Pre-print version of the paper. Published in UAIS Journal, 2007

Using RSS feeds for effective mobile web browsing

John Garofalakis

University of Patras, Computer Engineering & Informatics Department

26500 Patras, Greece

Computer Technology Institute

26500 Patras, Greece

+302610997866

garofala@ceid.upatras.gr

Vassilios Stefanis

University of Patras, Computer Engineering & Informatics Department

26500 Patras, Greece

Computer Technology Institute

26500 Patras, Greece

+306973281930

stefanis@ceid.upatras.gr

Abstract: While mobile phones are becoming more popular, wireless communication vendors and device manufacturers are seeking new applications for their products. Access to the large corpus of Internet information is a very prominent field. However the technical limitations of mobile devices pose many challenges. Browsing the Internet using a mobile phone is a large scientific and cultural challenge. Web content must be adapted before it can be accessed by a mobile browser. This work presents a new methodology that uses Really Simple Syndication (RSS) feeds for the adaptation of web content for use in mobile phones. This methodology is based on concrete design guidelines and supports different viewing modes. The mobile tool provided using the RSS feeds is evaluated based on user centered evaluation and the results are presented.

Keywords: Mobile devices, RSS feeds, content adaptation, mobile browsing, web browsing, quality evaluation.

1. Introduction

During the past decade the world has witnessed a revolution in the field of wireless networks and mobile devices. The term 'mobile device' refers to a device specially designed for synchronous and asynchronous communication while the user is on the move. It includes a wide variety of appliances such as PDAs and mobile/smart phones. Tablet PCs are also included in the category of mobile devices, however they are bigger in size and more costly to acquire. Mobile phones are by far the most popular mobile devices. Among the many facilities they provide is access to the Internet, however browsing using a mobile phone is not as easy as browsing using a common desktop PC. Screen size and resolution, number of supported colors, entering text method, computation power, memory size, rate of data transfer and energy required for proper functionality are the main limitations for using a mobile device for browsing [11]. On the other hand, users demand access to information anytime, anywhere. Since within a few years, many of the devices accessing the Web will be mobile, there is a need for developing methods and techniques that will allow efficient web browsing using mobile devices for all user groups.

The problem of web browsing through a mobile phone has attracted much attention by the research community and the industry alike. One of the proposed solutions is to use a proxy server which adapts the web content for mobile devices on-the-fly and returns the result to the mobile user. In this paper we use the proxy server idea to develop a tool which uses RSS feeds (an XML-like notation) of a web site. Our methodology combines RSS feeds with simple content adaptation techniques (presented in section 5) in order to evaluate how RSS feeds can improve mobile browsing.

Quality, in our work, has two main facets in a mobile environment: perception and service. The latter characterizes the technical side of mobile networking and represents the performance properties that the underlying network is able to provide. The former provides a characterization of the user side of the mobile web browsing experience. That Quality of mobile applications impacts upon the success of the mobile idea is without doubt, as it plays a pivotal role in attracting and retaining users. Frustrated users will leave a site if they perceive it as being slow (for instance), causing lost revenue [8]. To compound the topic, mobile applications are likely to be compared to the users' actual web experiences in the physical world and their normal browsing interactions in that environment. In this paper we use news sites for two reasons: (a) RSS feeds refer widely in news sites, (b) news sites are popular with general content.

This work is structured as follows. In section 2, the design principles of this paper are presented. In section 3, content adaptation approaches for mobile devices are presented, while in section 4, the RSS approach is discussed. Section 5 provides a complete presentation of the application that we have developed and in section 6 the quality evaluation method is presented. Finally, section 7 analyzes the quality evaluation results and in section 8 we conclude.

2. Design principles for mobile applications

The limitations of mobile devices force developers to be very careful when designing applications. In order to develop mobile web applications certain guidelines should be followed such as the XHTML Guidelines for Creating Web Content [27]. These guidelines are also presented in the Mobile Web Best Practices 1.0, Basic Guidelines, W3C Candidate Recommendation [15]. In the next paragraphs of this section the design principles of this paper are presented.
The interface of our tool is based on a consistent and easy to learn navigation model. Mobile users' requirements are different from PC users. Mobile users do not browse the Web in the classic way but they demand quick access to specific information. Therefore, in the presented tool content with a lot of graphics, animations and different colors is removed. For example, instead of an image the alt attribute information is presented, if it is provided.
Text inputting supports user's interaction with mobile web application. For the majority of mobile users, using a keyboard of a mobile phone, even with T9 support, is a slow process [23]. In order to limit text inputting the only form that the user has to fill is the form with the URL that she/he wants to visit.
Navigation hierarchy ensures the navigability of web applications and facilitates web browsing. A navigable web site is this where finding services require only a few clicks to appropriate hyperlinks. If the navigation model is too complex (e.g. a deep tree of hyperlinks), the user may get lost and frustrated. Developers also need to avoid creating links for functions already implemented by the system. For example, if we create hyperlinks for going to the previous page in a mobile web application, a function already implemented by the browser, valuable space is misused in an already small screen; on the other hand, special links are needed so that the user can browse back to the main page or other important pages in a click.
A designer of mobile web applications should have in mind that the applications refer to devices with small screen while at the same time these applications should be also user-centric; reaching a consensus there is difficult. Every page has to contain the main content starting from the beginning. The need for vertical

