What Did They Do? Understanding Clickstreams with the WebQuilt Visualization System.

Sarah Waterson, Jason I. Hong, Tim Sohn, Jeffrey Heer, Tara Matthews and James A. Landay
Group for User Interface Research, Computer Science Division
University of California, Berkeley
Berkeley, CA 94720-1776 USA
+1 510 642-3437
{waterson, jasonh, tsohn, landay} @cs.berkeley.edu
jheer@parc.xerox.com
matthewt@seattleu.edu

ABSTRACT

This paper describes the visual analysis tool WebQuilt, a web usability logging and visualization system that helps web design teams record and analyze usability tests. The logging portion of WebQuilt unobtrusively gathers clickstream data as users complete specified tasks. This data is then aggregated and presented as an interactive graph, where nodes of the graph are images of the web pages visited, with the arrows representing the transitions between pages. To aid analysis of the gathered usability test data, the WebQuilt visualization provides a semantic zooming feature, allowing the designer to understand the test results at the gestalt view of the entire graph, and then to drill down to sub-paths and single pages. The visualization highlights important usability issues, such as pages where users spent a lot of time, pages where users get off track during the task, navigation patterns, and exit pages, all within the context of a specific task. The WebQuilt visualization can aid remote usability testing on a variety of internet-enabled devices by providing a way to identify potential usability problems when the tester cannot be present to observe and record user actions. The visualization can also be a tool to aid traditional testing, as it can provide aggregate views of the research participants' interactions.

Keywords

Usability Evaluation, Log File Analysis, Web Visualization, Semantic Zooming, Remote Usability Evaluation

1. INTRODUCTION

There is good evidence that web site usability remains a serious problem today. Studies have shown many web users find it difficult to locate products they want to purchase and often give up [1, 2]. The usability of a web site is one key to its success, particularly for increasing sales and repeat business on e-commerce sites [3]. However, gathering and analyzing the appropriate usability data is a complicated matter. Traditionally usability testing and data gathering is done by bringing users into usability labs and observing their actions as they perform certain tasks. This provides important qualitative data, but is expensive, time consuming and tends to reflect only a small number of users. Analysis of server logs is frequently used to quantitatively understand what large numbers of users are doing on a site, but it is difficult to comprehend the actions [4, 5], goals, and motivations of individual users.

In the past few years, a number of online survey and logging tools [6, 7] have been developed in an attempt to quickly gather more data than traditional lab style testing that is of higher quality than the data provided via server log analysis. These tools, referred to here as remote usability testing tools, recruit users to perform specific tasks on web sites from wherever the user happens to use the Internet. Qualitative data and demographics are gathered using online questionnaires, and the research participants' clickstreams are logged. A clickstream is record the path the user took through the web site while performing the task, along with the times he or she spent on pages, and any data they entered and received.

While remote usability testing can provide access to more users, more qualitative information, and goal-oriented clickstreams, they are often problematic to set up, maintain, and limited in the types of testing they can record. Tools that gather data on the client-side require users to download special software, often browser and operating system specific, or require the user to modify system settings that they may be unwilling or uncomfortable doing. Logging done from the the server avoids the specialized software problems of client-side logging, but limits the testing to the sites on the server. Competitive analysis is not possible, and standard server logs do not provide the necessary data for gathering usability information, so logging of usability data is directly tied to serving content, potentially causing problems when modifications are necessary. Beyond the practical matters of deploying and running the tests, the problem of aggregating, interpreting and analyzing the gathered data remains. The current tools need to advance beyond obtrusive data gathering, and provide more sophisticated analysis tools in order to answer usability questions deeper than which pages received the most hits, or how may shopping carts are abandoned. While these factors are important to designers in understanding the usability of their site, they must be able to be studied in the larger context of the task and the users intentions.

In this paper we discuss our research into the visualization of remote web usability log data generated by WebQuilt, a proxy-based usability logging system [8] that can unobtrusively gather tasked-based clickstreams from a variety of internet-enabled devices such as PDAs, WAP phones, and desktop computers. Designed to support existing online survey and participant recruitment tools, the WebQuilt logging system can also act as its own stand alone remote usability testing and analysis system. A web designer would first set up several tasks. Next, he or she would recruit participants to perform these tasks, sending a starting URL via email that directs the participants browser to the intended site through the WebQuilt proxy logger. As the tasks are performed, the proxy automatically collects the data. Demographic, pre- and post-task questionnaires, as well as instructions or task descriptions can be incorporated into the proxy web pages the user is required to pass through. The rest of this paper will discuss related visualization tools, the WebQuilt visualization of this collected data, the visualization implementation, and our future work.

