurement, Documentation, Human Factors.

Keywords
Children’s search behaviour, data logging.

1. INTRODUCTION
Interactive technology plays an important part in children’s lives. Every day, more children have access to the internet and more information becomes available for them through the internet. The question is if existing search technologies support children in effective and/or enjoyable information-seeking.

Children’s search behaviour has not had a lot of attention in research over the past few years. It is quite interesting, however, because children’s search behaviour differs from the behaviour of adults in many ways. It also differs between various age groups. For example, young children who have just learned to read, may search using only one or a few words, may make more errors when typing and may benefit from images while browsing the results. Therefore, examining children’s search behaviour to design search technologies that support children in effective and enjoyable information-seeking, is an important research topic. We will discuss this in more depth in the next section.

Currently, the search behaviour of children is mostly examined in an experimental setting using additional equipment like eye-tracking devices to observe and record search behaviour [10]. This produces a large amount of useful high-quality data, but children and their parents need to travel to a place where the experiment can be held and the method is very time consuming. The number of children that can participate in such experiments is thus limited.

We were interested in finding a way to make it easier for children to participate in research on search behaviour and to gather data from a far more larger group of children. Therefore, we developed an application that can monitor children’s search behaviour on a large scale. This application may be installed on any computer, and may also be accessed through the internet. Using a network of libraries, primary schools and interested parents, we will be able to measure children’s search behaviour all over The Netherlands. In our research, we are mostly interested in children from eight through twelve years old.

To our knowledge, there are no studies conducted that use log analysis techniques to examine children’s search behaviour on a large scale. Measuring search behaviour on a large scale, using deep log analysis techniques, is far more common with research on adults’ search behaviour. These studies, however, are mostly aimed at evaluating the usability of specific websites or applications and not aimed at examining what principles are underlying on the search behaviour of the adult users. For example, Nicholas et al. [16] evaluate the usability of digital scholarly journals using deep log analysis techniques.

Although there are limitations to what one can measure with log analysis as compared to experiments in a controlled environment, this is offset by the fact that a far larger group of children can be involved. We believe that quantitative data from this type of research can provide interesting hypotheses that can be examined in more depth in experimental settings. This makes the research into the application useful.

This report follows the steps that were needed to develop the application to monitor children’s search behaviour. First, we determined which variables are useful to measure in assessing children’s search behaviour (Section 4). After that, we compared the variables that we wanted to measure with existing applications or systems, to see which one matched our wishes best (Section 5 and 6). Even the best match did not offer the optimal solution for our research goals. Therefore, we needed to adapt the best matching system to our situation. Finally, the characteristics of the application that we developed to assess the usage of information retrieval systems by children, will be discussed in Section 6.
Before discussing the development of the tool in more depth, we will first discuss how children are different from adults and what problems these differences cause for children in using digital technologies. By discussing these problems, we want to stress the importance of developing search technology that is tailored to children’s specific needs and skills. We conclude that more research is needed to examine children’s search behaviour and the developed tool presented in this paper can be used for this type of research on a large scale.

2. PREVIOUS RESEARCH ON CHILDREN’S SEARCH BEHAVIOUR

Design principles that are applied on search interfaces for children are often a projection of adult’s vision about children’s preferences. Interface designers take adult media content and attempt to make it ‘childlike’, by simplifying the content and adding more visual design (e.g. lots of colours) and multimedia (e.g. videos and animations) [17]. Also, children are often thought to be as web savvy as adults and sometimes they are even thought to understand technical terminology better than their parents. Are children that web savvy as most people think? Do they find information that is easily using search technologies that are initially designed for adults?

2.1 Why do children experience problems?

Developmental psychologists show why children are fundamentally different from adults. Children are not just little adults that only lack knowledge and experience in comparison to adults. They fundamentally experience and understand the world different than adults [9]. Jean Piaget [18], for example, described in his earlier works how children’s cognitions evolved during a series of four stages from sensorimotor (birth to 2 years), to preoperational (ages 2 to 7), to concrete operational (ages 7 to 11) to finally, the formal operational stage of development (ages 11 and up). Contemporary research recognizes that all children develop differently, but that Piaget’s general characterizations of children still remain useful. Also when examining children’s interactions with digital technology.

Because of these fundamental differences between children and adults, children can experience all kind of problems while exploring digital environments and using digital technologies.

2.2 Dexterity

Children’s motor skills are not equal to that of adults. Therefore, traditional input devices can be difficult for children to use. They have difficulties holding down the mouse button for extended periods or to drag-and-drop objects using the mouse [12]. Also typing can be difficult for children, because they have to ‘hunt and peck’ on the keyboard for the correct keys [7].