scrolling should be kept to a minimum; horizontal scrolling is generally not recommended at all. Every modified web page should contain the tag <title> so that the user may read a short description of the page at a glance. The browsers usually put the content of the <title> tag to a visible place. Therefore, the order of the html elements and the title of the specific page do not change.
Color usage is also important. Using colors obviously gives a pleasant and friendly interface, but a too colored screen confuses. All the pages of the application must have the same colors so the user can feel that he/she is navigating in the same environment. By removing background images, background colors and text colors the readability of the content is facilitated.
The use of images in Internet applications is common. Nevertheless, using images in mobile web applications significantly increases download and response time and thus, usage cost.
The interested reader may find the detailed guidelines in [15],[16],[25],[28].

3. Content adaptation in mobile devices

There are four general approaches for adapting web content for small screen devices: device-specific authoring, multi-device authoring, automatic re-authoring, and client-side navigation [1]. The first two approaches obtain high quality results by authoring device-specific web content. Having two versions of the same web content (one for regular and one for mobile users) is costly and the vast majority of web sites do not use it. This makes most of the Internet actually inaccessible for mobile phone users. On the other hand, client-side navigation and automatic re-authoring techniques do not require the collaboration of page authors and are therefore more widely applicable.
In client side browsing, the whole content is delivered at the browser and then the browser decides how to present it. For example, the Opera Browser [17] applies the Small Screen Rendering technique where the content is presented in a single column, so horizontal scrolling is not necessary. All client side browsing methods require the full code of a web page to be downloaded to the mobile device. This presupposes the existence of enough main memory to the mobile device for storing and running the page code. Furthermore, the user is usually billed according to the amount of downloaded data, so downloading large content is costly.

Automatic re-authoring fall into two main categories: page scaling and page reformatting. Page scaling includes scaling the page and return the result to mobile device as a thumbnail. An example of this approach is the SmartView system [13]. SmartView creates an image map thumbnail of the page based on its semantic content. The user is able to zoom at the preferred block of the thumbnail and read the content of the web page. Another example of a browser of this category can be found in [9], where a mobile browser based on Open Source Software components is presented. The main problem with this approach is the size of the thumbnail: the screen size of mobile devices does not allow the text to be readable in many cases. Another approach tries to solve this problem by combining the previous method with text summarization techniques [10]. The main idea behind this approach is for the first level of the thumbnail to contain only some keywords from every text unit and not the full text of the web pages. When the user zooms at a specific area she/he is able to read the full text contained there. These methods are mainly intended for PDA users and generally for mobile devices with relatively large screens. Small mobile phone screens are not able to present a satisfactory view for the original thumbnail and as such they are not recommended to be used with the techniques mentioned above. The typical screen resolution of a mobile phone is between 128x128 and 176x220 pixels. A web page, with rich content, in a 1024x768 pixels resolution usually requires scrolling when using a 17 inch monitor. It is obvious that the thumbnail for the mobile phone will be so small that the user will not be able to read text, especially if he/she is a first time visitor (and thus unfamiliar with the web site content organization). Finally, using images obviously increases user satisfaction but requires significant computational and power resources from the mobile device.

Examples of techniques based on page reformatting include the Power Browser [4], where images and white space are removed, and the WEST browser [2], which uses flip zooming, a visualization technique that breaks pages into screen-sized tiles and presents them as a stack. Difficulties with recognizing layout and leveraging the desktop browsing experience are common to all these approaches, since they all have an impact on page layout.

This paper's approach presents a hybrid technique which is associated with the automatic re-formatting combined with RSS feeds with purpose to improve web browsing for mobile devices. The reasons that the above methodology has been selected are: (a) collaboration of page authors as specific and multi device authoring approaches demand cannot be provided in every case. (b) Online accessibility is desired instead of client based (client-side navigation) approaches (c) Device independence can be achieved instead of page scaling technique of page re-authoring approach.

