Bridging Waiting Times on Web Pages

Florian Alt, Alireza Sahami Shirazi, Albrecht Schmidt
University of Stuttgart
Institute for Visualization and Interactive Systems
{firstname.lastname}@vis.uni-stuttgart.de

Richard Atterer
Google Zurich
atterer@google.com

ABSTRACT
High-speed Internet connectivity makes browsing a convenient task. However, there are many situations in which surfing the web is still slow due to limited bandwidth, slow servers, or complex queries. As a result, loading web pages can take several seconds, making (mobile) browsing cumbersome. We present an approach which makes use of the time spent on waiting for the next page, by bridging the wait with extra cached or preloaded content. We show how the content (e.g., news, Twitter) can be adapted to the user's interests and to the context of use, hence making mobile surfing more comfortable. We compare two approaches: in *time-multiplex* mode, the entire screen displays bridging content until the loading is finished. In *space-multiplex* mode, content is displayed alongside the requested content while it loads. We use an HTTP proxy to intercept requests and add JavaScript code, which allows the bridging content from websites of our choice to be inserted. The approach was evaluated with 15 participants, assessing suitable content and usability.

Author Keywords
WWW; mobile device; waiting time

ACM Classification Keywords
H.5.4 **Information Interfaces and Presentation**: Hypertext/ Hypermedia – *User issues*

INTRODUCTION
Non-digital natives still remember the early days of the Internet when surfing the web was slow and the page loading time often exceeded the time spent reading it. Today, exponential growth in network bandwidth and computational power, as well as cloud computing, have made this problem much less pronounced. Especially in developed countries, broadband connections with good connectivity are pervasive, allowing web pages to be loaded very fast. Nevertheless, there are still complex operations, such as querying large databases, that are time-consuming. Many websites try to fill the gap between the user's request and the system's response by presenting system feedback or ads in order to keep the user's attention towards the website and to prevent them from clicking 'reload'.

MobileHCI'12, September 21–24, 2012, San Francisco, CA, USA.

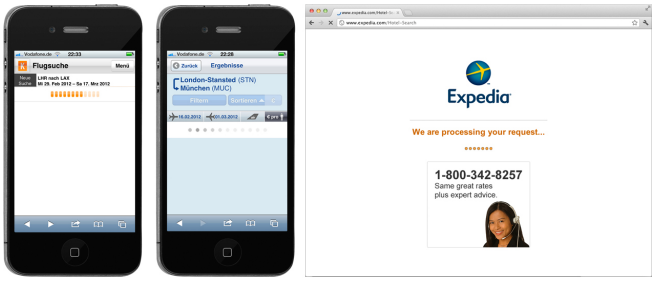

Figure 1: Websites that query large databases have to bridge waiting times. Current practice includes progress animations (left) and advertisements (right).

This problem is even more pronounced for the mobile Internet. Wireless connections strongly depend on the supported connection type (WiFi, UMTS, HSDPA, GPRS, EDGE) and on the number of people that simultaneously use the network. Hence, on average, data rates are lower during peak hours and in situations when we are moving (e.g., on the train). As a consequence, browsing on mobile devices is often cumbersome and the experience of surfing the Internet on a mobile device is very different from that on a PC or laptop [10].

A traditional approach of reducing waiting times is prefetching content that is linked from the current page. However, with highly linked pages and low bandwidth, this approach is not feasible on mobile devices. In this work, we aim at making Internet surfing on mobile phone more convenient by bridging the waiting time with useful content. In order to achieve this, we make use of the fact that the Internet connection is idle as soon as a webpage has been loaded and the user starts reading it. We use this time to preload content which can later be displayed when the next requested page is being loaded. However, the major challenge we are confronted with is the selection and presentation of appropriate content. Such content needs to require only little bandwidth to avoid affecting the overall performance. It should also not distract the user from his primary task and it should be tailored to the user's interests. Furthermore, it should be presented in such a way that the user's experience and the system's usability during Internet browsing is not negatively affected.

