Georg-August-Universität
Göttingen

Faculty of Economic Sciences
Institute of Information Systems
Chair of Application Systems and E-Business
Prof. Dr. Matthias Schumann

Working Paper 01/2011

Ed.: Matthias Schumann

Rasmus Voigts, Stefan Christmann
and Svenja Hagenhoff

Mobile Web Browsers

Arbeitsbericht
des Instituts für Wirtschaftsinformatik
Professur für Anwendungssysteme und E-Business
Georg-August-Universität Göttingen
Platz der Göttinger Sieben 5
37073 Göttingen

Working Paper
Institute of Information Systems
Chair of Application Systems and E-Business
University of Goettingen
Platz der Goettinger Sieben 5
37073 Goettingen, Germany

Tel. +49 (0) 551 / 39-4442
Fax +49 (0) 551 / 39-9735
www.as.wiwi.uni-goettingen.de
as@uni-goettingen.de

Table of Contents

Abstract

Accessing the World Wide Web via mobile end devices is increasing steadily. Mobile access is limited by device properties and radio networks. In the past, software developers responded to this by special standards for a special mobile web: WAP, HDML, WML, C-HTML and XHTML-MP. However, the World Wide Web Consortium follows the so-called One-Web approach and recommends using the general World Wide Web for all types of end devices. The browser plays a central role in this: the use of pages and applications on the World Wide Web can only be achieved if the display software supports up-to-date web standards. Consequently, mobile web browsers present the main bottleneck for mobile use of the World Wide Web. Their characteristics will therefore be examined systematically. The study shows that the current mobile end devices' web browsers facilitate mobile use of the general WWW, considering the attributes of the mobile operational environment. Specialised standards can therefore be omitted.

List of Figures

List of Tables

List of Abbreviations

AJAX	Asynchronous JavaScript and XML
API	Application Programming Interface
ARPU	Average Revenue Per User
BMP	Bitmap
C-HTML	Compact HyperText Markup Language
CSS	Cascading Stylesheets
DOM	Document Object Model
GIF	Graphics Interchange Format
HDML	Handheld Device Markup Language
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure
IP	Internet Protocol
JPEG	Joint Photographic Experts Group
JRE	Java Runtime Environment
LBS	Location-Based Services
NPAPI	Netscape Plugin Application Programming Interface
PC	Personal Computer
PNG	Portable Network Graphics
RIA	Rich Internet Application
RIM	Research In Motion
SOAP	-, earlier: Simple Object Access Protocol
SVG	Scalable Vector Graphics
SSR	Small Screen Rendering
TCP	Transmission Control Protocol
TLD	Top Level Domain
UMTS	Universal Mobile Telecommunications System
UAProf	User Agent Profile
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
WAP	Wireless Application Protocol
WML	Wireless Markup Language
WURFL	Wireless Universal Resource File

WWW	World Wide Web
XHTML	eXtensible Hypertext Markup Language
XHTML-MP	eXtensible Hypertext Markup Language – Mobile Profile
XML	eXtensible Markup Language

1 Introduction

The mobile use of the internet and especially the World Wide Web (WWW, short: web) as its most important service is increasing steadily. Statistics of the browser producer Opera show a growth in mobile web use of 224% between 2008 and 2009 (Tetzchner 2009; Zhang 2007). This growth is due to a drop in prices of mobile data services and the fact that more broadband mobile net accesses are available. Examples are the Universal Mobile Telecommunication System (UMTS), as well as mobile phones and smartphones with steadily increasing technical resources. Further continual development may be expected due to the related market potential, as mobile phones have a high diffusion rate in industrialised countries (e.g. Italy 123%, Germany 102%, Japan 80%, USA 77%; Central Intelligence Agency 2009). Mobile phones and similar mobile end devices are used on a daily basis and usually carried around ready for use. Many people are accustomed to using them, thus using the respective end devices is easy. More and more end devices are on the market that can be used comfortably, especially with touch-screen, which facilitates the use for less technophile target groups.

However, mobile end devices – not including notebooks as PC-equivalent mobile end devices – are clearly limited in comparison to desktop-PCs. Thus, the use of the World Wide Webs is also limited. Limitations are:

- **Display** (I): The screens of mobile end devices are usually 70% smaller than those of desktop-PCs (Bieh 2008). Whereas websites are generally designed for at least 800x600 pixels, the display of mobile end devices mostly only reaches 320x240 pixels (Bieh 2008).

- **Input** (II): Mobile end devices are more difficult to operate and only a few have complete keyboards. Most have only numerical keypads, supplemented with a few additional keys or touch-screens. Unlike desktop-PCs, no mouse or trackball is available. However, newer end devices now use intuitive modes of input, such as turning, tilting or shaking the device (Linjama et al. 2008; Crossan et al. 2008).

- **Resources** (III): Mobile networks are less powerful than wired networks or wireless computer nets. Furthermore, the life-span of accumulators is limited, so that end devices can only be used for a certain period of time, depending on utilisation. In addition, the lower calculating and working capacity affects the executable software, such as a web browser (Capin, Pulli and Akenine-Möller 2008).

These limitations show that the mobile use of the World Wide Web can not be similar to the stationary one. The question arises in which ways usage differs, which part of the WWW is suitable for mobile use and how the limitations mentioned before can be countered. These aspects will be discussed in the following article, beginning with a detailed description of the mobile web in the following chapter.

2 The Mobile Web

The mobile web is a service of the mobile internet. It can be defined as the usage of internet protocols and services via wireless networks. Four components are necessary for implementation: a basic infrastructure offering services (network level, 1), a mobile network for transfer (transmission level, 2), a mobile end device (device level, 3) and an application (application level, 4, see figure 1).