4. RSS Feeds

RSS is a lightweight XML format which summarizes web site information. RSS can be found as an acronym for, Really Simple Syndication, resource description framework (RDF) site summary. The RSS format was created to facilitate "channels" on Netscape Net center, and was made available to the general public in March 1999. In 2001, Netscape, then part of AOL, drops development and support for RSS, however different open source and private companies continue to develop the application. As a result, different non-compatible versions appear. Currently, on the web one of the three different RSS versions; version 0.91 [19], 1.0 [20] and 2.0 [21] can be used. Version 2.0 is developed within the open source community and is the most flexible and expandable version. These versions do not have major differences, but developers should be careful if they want their applications to support all versions.
RSS solves a lot of problems webmasters commonly face, such as increasing traffic, and gathering and distributing news. RSS can also be the basis for additional content distribution services [5]. The real benefit of RSS is that it leverages the web's most valuable asset, content, and makes displaying high-quality relevant news easiest. With thousands of sites now RSS-enabled and more on the way RSS has become perhaps the most visible XML success story to date. For example, feedster.com (a search and indexing engine for RSS feeds) indexes over twenty million RSS feeds, including more than 75,000 professionally published sources such as the BBC, CNET, The New York Times, and Wired [7]. If a site has RSS feeds, some displaying colored XML or RSS feed pictograms will inform the user. Generally, the user needs special software to read and manage RSS feeds (aggregator). However, the most popular browsers, such Mozilla, Firefox, Internet Explorer and Opera have build in support for RSS

feeds. Furthermore, Microsoft has announced that Windows Vista will have full support of RSS feeds.

RSS feeds can also be delivered to mobile devices. There are two categories of mobile RSS readers: client side readers [18] and web based readers [14]. In client side readers, the user has to install the application to his/her mobile device. Then, the user can give the URL of the feed as an input in order to view the RSS content already adapted for his/her mobile device. The Web based readers usually provide two complementary perspectives. In the first one, accessible from a desktop pc, the user can add the RSS feeds that he/she is interested in. Then, at any time she/he can visit a specific link to access the selected feeds from his/her mobile device. Some other web based readers require additional software to be installed in the mobile device [12]. The disadvantages of the above RSS readers are that the users have to know the URL of an RSS feed in order to access them. Furthermore, if users follow a link from an RSS feed, the mobile browser is redirected to the corresponding web page (designed for a desktop pc).

Of course RSS is not the only method used to mark-up content. Semantic Web technologies have long ago proposed structure description languages and combinations (RDF, XHTML+CSS etc.). However, Web sites do not make extended use of such technologies because it is expensive (in effort) to mark-up content. Many view the Semantic Web as small isles of marked-up content. Especially in our case (news sites), the majority does use RSS feeds. And why should they? RSS gets the job done; hyperlinks are available using only a few lines of XML. With the above statement we do not want to reduce the importance of mark-up technologies. It should be realised though that practical reasons (especially costs) often favour simple solutions.

A RSS is a XML file anyway. So, every RSS file has the root element <rss> where the version of the RSS file is defined. The only child of the <rss> element is the element <channel>. The element <channel> may contain any number of elements <item>, the main element in a RSS file. Every element <item> always contains the tags <title>, <link> and <description>. Beside the above tags, there are other tags supporting more functionalities, depending on the RSS version. Figure 1 gives an overview of the RSS

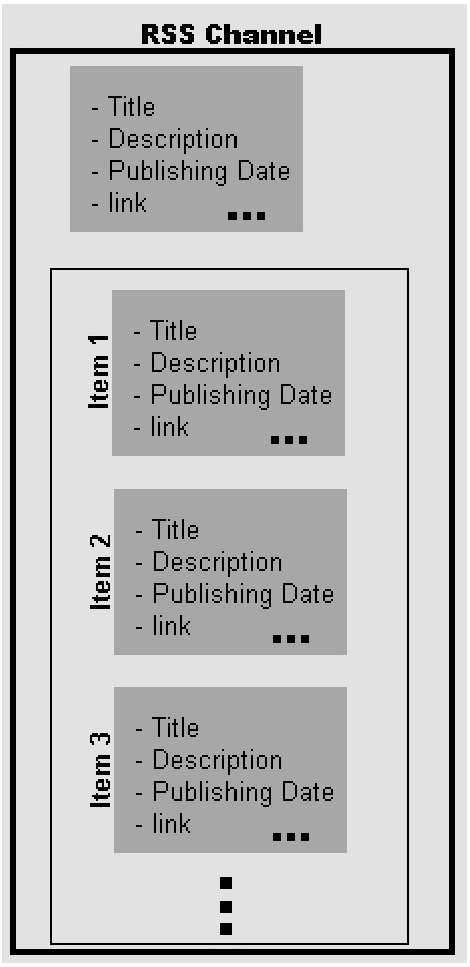


Figure 1

5. An RSS - based content adaptation technique

A standard way of processing web pages for viewing on small screen devices is through a proxy server that transforms pages on-the-fly. A proxy server is a program that receives web page requests (here from mobile devices), loads the respective pages, converts them, and serves them to the devices that requested them. In this way the proxy server, which usually runs on powerful servers, unleashes mobile devices from computational needs.