2.3 Problems with searching

There are several reasons why children have more difficulties with formulating search queries than adults. Children have less knowledge to base ‘recall’ on than adults [7, 10] and they rarely access their previous knowledge of the topic during formulating search queries [21]. They also do not have a very developed vocabulary as adults do and they have difficulties with correct spelling, spacing and punctuation, which is needed for most search engines to find relevant search results. Also, moving from natural language to a single keyword is more difficult for children than for adults [20].

2.4 Browsing problems

Browsing can also be more difficult for children than for adults. Children have difficulties to understand and select abstract terms; search tasks are more successful when concrete terms are used [7] and children find it easier to retrieve concrete terms than abstract terms [4]. Children can also have trouble understanding categories and finding the right category, because they have less domain knowledge and less vocabulary knowledge than adults [7]. The same problems occur with the use of metaphors from the adult world, such as file folders or filmstrips, which are unfamiliar to most children [9].

2.5 Interaction style

Children’s patterns of attention and interaction are quite different from those of adults. Traditional task-oriented analyses of activity does not support the playful, spontaneous nature of children’s interaction with technology [9]. Children are more reactive searchers and are more chaotic in their search performance than adults. They make more web moves, backtrack more often, loop searches and deviate more from their designated target [6].

3. RESEARCH OBJECTIVES

This paper reports on research to find a solution for measuring children’s search behaviour that can be carried out without the need for a completely tailored experimental setting.

Four questions can be identified that we will need to answer in order to find such a solution. Each of these questions will be discussed in the following sections.

1. Which variables are useful to measure to assess search behaviour of children?
2. Which of these values can be measured using only an application that can be installed or used on any computer?
3. What are the solutions for measuring search behaviour that already exist and which values do they measure?
4. Which of these solutions matches best with what we want to measure, and how does it need to be adapted to make it a perfect match?

4. MEASURABLE VARIABLES

We would like to measure everything that we are able to measure using only an application that can be installed or used on any computer without needing any extra equipment.

4.1 Variable groups

We can divide the variables that can be measured when people are using a search engine in four distinct groups.

1. Variables from measurements done directly on any computer involved in the search process. These are for example the number of clicks, amount of scrolling, speed of typing or which documents are retrieved.
2. Variables from measurements done by external equipment. An example of this are the values obtained by using an eye tracking device.
3. Variables obtained by user input. These are acquired by asking the user questions before, during or after the task.
4. Variables obtained by observation. The researcher sits near the user participating in the experiment and makes notes of everything that happens.
Research on the topic of information retrieval for children covering variables from groups 2, 3 and 4 is available. An example of a method that is aimed at obtaining user information from group 3 from children is The Fun Toolkit [19].

We are, however, in search of an application that can gather information about children’s behaviour when using search engines, that can be installed or used on any computer and without the needed interference of a researcher. We therefore can discard everything in group 2 and group 4. Also, since we want to directly observe behaviour, group 3 can be discarded as well.

Group 1 can be further divided into two distinct subgroups.

a. Variables which can only be measured locally on the computer being worked on. Examples are the number of clicks, amount of scrolling and speed of typing.

b. Variables which can also be measured on a server that the computer contacts. These are, for example, the documents that are retrieved or any other kind of server log.

Logging mechanisms on web servers are very common, so data from group 1b is readily available. We are primarily interested in measuring variables from group 1a. This data is not readily available; we need additional applications to gather it.

4.2 Overview of variables

There are several ways to gather a complete list of variables we are able to measure. The first approach is to find out which variables we are technically able to measure, using the input from devices usually attached to a computer. This will deliver variables such as mouse movements, clicks and keystrokes.

The second approach is to define what we would like to know and from there deduce which variables are needed exactly to be able to measure this.

Both approaches will deliver multiple layers of variables. Some that may be measured directly, others that can be deduced from existing measurements. To help perform both approaches, we will first study existing literature to find out which variables other researchers have chosen to measure in similar research.

4.2.1 Existing literature

Bilal has done measurements while children were performing search tasks [5]. She has divided the variables that can be measured into two groups: Transcribed Moves and Selection Actions. These groups are not distinct.

The Transcribed Moves consist of “moves that include all traversal behaviours” and are:

- searching
- browsing
- looping
- backtracking
- screen scrolling
- mouse movements
- exploratory moves

Selection Actions include only

- searching
- browsing (hyperlink activation)
- looping

Of these variables, only screen scrolling and mouse movements belong in group 1a. All the other variables can also be measured on the server that handles the requests (group 1b). Backtracking may not be measurable on the server when the client uses its cache, but this can easily be resolved by disabling the client’s cache.