The contribution of this paper is twofold: (1) We present a prototype implementation based on an HTTP proxy that allows the preferred content to be inserted into web pages. Furthermore, we identify suitable content. (2) In a qualitative evaluation with 15 participants, we examine the usability of our solution when surfing on mobile devices, focussing on the impact of the loading time, the content, and its presentation.

RELATED WORK

Modifying webpages for different purposes has been subject to research for many years. Atterer et al. employed UsaProxy to monitor the user's browsing behavior and enable collaborative browsing [3]. Alt et al. extended UsaProxy to allow arbitrary applications to be deployed on top of existing web pages [2]. Approaches that deal with transforming a website's appearance are mainly browser extensions (e.g., Greasemonkey, Platypus). WBI is a pre-Web 2.0 approach that observes user interactions by using different kinds of agents [5]. Chickenfoot [6] and WebL [11] provide a language to ease the manipulation of web pages. *Intermediaries* like proxies can modify the information flow between the provider and the consumer by adapting content based on user profiles and histories [12].

A lot of research has been conducted on enhancing mobile surfing. Buyukkokten et al. looked at browsing on a PDA with low bandwidth, small display, and slow CPU [8]. Efficiently downloading content on mobile devices was investigated by Vartiainen et al. [15]. Kim et al. explored use contexts and usability problems of the mobile Internet [10]. Ways to more effectively use small displays and to enhance their usability were presented by Buchanan et al. [7]. It was found that simplicity of content, navigation, and easy and well-arranged controls are most important factors.

Ultimately, researchers have investigated the cost of task interruptions. Mark et al. found that people compensate for interruptions by working faster [13]. Iqbal et al. reported that the visibility of suspended application fosters faster recovery and suggest to provide easy access to the suspended task [9].

Our approach builds on previous findings. Keeping users updated on the current status of the web page is not new. Some existing websites that include lengthy processes or querying large databases to prepare responses attempt to make the waiting time more interesting by showing progress animations or promotions (Figure 1). The purpose is to keep the user focused and to lower the risk that he leaves the page due to the long loading time. Furthermore, it prevents him from reloading the page as he thinks the page is unresponsive. These solutions are successful, but limited to individual websites. We are not aware of any existing work that universally applied this approach to arbitrary websites. As we believe that our work can unfold its full potential on mobile devices, due to the aforementioned limitation, we focus on mobile surfing.

BRIDGING THE WAITING TIME

Assuming the user surfs web page A and clicks a link leading to web page B, bridging content can technically be realized in various ways. We sketch three different alternatives.

A. Showing Bridging Content on Page A Bridging content can be loaded in the background while the user is surfing page A. As the user clicks on the link to page B, the bridging content is displayed. The bridging content is directly replaced with B's content by the browser once page B has completely loaded. No extra effort is required by the user to skip the bridging content. The idle time between the click and completely loading page B can be optimally used. There is no control over how long the content is shown.

B. Showing Bridging Content on Page B Bridging content can be loaded on page B prior to loading the rest of the page. For this approach, the bridging content needs to be small to be pre-loaded as quickly as possible. This has the advantage of having full control over how long the bridging content is shown. This is especially useful if the content on page B loads more quickly than expected. As a consequence, the time the bridging content is shown can be extended in order to allow it to be perceived by the user.

C. Hybrid Solution By setting HTTP headers, it is possible to preload content on page A so that it will be fetched from the browser cache on page B without further network access. Alternatively, the JavaScript history API can be used to make the browser display page B's URL and load the content of B while the JavaScript used to load the bridging content continues to run. From the view of the browser, no page load happens. Instead content from A is replaced with content from B via manipulating the DOM tree.

Suitable Content

Arbitrary content can be used as a gap filler between clicking a link and displaying the requested web page. However, in practice the possible types of bridging content depend on the envisioned time the user stays on a web page, as this has a direct impact on the amount of data that can be fetched. For instance, the user is likely to stay longer on a news website than on a search results page. In general, brief, non-complex information that can be quickly perceived and does not distract users from the main browsing task is most suitable. Furthermore, to succeed in displaying interesting information in an unobtrusive and non-distracting way, the user's interests and the context of the website should be taken into consideration.

