

Mobile Web for Pervasive environments – design web experiences for multiple mobile devices

Thomas Riisgaard Hansen
University of Aarhus
Aabogade 34
8200 Århus N, Denmark
thomasr@cs.au.dk

ABSTRACT

In this paper we present an architecture for designing web pages that uses multiple mobile and stationary devices to present web content. The architecture extends standard web technology with a number of functions for expressing how web content might migrate and use multiple displays. The architecture is developed to support desktop applications, but in this paper we describe how the architecture can be extended to mobile devices by using AJAX technology. The paper also presents an implementation and presents a number of applications for mobile devices developed with this framework.

Keywords

Mobile web pages, Distributed display environment, Ubiquitous web, Multi-display web pages, Interactive spaces, Web browsers, Multi-device interaction, Collaborative interaction

1. INTRODUCTION

Interfaces have traditionally been designed to support one display only. While multi-display setups are becoming more common in desktop and home environment the secondary display is mainly used as an extended desktop expanding the screen estate. Nevertheless, the number of displays and devices in offices and homes are expanding rapidly and most of them provide the ability to display web content. With the SpaceExplorer architecture [7] we developed a tool which allowed web designers to design multi-display web pages. Multi-display web pages use more than one device to present its content. The SpaceExplorer prototype was developed as a browser plug-in and worked in a completely distributed peer-to-peer manner.

However, development of browser plug-in for mobile devices provides several challenges. First, developing plug-ins for the mobile browser proves challenging due to various security models. Secondly, each manufacturer has its own implementation of the browser for their platform and supporting several mobile devices would require several code bases for each platform.

To address this issue this paper reports on how to support multi-display web devices by using a centralized server approach and Ajax. Using AJAX allows support for every mobile platform that

supports this Javascript function. AJAX and JavaScript also have a number of limitations concerning response time and the ability to control 3rd party web content. The paper will briefly present related work, present the SpaceExplorer foundation and the AJAX extension for mobile devices. Finally, the paper presents a number of web pages developed with the framework and concludes on mobile web development for mobile devices.

2. RELATED WORK

Research in multi-display systems has a long tradition [10]. Nevertheless, recently it has received extensive attention due to the increase in the number of devices and displays present in our everyday environment [8].

One research direction has been to create interfaces that *adapt to the device they are presented on* [2], [11], [5]. This research direction is characterized by describing the interface in some high level XML-format and applying a set of transformations depending on the device. These projects, nevertheless, mainly focus on adapting the interface to *one* device.

More relevant for this line of research is projects working with *applications that spans multiple devices*. The Pebbles project works with a shared application running on a public display and PDAs [12]. A similar application is designed and tested for a clinical setting in [1], Eriksson et al presents and application running on large wall displays and mobile phones for picture sharing in public places [3], Forlines et al has work on adapting a single user application (Google Earth) to multiple displays [4]. Finally, Zang et al an present an application for designing presentation that spans multiple displays through an authoring tool for distributed environments (based on time lines) [14]. All these projects presents interesting findings concerning how to design for multiple displays, but the solutions are all based on dedicated software and often focused on a particular domain.

Finally, a number of highly related projects have worked with the similar challenge of using *web technology to facilitate multiple displays*. The WebSplitter project works with how to split a web site into smaller pieces that can be presented to different users or to the different users' devices [6]. The project presents XML-web pages split between the devices according to a policy file. The architecture is based on a proxy server and a service discovery database. A similar proxy approach is used in the Multi-Browser [9] and the project by Vandervelpen et al [13]. However, where the MultiBrowsing uses a small butler application Vandervelpen et al uses XMLHttpRequest object to interact between the gateway and the devices. Nevertheless, the last project still needs a small client running on the device for device discovery.

The related work presented provides a broad view of the area and even though not all projects address exactly the same research

question as this paper, they all present ideas and solutions needed for multi-device interaction to succeed. Related to this line of research are also research into the mobile web and the ubiquitous web [15]. However, the focal point in this paper is on multi-display and device interaction and hence the focus is concentrated on this line of research.

3. THE SPACE EXPLORER ARCHITECTURE AND PROTOTYPE

The SpaceExplorer architecture main goal is to extend the browser with a set of additional functions for designing web pages that facilitates more than one display. The architecture consists of a device manager component, which holds a list of nearby devices and an expression component, responsible for distributing web content to nearby browsers on other displays. The expression component offers two new CSS-style properties and a new JavaScript function. Figure 3, summarizes the basic architecture and it is further elaborated in [7].

The first, the SpaceOpen property will, when applied to a CSS-class, load additional web content in the browser on one or more other displays present when the current page is loaded.

The second, the SpaceLink property will augment web link (<a> tags) and load the link on the display in the ensemble that provides the best match for the requirement specified in the SpaceLink class.