Schacter et al. have also performed analyses while children were performing search tasks [21]. They have logged the following variables:

- time spent on each web page
- time spent on each task
- total number of mouse clicks per task
- keywords entered
- URLs of all web pages visited

From these variables, the mouse clicks clearly belong in group 1a.

Kalsbeek and De Wit have, in cooperation with a primary school teacher, constructed a list of errors commonly made by children when they use search engines [13]. The list also contains a few differences in search behaviour of children when compared to adults. It may be possible to automatically measure some of these errors. A small excerpt from the list that Kalsbeek and De Wit made, is shown below. This list contains the items that can be detected automatically by software.

Table 1: Classes of differences in search behavior between children and adults (excerpt from [13])

<table>
<thead>
<tr>
<th>Class</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number words</td>
<td>W8, 4u, xs4all</td>
</tr>
<tr>
<td>No vowels</td>
<td>hill wrld (hello world)</td>
</tr>
<tr>
<td>Special characters</td>
<td>€pe (Europe)</td>
</tr>
<tr>
<td>Smileys</td>
<td>:) , ;-)</td>
</tr>
</tbody>
</table>

4.2.2 Defining variables

Using this knowledge, we can now define a complete list of variables that we can measure. Our objective is to make a generic tool for measuring behaviour. The second approach of defining variables is therefore less feasible in this situation. We cannot explicitly define what we want to know, because this will differ for each experiment we will conduct using this tool.

First, we will build a list of variables we can technically measure using the first approach. This is the bottom-most layer of variables; the ones we obtain by directly measuring data. We will call these “device data”.

Both Bilal and Schacter have measured mouse movements and clicks. These clearly belong to device data and should therefore be in our list. Both researchers also keep track of which pages are visited. These can directly be obtained from server logs and therefore also belong to this group.

We can directly measure what the user types on the keyboard and what information is on the screen. Below we give an overview of all device data variables.

<table>
<thead>
<tr>
<th>Level 1: Device data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keyboard input</td>
</tr>
<tr>
<td>Keystrokes</td>
</tr>
<tr>
<td>Mouse input</td>
</tr>
<tr>
<td>Movement (coordinates)</td>
</tr>
<tr>
<td>Clicks (coordinates)</td>
</tr>
<tr>
<td>Scrolling</td>
</tr>
</tbody>
</table>
A timestamp is stored with each piece of device data so that afterwards we can reconstruct exactly what happened at what moment in time. This data can be used directly to deduce a lot of other information. This will be called “directly derived data”.

Schacter [21] mentions that he has logged the used keywords. These can be derived from the keystrokes a user has entered. Also logged by Schacter is the time spent on each page, that can also be derived from the server logs.

An overview of all directly derived variables is given below.

<table>
<thead>
<tr>
<th>Level 1: Device data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Screen data</td>
</tr>
<tr>
<td>Server logs</td>
</tr>
</tbody>
</table>

When we have obtained the values for these variables, these can be used to derive even more information. Some examples of information that can be obtained are given below.

<table>
<thead>
<tr>
<th>Level 2: Directly derived data</th>
</tr>
</thead>
<tbody>
<tr>
<td>From keyboard input</td>
</tr>
<tr>
<td>Words entered</td>
</tr>
<tr>
<td>From mouse input and screen data</td>
</tr>
<tr>
<td>Buttons and links clicked</td>
</tr>
<tr>
<td>Items hovered over with cursor</td>
</tr>
<tr>
<td>From server logs</td>
</tr>
<tr>
<td>Amount of time spent on each page</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Level 3: Indirectly derived data</th>
</tr>
</thead>
<tbody>
<tr>
<td>From words entered</td>
</tr>
<tr>
<td>Spelling errors</td>
</tr>
<tr>
<td>From amount of time spent on page and clicks</td>
</tr>
<tr>
<td>Path taken through web site</td>
</tr>
<tr>
<td>From links clicked and items hovered over</td>
</tr>
<tr>
<td>Links the user may have hesitated to click on</td>
</tr>
</tbody>
</table>

A variable on level 3 may be an answer to the main research question of an experiment. This question is the starting point for the second approach to obtain all variables: define what we want to know and then deduce which variables are needed. We would then have a research question, find out which variables from level 2 are needed for this and finally, we know which variables from level 1 we need to measure to obtain the information we want.

5. MEASURING VARIABLES

Now that we know which variables we can measure, we need to find an application that is suited best to measure all of these.