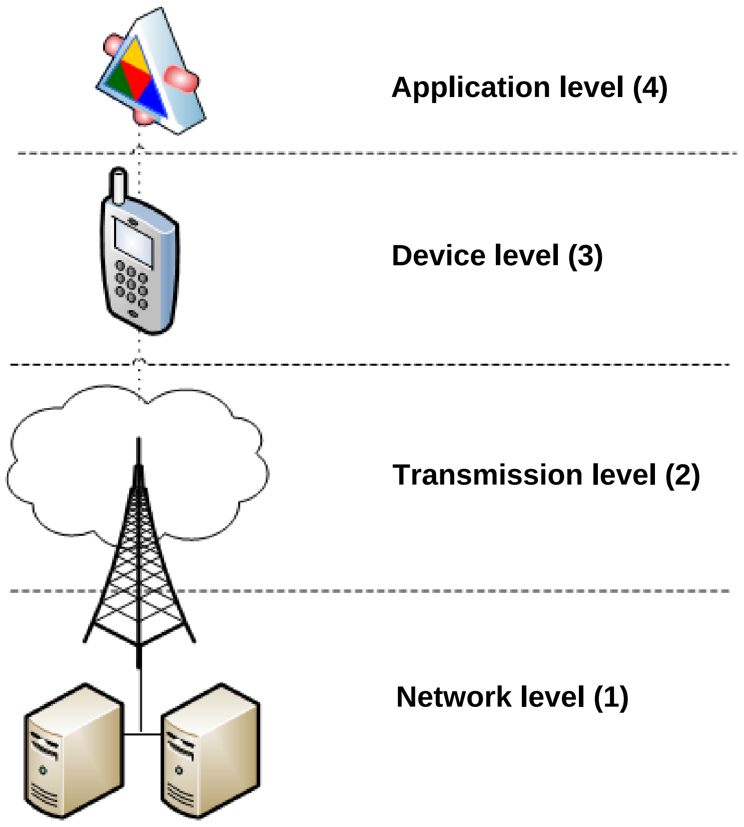

Figure 1: Architecture of the mobile internet

The network level (1) is that part of the internet which represents the server-side in the client-server-model. It can be identical with the server systems of the stationary internet, or it can offer specific services and protocols for the mobile internet. This component of the mobile internet is well-tried and fully developed. The transmission level (2) corresponds to the data transmission of the network level to mobile end devices and vice versa. It is concretised through standards of mobile communication, which allow transfer of data. This layer also poses little challenge to the mobile internet, as broadband wireless computer networks are more and more widespread. The multitude of mobile end devices found on the device level (3) will be limited to the group of handhelds in the following, as these are generally carried around ready for use (Roth 2005). The device level shows a high degree of heterogeneity, especially through a multitude of input technologies. With regard to the mobile web, this heterogeneity can be solved by operating systems for mobile end devices: they function as an abstraction-layer between hardware and application software. The application level (4) represents software products that use wireless internet access. A mobile web browser is used in the mobile web.

Web browsers are end user programmes that display (X)HTML-based content (Miller, Vandome and McBrewster 2009) and thus content of the World Wide Web (see Klau 1995, p. 275). From a technical perspective, these application softwares transfer (X)HTML files, as well as linked media files, such as pictures or videos via the HTTP-protocol. They also display these files with the help of stylesheets such as CSS (“rendering”) and enable loading of plug-ins (e.g. flash-video, Java applets), as well as the execution of client-side scripts (such as JavaScript applications). Meanwhile, their functions exceed the original design of the WWW’s inventor, Tim Berners-Lee by far (Berners-Lee, Fielding and Frystyk 1993; Berners-Lee and Cailliau 1994). The most important part of a browser is the rendering-engine, which carries out the interpretation and depiction of the (X)HTML-source code. It can be either producer specific or open-coded and thus be used in more than one browser (e.g. Gecko or KHTML). Browsers are generally used through menus and toolbars. They often have additional functions such as the management of favourite sites, which can often be enhanced by plugins (Williams and Tollett 2004).

The web browser’s functions determine how well and to what degree the mobile web can be used. In principle, two approaches can be discerned: the Multiple Web-Approach and the One Web-Approach. Both will be presented in the following.

2.1 Multiple Webs-Approach

The Multiple Web-Approach splits the World Wide Web into pages that can be displayed by mobile end devices and pages that can not. According to this definition, the mobile web is that part of the WWW that can be used with mobile end devices (Alby 2008; see Figure 2). Special standards are applied with this definition in order to implement the mobile web. The WAP-Forum already began developing the Wireless Application Protocol (WAP), which enables the use of special web content on limited mobile end devices, in 1997. In the beginning, the Handheld Device Markup Language (HDML) was used for creating such content, whereas later the Wireless Markup Language (WML; Bieh 2008; Hong, Jun and Kwak 2006) was used.

WML used so called “cards” that were combined in “decks” the user could flip through. WML is based on the meta-language XML. It has, however, no relation to the description language XHTML primarily used in the stationary web.

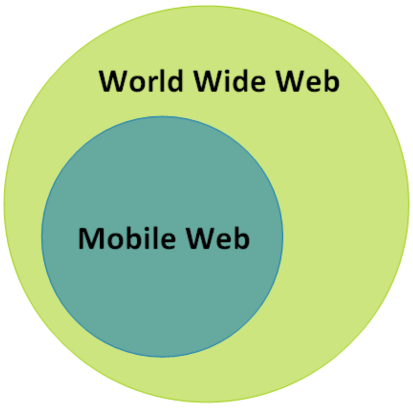

Figure 2: The mobile web as part of the WWW

It used a simple scripting language based on JavaScript – WMLScript – for dynamisation on the client-side (Jamsa 2001). However, the Japanese i-Mode-Standard uses an HTML-sub-set instead of WML: Compact HTML (C-HTML). In the last step, XHTML was also applied to mobile end devices. XHTML, a XHTML-based advancement of HTML, provides a reduced version called XHTML-Basic. A version suitable for mobile use was in turn derived from it: XHTML-Mobile Profile (XHTML-MP, Kaikkonen and Roto 2002).

The Multiple Web-Approach solves the specific problem of mobile end devices by providing reduced versions of services and protocols. Furthermore, the approach of dedicating certain areas of the WWW to mobile use exists. Therefore only web providers whose content meets the requirements of mobile end devices receive a .mobi domain. Users of web sites in the .mobi domain can thus be sure that the web pages behind such addresses are actually suitable for mobile use (Bieh 2008).

