BrowsingViz: Visualizing Web Browsing Behaviours for HCI Research

J. Karen Parker

University of British Columbia
201-2366 Main Mall
Vancouver BC Canada V6T 1Z4
parker@cs.ubc.ca

ABSTRACT

This paper introduces BrowsingViz: visualization software developed for an HCI researchers' dataset of web browsing logs. Analyzing such a dataset - which combines automatically logged data with user-provided qualitative information – can be a daunting task. Using an iterative, participatory design approach we developed software that uses established InfoVis techniques such as colour and spatial layout to simplify the task of analysis. BrowsingViz provides two layout options for the data (compressed and temporal), which allow the end user to both see patterns and gain an overall understanding of the dataset. Additional utilities such as tool tips provide the user with a rich array of information for each data point. We believe our software is an effective tool for the analysis of this dataset and stands up well to several scenarios of usage. Furthermore, the techniques used in BrowsingViz could also be effectively applied to other similar research data.

CR Categories and Subject Descriptors:
Additional Keywords:

1 INTRODUCTION

In HCI, data analysis can be a complex task. Researchers often use a combination of logging and qualitative methods to record data, resulting in a daunting amount of information which must be sifted through in order to do a thorough analysis.

This data overload is a common problem, as evidenced by a recent workshop at CHI 2005, which aimed to tackle the problems associated with combining data logging and qualitative methods [8]. Kort and de Poot identified several key tasks that are difficult to achieve with existing data analysis solutions: detecting patterns in behaviour, comparing patterns between users, and combining patterns in behaviour with qualitative data [workshop ref].

Hilbert and Redmiles suggest, among other things, "[visualizing] the results of transformations and analyses of event streams so they can be explored with more ease" [4]. So, some form of information visualization may be able to aid HCI researchers in their data analysis problems.

Unfortunately, in trying to create a visualization solution that is good for everybody, we may end up with a solution that suits nobody terribly well. In an attempt to avoid this pitfall we take a somewhat narrow approach to visualizing HCI log data, focusing solely on the visualization of web browsing behaviour.

To this end, we propose BrowsingViz, a software visualization tool to aid in the analysis of web browsing behaviour. Through established InfoVis techniques such as the use of colour and spatial positioning, and features such as tool tips and flexible display settings, BrowsingViz effectively aggregates and displays our target dataset.

This organization of this paper is as follows: First, we present related work in the area of our problem domain, as well several areas of related InfoVis research; Then, we describe the BrowsingViz system, including the dataset, implementation details and InfoVis techniques used; This is followed by our results, which cover scenarios of use, performance, and evaluation; Then, we discuss the lessons we learned through the BrowsingViz development process, and touch on several strengths and weaknesses of our software; Finally, we present our conclusions and plans for future work.

2 RELATED WORK

2.1 Problem domain

Recent research into web browsing has begun to examine user behaviour at a much deeper level than simple navigation. In particular, several researchers at Dalhousie University are conducting examining web browsing behaviour by logging low-level browser events in order to gain information about users' web browsing habits [3,7]. By combining browser events with user-provided data (e.g. the task a user was trying to accomplish when they were at a particular page), these researchers hope to reveal interesting trends and patterns in web browsing behaviour.

BrowsingViz is designed specifically to visualize data from one of these researchers' studies: Hawkey's "Privacy Gradients for Web Browsing." (Figure 1)

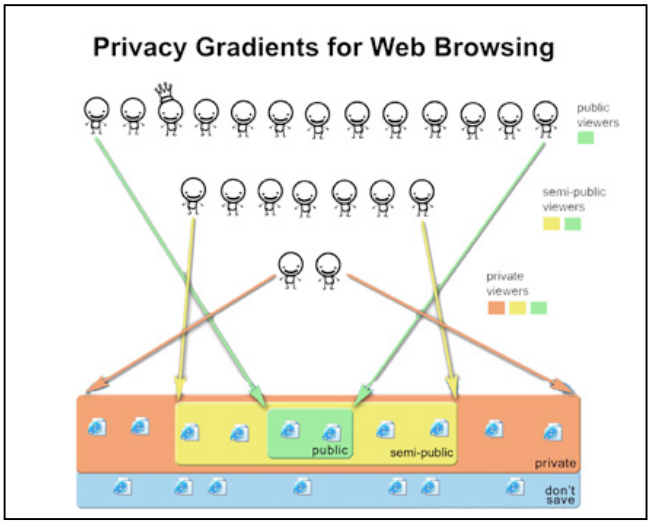

Figure 1. Privacy Gradients for Web Browsing: An overview.

2.2 Web browsing visualization

Most previous InfoVis research into web browsing has been concerned primarily with mapping users' navigation through individual websites. The main aims of this type of visualization

include improving website usability [5], and characterizing how users navigate complex information spaces [1].

Berendt and Brenstein offer STRATDYN, an analysis tool for providing meaningful qualitative and quantitative data about users' web browsing behaviours *on a specific site* [1]. Rather than being concerned with the bigger picture of a user's browsing behaviour, the authors present STRATDYN as a server-side tool for drawing maps of users' navigation through a web site's pages [1]. These maps can then be used to identify and categorize sequences of movements that are important in characterizing web navigation. The authors were able to use the STRATDYN tool to analyze data from a study that looked at the effect of attention on navigation performance [1].