2. RELATED WORK

Usability log visualization has a long history. Guzdial and colleagues review and introduce several desktop-based, non-web systems in [9]. A variety of web and Internet related visualization systems have been developed, including displaying an individual user's history in a web browser [10], web site mapping [11], web connectivity [12], information scent [13], and web traffic [14]. The WebQuilt system differs from these visualization systems in that it is targeted specifically at understanding task-based usability problems.

A few commercial and research systems for web usability logging and visualization have been developed and share usability goals similar to WebQuilt. The National Institute of Standards and Technology has developed VISVIP [15],an extension to their usability logging tool WebVIP [16], which shows individual paths overlaid on top of a 3-D visualization of the web site. While compatible with many operating systems and browsers, the WebVIP logging system requires entire sites to be downloaded for analysis, and not just the relevant pages, a problem WebQuilt avoids by using the proxy to record only the visited pages. Visual Insight's eBizinsights [17] is a sophisticated server log visualization tool for generating custom reports and interactive charts. Designed mostly for management and marketing reports rather than designers, the eBizinsights software also relies on server-side data gathering. Vividence ClickStreams [18] displays individual and aggregate user paths through a web site as a hyperbolic tree, using path color and weight to indicate task success and traffic. Blue Martini's ClickViz [19] provides a graph of icons representing web pages and the flow of traffic, with additional controls to modify the display via filtering and sorting. While these visualization systems all analyze web usability logs, they all vary in the types of data they can gather and present, as well as in the levels of control and interaction with the data. The visual analysis tool of the WebQuilt usability log data expands on the successes of some of these systems, incorporating some visual elements similar to ClickStreams and ClickViz such as path traces representing traffic flow, and offers new, powerful methods for display and interaction.

3. THE VISUALIZATION

The WebQuilt visualization aggregates the data from multiple participants into an interactive graph, an example of which is shown in Figure 1. Visited web pages are nodes in the graph and are represented by color blocks in this view. Arrows are used to indicate traversed links and where participants hit the back button. Thicker arrows indicate more heavily traversed paths. Arrow color is used to indicate the average amount of time spent before traversing a link, with colors closer to yellow meaning short amounts of time and colors closer to red meaning longer amounts of time. As an option, a designer designated "optimal" path can be indicated and is rendered as a thick blue highlighting along the path (e.g. the top path in Figure 1). The web page nodes also use color to differentiate entry and exit points. The nodes appearing in green are the entry nodes, and typically there is only one entry node as usability tests start all participants at the same point. Cyan nodes are web pages where participants exited the task, and the dark gray nodes are neither entry nor exit points.