The central problem of the Multiple Webs-approach is user acceptance. Users are usually not willing to use a restricted web, they want the entire range of available information and functions (Bieh 2008).

2.2 One Web-Approach

The One Web-Approach is the direct opposite of the Multiple Webs-approach. It postulates that the entire web should be used by different end devices, whereas optimisation for specific end device groups, such as handhelds, is acceptable (Bieh 2008). Thus, the mobile web is identical with the stationary WWW, with different end devices being used. It is therefore defined as the usage of the WWW by mobile end devices via wireless communication networks. The One Web-approach was defined by the World Wide Web Consortium (W3C) in the context of its Mobile Web Initiative (W3C 2010). Adaptation of pages is realised especially through the Standard User Agent Profile (UAProf) or the Wireless Universal Resource File (WURFL). Servers try to identify the mobile end device as accurately as possible and to transfer adapted web pages to it.

The advantage of using one World Wide Web and of adapting via identification of the mobile end devices is the WWW's diversity in information and functions. Mobile surfing becomes more attractive as the usability widens. However, the adaptation of web pages to the multitude of end devices, which is sometimes necessary, poses a great challenge to web developers. However, many different ways of optimising on client and server-side have been developed (Christmann et al. 2009), such as content adaptation (Zhang 2007; Brusilovsky/Maybury 2002; Buyukkokten/Garcia-Molina/Paepcke 2001) or server-side adaptations to the amount of data transferred (Bharadvaj/Joshi/Auephanwiriyakul 1998).

2.3 The Keyrole of the Web Browser

The comparison of the two approaches shows that the One Web-approach is more appealing to the user and that the Multiple Webs-approach was only developed because of the limitations mentioned in chapter 1. The question arises whether today's technology renders it possible to utilise almost all websites, considering the mobile web's characteristics and the limitations of mobile end devices.

As described in chapter 2, components on four levels are necessary for the mobile web: the server infrastructure, the radio network, mobile end devices and a mobile browser. In the One Web-Approach the server-side is identical to the stationary internet, provided no specific adaptations are made to end devices (e.g. different applications for different display sizes or input possibilities). However, this area will not be discussed further as only very few pages are optimised in such a way, and this is not likely to change in the future. On the next two levels, there are both broadband networks, such as UMTS, and high-performance end devices, such as the iPhone (600 Mhz-processor, 256 MB RAM; in model "3GS"). Thus, these levels do not pose great problems. Focus is therefore laid on how the software deals with different end devices and fluctuations in the network connection.

The last component to be examined is the browser. It must fulfil various requirements in order to be able to display as many websites as possible. Ideally it should also be optimised to mobile usage. As there are hardly any scientific studies on web browsers for mobile end devices, the range of functions of these application systems is insufficiently covered. However, the web browser is the central component in using the mobile web. It is the link to the user; its abilities define the possibilities of displaying web pages, and it can also compensate for limitations imposed by the other components. Thus, the web browser is the bottleneck in mobile WWW-usage.

The question therefore arises how efficient mobile web browsers are and whether they facilitate the use of the general WWW. In the following paragraphs requirements of web browsers for mobile end devices will be defined and the four browser systems dominating the market will be analysed accordingly.

3 Requirements on Mobile Web Browsers

Mobile web browsers must fulfil different requirements, which will be presented here using the layer model described in chapter 2, starting at the top. The possibility of accessing the stationary internet exists in both layers (1) and (2), due to the UMTS-mobile communications standard. Websites originally designed for the stationary internet can be accessed by mobile end devices, because the data transfer technologies of the UMTS-standard support the utilisation of HTTP and TCP/IP protocols of the stationary internet. In order for the users to be able to use these websites, mobile web browsers have to support the display and reproduction of websites and multimedia content of the WWW just as widely as stationary browsers. This is shown in table 1 as requirement (a), which is the main function of web browsers.

On level (3) of the mobile communication model (device level) requirements on mobile web browsers can be deduced from the heterogeneity of mobile end devices. The multitude of input and output possibilities have to be supported by mobile web browsers in order to guarantee user-friendliness. This is shown in table 1 as requirement (b).

The characteristics of the mobile internet have to be taken into consideration when using mobile data connections. These characteristics include, for example, the limits to data transmission. Reduced bandwidth and possible connection problems in insufficiently covered areas such as tunnels have to be anticipated. A mobile web browser should therefore minimise the effects of these limitations. This is shown in table 1 as requirement (c). The resulting requirements are presented in table 1.

Table 1: Abstract requirements on mobile web browsers

Requirement (a) *The correct display and reproduction of web sites and multimedia contents of the WWW.*
Requirement (b) *The support and consideration of different input and output modes in a user-friendly form.*
Requirement (c) *Considering the characteristics of the mobile data connection in the form of available bandwidth and possible connection failures.*

The next section presents these requirements in more detail, together with possible solutions. For the sake of clarity, these are presented in a morphological box. Whether and how well these requirements are met will be discussed later.

3.1 Display and Reproduction of Web Content (a)

In order to guarantee the correct display of websites, the technologies needed have to be supported by a mobile web browser. The technologies mainly used for displaying websites are (X)HTML (content), CSS (formatting) and JavaScript (client-side adaptation; Flanagan 2006). In terms of the One Web-approach (see chapter 2.2) the standards of the respective technologies of the stationary WWW should also be supported by mobile web browsers. It can also be concluded that the standards of the used basis technologies of Rich Internet Applications (RIA) also have to be supported. These technologies, integrated as plug-ins or JavaScript-classes, enable user interfaces and are equal to those of PC-applications (Xiao et al. 2009; Paulson 2005; Garrett 2005). By supporting the standards of RIA-technologies such elaborate web applications should be correctly displayed on mobile end devices, and also be usable.