Finally, a new JavaScript function, RemoteScripting, allows a web page to request the executing of a piece of JavaScript on one of the other displays nearby.

Figure 1, shows a use case. A page is loaded on display 1. If the user moves the mouse over a picture the RemoteScripting function is called and the location of the picture is displayed on display2. Finally, each picture has a link to a full size version of the picture. This link has a CSS-class associated with the SpaceLink property and each time the picture is clicked the content is loaded on display3. Figure 2 shows the code required for this example.



Figure 1. An example of a multi-display setup with an overview display, a display showing the location of the photo and a display showing the photos in full resolution.

The device manager component is responsible for detecting new devices, handle devices that go offline, managing the description of the different devices capabilities and ensuring communication between the devices. The device manager uses multicast to discover nearby devices. If a new device is found, the devices are added to the list of present devices. Because, multi-cast is used for discovery only devices within the network are found. The protocol is a simple proprietary discovery protocol, but it can be extended to use e.g. Bonjour, Zeroconf or Universal plug and play technology (UPnP).

Each device is represented by three core abilities: Its name, position and a list of capabilities. The name identifies the device. It can be unique, but it is not a requirement. The position describes the current location of the device and capabilities denotes a list of the device's properties along with a number specifying how good the particular device is at that property e.g. sound=10, video=30, mobility=20. Currently, there is no fixed ontology of properties and the user is free to add whatever properties they need (e.g. a capability could be called myfavoritetv=20).

```
<html><head><title>SpaceTravel</title>
<script src="SpaceExplorer.js"
type=text/javascript language="JavaScript">
</script>
<style type="text/css">
.PictureDisplay {
SpaceLink:cap=picture#;
}</style>
</head>
<body><table> <tr><td>
<a href="NewYork.jpg" class="PictureDisplay">
 src="NewYork.jpg" width="259"
height="194"></a>
</td></tr> </table></body></html>
```

Figure 2. A full example of a web page that demonstrate the SpaceLink and Cross -Device scripting functionality

A simple query mechanism specified which display to open content on and is used by the SpaceOpen, SpaceLink and RemoteScripting feature. A query can specify e.g. that the content should open on the display that best match a set of capabilities e.g. cap=sound. The query can also ask for the device with a specific name, a specific position or with the best match of a set of capabilities.

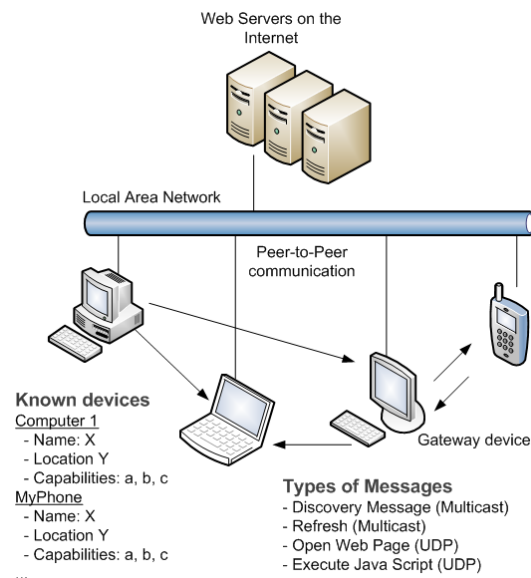


Figure 3. The overall SpaceExplorer Architecture. The content is stored on web servers. Devices on the local area

network use peer-to-peer communication to maintain a list of known devices and their properties. Multicast messages are used to update this list and UDP messages are used to open web content and execute JavaScript on remote devices.

To test the feasibility and properties of the suggested architecture, and to get feedback about web design for multiple devices, the architecture has been fully implemented. The architecture is currently implemented as a Browser Helper Object / Toolbar in Internet Explorer. The reason for choosing Internet Explorer is that it has the biggest market share and supported extensive customization, but the architecture can also easily be implemented as an extension to e.g. Mozilla Firefox. The toolbar integrate seemingly into the browser and no other program is needed on the computer for the system to work (Figure 4 shows the toolbar).

The first time the toolbar is installed the user need to configure the name of the device and add some capabilities to the device. These information is then stored in the registry on the local device. From the toolbar the service can be started and stopped. The display field shows the current number of devices in the ensemble and the collaborate button can be used to open web pages on some of the other devices.

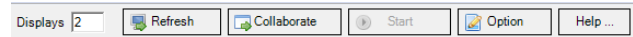


Figure 4. The Space Explorer Internet Explorer Toolbar. Shows the available display, refresh for updating the information, collaborate for sending content to other devices, start for starting the system, option for adding capabilities and help for help.