A brief description of our application follows [3]. If our application detects RSS feeds in a web page, the user is offered the choice of using the RSS feeds of the current page. If the user agrees and chooses one of the available RSS feeds, the system reads every <title>, <link> and <description> element from the feed, adds the application's menu and returns the XHTML – MP content to the user. If the page has no RSS feeds or the user does not want to use them, the proxy server adapts the web content for mobile devices, adds the application's menu and returns the web content to the user as described previously. The technique used to detect and exploit the RSS feeds is presented in the following section.

5.1 Implementation of the adaptation technique

There is no need for configuring the device prior to the use of the application. The user has to start the browser of the device and provide the URL of the server that hosts the proxy server application. The content that will be returned to the user is the main XHTML – MP page of our application, with a total size of 0.65 KB

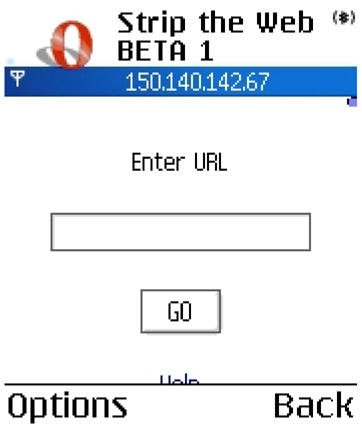


Figure 2

In the text box of the main page, the user provides the URL of the web site he/she desires to browse. This is the only part where the application needs text inputting. After this step, the user uses the device browser's interface. When the user hits the "GO" button, the proxy server creates a connection with the given URL, gets the HTML code, transforms it and returns the new code to the mobile device. Details like the "http://" string in the URL do not concern the user, since we want to achieve the minimum keyboard usage. Also, if the web server redirects the original URL (code 302 at HTTP response) the application finds and follows the new URL without disturbing the user.
The proxy server has now saved the html code of the web page that the user requested. In this point, a parsing algorithm removes the tags <script>, <noscript>, <style>, <link>, <iframe>, <object> and <embed>. The <script>, <noscript> tags (e.g. pop up windows) and <object>, <embed> (e.g. flash animations) are not supported by most mobile browsers. The algorithm also removes the comments from the html code. Comments do not affect the final result because browsers ignore them. If the final code includes comments, the

mobile user pays for useless data. Forms are also removed. This action decreases the usability of the application in many sites since it suspends the use of searching, login, online transactions and generally web applications that involve interaction of this type with the user. However, the purpose of the application is focused on cost effective browsing and not facilitating more complicated on-line transactions. Despite the fact that forms are very important our tool focuses on the text content of every web page. Furthermore, our application focuses on delivering to mobile users the content of a web page and not on implementing the functionalities of any web site for mobile devices.

The use of tables in the web is a common practice because they summarize content elegantly. However, in mobile devices the use of tables is a problem because of the lack of space on the screen. Our tool solves this problem by removing tables and presenting their content in a new line for every cell. If nested tables in a cell are used, the procedure is applied recursively. Similar techniques are applied from other mobile browsers too, because we want to avoid the horizontal scrolling.

For further processing of the code, we need some knowledge about its structure (for example if one tag is nested in another). Our proxy is implemented using PHP and DOM. The DOM interface of PHP follows the DOM Level 2 standard [6].

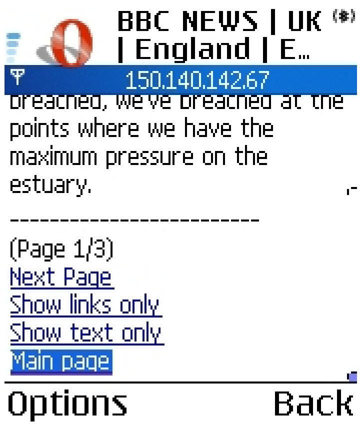
The next step is to remove the images from the web page. From the DOM tree, we locate all <img> tags by using the getElementsbyTagname PHP function. If the image is at the same time a hyperlink to another page, removal of the image is done in such a way that hyperlink information is not lost.

5.2 Managing hyperlinks

One of the design goals of our approach is also to ensure transparency. When a page has been served to the user, she/he can browse to new pages by following the hyperlinks or by giving a new URL to the main page of the application. So, the destination of each hyperlink in the transformed web page must be changed in a way that requests are redirected through the proxy. For example, if a hyperlink points at http://www.springer.com the new hyperlink must be change to the following format: http://proxy_address/main.php?url=http://www.springer.com. Of course, a hyperlink may point to things other than web sites. For example, a hyperlink may point to a file that is hosted at the same server using a related path such as <a href = ”../test.html”>Test page</a>. In this case, the hyperlink points to the file test.html which is located at the directory which is a level higher from the current one. The proxy server that we have developed checks all those cases.