Hong and Landay's WebQuilt [5] also visualizes web browsing behaviours, but for the purposes of usability testing, rather than research data analysis. Again, the focus is on navigation paths through the website, rather than the attributes of the pages being visited. Meant as a usability tool for website designers, WebQuilt is very path-centric, mapping common user paths through a site, a well as optimal oaths [5].

In contrast to this previous work in web browsing visualization, which takes a server- or page-centric view of a user's navigation, we are interested in a user's *whole* browsing experience, *not* their path of navigation over specific web pages or servers.

2.3 Log Data Visualization

Previous work in visualizing complex systems may provide some insight into the HCI log data problem. Bosch et al. state that "because of the complexity of computer systems, the analysis process is a highly unpredictable and iterative one: an initial look at the data often ends up raising more questions than it answers" [2]. The authors' approach to this problem is to create a tool that is as flexible and customizable as possible. The result – Rivet – is able to handle a variety of computer system data [2].

While a highly flexible, customizable interface for visualizing similar-but-different data is a noble goal, it places a lot of the work on the shoulders of the end user, who must spend a great deal of time customizing the tool to their dataset. In this paper we take the opposite approach to visualizing web-browsing behaviour by choosing to support only a very specific dataset. While this limits our software's applicability to other, similar, types of data, it also makes BrowsingViz highly useful for our single target end user.

3 BROWSINGVIZ

3.1 Dataset

Our dataset consists of two groups of individual files from a user study that investigated issues of privacy surrounding web browsing behaviour. Each participant in the study had his or her web browsing behaviour logged for one week. During this time, two data files were generated: browsing log and browser events.

3.1.1 Browsing log file

This file contains a list of every single page the participant visited during the one-week study period. For each page, the following information is present (representing one line of data):

WindowID	unique ID for the browser window
Date	day, month, and year
Time	time of day
Page Title	title of the web page
URL	address of the web page
Privacy Level	user-defined privacy level (4 possible)
Location	user's location (e.g. school/work/home)
Category	user-defined category
Other	secondary category
Secure	whether or not the page was secure (https)

3.1.2 Browser events file

This file contains a list of captured web browser events. These events occur during the same time span as the browsing log file, but are not associated with specific web pages:

WindowID	unique ID for the browser window
Date	day, month, and year
Time	time of day
Event	four types: - window opened - window closed - window lost focus - window gained focus - page loaded - navigation started

3.2 Implementation details

Our software was built in Eclipse 3.1.1 for OS X with Java 1.4.2. We made heavy use of the Swing libraries and also took advantage of two third-party libraries: OsterMiller[9] provided a suite of classes to parse our CSV data files, and Joda-Time[6] offered a very usable alternative to Sun's horrid date and time implementation. With the exception of the aforementioned libraries, we wrote the entire application. We did not use any InfoVis toolkits.

Cross-platform compatibility is a major concern for us, since our target end user runs Windows. Our software has been tested on Eclipse 3.1.1 for Windows XP, and runs well.

3.3 InfoVis techniques

3.3.1 Colour

Ware tells us that colour "is excellent for labelling and categorization" [10]. We take this to heart in BrowsingViz, using colour as one of the primary methods for identifying attributes of our data. We use for two primary purposes: to display privacy levels, and to display page categories.

Privacy levels

Colour blindness is an important consideration in selecting colours for visualization applications [10], however in the case of our software this was not a priority. We had a very specific target end-user who already has a set of colours associated with the dataset. Each page in the dataset has an associated privacy level which maps to one of four colours:

Privacy Level	***Colour***
Public	Green
Semi-Public	Yellow
Private	Red
Don't Save	Blue

Since our end user already had strong associations between the data and these colours, we chose to mirror these mappings in our software (Figure 2).

Figure 2. End user's original data colour mappings in use

Categories

In addition to its privacy level, each page in our dataset is labelled with at least one category. Some pages also have an additional category field called "other." Each data file defines its own categories, so it is impossible for us to define a global category colour scheme. Furthermore, some files define upwards of 20 different categories. Automatically assigning this many colours is difficult not only in terms of coming up with distinguishable hues, but would also require a huge amount of cognitive effort on the part of our user – even with a legend for reference - when she wanted to figure out which colour mapped to a specific category.

Instead, our approach is to put the colour assignment in the hands of the end user. When a data file is loaded, we assign the same default colour (black) to every category. The user can then use the settings panel to modify as many of these category-colour mappings as she wishes to. (Figure 3)

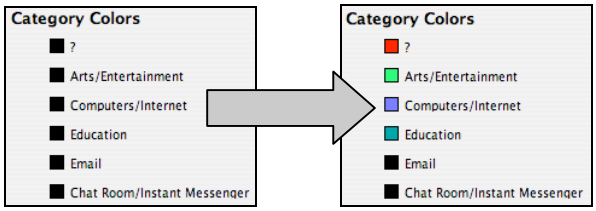

Figure 3. End user controls display colours for categories.

This is useful because the end user can choose colour mappings that are meaningful to her. It also makes it easy to visually "collapse" categories. For example, if our end user wanted to group the Computers/Internet and Email categories in the figure above, she could simply set the same colour for both categories.

3.3.2 Spatial position

At first glance, the temporal nature of our dataset seems to easily lend itself to a time-series visualization. However, we have found that a slightly different type of layout is also quite useful. Our software implements two layouts: temporal and compressed.