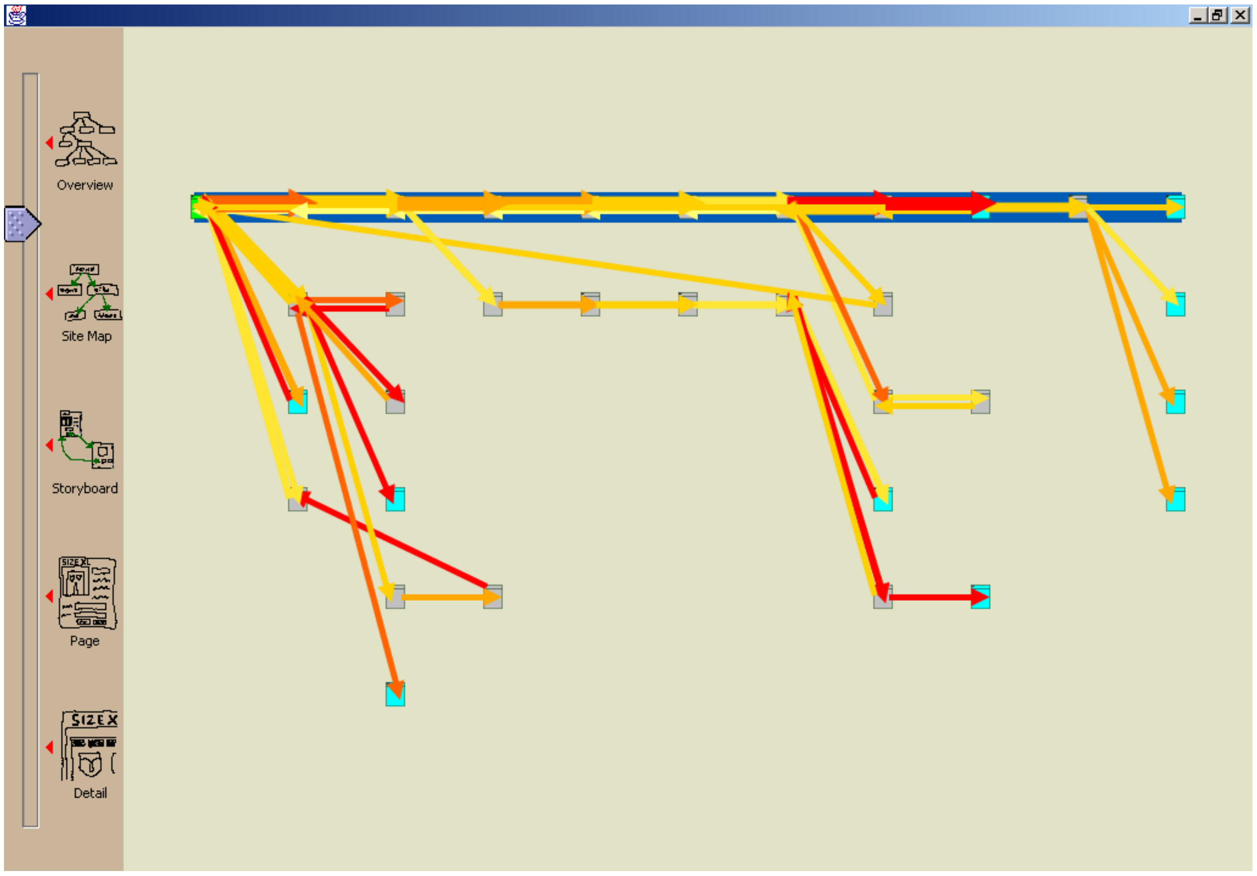

Figure 1: The WebQuilt visualization. Nodes are visited web pages, and arrows are the traffic between the pages, with thicker arrows representing heavier traffic. Color is used to indicate time spent on a page before transitioning. The closer the arrow to red, the longer spent in transition. The designer's path is highlighted in blue. Entry pages are green, and exit pages cyan. The zoom slider interface along the left hand side is used to change the zoom level.

On the left-hand side of the interface is a slider used for zooming in and out of the graph to show the web pages and paths at varying levels of detail. Figure 2 shows a zoomed in view of the left-hand side of the graph in Figure 1, including the green start page (the node furthest to the left in the figure). This level of detail, the storyboard level, shows the interaction between a handful of pages. Instead of color blocks, the nodes are now thumbnails of the actual web page viewed, and the details of the path transitions are more distinguishable. Zooming in further to the start page in Figure 3, the entire page layout is clearly visible, as is the URL for the page. Additionally, the arrows out of the page have been connected to the links that users actually clicked. Both arrows in and out of the page become more translucent at this level, so as not to cover design details in the page below. For any selected links that required scrolling to reach, the resulting out-bound arrows originate from a location on the edge of the image.

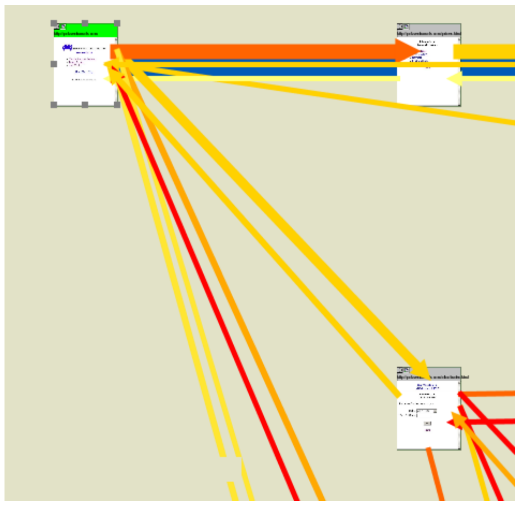

Figure 2: A zoomed in view of the graph in Figure 1. Web pages are now thumbnails and path transitions are more distinguishable.