In order to develop clear assessment criteria, the technology standards to be supported for the correct display of web applications have to be defined. A comparison of web browsers used on the stationary internet is suitable for this. Technology standards can be derived from these properties, which have to be supported by a mobile web browser. Market shares indicate how frequently a stationary web browser is used. These can be determined by analysing different sources (Webhits 2010; Browserstatistik 2010). Even though information is differentiated, one can see that the following browsers are leading in the market of the stationary WWW: Microsoft Internet Explorer, Mozilla Firefox, and Apple Safari. The relevant standards are presented in table 2.

Table 2: Supported standards of stationary webbrowsers

Browser Technology	**Mozilla Firefox 3.5.3**	**MS Internet Explorer 8.0**	**Apple Safari 4.0.3**
Standards for the display of web applications			
(X)HTML for structuring	HTML 4.01, parts of HTML 5, XHTML 1.0 / 1.1		
CSS for formatting	CSS 1 / 2.1, parts of CSS 3		
JavaScript for dynamisation	JavaScript 1.8	JavaScript 1.5	JavaScript 1.7
Additional standards for displaying RIAs			
Plug-in interface	NPAPI	ActiveX	NPAPI
AJAX	XML 1.0, DOM 1/2, parts of DOM 3, XMLHttpRequest, SOAP 1.1		XML 1.0, DOM 1/2, parts of DOM 3, XMLHttpRequest, SOAP 1.2

The literature (Hammond 2009) and the comparison of existing web standards of the W3C, as well as the specifications given by the producers, show that hardly any web standard is completely supported by the browsers listed. According to Hammond a standard is partly supported if less than 50% of all defined properties are correctly executed. A standard is supported if more than 50% are correctly executed (Hammond 2009). This paper uses this classification for assessing the support of web standards by mobile web browsers.

By comparing the versions or variation of standards the minimum of standards can be derived that have to be supported by a mobile web browser at present. A common denominator of the different standards has to be determined which has to be supported by mobile web browsers.

If different standard versions exist, the older version with the lower version number is the common basis. If different technologies exist, only those need to be supported that are also supported by the three mentioned stationary web browsers. The only exception is the support of the interface technology for plug-ins:

Table 3: Requirements on mobile web browsers for displaying websites

Web standards for the display of web applications					
(X)HTML	HTML 4.01	XHTML 1.0	XHTML 1.1	Parts of HTML 5	
CSS	CSS 1	CSS 2.1	Parts of CSS 3		
JavaScript	JavaScript 1.5				
Technology standards for display/use of RIAs					
Plugin interface	NPAPI or ActiveX				
AJAX	XML 1.0	DOM 1	DOM 2	Parts of DOM 3	
	SOAP 1.1	XMLHttpRequest			
Other standards to be supported					
Graphics	JPEG	GIF	PNG	BMP	SVG
Data transfer	HTTP(S)-Upload	HTTP(S)-Download			

The standards NPAPI (Netscape) and ActiveX (Microsoft) are competing technologies. However, as they have been coexisting since the end of the nineties (Giannandrea 2009) most manufacturers offer their plug-ins for both interfaces. The plug-ins “Adobe Flash Player” and “Java Runtime Environment” (JRE) are widely distributed and have high user numbers. Therefore, emphasis is laid on their support and plug-ins are compatible with both interfaces. It is therefore not relevant whether a mobile web browser supports NPAPI or ActiveX(Adobe 2009; Sun 2009).

In addition to the plug-in-interfaces, the technology pack AJAX also has to be supported, as it is widely distributed in the stationary web. Asynchronous JavaScript and XML is a combination of technologies that carry out Rich Internet Applications without plug-ins. It uses the available scripting language JavaScript, the Document Object Model (DOM) for addressing elements on a (X)HTML-page, as well as XML, SOAP and the JavaScript-class XMLHttpRequest for the asynchronous data transfer between client and server.

In addition, fundamental functions, such as the display of prevalent image formats, e.g. JPEG, GIF, BMP, PNG and SVG, have to be possible. File formats not supported by mobile web browsers or a supporting programme have to be transferred onto the device via HTTP-download, in order to be displayed by local applications. Furthermore, uploading files via HTTP, as well as the use of encoded connections with HTTPS (e.g. for mobile banking) should be enabled.

3.2 User-friendly Input and Output (b)

The different mobile end devices' input options have to be supported by mobile web browsers. Navigation on and between the different websites should be possible with as few and simple inputs as possible. One possible solution is the function of a "virtual mouse", which is operated via touch-screen or joystick. Navigation via buttons is possible provided there is a function to highlight interactive (X)HTML-elements, such as hyperlinks and form-elements. The selection jumps to the next or the previous element on the touch of a button. If an element is selected, an action follows after further confirmation by hitting a key. Examples for such actions are the opening of hyperlinks or sending forms. Alternatively a browser could also assign "access keys" to activate a certain link on the display by hitting a certain key-number.

Pushbuttons on the display are another form of navigation-aid. These can be equipped with desktop-web browser functions. However, the already limited display size is a drawback as only little space is left for displaying websites. If the mobile end device is equipped with a keyboard, this problem can be solved by assigning functions to certain keys. Alternatively, these controls could be blended in and out.

A text search function facilitates navigation on text heavy websites. As mobile end devices have limited input modes, entering data into forms is a further requirement. A function to fill text boxes automatically can also reduce user input. Examples are the automatic insertion of the protocol or the Top-Level-Domain (TLD) when entering the URL, as well as the completion of text boxes with input already entered. Passwords could also be saved like this.

In comparison to desktop PCs, the display of web content on mobile end devices is limited due to their output possibilities. The width of the monitor in particular hampers the user-friendly display of websites. A mobile web browser thus has to have functions that reduce this drawback, without horizontal scrolling of a website. There are several solutions to avoid horizontal scrolling. The browser could provide an overview the user can zoom in to (zooming function). End devices with a touch-screen capable of multi-touch are especially user-friendly.
Another solution is the so called "Small-Screen-Rendering" (SSR). Here the requested websites are processed by stationary servers before transmitting them to the mobile end

device. Web content is compressed so that it can be displayed on small monitors. By displaying the websites in landscape format, the rectangular form of displays could be optimally used.