Temporal layout

The temporal layout shows a user's browsing behaviour over time (Figure 4). Each browser window is drawn on a separate line of the Y-axis. Each individual page is represented by a rectangle that is proportional in length to the amount of time it was open in a browser window. The location of each page is mapped along the X-axis (time).

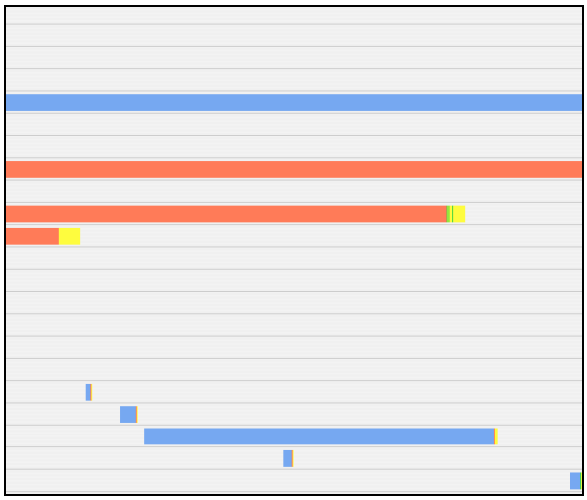

Figure 4. Temporal layout

However, the nature of the web browsing data presents some problems with this arrangement. Each data file spans a week and some pages only span a few seconds. If we represent each second as a single pixel on the X-axis we end up with an extremely large visualization spanning hundreds of screens-lengths. While this provides a realistic picture of the data over time, it is impossible for our end user to see more than a few minutes' worth of browsing behaviour at a time.

Compressed layout

To address the problems presented by our temporal layout, we introduced the compressed layout (Figure 5). Each page in the compressed layout is of equal width, regardless of the length of time it represents. The Y-axis of unique browser windows remains, but pages are no longer mapped to time on the X-axis; instead, they are simply placed one after the other in the order they appeared in the browser window.

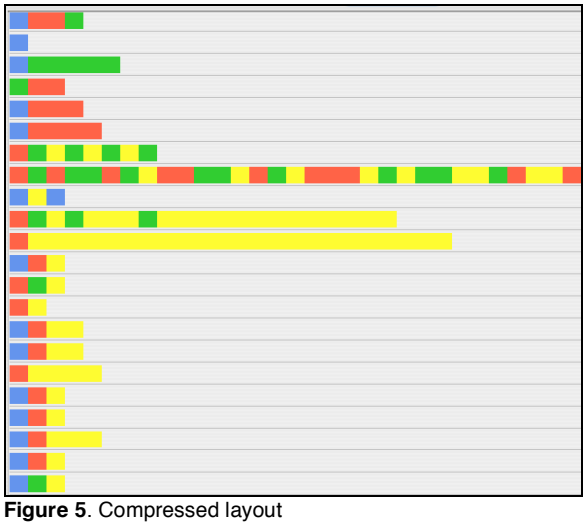

Figure 5. Compressed layout

Unlike the temporal layout, which spaces data far apart and makes some individual pages too narrow to see, the compressed layout gives each page enough screen real estate to easily be seen by the end user. It also lines the browser windows up in a vertical row so that they may be easily compared.

3.3.3 Navigation and zooming

We originally envisioned an interactive zooming feature for our application. Users would be able to drag and select a portion of the time-series, and the display would zoom horizontally, displaying only the selected data. However, limits on our time and programming knowledge resulted in this feature not being implemented in the current version of the BrowsingViz software.

Instead, navigation is currently handled with horizontal and vertical scrollbars. If the visualization is larger than the available display space, the user can scroll vertically to see additional browser windows, and horizontally to see additional pages.

We considered displaying multiple browser windows on one line to help alleviate the need for vertical scrolling, however our end user adamantly prefers individual lines for each window because it is easy to make comparisons between individual windows this way.

We hope to improve on the navigation and zooming capabilities of our software in the future, perhaps even offering a Focus+Context interaction technique.

3.3.4 Additional features

In addition to the InfoVis techniques discussed above, out software has several other useful features. These include: page break markers to delineate transitions between pages; category visualization bars to provide a secondary visual piece of information; tool tips to provide rich textual data for each page; and behind-the-scenes aggregation of previously disconnected data.

Page break markers

If multiple pages with the same privacy level appear beside each other, it is difficult to tell where one page ends and another begins. In order to make it easier for the user to discern page transitions, we provide the ability to turn visual "page breaks" on or off. (Figure 6)

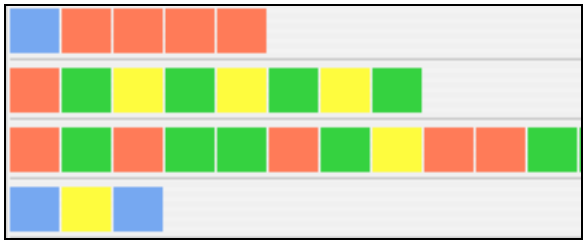

Figure 6. Compressed layout with page break markers

With page breaks turned on, the user can instantly tell where page changes occur in a bar of browser window data.

Category visualization bars

The main aim of our visualization is to display the privacy level of each page, but the user may sometimes want to see additional visual information such as a page's category. Our software allows the user to turn two additional display bars on and off: one for the primary category, and one for the "other" category. With these bars turned on, and colours assigned for specific categories, the user can explore relationships between a page's privacy level and some of its other attributes.