5.3 Creating sub pages

When the parsing algorithm has finished with the content adaptation process, it checks the size of the new page. If the size is bigger than 10KB then it sends, through the proxy, the content in sub-pages with size of about 10KB. If the point where the page has to split is in the middle of a paragraph, the splitting point is moved so that the whole paragraph appears in the same sub-page. The 10KB limit may seem small for new mobile devices however, with this limit we are sure that there will be no memory problems with any device that supports WAP 2.0. One other reason for creating sub-pages is that larger pages may lead to longer response times. Furthermore, a page may contain content which is useless to the user or may be an intermediate page for another destination. In this case, the user may ignore this page having paid only for the 10 KB of the first sub-page. However, even with the use of sub pages the user cannot avoid vertical scrolling for web pages with reach content. The 10 KB limit can be easily changed or cancelled by the administrator of the proxy server.
At the end of each page or sub-page a menu is added that allows access to the other sub-pages that may exist or returns to the main page of the application to view only the links or only the text of a page.


BBC NEWS | UK | England | E...
150.140.142.67
breached, we've breached at the points where we have the maximum pressure on the estuary.

(Page 1/3)
Next Page
Show links only
Show text only
Main page

Options Back

Figure 3

These functionalities are described in detail later in this section. The menu is at the end of each page but is quickly accessible from the user by pressing the 0 key in the keyboard. This function is implemented using the attribute accesskey at the tag <a>.

When the user is navigating in a site she/he usually browses through hyperlinks ignoring most of the content, until the proper information is found. To speed up this process the application provides the user with the option to view only the hyperlinks of a page. Similarly, the user has the option to view only the text of a web page. This functionality may be useful if a page contains for example, an article. Therefore, text is the most important part in this case. Of course, the user may switch from one mode to another or return to the original viewing mode.

If the user requests a page that includes RSS feeds then the application does not present the content of the page immediately but recommends the user to browse the site through the information that the RSS feed contains. By default, the RSS feed includes all the latest and interesting information that the site contains. In order to discover if a page contains RSS feeds, the parser checks all the <link> tags before they are removed from the content adaptation process that we described previously. If the attribute type of the tag <link> has the value "application/rss+xml", then the attribute href of the same tag contains the location of the RSS feed's XML file. The parser returns the title of the feed as it is defined by the attribute title of the tag <link>. If a page contains more than one RSS feeds, the parser returns all the feeds in the order that they are detected in the web page code. The application recommends the user to continue browsing through the available RSS feeds. However, the user can choose to browse the site through the process that we have described previously.

An alternative approach to serving chunk of content would be to serve content hierarchically, for example we can create a list with the headings. Despite the fact that HTML provides the tags for creating headings, web designers use also other techniques to form the text such as CSS style sheets. So, we cannot rely on this technique for every web site.

5.4 Handling RSS feeds

If the user selects to browse based on RSS feeds as recommended, an XML parser is used to read the RSS feed. For the implementation of the RSS parser we use the xml_parse_create function from the XML library of PHP. The XML parser works with all three versions of RSS feeds, which are versions 0.91, 1.0 and 2.0. From every RSS feed, the application returns the title, the description and the link for every element <item> of the feed. Just like in every page, users may only view the links or the text of the page.

We have developed our server using PHP (php.net). The only requirements for the application is a web server with PHP 5 support or newer. All the functions that have been used are included to the default installation of PHP. For our tests we have used the Apache 1.3.33 as a web server and the PHP 5.0.4.

From the client side, the only requirement is a WAP 2.0 compatible browser, a browser that most of the devices today have. The pages of the application are XHTML-MP valid, so we are sure that the content the proxy server sends is presented correctly in any category of mobile devices.

6. Evaluation Method

Quality's evaluation of mobile applications as long as the end user's requirements are concerned, gives emphasis on the systems' usability and in the level of user's satisfaction [26]. It is within this framework that the end users, who are also the customers of a mobile network, will specify the relative significance for each function that the mobile system provides to the user.

The end-user (abstracted through a correct sample, sufficiently large and diverse) is the best judge (when properly questioned) of the degree to which their needs are satisfied, and thus of the quality of the system viewed as service [22]. However, the end user must be questioned about very specific aspects because otherwise:

- their opinion can be uninformed and biased. For example each end user has different perception of mobile usability based on his/her previous experience of mobile systems,
- we do not obtain specific information about what exactly is good and what is not in the system (the use of the appropriate measurement scale supports end users evaluation results).

For the presented study, inquiry methods were selected. Inquiry methods are methods that require user participation but not a laboratory. These methods are implemented after the users have used the mobile systems in their own environment under real use conditions. Examples of such methods are user interviews, focus groups and surveys. The user interviews are based on the direct contact between the evaluator and each user and request that the users' opinion of the system is recorded in detail. The focus group method is implemented in a similar manner and involves a small group of five to ten users who interact with an expert usability evaluator for evaluating the system. The surveys aim at recording the user's opinion of the web system and measuring his/her opinion from interacting with it. In this evaluation study we used user interviews and surveys as well. We used these methods in order to ensure the validity of our results. User interviews offer information about users' behavior during the evaluation process and give significant feedback of users' comments and remarks. On the contrary, surveys using questionnaires offer results for specific features of mobile browsing. For example, features such as tool's navigability, response time, number of links, mobile browsing process and content's presentation are evaluated from the user.