3.3 Consideration of Data Transmission (c)

As the bandwidth of wireless data transmission is limited, more time is needed to transmit web content than in broadband internet connections. The resultant waiting time reduces user-friendliness. In order to decrease transmission time and possibly also data transfer costs, the data could be compressed.

The SSR method can optimise the code of websites or reduce the size and quality of images. This method requires the use of a proxy-server, which optimises requested web content before transmission (Christmann et al. 2009).

Another way of reducing waiting time during transmission is to load web content in the background. Other content could be transferred in the time a user spends looking at one website. “Tabbed Browsing” opens clicked hyperlinks in a separate tab. The requested content can be viewed later on.

A mobile web browser has to counter possible connection problems through appropriate functions. Saving data is one possible solution. By transferring and saving entire websites together with multimedia content on the mobile end device when it is connected (“caching”), these can still be accessed when the connection is broken.

3.4 Summary of the Requirements

Table 4 summarises the requirements for the display of web sites and the subsequent textual requirements.

Table 4: Overview of requirements on mobile web browsers

Requirements for displaying web sites	**Web standards for the display of web applications**					
	(X)HTML	HTML 4.01	XHTML 1.0	XHTML 1.1	Parts of HTML 5	
	CSS	CSS 1	CSS 2.1	Parts of CSS 3		
	JavaScript	JavaScript 1.5				
	Technology standards for display/use of RIAs					
	Plugin interface	NPAPI or ActiveX				
	AJAX	XML 1.0	DOM 1	DOM 2	Parts of DOM 3	
		SOAP 1.1	XMLHttpRequest			
	Other standards to be supported					
	Graphics	JPEG	GIF	PNG	BMP	SVG
	Data-transfer	HTTP(S)-Upload	HTTP(S)-Download			
Requirements of input and output	**Navigation**	Virtual mouse with touch-screen/ joystick	Marking of (X)HTML elements	Push buttons		
	Input	Text search	Auto-completion			
	Output	Zooming	Display in landscape-format	Small-Screen-Rendering		

4 Analysis of the Characteristics of Mobile Web Browsers

In this chapter mobile web browsers are presented and evaluated according to the morphological box.

4.1 Market-oriented Preliminary Considerations

As the state-of-the-art of mobile web browsers is to be presented, only providers of mobile web browsers which currently play a crucial role on the market of mobile internet usage are considered. All browsers of manufacturers that make up 80% of the browser market for mobile end devices are selected. According to statistics (StatCounter 2009), these are the browsers from Opera, Apple, Nokia and Research in Motion (RIM). The first step when looking at the manufacturer is the determination of the browser to be examined. The manufacturers generally provide many browsers or several browser versions simultaneously. In this case, the adequate product is determined through argumentation. In the next step the selected web browser is tested against the requirements. Requirements fulfilled are marked in green (light), requirements not met in red (dark). The results are explained and additional browser-specific solutions are given.

4.2 Opera Mobile

Opera is the market leader in mobile web browsers and offers two alternative products for the mobile internet: Opera Mini and Opera Mobile. Opera Mini is designed for the use with mobile phones and smartphones. As Small-Screen-Rendering is used, only compressed content is sent to the mobile end device. The used Java-basis and the related high platform independence of the Opera Mini are the reason for the high share in the market of Opera browsers.

As this paper examines the current technical possibilities of mobile web browsers, the Opera Mobile with the function of complete client-sided rendering is examined. As Opera recommends the use of Opera Mobile for smartphones, its most recent version 9.7b1 will be examined (Opera-I 2009).

Opera Mobile 9.7 uses the latest version of the proprietary rendering engine “Opera Presto 2.2”, which is also used by the desktop versions of the web browser “Opera 10x” (Mills 2009). Presto achieved top marks in the so-called Acid3-test of following the W3C-standards. This is an indication for the good quality of this rendering engine. The documentation of the Presto-engine also shows that all web standards examined in the requirements-model are supported. The support of plug-ins is guaranteed by components of the NPAPI-interface (Opera-III 2009). Furthermore, this documentation shows that all standards of the AJAX-technology listed in the requirements-model are supported. The required XMLHttpRequest-interface is also provided (Opera-IV 2009). Requirement (a) of displaying and reproducing websites is fully met.

Concerning requirement (b) of user-friendly input and output, the mobile web browser meets almost all of the criteria (Opera-I 2009). Currently, only mobile end devices with touch-screens are supported. This problem will probably be solved, as earlier versions already supported devices without touch-screens. Furthermore, the version examined in this paper is still in its beta-version, i.e. in the testing phase (Opera-II 2009). Opera Mobile does not support a text search function (Opera-I 2009). The function “Opera Turbo” can be activated optionally. This is identical to Small-Screen-Rendering. SSR facilitates the use due to its more clearly laid out display of web sites. Solutions for limited data transmission (requirement c) are also provided.

Table 5: Evaluation table for Opera Mobile 9.7b1[1]

Requirements for displaying web sites	**Web standards for the display of web applications**					
	(X)HTML	HTML 4.01	XHTML 1.0	XHTML 1.1	Parts of HTML 5	
	CSS	CSS 1	CSS 2.1	Parts of CSS 3		
	JavaScript	JavaScript 1.5				
	Technology standards for display/use of RIAs					
	Plugin interface	NPAPI or ActiveX				
	AJAX	XML 1.0	DOM 1	DOM 2	Parts of DOM 3	
		SOAP 1.1	XMLHttpRequest			
	Other standards to be supported					
	Graphics	JPEG	GIF	PNG	BMP	SVG
	Data transfer	HTTP(S)-Upload	HTTP(S)-Download			
Requirements for input and output	**Navigation**	Virtual mouse via touchscreen / joystick	Marking of (X)HTML elements	Push-buttons		
	Input	Text search	Auto-completion			
	Output	Zooming	Display in landscape-format	Small-Screen-Rendering		
Requirements of limited data transmission		Small-Screen-Rendering	Tabbed Browsing	Saving of websites		

"Opera Turbo" enables both faster data transmission and the simultaneous opening of websites ("Tabbed Browsing") (Opera-I 2009). The function of saving entire pages on mobile end devices enables the viewing of static content without an internet connection. Opera Mobile also offers Google Gears (Mills 2009). This web browser add-on enables extensive offline usage of web applications through different interfaces (API) and an integrated data base. However, these have to be optimised for offline-use (Google-I 2009). Google Gears also has an interface with which the position of the mobile end device can be determined via GPS - a meaningful and necessary enhancement when using Location-Based Services (LBS; Google-II 2009). All of the criteria supported are shown against a light background in table 6.