Tool tips

The large amount of data available for each individual page is impossible to display persistently as part of the visualization. In order to give the user access to all of this data we have implemented tool tips. Whenever a user's cursor enters a page on the display, a tool tip will appear. A standard UI tool tip disappears after a short amount of time, but our tool tips persist until the user's cursor exits the bounds of a particular page. These tool tips allow us to provide the user with a rich array of data for each page without overwhelming the display with information (Figure 7).

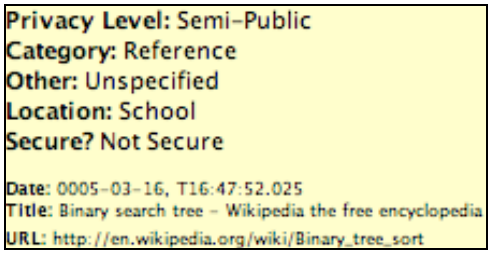

Figure 7. Sample tool tip

Data aggregation

As described in section 3.1, our dataset contains two sets of files: browsing log files, and events files. The events file contains information that could be very relevant to the browsing log file. Prior to the development of BrowsingViz our end user had no useful way of combining these two files. Our software aggregates the information from both files, providing additional context to the data.

The browsing log file specifies when each page was opened, but not when it was exited or closed. If we sort the list of pages in each browser window chronologically we can deduce that the exit time for each page is the same as the open time for the next (chronological) page. However, this method fails when we reach the last page in the window. We have no idea when this page was closed. Luckily, out events file contains close events for each browser window. This gives us the exit time for our final page.

The focus information from the events file could be used to provide heightened context for our data. Though not currently implemented in the software, we could imagine providing visual cues on the timeline showing which window currently has focus.

Thus, aggregating the browsing log and events file provides us with more information about page exit times, and could provide other even more useful contextual information if further event-centred features were implemented.

4 Results

4.1 Scenarios of use

Although our dataset is very specific, our intended end user has several tasks she'd like to accomplish with our software. We outline three examples of this below.

4.1.1 Scenario 1: Looking for privacy level patterns in browser windows

Before she had our visualization tool, one of the patterns our end user *was* able to glean from her dataset was that users often browse long runs of one privacy level in the same browser window, and use a different window for browsing pages of another privacy level. She has just collected data from a new participant and wants to quickly see if this is the case for their data as well. She opens the data file in our visualization tool, and adjusts the settings so that only the privacy bar is visible, and the layout is compressed rather than temporal.

She then switches to the visualization panel and scrolls vertically through the visualization until she spots a run of pages that is almost completely green, except for one anomalous red block in the middle. She moves her cursor over the mouse to get more information about the anomalous page (Figure 8).

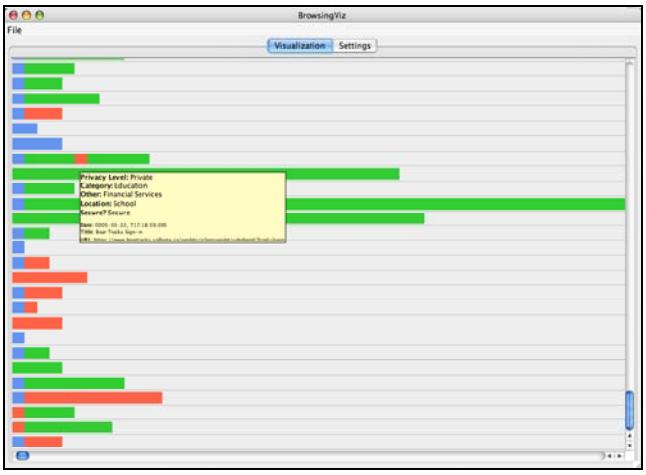

Figure 8. Looking for privacy level patterns in web browser windows. A larger version of this figure is available on page 8.

4.1.2 Scenario 2: Examining the interaction between a page's privacy level and its category

Because each participant's log file has a different set of categories, examining anything related to these categories can be a tedious job. Using our software, the end user wants to be able to examine how different categories match up with privacy levels.

She opens the data file in BrowsingViz, makes sure the category visualization bar is turned on and then examines the list of categories on the settings panel. The categories start out uniformly colour-coded to black.

She wants to see whether the participant tended to label Health-related sites as private, so she sets the Health category colour to teal. She also decides that e-mail and chat/instant messaging are similar and she wants to collapse them into one category, so she colours them both purple (Figure 9).

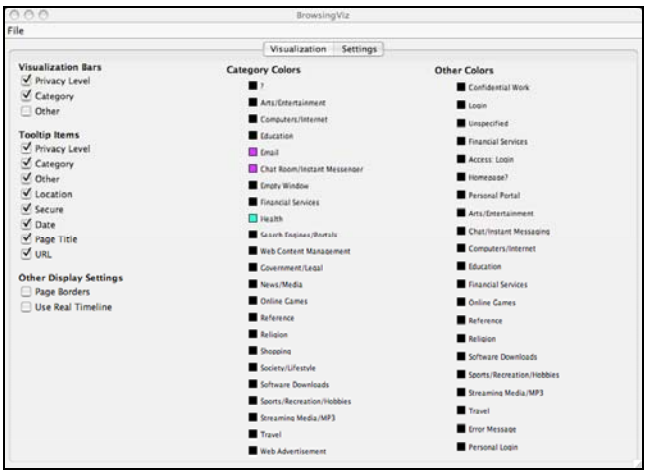

Figure 9. Adjusting category bar colours on the settings tab. A larger version of this figure is available on page 8.