Alt et al. [1] identified news, weather, and entertaining content (cartoons, etc.) as most promising forms of micro entertainment based on a web survey with 127 participants. In times of social networks, we also see potential in displaying the latest feeds from networks such as Facebook or Twitter. Ads as the most obvious business model might also be considered. With knowledge about user interests, ads can be personalized to fit the viewer's interest. For our prototype we opted to support the following types of information: weather, RSS news feeds, Facebook news feeds, and contextual ads.

Presentation of Content

Finally, we reflected on the *presentation* of the content. We implemented two visualization modes. First, in *time-multiplex mode*, we use the entire screen to display the content during the idle time while the requested page is loading. As soon as the requested page has been completely loaded, the bridging content disappears and the requested content becomes visible. In *space-multiplex mode*, content is displayed alongside the requested content as it is loading (see Figure 2).

PROTOTYPE

We use an *HTTP proxy* to intercept HTTP requests that occur when the browser loads a web page, and insert JavaScript into the HTML code to provide the required functionality. At the same time, a *Content Creator* module handles choosing and requesting the bridging content, inserting it into the

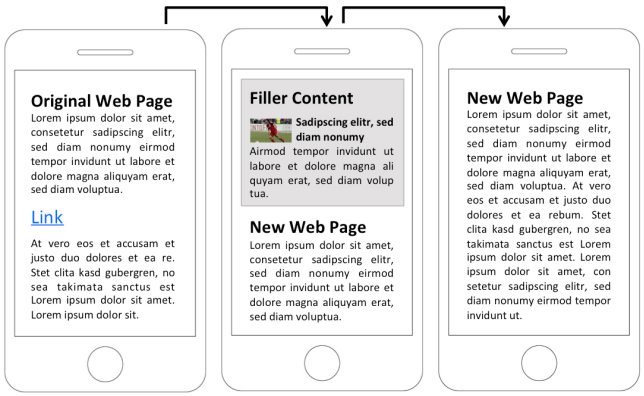

Figure 2: In space-multiplex, the bridging content is displayed alongside the new page as it is loading.

HTTP response together with the associated JavaScript code. Furthermore, an *administration (web) interface* is provided that allows the bridging content to be personalized. In the client (browser), the JavaScript is responsible for displaying the bridging content in time- or space-multiplexing mode.

For the prototype implementation, we used four different types of content: weather (from WorldWeatherOnline.com based on the IP address), RSS new feeds, Facebook news feeds, and contextual ads. Content is shortened to 100 characters due to the limited display size on mobile devices. Each content title is a hyperlink which points to the full article.

The prototype allows each user to specify (a) the content type, (b) the RSS feed category, and (c) the multiplexing mode via an admin interface. For each type of *content* the user can select whether the respective type of content should be displayed at all. Furthermore, he can specify a priority, which affects how frequently it occurs. Similarly, the user can choose his preferred *RSS feed category* (sports, computers, etc.). Based on this, content from the associated RSS feeds is used. Finally, the *multiplexing mode* can be selected.

EVALUATION

In a lab study with 15 participants, we were interested in (a) how suitable the approach is for different devices (smartphones, tablets), (b) whether there is an effect of the loading time on the perceived usability, and (c) the preferred presentation mode. We used a within subject design. The users had to perform search tasks for products on the Amazon web page (e.g., "What is the price of the iPad 16GB?"). The users performed the search tasks on two mobile devices – a Samsung GalaxyTab and an HTC Desire HD smartphone – and for each device in time-multiplex, space-multiplex, and a baseline mode (no bridging content). This resulted in 6 conditions that we counter-balanced between the participants. The search tasks were read aloud by the experimenter. As soon as participants found the answer, they informed the experimenter. The search tasks were performed for 5 minutes per condition. To reduce the complexity, we only used non-personalized RSS feeds in the study. We selected 200 RSS feeds and separated them into 20 categories prior to the study. Ten categories were used in each mode.