[1] Requirements fulfilled are marked in green (light), requirements not met in red (dark).

4.3 Apple Safari

Apple delivers its mobile end devices (iPhone, iTouch and iPad) including the proprietary web browser Safari (Apple-I 2009). Due to the strict specifications set by Apple, there are presently no real technologically and functionally adequate and therefore competitive alternatives running on both devices (Witte 2009). As the mobile version of the Safari-browser is linked to the installed operating system “iOS”, the latest version “iOS” 3.1 will be considered (Apple-II 2009).

Just like the desktop versions, the mobile version of the web browser Safari uses the rendering-engine Webkit (Apple-I 2009). Even though Apple is directly involved in the development of the Webkit-engine, it is an open source project. It is therefore available to many other web browser developers free of charge. The Webkit-rendering engine also receives the highest score in the Acid-3 test (Stachowiak 2009). The documentation shows that support of the required web standards is very good (Apple-III 2009).

The DOM-reference shows that Webkit belongs to the most advanced rendering-engines and that this also applied to the mobile version of the Safari-web browser (Apple-IV 2009). All of the criteria listed in the requirement model concerning the displaying of websites (requirement a) are met by the Safari-browser, with the only exception of supporting plug-ins. Apple argues that computationally intensive plug-ins, such as the Flash Player, exceed the possibilities of mobile platforms. Less intensive alternatives like Flash Lite are described as insufficient (IDG 2009; Apple-V 2009). The iOS-version of the Safari-browser was explicitly developed for a special mobile end device (Apple iPhone) and its handling is only adapted to the iPhone’s properties.

The general criteria for the handling of Smartphones (requirement b) formulated by the requirement model are not met completely. However, Apple meets the unfulfilled criteria by their own functions, such as the “multitouch” function, which facilitates navigation through the touchscreen by touching two points simultaneously (Apple-V 2009).

Apple Safari Mobile does not take adequate account of broken or limited connections (requirement c). Even though up to eight websites can be opened simultaneously, storing entire websites is only possible using an additional program (Titcomb 2009). No service facilitating SSR is provided. The results are summarised in table 7.

Table 6: Evaluation table for Apple Safari iOS 3.1[2]

Requirements for displaying web sites	**Web standards for displaying web applications**					
	(X)HTML	🟩 HTML 4.01	🟩 XHTML 1.0	🟩 XHTML 1.1	🟩 Parts of HTML 5	
	CSS	🟩 CSS 1	🟩 CSS 2.1	🟩 Parts of CSS 3		
	JavaScript	🟩 JavaScript 1.5				
	Technology standards for displaying/using RIAs					
	Plugin interface	🟥 NPAPI or ActiveX				
	AJAX	🟩 XML 1.0	🟩 DOM 1	🟩 DOM 2	🟩 Parts of DOM 3	
		🟩 SOAP 1.1	🟩 XMLHttpRequest			
	Further standards supported					
	Graphics	🟩 JPEG	🟩 GIF	🟩 PNG	🟩 BMP	🟩 SVG
	Data-transfer	🟩 HTTP(S)-Upload	🟩 HTTP(S)-Download			
Requirements for input and output	**Navigation**	🟥 **Virtual mouse via touchscreen/ joystick**	🟥 **Marking of (X)HTML elements**	🟩 Push-buttons		
	Input	🟥 Text search	🟩 Auto-completion			
	Output	🟩 Zooming	🟩 Display in landscape-format	🟥 Small-Screen-Rendering		
Requirements due to limited data transmission		🟥 Small-Screen-Rendering	🟩 Tabbed Browsing	🟥 Saving of websites		

4.4 Nokia Maemo

Nokia offers several mobile web browsers, similar to Opera. Their mobile end devices use web browsers produced by Opera and proprietary developments. The most powerful browser used on Nokia devices is part of the operating system Maemo. According to Nokia, Symbian will continue to be the operating system on the market for limited mobile phones (Nokia-I 2009). The more powerful Maemo will be used with Smartphones (e.g. Nokia N900). As

[2] Requirements fulfilled are marked in green (light), requirements not met in red (dark).

Maemo’s technology is superior and as Nokia has announced to use it on better equipped devices, this paper will focus on the newest version 4.1.3 and the corresponding Maemo-browser.

The Maemo browser is based on the Gecko-Rendering-Engine of the Mozilla Foundation. This rendering-engine is also used by the Mozilla Firefox desktop-browser and other web browser producers, due to repeated licensing. The used Rendering Engine Gecko version 1.9 supports all of the web standards for displaying web applications that are listed in the requirement model. Of all the browsers examined, the Maemo browser is the only one which implements the complete NPAPI interface and other standards. Using all of the Firefox plug-ins is therefore possible (Murtazin 2009). RIAs are also supported by a complete compatibility with AJAX. Thus, all of the requirements on depicting WWW-content (a) are supported entirely (Nokia-III 2009).

The handling of the Maemo browser is very good. In addition to all of the identified solutions, other useful functions are implemented. Different modes of display can be selected, JavaScript-popup-windows can be activated and deactivated and the websites’ metadata can be shown (Murtazin 2009). Furthermore, the Maemo browser also offers a function that highlights and copies text on websites Nokia-III 2009). However, the Maemo operation system is presently only used with mobile end devices equipped with a touch-screen. It is not clear whether other input media will be supported. The Maemo browser used on Nokia devices fulfils most of the requirements (requirement b).