She switches to the visualization panel and scrolls through the information, looking for blocks of purple or teal in the category bars. This allows her to make interesting observations, such as the fact that this user generally marks all email and chat browsing as private. (Figure 10)

Figure 10. Looking for relationships between privacy level and category using coloured bar visualizations. A larger version of this figure is available on page 9.

4.1.3 Scenario 3: Exploring a participant's web browsing behaviour over time

While the compressed layout has been useful for a variety of data examination tasks, our user now wants to form a general impression of when a particular participant does their private browsing. For example, do they mostly browse private pages late at night?

She opens the data file in BrowsingViz and adjusts the settings so that only the privacy bar is visible, and the layout is temporal.

She switches to the visualization panel and scrolls through the information, looking for large blocks of red (private) pages. When she finds one, she moves her mouse over it to bring up a tool tip that shows her the date and time for that data, among other information. (Figure 11)

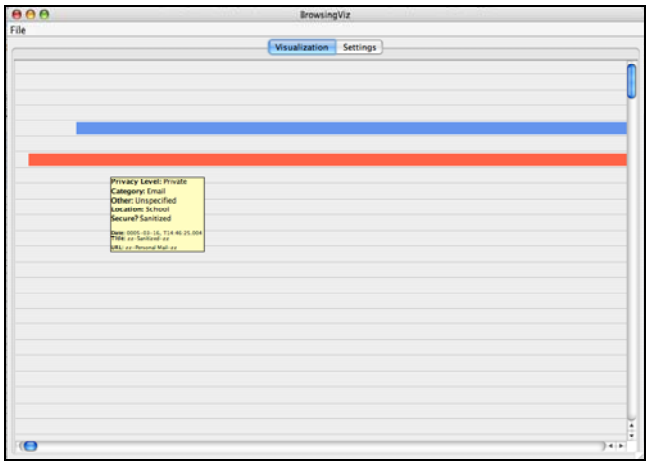

Figure 11. Exploring a participant's web browsing behaviour over time. A larger version of this figure is available on page 9.

4.2 Performance

One of the obvious bottlenecks for any InfoVis application is the reading and parsing of data. At a couple-thousand lines of comma-separated text per file, the data files our software uses are relatively small. Since our application only deals with one file at a time, it is easily able to read in and parse the data with very little delay.

Another possible cause of lag in InfoVis applications is rendering. Again, since our dataset is relatively small (compared to those which use hundreds of thousands of data points) this is not a major problem for our software. While some parts of the BrowsingViz interface are currently a bit sluggish to redraw, we believe this to be a symptom of the preliminary nature of our application code. We are confident that this will be remedied once we have had some time to optimize the UI code.

4.3 Evaluation

While we did not perform any formal user evaluations on our software, we followed a continuous cycle of iterative, participatory design with our end user, who was heavily involved in design decisions throughout the development process.

In the early stages she provided us with information about tasks she wanted to accomplish with the data. She also joined us in a low-tech prototyping session in which we used whiteboard sketches to generate ideas for the BrowsingViz interface.

As development progressed, we polled her regarding the interface design and the importance of certain features. Her feedback was invaluable, and we expect to continue to iterate with her in the future as we continue to develop the project.

5 Discussion

5.1 Lessons learned

The development process of BrowsingViz was the first time we had ever built such an interface- and interaction-heavy piece of software, and it was a learning experience. Two lessons, in particular, stand out:

5.1.1 Simple visualizations are not always simple to implement.

If something looks simple and intuitive, someone probably spent a lot of time making it look that way! Since we decided not to build on any existing InfoVis toolkits, a great deal of work went into creating the visualizations and interactions for BrowsingViz. We were not able to complete all the features we had originally planned, and had particular trouble with the temporal layout, as well as the navigation and zooming functionality. In the future, we wouldn't be so quick to rule out toolkits.

5.1.2 Users don't always know what they want

This is not meant as an insult to our end user. Rather, it is a caution that you cannot expect to ask a user what they want in an application, go away and build it, and come back and present them with a finished product. The iterative, participatory development process was key to the success of our design because we were able to alter our plans based on frequent feedback from our end user. Often, we tried several different approaches to a particular issue before hitting upon one that resonated. Without the iterative approach, we may never have reached such resonations.

5.2 Strengths

While there are plenty of improvements that could be made to our software, we believe that the current version of BrowsingViz does several things quite well.

5.2.1 Use of existing dataset attributes

The software takes existing attributes of the dataset and uses them effectively in visualizations. In particular, the colour codes associated with the privacy levels in the dataset are put to good use, as is the chronological nature of the data.

The privacy colour codes defined by our user translate well to the bar visualization format. Our user is able to glean the privacy level of pages in our visualization instantly primarily because this set of colours is so deeply ingrained in her mind.

Additionally, the chronological information implicit in our dataset makes for an easy-to-understand spatial layout. Even in the non-temporal, compressed BrowsingViz layout we use information about the chronological order of pages to determine order on the screen. This juxtaposes pages that were visited sequentially and allows our use to spot temporal patterns.