As our prototype loads the bridging content on the requested page, we were not only able to control how long the content was shown, but also to measure how long it was visible to the participants (e.g., when they scrolled away or clicked to close it). During the study, we logged user interaction (user ID, current website, device, loading time of the website). Additionally, for space-multiplex mode, we logged the time until users scrolled away from the bridging content. For time-multiplex mode, we recorded whether content was closed prior to the page being loaded or if content was brought back. Usability was measured using an SUS questionnaire [4] for each condition. Finally, a semi-structured interview was conducted.

FINDINGS

In total, 15 people participated in the study (11 male, avg. age 25.8 years). All were students, 12 owned a touchscreen phone and 8 had unlimited data access on their phone. Note that despite the limited demographics, our sample of young people nevertheless represents one of the main target groups.

The analysis focuses on the mobile phone as we assumed the surfing experience on the tablet to be more similar to the PC. In order to assess the perceived *usability*, we compared the SUS scores for all conditions. For the phone, the baseline condition (86.9) was rated slightly higher than space-multiplex (85.2) and time-multiplex mode (79.6). Further analysis revealed no significant differences between all conditions, which suggests that our approach does not have any negative influence on usability for mobile surfing.

We were also interested in the effect of the *loading time*. The analysis of the log file showed that the average loading time on the mobile phone (avg.=4.5s, STD=5.09s) was much higher than on the tablet (avg.=1.9s, STD=2.03s). A Pearson correlation analysis revealed a negative correlation between the loading time and the SUS score for the phone in the baseline condition ($r=-.56$, $p<.04$). This is a strong indicator that for long loading times, browsing on a mobile device was perceived to be less usable. Interestingly, no significant correlation was found for time-multiplex and space-multiplex mode. Consequently, we believe that our approach has the potential to overcome the perceived low usability of mobile surfing. For the tablet we did not find any significant correlation between the conditions and the SUS scores.

With regard to the *content*, we were interested in how much time people would spend with the bridging content. For space-multiplex mode, we logged the time until users scrolled away from the content (which was possible at any time). On the phone, this happened on average after 5.05s (STD=9.38s), on the tablet after 2.9s (STD=3.9s). When subtracting the average loading time, participants spent 2.43s longer with the content on the phone than in the baseline condition, on the tablet 2.27s. These results are interesting, e.g., for advertisers, as there is a high chance that users indeed perceived the content. Also, these values are likely to increase if the content is tailored to the users' interests. In time-multiplex mode, users were able to hide the content before the page finished loading in the background, and they were able to redisplay the content after it had been closed. On the phone, the bridging content

was never hidden, but in 8 cases it was redisplayed and perceived for 8.22s on average before closing it again. On the tablet, the bridging content was closed 23 times. In 8 cases, content was redisplayed and perceived 15.9s on average.

The *qualitative feedback* showed that the system was very intuitive and usable. For the time-multiplex mode, participants felt that more feedback on the current loading status would be useful. All would be interested to get personalized content.

DISCUSSION AND CONCLUSION

We presented an approach for bridging loading times on web pages based on personalized content. We described 3 approaches on preloading content and discussed advantages and disadvantages. Our prototype embeds the bridging content either in a time- or space-multiplexed manner. Findings from our evaluation suggest that both approaches are most suitable if the connection speed is low and the screen is small. No negative experience or impact on usability were observed. Yet, no conclusion can be drawn as to which approach is more usable: space-multiplexing gives the user more control as she can immediately perceive a partially loaded page, whereas time-multiplexing only notifies her as the page is loaded.

We should mention that the study design was *not* in favor of our approach. First, the users had a directed task. Hence, a rather low interest in the additional information can be assumed. Nevertheless, the additional content was explicitly followed in 16 cases. We believe that the full potential of the approach will only become evident for activities where users do not follow a certain goal, such as reading news websites. In these cases, they will be more open to follow and perceive the additional content and links. Second, in order to reduce complexity and not to bias the participants, we did not exploit the adaptive capabilities of the system. However, we see a lot of potential in tailoring content to the users' interests.