The Browser plug-in works for all newer version of Internet Explorer and works on Windows XP and Windows Vista, however in the larger evaluation some compatibility issues surfaced as discussed in the evaluation section. It is freely available for download at (<http://www.daimi.au.dk/~thomast/SpaceExplorer>).

4. MOBILE SUPPORT

To support mobile browsers we developed a small embedded webproxyserver in the toolbar, we package each web page in a AJAX frameset that checks for updates and rewrite every link to point to the proxy server. Mobile and other non-Internet Explorer browsers can connect to the peer-to-peer network by contacting the device with the toolbar through the toolbars proxy (e.g. <http://proxydeviceurl:proxydeviceport>). The mobile browser is hence not automatically discovered, but the user need to explicit connect it to the network.

The first time the mobile browser connect to the address a small web page will be presented where the user can select a name, a set of capabilities and select how often to send an AJAX update request (see Figure 5). This information is stored in the small web server running within the toolbar. The next page the user is presented with is a small page showing the connected devices and which allows the user to open pages on the mobile device or on one of the devices in the ensemble. If another device is selected the *open request* is send to this particular device through the proxy gateway. The presented web page on the mobile device is also embedded in a frameset that host a small AJAX script which regularly checks the gateway for update. If another device sends an *open web page request* to the mobile device this request is first

stored on the proxy gateway and the next time the mobile device checks in the new web page is send to the mobile device.

Every time a web page is opened on the mobile browser the page is first parsed on the gateway device. The gateway device will replace all links in the current document with new links that points to the gateway and if the SpaceLink or SpaceOpen style is used the gateway will translate these link accordingly. Because, no additional component is installed on the mobile client the only way to implement the SpaceExplorer functionality is to use the proxy gateway. The advantage is that web pages can be viewed that does not support the framework. The disadvantage is that the gateway only translates HTML-links and not e.g. links activated through JavaScript or other means. Embedding every web page in a AJAX frameset that checks for updates and parse and translate the web pages in the gateway allows other browsers to join the ensemble and use the SpaceExplorer features without having to install additional software.

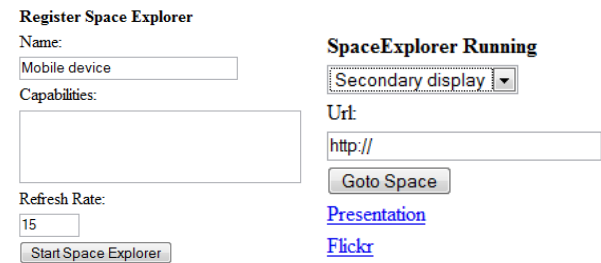


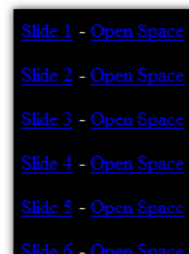
Figure 5. The mobile pages for registering the device with the ensemble through the proxy gateway.

5. MOBILE UBIQUITOUS WEB PAGES

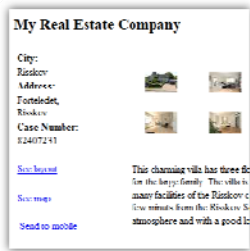
Based on the technology we developed a number of ubiquitous mobile web pages.



SpaceFlickr – The SpaceFlickr page allows the user to view pictures from a Flickr account on the mobile phone. When a photo is loaded on the mobile phone a large scale version of the picture is shown on a nearby display, if a second screen is present the next picture in the slide show is shown on this screen. Finally adding a third display will show the previous picture on this display. (Less than 45 lines of CSS/HTML/PHP)



SpacePresentation – This small web page shows a list of link to slides prepared for a slide show on the mobile device. Clicking a slide will show that slide/page on a nearby computer (e.g. a computer with a projector connected) (less than 35 lines of CSS/HTML)



SpaceRealEstate – This web page shows a real estate home page. Clicking on the different pictures will open the pictures on other displays. Clicking on the “send to mobile” will push the selected web page to the mobile device for further reference (less than **100** lines of CSS/HTML – mainly due to page design elements)

Figure 6. Three selected small applications

Figure 6 shows three of the developed applications along with the number of HTML/CSS/PHP lines of code to realize the described behavior. In general, the developed application required very few lines of code to present advanced behavior by using the three expression features (SpaceLink, SpaceOpen and cross-device scripting) and only very few lines related to the multi-device distribution of content (less than 10 in all three applications).

6. CONCLUSION

Web technology today is mainly about presenting content on one display. In this paper we present the SpaceExplorer architecture, a tool for exploring how to design web content that is inherently designed for more than one display.