Applications that monitor and log user actions do already exist. Broadbent et al. [8] have developed a test case that measures a number of metrics for information retrieval systems. Muresan and Bai [14] have developed a methodology that is aimed at designing the user interface, logger and log analyzer in such a way that as little useful data as possible is lost. More methods like this exist. However, none of them is tailored for usage with children.

There are keyloggers and mouse recorders available for general use, like for example the Keyboard and Mouse Recorder [1]. These tools have, however, not been specifically designed for usage as information source for analyses but rather to simplify computer tasks or for less decent things like stealing passwords.

Also, tools that record exactly what is happening on the screen are widely available. These tools only capture the pixels that are present on the screen and do not log any additional information. This makes analysing these videos a cumbersome task.

Web server logs record which pages have been visited by a person. Unfortunately, that is all they do. They do not register what a user does while on a page, and it is also not always possible to deduct a path the user followed through the site based solely on the web server logs.

WebQuilt tries to solve this problem by having the user visit web pages through a proxy [11]. This proxy records the path a user takes through the site. WebQuilt is able to visualise this information as well; one can easily see which path was chosen most for a certain task. What happens while the user is on a web page is unfortunately not visible when using this tool.

Mueller and Lockerd have developed Cheese, a tool that tracks mouse movement activity on websites [15]. This tool uses embedded scripts to automatically send mouse movement data to the server so that it can be stored there. They have manually visualised this information and evaluated the data they have collected.

Cheese only tracks the position of the mouse on the screen. It does not log what is on the screen and also does not log keypresses. Also, visualisation of what happens is not automatically done by this system.

MouseTrack, another tool specifically designed for websites, does have visualisation options [2]. This tool does not focus on logging mouse clicks, but rather on mouse browsing paths within a website. It is able to display these paths using a variety of visualisation options.

Another tool that is geared towards visualisation of mouse movement on websites is (smt) Simple Mouse Tracking [22]. This tool allows the researcher to exactly replay any user’s mouse paths over the original web pages. It is also able to deduce from its logs information about “the user’s skills, how he uses the web interface, if he is an impatient person, etc.”

These two tools, however, do also not log keyboard input. A tool that does log keyboard input is UsAProxy [3]. The aim of this tool is to log as much as possible while being as unobtrusive as possible to the user. It logs mouse movement, mouse clicks and keyboard input. It also logs which element of the document has been clicked on or has been typed into. Unfortunately, it does not offer visualisation options.

6. ARCHITECTURES

From the applications listed above, we can extract three architectures that are common for applications like this. These architectures are:

- completely client-side,
- completely server-side,
- using a proxy between the client and server.

We will now further investigate these different architectures.

6.1 Completely client-side

This category consists of applications that only need to be installed on the client and do not need a server to retrieve information from or store information on. The Keyboard and Mouse Recorder [1] is an example of an application that is completely client-side.
The advantage of this kind of applications is that they usually can be installed on any computer. There are no further requirements; install the application and we can immediately start measuring. Another advantage is that we can obtain usage information from all applications running on that computer, and not just for one website or application as would be the case when using a server-side or proxy solution.

The large disadvantage of applications like this is that they can only see where the mouse is on the screen -its actual coordinates- but they cannot see, for example, whether the user is hovering over a link, clicking a button or just clicking randomly.

This is also the case for key loggers. They can see in which application the user is typing, but they cannot see in which field the user is entering information. For our purpose, this is vital information.

There are a number of ways that we could get this additional information using only client-side applications. The first is to also record what is happening on the screen, for example, by using a video recorder. This means that the researcher needs to watch the video to deduce information out of it. When used in combination with a key logger or mouse recorder, we can use the information from these applications to quickly fast forward to potentially interesting moments in the video. This reduces the amount of work needed to analyse the video. It is, however, still not an ideal solution.

The second way is to develop a browser plug-in. Such a plug-in will not have the advantage of being able to capture information about all applications running on the computer. However, it will have complete access to the current web site the user is visiting. It can determine exactly what part of the site the user is interacting with; for example, which link he is hovering over with his mouse, or which input box he is typing text into. We have, unfortunately, not been able to find such a plug-in readily available.

6.2 Completely server-side

Completely server-side may be a misleading title, since we will always need scripts running on the client to collect and send information about, for example, mouse movements and key presses. However, completely server-side implies that we do not need to install anything on any client computer that we use for measurements.

Simple Mouse Tracking [22] is an example of a logging application that is completely server-side. The web pages that (smt) is used on, need to have been altered to contain a piece of JavaScript-code which takes care of mouse tracking.

Cheese [15] is another tool that uses this approach. The authors have also embedded scripts in their web pages that collect information about mouse movements.