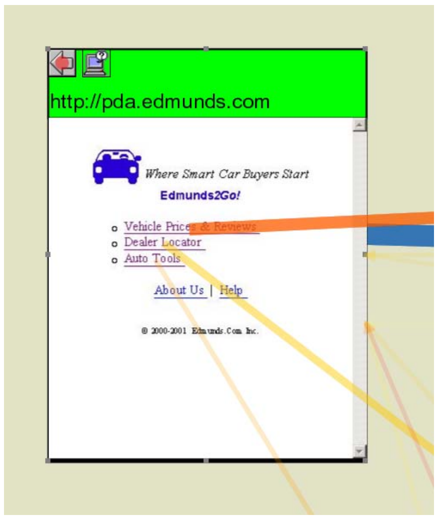

Figure 3: The graph in Figure 1 zoomed to the page view. Page design is visible, arrows originate from clicked links, and the URL is along the top. Icons on the top are also used to indicate browser back button actions and actions the system cannot associate with locations.

Along the top of the page view are two icons. The left pointing arrow icon represents a browser back action. If any user clicked the back button during the task while on the page, a red dot is placed on this arrow button to indicate that action, as shown in the example in Figure 3. The other icon, a computer with a question mark, will have a dot if the user clicked on a link with a location currently undeterminable by the WebQuilt logging system, for example the locations of form submission buttons and fields are not yet recorded in the system.

Navigation of the graph is currently done by selecting nodes, zooming in and out, and performing navigational gestures on the visualization itself to shift different portions of the graph into the screen display. Gestures, a common interaction technique for pen-based interfaces, allow the user to draw a stroke, such as line from left to right, across the visualization. This left-to-right gesture is interpreted by the system as an action to move the graph to the right. Gestures need not be performed with a pen input device. Holding down the right mouse button and drawing strokes performs the same action. Gestures can also be used for selection of graph elements. The navigation rationale and implementation is further discussed in the System Implementation section.

4. VISUAL ANALYSIS

The screenshots in Figures 1-3 show an example visualization of ten internet-enabled PDA usage traces, where users were asked to use the Edmunds.com PDA web site (http://pda.edmunds.com) to find a specific piece of safety information on the latest Nissan Sentra model, as well as the address of the closest local Nissan dealer. The pages along the blue highlighted path at the top represent the optimal, designer-defined path. By looking at the thickness of the lines, one can see that many people took the optimal path, but a number of people took longer paths, or paths much different from the one the designer intended them to follow. Tracing some of these longer paths, one can also see where users come to a page, and decide to backtrack, either via the back button or a link.

How and why do the user traces differ from the designer's path? And why are these deviations interesting? The semantic zooming in the WebQuilt visualization is a key feature for analysis, providing various levels of detail and supporting both the overview of the test and the ability to drill down for specifics. Looking closer at the deviations on the left side of the visualization, indicated in Figure 4(a), we can see that some users chose to find the Nissan Dealer, then return to the start page to find the safety information. Designers can use behavior information such as this recognize places where better navigation may shorten this path, for example by providing a link to the Nissan models from the Nissan dealers page would remove a few unnecessary transitions. Two of the ten users following the path in figure 4(b) looked for the safety information, but not on the latest (the 2002) Nissan model. Rather they investigated the 2001 model. Designers see the behavior of users that may seem to think they are finding the correct information, when in actuality they are not. Perhaps the designer will want to consider making the 2002 link more prominent, or provide some cues to the user about other information, such as links, on the 2001 pages indicating newer models exist to prevent further confusion. Most of the cyan exit points show locations of Nissan dealers relative to the zip code entered by the user, or the safety pages including the information sought. Only one of the ten participants left the task on page not related to the task. This could indicates a positive statement about the usability – that most users, even if they incorrectly finished the task (using the 2001 model instead of the 2002) were able to successfully navigate the site.