5.2.2 Recognizing patterns and overviews

As hinted at in the previous section, the visualizations we have created allow the user to accomplish two important tasks: spotting patterns in the data, and forming an overall impression of the data.

The use of coloured bars allows the user to see, at a glance, patterns and anomalies in the dataset. In the temporal view, these bars are mapped over time, allowing the user to look for patterns not only in colour-coded information such as privacy level and category, but in temporal attributes such as time of day and length of page visits.

The two layouts (compressed and temporal) also offer unique overviews of the data. The temporal layout allows a user to form an impression of a users' browsing behaviour over time, but requires a lot of scrolling on the part of the user. The compressed layout minimizes the need to scroll, and while it loses much temporal information, it shows the ordering of pages in each browser window, thus preserving some basic information about chronology.

5.2.3 Data aggregation

Finally, we have successfully aggregated two related – but previously separate - data sets: page visit data and browser event data. Prior to our work on this, the user was not able to effectively use the data from the event files. Now, our visualization uses the event data to refine details about page length. And in the future, we expect to more fully take advantage of the event data by displaying important events on the timeline.

5.3 Weaknesses

BrowsingViz is a work in progress, and there are several major improvements that must be made before it can reach its full potential. In particular, we need to work on the navigation and zooming functionality, the juxtaposition of the settings and the visualization, and the applicability of our software to other datasets.

5.3.1 Navigation and zooming

A week's worth of web browsing behaviour is a whole lot of web pages, and when representing that data on a timeline scaling is an issue. Some pages are visited for mere seconds, so trying to make every page on a weeklong timeline visible is problematic. BrowsingViz handles this poorly. It generates extremely large

timeline visualizations and requires the user to scroll through them in order to see all of the data. Additionally, if there are long time gaps between browsing sessions, the user may find herself staring at an empty screen at some point, with little contextual information to fall back on for help. In future versions of the software we hope to tackle this problem with a focus+context approach, allowing the user to see a week's worth of browsing history on the screen while zooming in on specific section of the timeline to reveal further details.

5.3.2 Juxtaposition of visualization and settings

In an attempt to maximize the screen real estate available for our visualization, we ended up putting all of the settings on a separate tab. While this leaves us lots of space for displaying the visualization, it also means that the user must switch to the settings tab every time she wants to modify the display, and then switch back to the visualization tab after making the desired modifications. This lack of tight visual coupling between settings and visualization makes it difficult for the user to see the effect her settings changes have on the visualization. One possible solution to this problem would be a pop-up settings window. This might occlude part of the visualization, but only temporarily.

5.3.3 Dataset limitations

Currently, BrowsingViz is designed to work with the very specific dataset made available to us by our end user. However, there are other HCI researchers doing similar work in the area of web browsing, with similar datasets to analyze. We believe that these researchers could also benefit from the types of visualizations our software provides. While the software could be easily hard coded with the characteristics of some other dataset, overall flexibility of use is a far superior goal. In the future we hope to be able to extend our software to give users more control over the types of data that can dealt with, and the ways in which it can be displayed.

6 Conclusions and Future Work

In this paper we have presented BrowsingViz, a visualization tool built specifically to aid a researcher's analysis of logged web browsing behaviour. The preliminary version of our software accomplishes our goals of allowing the user to find patterns, and to gain a better overall understanding of the data.

The current version of BrowsingViz is fairly preliminary, and there are several additional and improvements that would make it an ever better tool. We plan to make several modifications to the temporal layout, including adding a focus+context zooming technique to improve the visibility of data, and integrating focus events from the event data file onto the display in order to provide our user with a fuller picture of participants' browsing behaviours.

Going beyond out current limited dataset, we hope to extend our software to work with other, similar datasets. In particular, we plan to support the work of another researcher at Dalhousie who is studying the implications of task on web browsing behaviour [7]. Our eventual goal with BrowsingViz is to make it flexible enough to be used by anyone who is studying logs of web browsing behaviour.

7 Acknowledgements

We would like to thank Kirstie Hawkey for providing us with the dataset for this project, and for her insightful comments and suggestions during our iterative design process.

References

[1] Berendt, B. & Brenstein, E. (2001). Visualizing Individual Differences in Web Navigation: STRATDYN, a Tool for Analyzing Navigation Patterns. In *Behavior Research Methods, Instruments, & Computers*, 33, pp. 243-257.

[2] Bosch, C.S. et al. (2000). "Rivet: A Flexible Environment for Computer Systems Visualization." In *Computer Graphics*, February 2000, pp. 68-73.

[3] Hawkey, K. and Inkpen, K.M. (2005). Privacy gradients: exploring ways to manage incidental information during co-located collaboration. In *Extended Abstracts of the Conference on Human Factors in Computing Systems*, CHI 2005. Portland, OR, USA. pp. 1431-1434.

[4] Hilbert, D. and Redmiles, D.F. (2000). "Extracting Usability Information from User Interface Events," In *ACM Computing Surveys*, Dec. 2000, pp. 384-421.

[5] Hong, J.I. and Landay, J.A. (2001). "WebQuilt: A Framework for Capturing and Visualizing the Web Experience." In *Proceedings of The Tenth International World Wide Web Conference (WWW10)*, Hong Kong, May 2001, pp. 717-724.

[6] Joda. (2005). *Joda-Time*, Version 1.1, August. 2005.