The respondents mix includes 20 expert internet users, 23 – 39 years old, 8 of them female [24]. We have selected expert Internet users and occasional mobile Internet users with familiarity of internet services, like search, navigation map and RSS. This variation enables the breadth in data collection and provides multiple interpretations about the system's quality.

The users were asked to assess specific information sites during each mobile session. As a mobile session for each user we define an evaluation scenario where the users were asked to find specific articles of world news. Because of the 'nature' of RSS application where the most recent information is included, we selected news titles from recent stories and top stories as well. So, the users were asked to find the easiest, in their opinion, way to the information. For the evaluation process we have used the Nokia N 70 mobile phone. The N70 has a screen with resolution 176 x 208 pixels and supports 262.144 colors. The phone can also connect to 3G networks for high rate data transfers using the Opera Mobile 8.51 browser. In order to avoid operability issues for the Nokia N70, help about the functionalities of the device was provided during the evaluation process. During each evaluation session, two different navigation procedures were selected from the users in order to reach specific information: (a) The users navigate through the Opera browser (b) the users use the RSS application. During each session each user was asked to find one recent article from three popular news sites (cnn.com, news.bbc.co.uk and guardian.co.uk) using two different ways. As recent we define an article that can be found in the main page of the site and in the

RSS feed as well. Each user visited each of the above web sites at least once from his/her PC. The users did not browse each site but they viewed a snapshot of the first page in order to evaluate the re-formatting of the web content.
Using the Opera Mobile Browser, each user typed the URL of the web site to the mobile phone and tried to find the information that we asked for using only the browser's interface.

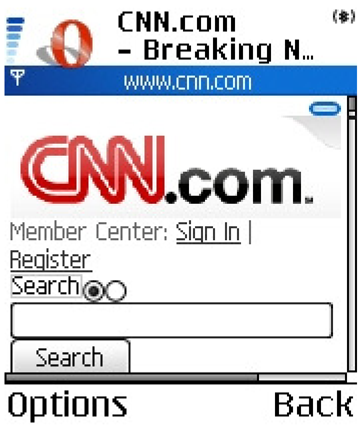


Figure 4

This first part of the session had a time limit of 3 minutes. This was not announced to the user in order not to have any time pressure. In the second part of the evaluation session, we asked the same user to find the information by using our application.

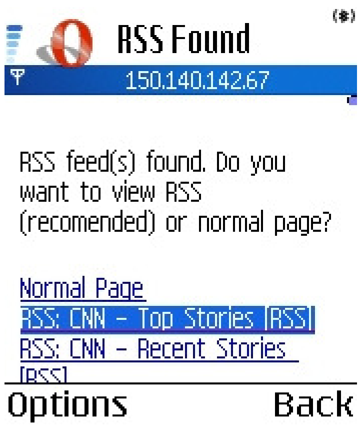

RSS Found
150.140.142.67
RSS feed(s) found. Do you want to view RSS (recomended) or normal page?
Normal Page
RSS: CNN - Top Stories [RSS]
RSS: CNN - Recent Stories [RSS]
Options
Back

Figure 5

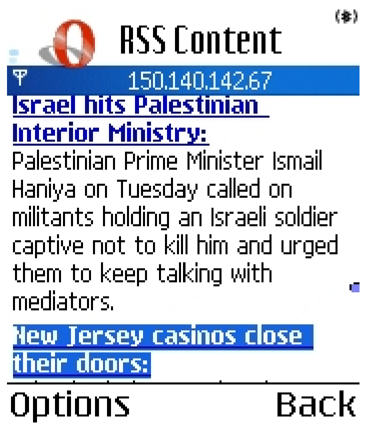

RSS Content
150.140.142.67
Israel hits Palestinian Interior Ministry:
Palestinian Prime Minister Ismail Haniya on Tuesday called on militants holding an Israeli soldier captive not to kill him and urged them to keep talking with mediators.
New Jersey casinos close their doors:

Figure 6

Again there was a time limit of 3 minutes.

By the end of this evaluation session the user was asked to assign a value using a five-grade Liker-type scale of (1 = not important, 2 = important, 3 = average important, 4 = very important, 5 = critical) to specific features of the mobile session. The user answered to specific questions and selected one answer for each question. At the end of the session the users discussed with the evaluators their impressions and expectations of the application.

Final editing of the results has been conducted from the writers of the research. The issues that this quality evaluation study examined were related with the navigability and the accessibility of the mobile system, the time behavior, content adaptation issues and RSS functionality.