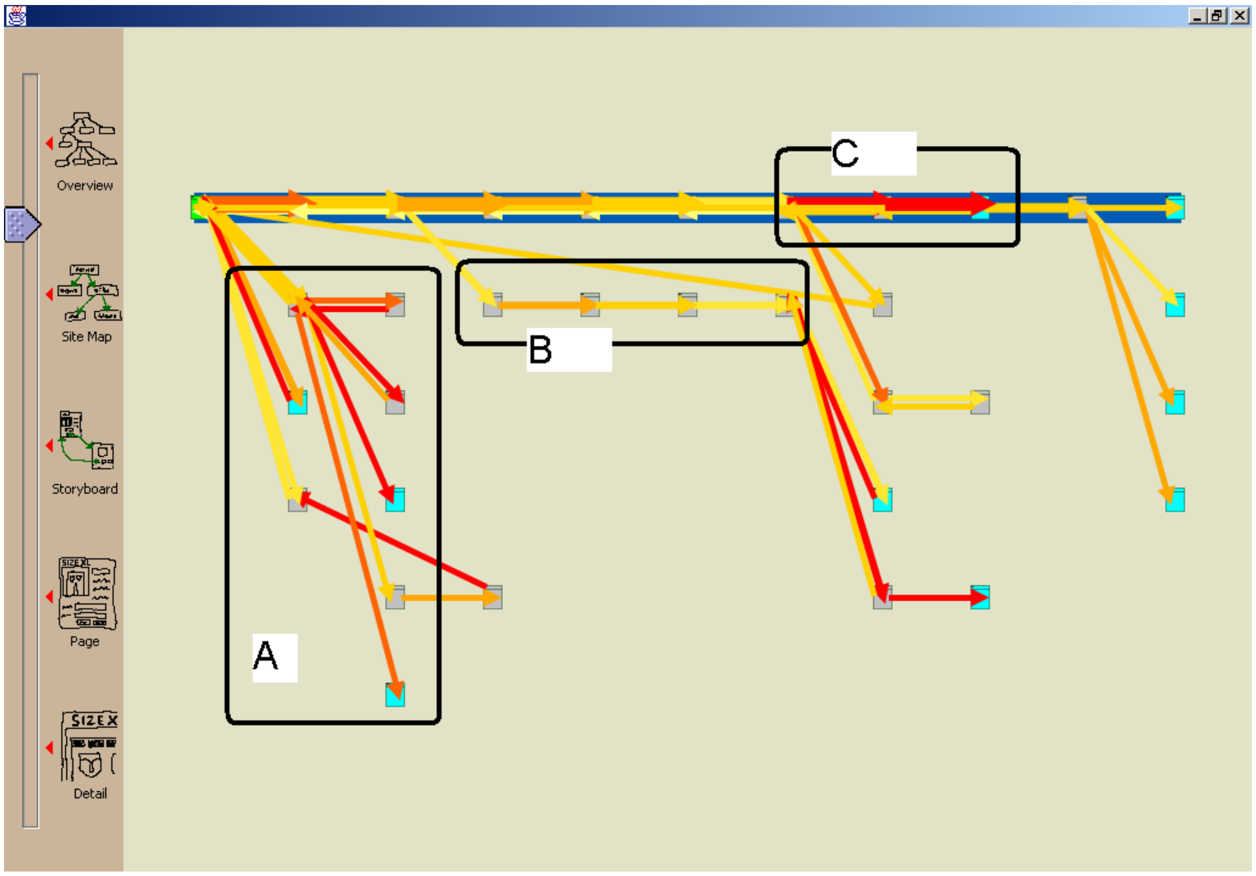

Figure 4: (a) Some users chose to find the dealer information before the safety information. (b) Most users investigated the most recent Sentra model, the 2002, as the task asked. One user, however, investigated the 2001 model. (c) Users spent lots of time on these text filled pages looking the next link.

In this example there are also several red arrows, which indicate that people took a long time, in this case, over a minute, before going to the next page. For example, from the overview of the entire task, a viewer can see thick red arrows in Figure 4(c). Zooming in, we can see that both pages are text-heavy, with long descriptions and lists. It took most of the users considerable time to read through the text and decide where to go next, though they all eventually found it. Because these two pages are in the designers path for this task, the designer may want to reconsider how the information is organized on these pages, so as to be more prominent if this is a common or important task.

Providing the context of the task and a framework to add more details when needed, this visualization offers a number of simple, but very useful, quick and powerful analysis of the user experience. Figure 5 shows the results of a simple information finding task trace of six users. Notice how most users found the information in two clicks (the top paths). One user, however, displayed "ping-ponging" behavior, quickly examining every available link on one page before finding the appropriate page. Our visualization allows interesting user behaviors and actions to be identified and analyzed directly with the design of the web pages Designers and usability experts can interpret their data, and provide diagnosis. In this situation, perhaps more descriptive links rather than brief names and cryptic numbers could avoid user frustration.