Requirement (c) of considering connection problems is met by all possible solutions, with the exception of the SSR. In addition, it provides functions such as a “browsing history”, which offers an option to save all visited websites. An integrated download manager makes it possible to download data simultaneously or stall a download (Murtazin 2009). Table 8 summarises all of the evaluated requirements.

Table 7: Evaluation table for Nokia Maemo-Browser 4.1.3[3]

Requirements for displaying web sites	**Web standards for the display of web applications**					
	(X)HTML	🟩 HTML 4.01	🟩 XHTML 1.0	🟩 XHTML 1.1	🟩 Parts of HTML 5	
	CSS	🟩 CSS 1	🟩 CSS 2.1	🟩 Parts of CSS 3		
	JavaScript	🟩 JavaScript 1.5				
	Standards for displaying/using RIAs					
	Plugin interface	🟩 NPAPI or ActiveX				
	AJAX	🟩 XML 1.0	🟩 DOM 1	🟩 DOM 2	🟩 Parts of DOM 3	
		🟩 SOAP 1.1	🟩 XMLHttpRequest			
	Other standards to be supported					
	Graphics	🟩 JPEG	🟩 GIF	🟩 PNG	🟩 BMP	🟩 SVG
	Data-transfer	🟩 HTTP(S)-Upload	🟩 HTTP(S)-Download			
Requirements for input and output	**Navigation**	🟥 Virtual mouse via touchscreen/ joystick	🟥 Marking of (X)HTML elements	🟩 Push-buttons		
	Input	🟩 Text search	🟩 Auto-completion			
	Output	🟩 Zooming	🟩 Display in landscape-format	🟥 Small-Screen-Rendering		
Requirements of limited data transmission		🟥 Small-Screen-Rendering	🟩 Tabbed Browsing	🟩 Saving of websites		

4.5 Research in Motion BlackBerry-Browser

Research in Motion (RIM) call their mobile end devices „BlackBerrys“. They use the proprietary operation system “BlackBerry OS”. The web browser developed by RIM for the mobile internet is called “BlackBerry browser”, and it is linked to this operation system. Its newest version 5.0 beta is examined (RIM-I 2009).

[3] Requirements fulfilled are marked in green (light), requirements not met in red (dark).

Table 8: Evaluation table for BlackBerry-Browser 5.0 beta[4]

Requirements for displaying web sites	**Web standards for the display of web applications**					
	(X)HTML	🟩 HTML 4.01	🟩 XHTML 1.0	🟩 XHTML 1.1	🟩 Parts of HTML 5	
	CSS	🟩 CSS 1	🟩 CSS 2.1	🟩 Parts of CSS 3		
	JavaScript	🟩 JavaScript 1.5				
	Technology standards for displaying/using RIAs					
	Plugin interface	🟥 NPAPI or ActiveX				
	AJAX	🟩 XML 1.0	🟩 DOM 1	🟩 DOM 2	🟥 Parts of DOM 3	
		🟩 SOAP 1.1	🟩 XMLHttpRequest			
	Further standards supported					
	Graphics	🟩 JPEG	🟩 GIF	🟩 PNG	🟩 BMP	🟩 SVG
	Data-transfer	🟩 HTTP(S)-Upload	🟩 HTTP(S)-Download			
Requirements for input and output	**Navigation**	🟩 Virtual mouse via touchscreen/ joystick	🟩 Marking of (X)HTML elements	🟩 Push-buttons		
	Input	🟩 Text search	🟩 Auto-completion			
	Output	🟩 Zooming	🟩 Display in landscape-format	🟩 Small-Screen-Rendering		
Requirements of limited data transmission		🟩 Small-Screen-Rendering	🟥 Tabbed Browsing	🟩 Saving of websites		

RIM also uses a proprietary rendering-engine with its BlackBerry browser. It was created especially for the mobile internet. The supported standards and the browser functions can be found in a RIM document (RIM-I 2009). The support of Google Gears is especially worth mentioning. The NPAPI-interface for installing plug-ins is not available. However, the rendering engine does offer many formats of multimedia, for which usually plug-ins developed by other producers are used. Unfortunately, the widely used Adobe Flash-plug-in cannot be installed. Therefore many RIAs cannot be used. However, some flash files can be saved onto the mobile end device and converted into a supported format. The DOM-3-standard is also not supported; however, AJAX applications can be used, because DOM 1 and DOM 2 are

[4] Requirements fulfilled are marked in green (light), requirements not met in red (dark).

available. However, not all of the criteria on displaying web content (requirement a) are fulfilled.

Compared against the criteria set by the requirement model, the BlackBerry browser is very user-friendly (requirement b). All of the mentioned solutions are implemented. Regarding connection problems (requirement c) the BlackBerry browser does not offer a function that allows simultaneous opening of websites. However, other useful functions are implemented. For example, entire filled out forms can be saved as data sets and transmitted again after connection is restored. Launched downloads can be minimised and carried out in the background.

Table 9 is arranged according to the evaluation model presented in chapter 4.1.5.

4.6 Comparison of mobile Web Browsers

Requirement (a) of displaying web sites and multimedia content correctly is fulfilled by all of the examined web browsers. The required support of web standards is also met by all products. However, it must be kept in mind that it differs between the different products, due to the 50% rule set up by Hammond (2009). The number of supported properties of the newer web standards HTML 5 and CSS 3 also differs. In order for web applications to be used widely, it has to be checked in the respective web browser’s documentations if a property is supported or not. Even though all of the examined products possess a JavaScript-engine, not all standard properties are supported. Web developers should therefore use these technologies only to a limited extent, especially where essential parts of a web application are concerned.