7. Evaluation Results

7.1 Using the Opera Mobile Browser

The time limit of three minutes was set for each user to find the specified article in each site, including the response and data transfer time. 50% of our users could not reach the information within the time limit. One of the main reasons is that the users didn’t scroll enough to see the whole content of the page (a hyperlink for the article could be found at the main page) or they tried to use the search textbox of the sites so it took them enough time to type a proper query. This time delay, which depends on the specific device keyboard, affects the total mobile session time. 70% of the users complained about the response time even though we had a 3G connection (download speed up to 384 kbps).

7.2 Using the RSS proxy based system

We have again the 3 minutes limit for this part of the mobile session. None of our users went over this time limit. When the user typed the URL of the target-site to the text box of the main page (figure 1), our system informed the user that the site contained RSS feeds and proposed the user to use it for further browsing. 80% of the users agreed with this proposal and used the RSS feeds and they managed to find the article. If the site contained two or more RSS feeds (like cnn.com) the users chose a feed. When they could not find the specific information to at the first RSS feed, they tried to view the second feed. Some users just pushed the "Back" button of the browser and choose the other feed, but some other users (a 30%) clicked on the "Main page" link (figure 3). As a result they were transferred to the main page of the application. So, in the future we have to implement a simple way for the user to switch between the different RSS feeds of the same site. The other 20% chose to view the normal page but they managed to find the article also. The 90% of the users where very pleased with the response time of our system. This was expected because the max size of each page send to the user's mobile was only 10 KB.

One other interesting result of our experiments is that 70% of our users answered that they do not want to view images through our system. This percentage must be viewed in combination with the characterization of our system as 'useful' by the 70% of the users too [table1]. So, we confirm that the mobile users do not browse the web in the same way as desktop users and that the high priority for them is the access of the right information with low cost. At this point we have to mention that the cost of mobile browsing is not flat rate. The user pays for every KB that she/he downloads and they had this in mind when they answered that question. Finally, we asked the users how many links they would like to see in a RSS file for the best viewing experience from a mobile device [table2]. The 60% answered from 5 to 10, a 30% answered from 10 to 15 and 10% answered a maximum of 5. Then, we studied twenty RSS feeds from some popular, according to Alexa.com, news sites. 80% of those RSS feeds have more than 10 links in a single RSS file, 35% have 10 to 15, 10% have 15-20 and 35% have over 20 links. So in order to comply with user's expectations in our future work will present only the information of the <link> tag from the RSS feed and not the description only for large RSS feeds.

During the evaluation process we measured the transferred data to the mobile device for each mobile session. When the users browse through the Opera browser we had an average of 643KB for CNN, 389 KB for BBC and 387 KB of transferred data for Guardian. If the users used the RSS feeds application we measured 22.4 KB, 28.7 KB and 21.8 KB respectively. Finally, when the users did not use the RSS feeds we measured 30.5 KB, 27.1 KB and 29.2 KB of transferred data for each experiment. We notice that in the BBC case the amount of data when we used RSS feeds is larger than when we did not used it. This happens because the RSS file of BBC is too big compared to the other two sites, since it contains 34 links. RSS is usable due to its simplicity (5-10 links); Large RSS feeds are similar to large web pages with many hyperlinks; useful but confusing for general use.

When we asked the general opinion of the users about the RSS application they characterized it as useful (70%), stable (20%) for browsing the web and usable (10%). The users had to choose only one answer in order to characterize the system as "usable", "fair", "reliable", "stable" or "useful".

Finally, we did not emphasize at time measurements for two reasons. First, we were more interested to see how the users evaluate our tool, and second in a mobile application, time is a parameter that depends on many factors, such as the network load, and it does not depend exclusively on the design of the application.

8. Conclusions

In this paper we presented a method that allows cost-effective web browsing for users of mobile devices. This method exploits mainly the existence of RSS feeds, metadata that summarize information. We concluded that the existence of RSS feeds improves significantly the content presented to the mobile user decreasing at the same time the size and access cost especially in news and information sites. Furthermore a quality evaluation process is presented based on surveys and user interviews. The conclusions that were drowned from our surveys represent users' perception of mobile systems quality and provide the user oriented character of the RSS application. Although, using surveys and user interviews is a representative evaluation tool further improvement of the application is needed based on user's preferences. This study employed a quality evaluation to validate mobile applications; future quality evaluation studies on the topic will have to extend the reliability and validity of the findings.

We consider that this survey gives only preliminary results, indications on the usefulness of our approach. Further work is needed on the issue of using web shortcuts as mobile shortcuts; in any case our survey showed that we achieved an economy of resources: we used already marked-up content intended for web use to facilitate mobile navigation. Specific guidelines were followed, however we did make our own assumptions were necessary. One assumption was to remove various tags, which reflects the importance of combining RSS feeds with more sophisticated content adaptation algorithms.