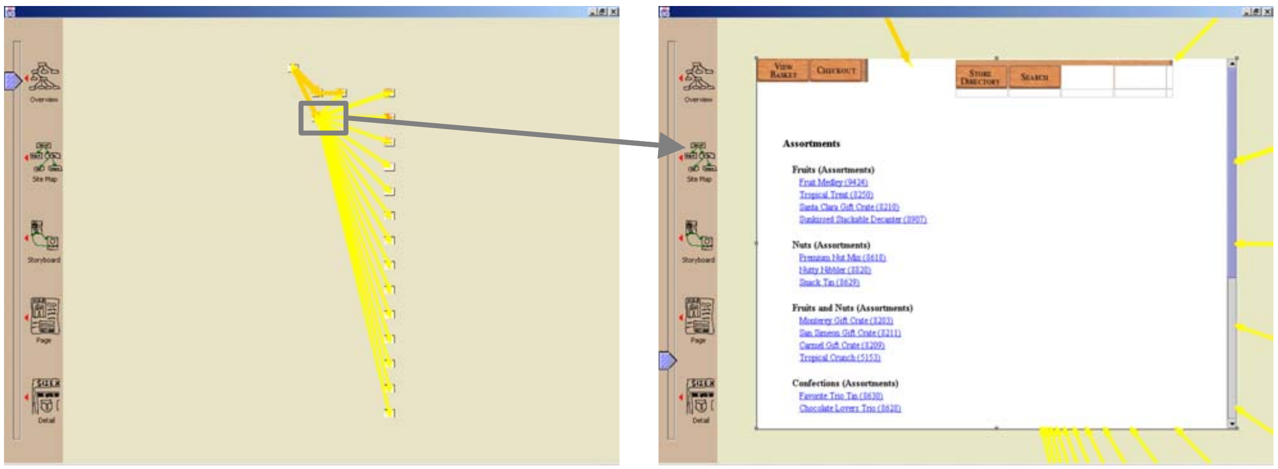

Figure 5: In this example, six users were given the task of finding a piece of information about the Casa de Fruita company on the company web site. The thicker orange arrows at the top show that most of the users found this information in two clicks. One user, however, quickly "ping-ponged" between sitemap and many other pages before finding the information.

5. SYSTEM IMPLEMENTATION

In the previous section, we demonstrated how potential usability problems issues can quickly be found and interpreted by the designers. Creating this visualization, from the data gathering, to the aggregation and user action inference, to displaying the interactive graph, is the topic of this section.

5.1 Gathering Data

At the heart of any data analysis system is the data itself. Gathering quality remote usability data is difficult, as the data is highly dependent upon the logging method used. Remote web usability logging can be broadly defined in three categories: client-side logging, server-side logging, and proxy-based logging. Client-side logging can capture a rich set of interaction data, such as scrolling and mouse movement, but is limited to specific platforms and requires users to download/install specialized software. Server-side logging requires no adjustments or installations for the end user, but often requires adding special tools to the server, limits testing to sites owned or controlled by the tester, and cannot capture richer user interactions. The WebQuilt logger is a proxy-based logger build using Java Servlet technology. As an intermediary to the client and server, it has the advantage of being able to gather information unobtrusively, without requiring the client or server to have any special software configurations. It can be used to test any web site, and is compatible with a range of operating systems, browsers, and devices. However, it is difficult for a proxy to capture rich user interactions.

The logging process works as follows. The WebQuilt proxy intercepts each user page requests and relays the request to the content server. When the requested content returns to the proxy, it is modified to redirect the links through the proxy, a copy of the content is cached on the proxy machine, and the action is logged in a file format similar to a server log, but specialized to include information necessary to recreate the experience for usability analysis. This includes keeping track of the links clicked on each page according to the Document Object Model (DOM), adding transaction ordering identification to pages to recreate the clickstream, and tracking the use of frames. For more information on the implementation of the WebQuilt proxy logger see the descriptions in Hong and Landay 2001. Additional proxy functionality includes rendering a JPEG image of the web content being sent to the participant, and the saving of the DOM for reference. Figure 6 shows this process.