[7] Kellar, M. & Watters, C. (2005). Studying User Behaviour on the Web: Methods and Challenges. In *CHI 2005 Workshop on Usage Analysis: Combining Logging and Qualitative Methods*, Portland, OR.

[8] Kort, J. and de Poot, H. (2005). "Usage Analysis: Combining Logging and Qualitative Methods", in *CHI 2005 Workshops*, April 2-7, 2005, Portland, Oregon, USA.

[9] Ostermiller, S. (2005). *Ostermiller Utils*, Version 1.05.00.

[10] Ware, C. (2000). *Information Visualization: Perception for Design*. San Francisco: Morgan Kaufmann Publishers.

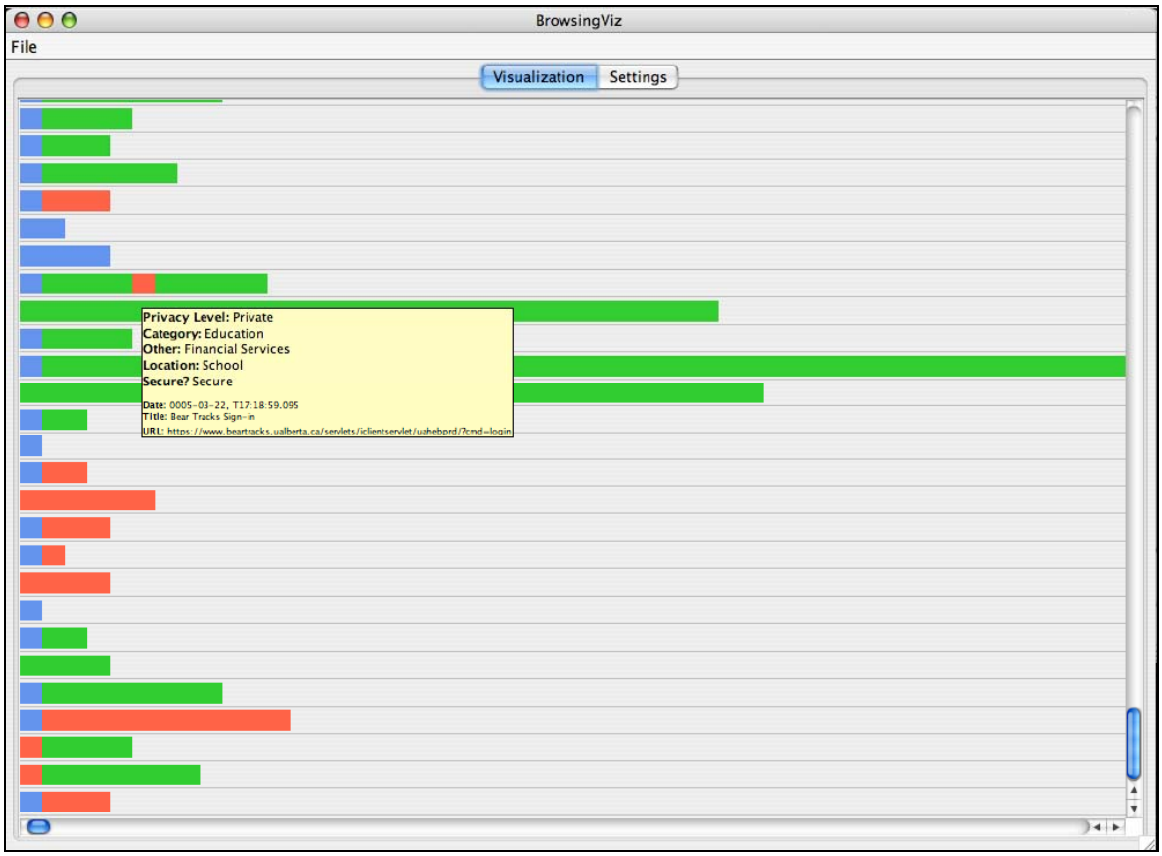

Figure 8. Looking for privacy level patterns in web browser windows.