Our tool is not suitable for accessing on-line transaction systems since forms are omitted. This was not in our intention anyway as we targeted news sites. However, mobile commerce is too big of a market to be ignored. Our future work will target this area also.

9. References

[1] Bickmore, T., and Schilit, B., Digestor: Device-Independent Access to the World Wide Web. In Proc. Seventh Intl. WWW Conference 1997, pp. 655–663.

[2] Bjork, S., Bretan, I., Danielsson, R., and Karlgren, J. WEST: A Web Browser for Small Terminals. In Proc. UIST'99, pp. 187-196.

[3] Blekas, A., Garofalakis, J., and Stefanis, V., Use of RSS feeds for Content Adaptation in Mobile Web Browsing, International Cross-Disciplinary Workshop on Web Accessibility, WWW2006, pp. 79 – 85.

[4] Buyukkokten, O., Gracia-Molina, H., Paepcke, and Winograd, T. Power Browser: Efficient Web Browsing for PDAs. In Proc. CHI 2000, pp. 430-437.

[5] Curran, K., McKinney, S., Scheduled RSS feeds for streaming multimedia to the desktop using RSS enclosures, Information Management & Computer Security, Volume 14, Number 1, 2006, pp. 65-74.

[6] Document Object Model Specifications, http://www.w3.org/DOM/DOMTR.

[7] Feedster.com statistics, http://feedster.blogs.com/corporate/overview/index.html.

[8] Ghinea, G., Angelides, M., A User Perspective of Quality of Service in m-Commerce, Multimedia Tools and Applications, Volume 22, Issue 2, 2004, Pages: 187 – 206

[9] Grassel, G., Geisler, R., Vartiainen, E., Chauhan, D., and Popescu, A., The Nokia Open Source Browser, at MobEA IV, WWW2006.

[10] Heidi, L. and Baudisch, P. Summary thumbnails: readable overviews for small screen web browsers. In Proc. SIGCHI 2005, pp. 681 – 690.

[11] Kaikkonen, A., and Roto, V., Navigating in a mobile XHTML application. In Proc. SIGCHI 2003, pp. 329 – 336.

[12] Lite Feeds. http://www.litefeeds.com

[13] Milic-Frayling, N. and Sommerer, R. Smartview: Enhanced document viewer for mobile devices. Technical Report MSR-TR-2002-114, Microsoft Research, Cambridge, UK, November 2002.

[14] Mobile RSS – RSS news for the mobile user. http://www.mobilerss.net

[15] Mobile Web Best Practices 1.0, Basic Guidelines, W3C Candidate Recommendation 27 June 2006, http://www.w3.org/TR/2006/CR-mobile-bp-20060627/

[16] Mobile Web Initiative, http://www.w3.org/Mobile/

[17] Opera Mobile Browser, http://www.opera.com/products/mobile/

[18] RSS Reader MIDlet. http://www.brothas.net/rssr/

[19] RSS version 0.91 Specifications, http://my.netscape.com/publish/formats/rss-spec-0.91.html.

[20] RSS version 1.0 Specifications, http://web.resource.org/rss/1.0/spec.

[21] RSS version 2.0 Specifications, http://blogs.law.harvard.edu/tech/rss.

[22] Stefani A., Stavrinoudis D., Xenos M., "In-Depth Analysis of Selected Topics Related to the Quality Assessment of E-Commerce Systems", Proceedings of the 2nd IEEE International Conference on E-Business and Telecommunication Networks, ICETE-2005, Reading UK, Vol. 1, pp. 122-128, October 3-7, 2005.

[23] T9.com http://www.t9.com.

[24] Understanding Mobile Human – Computer Interaction, S. Love, Elsevier, 2005

[25] Web Accessibility Initiative, http://www.w3.org/WAI/

[26] Xenos M., Technical Issues Related to IT Governance Tactics: Product Metrics, Measurementsnd Process Control, In Strategies for Information Technology Governance, Wim Van Grembergen (Ed.), Idea Group Publishing, ISBN 1-59140-140-2, Chapter IX, pp. 216-244, 2003.

[27] XHTML Guidelines For Creating Web Content, 2005, Nokia Corporation.

[28] XHTML Mobile Profile and CSS Reference, 2003, Openwave Systems

Figure 1: The RSS Channel Structure

Figure 2: Main page of the application

Figure 3: Application's menu

Figure 4: Main page of CNN from the Opera Mobile Browser

Figure 5: Proposing using the RSS feeds

Figure 6: Reading an RSS feed from our application

Table 1. System's characterization.

Characterization	Usable	Fair	Reliable	Stable	Useful
Percentage of users	10%			20%	70%

Table 2. Comparing the number of links of RSS feeds.

	Number of links to RSS feeds (y)

	y<5	5≤y<10	10≤y<15	15≤y<20	20≤y
Users preference	10%	60%	30%		
20 popular news sites	5%	15%	35%	10%	35%