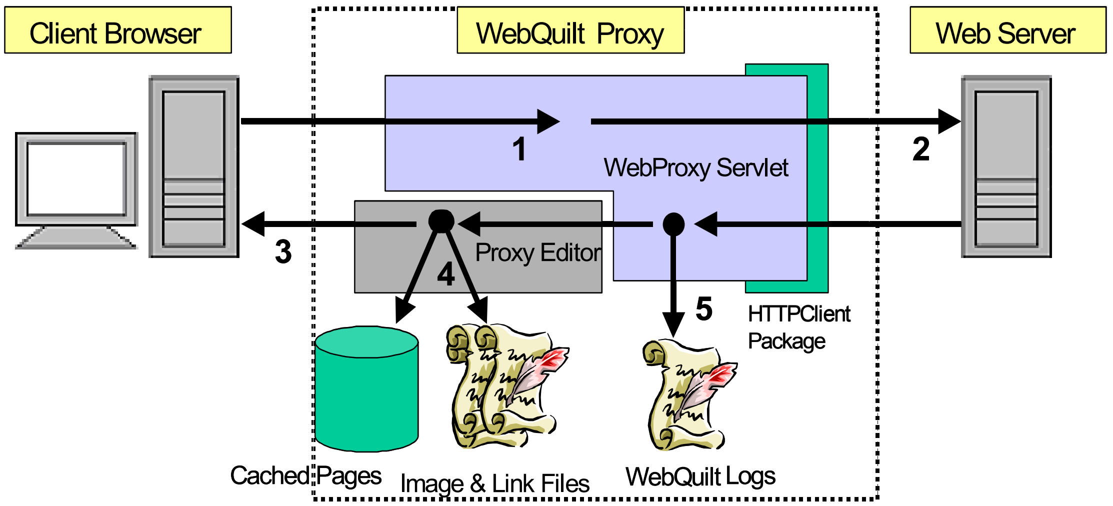

Figure 6: Proxy architecture overview and sequence of operations. In step one, the client request is processed. In step 2, the requested page is retrieved. In step 3, all of the links are redirected to the proxy, and the web page is returned to the client. In step 4, the web page is cached, the JPEG image of the page is created, and the DOM link information is saved. In step 5, the entire transaction is logged.

5.2 Inferring Actions, Aggregation and Layout

After all of the data for a task is collected, it is processed to infer where users clicked the back button of the browser. It should be noted that given our logging approach, the system can only be certain that the back button was pressed, but it cannot be certain of combinations of forward and back buttons. The processed files are then aggregated into the graph structure.

The current graph layout algorithm is an edge-weighted, depth-first traversal of the graph, displaying the most trafficked path along the top, and incrementally placing the less and less followed paths below it. This algorithm also uses grid positioning to help organize and align the distance between the nodes. The graph layout system has been designed to be extremely flexible, and as better algorithms are explored, they can be easily plugged-in.

5.3 Display

Nearly all of the WebQuilt visualization system has been written in Java. Except for the rendering of HTML into JPEG images, the entire system has been built on SATIN – a toolkit for pen-based interfaces [reference]. While the WebQuilt visualization is not intended for pen interfaces alone, SATIN provides excellent support for gesture interaction, zooming, and multiple views of objects.

Rendering the images of the web pages proved to be a research task in itself. We initially tried Java-based HTML browsers such as NetClue's Clue Browser [20] and WindRiver's IceStorm [21] browsers. Due to distribution and financial reasons, we were unable to provide high-quality renderings with these tools. Abandoning our idea of a purely Java-based and platform independent tool, we now use InfoZoom's JacoZoom []package to create Java callable wrappers around an ActiveX component containing a Microsoft Internet Explorer [] browser. The Java Native Interface (JNI) allows us to obtain a screen capture of the browser using native Windows code. Additionally, using Microsoft's specified DOM components, we are able to record the locations of all of the links in the web page. These link locations are referenced by the visualization, matching the logged DOM link IDs to the locations on the JPEG, effectively showing where users clicked.

5.4 System Limitations

A few disclaimers must be made on the re-rendering of pages as images. As we cannot infer the resolution of the end user's browser, our rendering is not entirely accurate. The same web page can also appear differently to users based upon the users browser. Our visualization is biased to the rendering of the most popular browser, Internet Explorer. Assuming participant's browsers and resolutions vary, we are merely selecting one rendition typical for many desktop systems. For devices such as PDAs and WAP phones, where screen resolution and space are critical concerns for designers, better methods will have to be investigated for understanding exactly what the users saw. However, the size of the browser screen used to re-render the content for the WebQuilt visualization can be modified by the designer in order to simulate a variety of device resolutions.

The WebQuilt visualization system is limited mostly by the logging technique. For example, the current proxy implementation cannot process links in JavaScript, though this is certainly technically feasible. The proxy also limits the type of data available for the visualization. Richer, more detailed user interactions like scrolling and mouse movement are not captured with this proxy. While this data can be gathered by instrumenting the proxied HTML with JavaScript, as in Edmonds 2001, it was a conscious design decision to avoid adding any extra information to the proxied data that may limit either the browsers and devices the WebQuilt proxy could log. The current implementation allows designers to explore the world of the Internet beyond the desktop, with PDAs and WAP phones, and still captures and displays a good deal of valuable usability data.