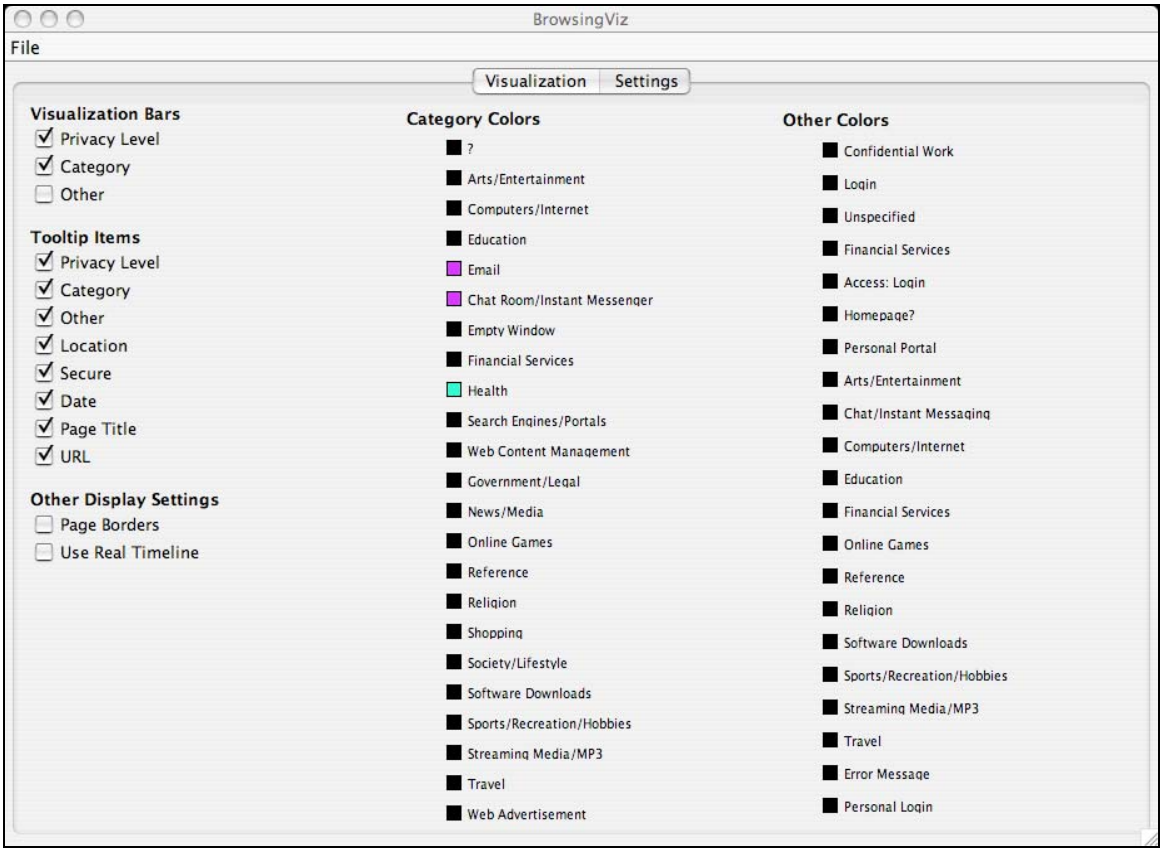

Figure 9. Adjusting category bar colours on the settings tab

Figure 10. Looking for relationships between privacy level and category using coloured bar visualizations.

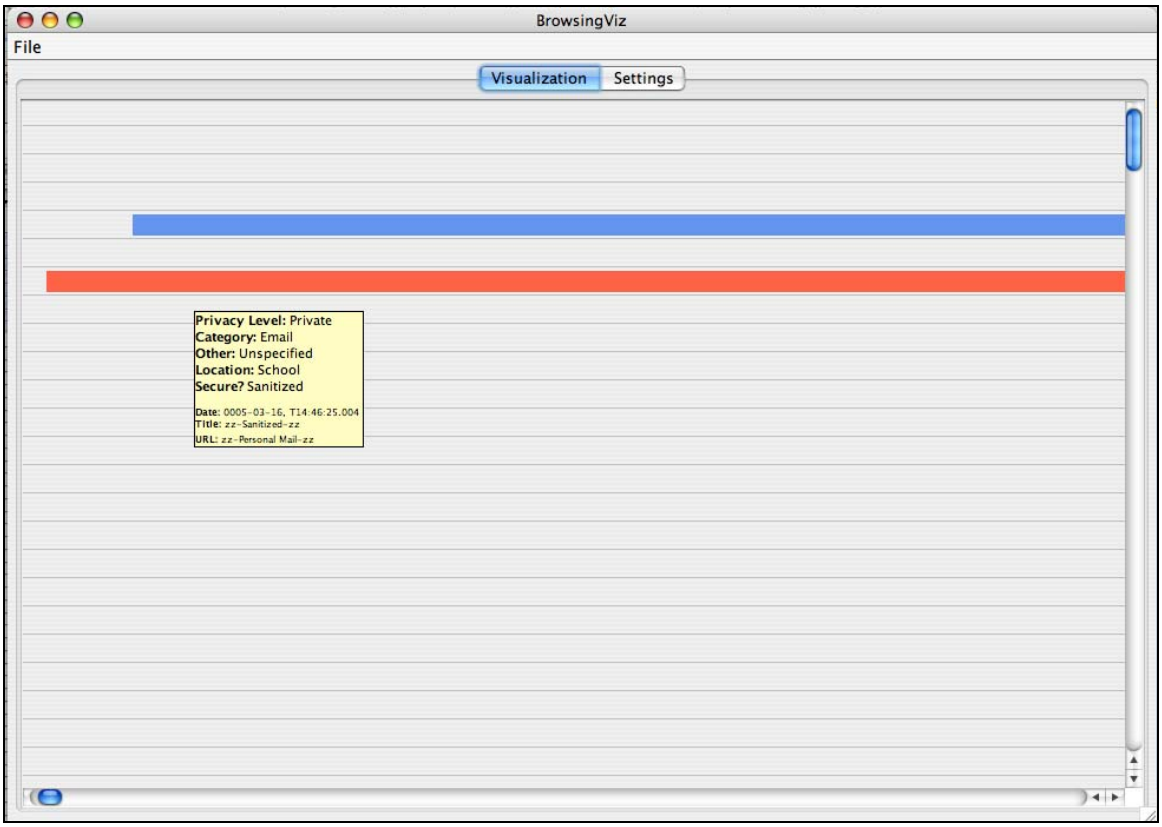

Figure 11. Exploring a participant's web browsing behaviour over time