Qualitative feedback revealed that the participants' primary concern was the distraction from the primary task. As mentioned above, this was probably due to the directed search task. We believe that this could be improved by considering the main content of the website when choosing the bridging content. Bridging content that aligns well with the requested web page is likely to minimize distraction (e.g., showing links to related articles as the user browses a news site). A future version could tailor the bridging content based on semantic information from the current page. Future research could investigate cases in which users find personalized content so interesting that they abandon their current task and whether this potentially leads to loss-of-activation or description errors [14].

Though our approach can be universally applied, we nevertheless see potential in incorporating it with certain websites, especially when facing long loading times. This way, the type of information can be tailored to the site. For instance, an online store like Amazon might choose to display ads related to the products the user looked at, and a travel site like Expedia might show a checklist where the user ticks off "hotel booked", "flight booked", or "car rented".

ACKNOWLEDGEMENTS

We would like to thank Markus Stumvoll for his help with extending the UsaProxy and during the study. The research leading to these results has received funding from the European Union Seventh Framework Program (FP7/ 2007-2013) under grant agreement no. 244011.

REFERENCES

1. Alt, F., Kern, D., Schulte, F., Pfleging, B., Shirazi, A. S., and Schmidt, A. Enabling micro-entertainment in vehicles based on context information. In *Proc. of AutomotiveUI '10*, ACM (2010), 117–124.
2. Alt, F., Schmidt, A., Atterer, R., and Holleis, P. Bringing web 2.0 to the old web: A platform for parasitic applications. In *Proc. of Interact '09*, Springer-Verlag (Berlin, Heidelberg, 2009), 405–418.
3. Atterer, R., Wnuk, M., and Schmidt, A. Knowing the user's every move: user activity tracking for website usability evaluation and implicit interaction. In *Proc. of WWW '06*, ACM (2006), 203–212.
4. Bangor, A., Kortum, P., and Miller, J. An empirical evaluation of the system usability scale. *International Journal of HCI 24*, 6 (2008), 574–594.
5. Barrett, R., Maglio, P. P., and Kellem, D. C. How to personalize the web. In *Proc. of CHI '97*, ACM (1997).
6. Bolin, M., Webber, M., Rha, P., Wilson, T., and Miller, R. C. Automation and customization of rendered web pages. In *Proc. of UIST '05*, ACM (2005), 163–172.
7. Buchanan, G., Farrant, S., Jones, M., Thimbleby, H., Marsden, G., and Pazzani, M. Improving mobile internet usability. In *Proc. of WWW '01*, ACM (2001), 673–680.
8. Buyukkokten, O., Garcia-Molina, H., Paepcke, A., and Winograd, T. Power browser: efficient web browsing for pdas. In *Proc. of CHI '00*, ACM (2000), 430–437.
9. Iqbal, S. T., and Horvitz, E. Disruption and recovery of computing tasks: field study, analysis, and directions. In *Proc. of CHI '07*, CHI '07, ACM (2007), 677–686.
10. Kim, H., Kim, J., Lee, Y., Chae, M., and Choi, Y. An empirical study of the use contexts and usability problems in mobile internet. *Proc. of HICSS* (2002).
11. Kistler, T., and Marais, H. Webl - a programming language for the web. In *Proc. of WWW '98*, Elsevier (1998), 259–270.
12. Maglio, P., and Barrett, R. Intermediaries personalize information streams. *Commun. ACM 43* (2000).
13. Mark, G., Gudith, D., and Klocke, U. The cost of interrupted work: more speed and stress. In *Proc. of CHI '08*, ACM (2008), 107–110.
14. Norman, D. A. *The Design of Everyday things*, 1 ed. Basic Books, Dec. 2002.
15. Vartiainen, E., Roto, V., and Popescu, A. Auto-update: a concept for automatic downloading of web content to a mobile device. In *Proc. of Mobility '07*, ACM (New York, NY, USA, 2007), 683–689.