6. FUTURE WORK

The WebQuilt visualization system shows much promise in helping web site designers and usability experts understand remotely collected usability data. A number of enhancements are currently being added to the system. These include better navigation support, customization controls, improved quantitative reporting, and most importantly, the facility to filter on a variety of factors and scale to larger data sets. Moving to larger data sets will require research into clever layout algorithms and heuristics for simplifying usability graphs. Combining the filtering techniques with the current visualization system's capacity for semantic zooming will hopefully provide an analysis framework that is both understandable and usable.

Additional plans include an evaluation of the tool with web site designers, as well as incorporating the demographic and questionnaire data provided by outside sources.

7. CONCLUSION

We have introduced and described the WebQuilt visualization system for analyzing remote web usability log data gathered by the WebQuilt proxy logger. We have shown that semantic zooming of clickstream data is an effective method for exploring and probing usability data, allowing a designer to probe the data for interesting issues. We have shown that using an innovative, unobtrusive system to log data can provide valuable usability information when used in conjunction with task-based testing. We have detailed the underlying visualization system, its implementation and layout, its limitations, and provided a number of suggestions for enhancement.

The WebQuilt homepage can be found at: http://guir.berkeley.edu/projects/webquilt.

8. ACKNOWLEDGEMENTS

We would like to send a special thanks to Tara Matthews for all of her hard work, and a shout out to Wayne Kao and Vibhav Mittal as well. Many thanks to the Group for User Interface Research for their feedback, ideas and work on improving portions of WebQuilt.

9. REFERENCES

1. *Shop Until You Drop? A Glimpse into Internet Shopping Success*. 1999, Zona Research.
2. Spool, J. and e. al., *Web Site Usability: A Designer's Guide*. 1998, San Diego, CA: Academic Press.
3. Tedeschi, B., *Good Web Site Design Can Lead to Healthy Salses*. 1999, nytimes.com.
4. Etgan, M. and J. Cantor. *What Does Getting WET (Web Event-Logging Tool) Mean for Web Usability?* in *Fifth Human Factors and the Web Conference*. 1999. Gaithersburg, MD.

5. Davison, B. *Web Traffic Logs: An Imperfect Resource for Evaluation*. in *Ninth Annual Conference of the Internet Society (INET '99)*. 1999. San Jose, CA.
6. NetRaker, *NetRaker Suite*. 2001.
7. Vividence, *Vividence Browser*. 2000.
8. Hong, J.I. and J.A. Landay. *WebQuilt: A Framework for Capturing and Visualizing the Web Experience*. in *Proceedings of the Tenth International World Wide Web Conference (WWW10)*. 2001. Hong Kong.
9. Guzidal, M. and e. al., *Analyzing and Visualizing Log Files: A Computation Science of Usability*. 1994, Georgia Institute of Technology.
10. Wexelblat, A. and P. Maes. *Footprints: History-Rich Tools for Information Foraging*. in *Proceedings of ACM CHI 1999 Conference on Human Factors in Computing Systems*. 1999. Pittsburgh, PA.
11. Huang, M. *Information Visualization in Web Site Mapping: A Survey*. in *World Multiconference on Systemics, Cybernetics, and Informatics*. 2000. Orlandlo, FL.
12. Carrière, J. and R. Kazman. *WebQuery: Searching and Visualizing the Web through Connectivity*. in *Sixth International World Wide Web Conference (WWW6)*. 1997. Santa Clara, CA.
13. Chi, E.H., P. pirolli, and J. Pitkow. *The Scent of a Site: A System for Analyzing and Predicting Information Scent, Usage, and Usability of a Web Site*. in *Proceedings of ACM CHI 2000 Conference on Human Factors in Computing Systems*. 2000. Amsterdam, Netherlands.
14. Lamm, S.E. and D.A. Reed. *Real-Time Geographic Visualization of World Wide Web Traffic*. in *Fifth International World Wide Web Confernce (WWW5)*. 1995. Paris, France.
15. Cugini, J. and J. Scholtz. *VISVIP: 3D Visualization of Paths through Web Sites*. in *International Workshop on Web-Based Information Visualization (WebVis '99)*. 1999. Florence, Italy: IEEE Computer Society.
16. NIST, *WebVIP*. 1999.
17. VisualInsights, *eBizinsights*. 2001.
18. Vividence, *Vividence ClickStreams*. 2000.
19. Brainard, J. and B.Becker. *Case Study: E-Commerce Clickstream Visualization*. in *IEEE Proceedings of Information Visualization*. 2001. San Diego, CA.
20. NetClue, *Clue Web Browser*. 2001.
21. WindRiver, *IceStorm Browser*. 2001.