The advantage of solutions that are completely server-side is that there is no need to install an application on every client that is used to measure variables. Measures will be done for all visitors that visit websites which have these scripts embedded within them.

The disadvantage is that the actual website we want measurements for needs to be altered in order to obtain these measurements. We have to manually embed JavaScript in these websites for these server-side solutions to work.

6.3 Using a proxy

We define a proxy as any entity which fetches the desired website for us and makes sure it is able to carry out measurements on the usage of this website. This entity may be an application, a separate server or something else. One could argue that an application, running on the same server as the application serving web pages, that acts as an extra layer between us and the application serving web pages, is a completely server-side solution. However, using our definition, such an application would be defined as a proxy.

MouseTrack [2] is an example of such an application. It consists of a PHP script that fetches the desired web page and enhances it with JavaScript, that is used to measure usage of the page and also to visualise this data.

UsaProxy [3] uses a different approach. This application is a full-fledged HTTP proxy that will forward any request to the actual web server. It will then modify the responses coming from the web server to contain JavaScript that will collect usage information.

The advantage of this approach is that UsaProxy can be installed, either on the client that is used to visit the web sites, or on the server which delivers the web pages. MouseTrack can only be installed on a computer that runs on a web server.

7. OUR APPLICATION

We have chosen to use UsaProxy for our efforts. This system is the most flexible of all discussed systems, supplies all necessary data and is open source.

UsaProxy embeds JavaScript in all web pages that are requested. This JavaScript is used to send information about events back to the application while the user is browsing sites. An event occurs whenever the user moves the mouse, clicks, types text, etc. To send this information back, UsaProxy makes use of the AJAX functionality that is present in all modern browsers.

UsaProxy logs this information in a log file. This file consists of one entry per event that happens. During our tests, the size of this log file grew with approximately 30 kB per minute.

There were, however, a few drawbacks that we had to attend, to make the system useful for our research on children’s search behaviour. First of all, UsaProxy could initially only measure events with an accuracy of a second. We have improved the accuracy of the system to one millisecond.

The second drawback was that UsaProxy is unable to record key presses on the delete and backspace buttons when using Internet Explorer. The user could delete letters from his query without UsaProxy noticing it. The number of words and the average length of words that we extract from the raw data is therefore an estimate. Firefox does detect these key presses. However, since we cannot be sure which browser is used when conducting experiments, we have chosen to disregard these key presses.

The third drawback was that the raw data the system delivers, was hardly usable for analysis. We needed to derive data from this that is more usable.

To solve this problem, we have written an application that takes a directory of log files and extracts useful information out of them. We have chosen to, at first, extract the following information from the log files. The level mentioned between parentheses is the level from section 4.2.2 that the variable belongs in.

1. Speed of typing (level 1)
2. Number of clicks per session (level 1)
3. Number of words per query (level 2)
4. Average length of words in the query (level 2)
5. Session length (level 2)
This set of variables is easily extractable from the log files and serves as a good starting point to assess the functionality of our application in this preliminary stage of our research.

We have defined a session to take place between consecutive queries. When a user returns to the start page to start a new search query, a new session starts. The information shown above is gathered and grouped per session.

Our application has been written in Java and can be run on any computer that has a Java Virtual Machine installed. The researcher can select a directory that contains log files which have been produced by UsaProxy. He can also enter what the start page is, so that the tool can identify where a new session starts.

The information that the application extracts from the log file is then stored in a comma separated values file. This file format is easily readable by Excel or other analysis applications.

In our research, we need to make sure that the information UsaProxy logs, is only generated by children using the websites, not by adults. This is where UsaProxy’s flexibility proves its usefulness. We can install it anywhere, also on a computer that we know is only used by children.

The combination of an adapted version of UsaProxy and our own application, that extracts useful information out of UsaProxy’s log files, is well suited for our research on children’s search behaviour. We think that this combination may also be useful for researching search behaviour of adults, but finding evidence for this is not the aim of our research.

8. CONCLUSIONS

In this paper, we presented the development of an application that can monitor children’s search behaviour on a large scale. Instead of examining problems children experience during information-seeking through high-quality research, this application gives us the opportunity to examine children’s information-seeking problems through high-quantity research. The application can provide information about children’s search behaviour on a much larger scale, that can be of high value for the research on developing search interfaces and search technologies that support children in effective and enjoyable information-seeking.

9. ACKNOWLEDGMENTS

This research is funded by the European Community’s Seventh Framework Programme FP7/2007-2013 under grant agreement no. 231507, called ‘PuppyIR’.

The research is also part of the research program called ‘The digital youth library’ that is funded by The Netherlands Public Library Association (VOB).

10. REFERENCES