A challenge with extending web technology by using plug-ins is to provide mobile support for these features. By using AJAX and a proxy gateway the paper shows how the functionality from the desktop browser plug-in can be transferred to mobile browsers. The paper also present a set of web pages developed with this technology that demonstrates how mobile web content can be pushed to e.g. large displays (SpacePresentation), how browsing mobile content on the mobile phone can trigger the loading of additional content on nearby large displays (SpaceFlickr) and how browsing for e.g. a new home on a computer in a real-estate store can facilitate the device switch from a stationary computer to a mobile device by clicking a link (SpaceRealEstate).

However, there are a number of challenges with the AJAX approach e.g. the need to constantly pull for updates. The tool in its current version mainly designed for exploring how web pages can be designed for setups with both mobile and stationary devices, but will also be fully usable in creating creative new web pages e.g. for museums or other setups. The plug-in is freely available at <http://www.daimi.au.dk/~thomasr/SpaceExplorer>.

REFERENCES

- [1] Alsos, O. A. and Svanæs, D. 2006. Interaction techniques for using handhelds and PCs together in a clinical setting. In Proceedings of the 4th Nordic Conference on Human-Computer interaction: Changing Roles, NordiCHI '06, vol. 189. ACM Press, New York, NY, 125-134.
- [2] Berti, S., Correani, F., Mori, G., Paternò, F., and Santoro, C. 2004. TERESA: a transformation-based environment for designing and developing multi-device interfaces. In CHI '04 Extended Abstracts on Human Factors in Computing Systems. ACM, New York, NY, 793-794.
- [3] Eriksson, E., Hansen, T. R., and Lykke-Olesen, A. 2007. Reclaiming public space: designing for public interaction with private devices. In Proceedings of the 1st international Conference on Tangible and Embedded interaction, TEI '07. ACM, New York, NY, 31-38.
- [4] Forlines, C., Esenther, A., Shen, C., Wigdor, D., and Ryall, K. 2006. Multi-user, multi-display interaction with a single-user, single-display geospatial application. In Proceedings of the 19th Annual ACM Symposium on User interface Software and Technology, UIST '06. ACM Press, New York, NY, 273-276.
- [5] Grundy, J. and Yang, B. 2003. An environment for developing adaptive, multi-device user interfaces. In Proceedings of the Fourth Australasian User interface Conference on User interfaces 2003 - Volume 18 ,ACM International Conference Proceeding Series, vol. 36. Australian Computer Society, Darlinghurst, Australia, 47-56.
- [6] Han, R., Perret, V., and Naghshineh, M. 2000. WebSplitter: a unified XML framework for multi-device collaborative Web browsing. In Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work, CSCW '00. ACM, New York, NY, 221-230.
- [7] Hansen, Thomas Riisgaard, SpaceExplorer – a light-weight architecture for designing web experiences for multiple displays and devices, in submission.
- [8] Hutchings, D. R., Stasko, J., and Czerwinski, M. 2005. Distributed display environments. interactions 12, 6 (Nov. 2005), 50-53. DOI=<http://doi.acm.org/10.1145/1096554.1096592>
- [9] Johanson, B., Ponnekanti, S., Sengupta, C., Fox, A., "Multibrowsing: Moving Web Content Across Multiple Displays," Proceedings of Ubicomp 2001, September 30-October 2, 2001.
- [10] Mano, Y., Omaki, K., and Torii, K. 1981. An intelligent multi-display terminal system towards: a better programming environment. SIGSOFT Softw. Eng. Notes 6, 2 (Apr. 1981), 8-14.
- [11] Mori, G., Paterno, F., and Santoro, C. 2004. Design and Development of Multidevice User Interfaces through Multiple Logical Descriptions. IEEE Trans. Softw. Eng. 30, 8 (Aug. 2004), 507-520.
- [12] Myers, B. A., Stiel, H., and Gargiulo, R. 1998. Collaboration using multiple PDAs connected to a PC. In Proceedings of the 1998 ACM Conference on Computer Supported Cooperative Work (Seattle, Washington, United States, November 14 - 18, 1998). CSCW '98. ACM, New York, NY, 285-294.
- [13] Vandervelpen, Ch., Vanderhulst, K., and Coninx, K. Light-weight Distributed Web Interfaces: Preparing the Web for Heterogeneous Environments. Proc. of 5th Int. Conf. on Web Engineering ICWE'2005 (Sydney, July 25--29, 2005). Lecture Notes in Computer Science. Springer-Verlag, Berlin, 2005.
- [14] Zhang, H.J., Liu, Q., Lertsithichai, S., Liao, C.Y., and Kimber, D. (2004), A Presentation Authoring Tool for Media Device Distributed Environments, Proceedings of ICME'2004, Taipei, Taiwan.
- [15] Web: Next steps: Ubiquitous Web (Feb 2007) <http://www2006.org/programme/item.php?id=w7>