The support of technology standards for running RIAs differs greatly between the tested mobile web browsers. The Maemo-browser is the only one offering complete support of the NPAPI-interface and thus the possibility of using any plug-in developed for this interface (Nokia-II 2009). Opera Mobile has integrated parts of the NPAPI-interface. The manufacturer Opera therefore lists Java and Flash Lite as supported plug-ins (Opera-III 2009). Flash Lite is a less computationally intensive version of the Flash player. It has been optimised for devices with limited resources, such as Smartphones. Apple uses this as an argument against supporting this plug-in. It describes alternatives, such as Flash Lite, as insufficient (IDG 2009). The BlackBerry browser offers at least the solution of storing flash-files and converting them

on the mobile end device. iOS and BlackBerry OS partly compensate for insufficient support of other multimedia plug-ins by aid-programmes (Apple-II 2009; RIM-II 2009).
All of the examined products support the AJAX-technology. However, the extent of this support differs greatly. Developers of RIA based on AJAX technology will have to read the documentations provided by the manufacturers carefully in order to decide whether the characteristics and elements of the respective technologies are indeed supported by the mobile web browsers.

In summary, requirement (a) is only entirely fulfilled by the Opera Mobile and Maemo-browser. Of all the tested products, the Maemo-browser supports plug-ins best and is fully equivalent to a stationary web browser (Murtazin 2009).

The requirement of supporting user-friendly input and output (b) differs greatly between the devices. A mobile web browser, such as Apple Safari, which is designed for a specific mobile end device, may have very good controls and support of this specific device. However, compared against the requirement model with its very general evaluation criteria it does not get a high ranking.

Table 9: Central results of the study

Requirement (a) *Display*	▪ Requirement is met by all browsers. ▪ Degree of support of web standards differs between the browsers. ▪ Some plug-ins such as Flash are not always available for mobile use. ▪ AJAX can be used on mobile devices.
Requirement (b) *Input / Output*	▪ User-friendliness differs greatly between the end devices. ▪ Lack in supported functions does not always entail less usability (Apple).
Requirement (c) *Data connection*	▪ Plug-ins facilitate the continuation of work when the connection breaks. ▪ Not all browsers support this.

The BlackBerry-browser’s good controls should be emphasised. Only the BlackBerry-browser and the Maemo-browser offer full-text search (RIM-II 2009; Murtazin 2009). Other mobile web browsers only use so-called bookmarklets as a supporting script. Requirement (b) is best fulfilled by the BlackBerry browser. Whether the Maemo operation system will in the future support other input modes besides touch-screens is unclear. If this were the case the Maemo-browser would be the most advanced technology. Opera Mobile has the best solution for addressing possible connection problems and limitations to bandwidth (c).

The support of Google Gears technologies is also very important for accessing web applications and RIAs without an internet connection. It is officially supported by Opera Mobile and BlackBerry browser (Mills 2009; RIM-II 2009).

The Nokia Maemo-browser’s development server shows that work is also being done on supporting Gears (Kinnunen 2009). Apple Safari’s performance is weaker as it only offers "tabbed browsing" function and does not compensate this by offering other useful functions such as a download manager or an extensive “browsing history”, as does the Maemo-browser.

5 Conclusion

The range of functions offered by web browsers for mobile end devices differs greatly, also in more recent versions. Therefore, not all web browsers are suitable for displaying internet pages. However, this study shows that implementation of the most important standards is possible and that web sites designed for stationary computers can also be used within the limitations of mobile use. The browser is the decisive component, which today is powerful enough.

It can be assumed that the use of highly efficient web browsers on mobile end devices will increase. The mobile use of the WWW has become an important advertising point for mobile phone network providers, as their ARPU (average revenue per user) from telephone calls has been dropping.

This study shows that the original multiple webs-approach has become obsolete, that specialised standards such as WML or C-HTML are superfluous and that they can be replaced by their equivalents from the stationary web. However, complex applications based on web technologies increasingly found in the stationary web have to be optimised. Applications should be optimised for mobile end devices if an application is used frequently and for a longer period of time or if superior interaction between end device and user is needed (e.g. in online banking, cf. Christmann et al. 2009).

Using web standards as a technological basis also makes it possible to create applications for mobile end devices that run on all operation systems. At present, mobile applications can generally only be run on the platform they were programmed for. Even platform-independent applications such as applications developed with the help of JavaME are not guaranteed to run on all end devices. However, web technologies can become the basis of mobile applications, as the expected distribution of high-performance web browsers on mobile end devices will take place.

References

Adobe (2009) Flash Player Version Penetration. http://www.adobe.com/products/player_census/flashplayer/version_penetration.html. Accessed 4 October 2009

Alby T (2008) Das mobile Web. Carl Hanser Verlag: Munich

Apple-I (2009) Safari Dev Center - Apple Developer Connection. http://developer.apple.com/safari/. Accessed 18 October 2009

Apple-II (2009) Apple - iPhone - New features in the iPhone 3.1 Software Update. http://www.apple.com/iphone/softwareupdate/. Accessed 10 October 2009

Apple-III (2009) Safari 4.0 Release Notes. http://developer.apple.com/safari/library/releasenotes/AppleApplications/rn-safari/. Accessed 12 October 2009

Apple-IV (2009) API Reference: WebKit DOM Reference. http://developer.apple.com/safari/library/documentation/AppleApplications/Reference/WebKitDOMRef/. Accessed 12 October 2009

Apple-V (2009) iPhone Human Interface Guidelines for Web Applications: iPhone and the User’s Environment. http://developer.apple.com/safari/library/documentation/InternetWeb/Conceptual/iPhoneWebAppHIG/iPhoneUserEnvironment/iPhoneUserEnvironment.html. Accessed 12 October 2009

Berners-Lee T, Cailliau R. (1994) The World-Wide Web. Communications of the ACM 37(8), pp. 76-82.

Berners-Lee T, Fielding R, Frystyk R (1993) Hypertext transfer protocol. http://graphics.cs.msu.ru/courses/wp_el00/Internet/HTTP/article.html. Accessed: 2010-02-07

Bharadvaj H, Joshi A, Auephanwiriyakul S (1998) An Active Transcoding Proxy to Support Mobile Web Access. In: IEEE Computer Society (eds) Proceedings of the The 17th IEEE Symposium on Reliable Distributed Systems. IEEE Computer Society Press, Washington, pp. 118-123

Bieh M (2008) Mobiles Webdesign – Konzeption, Gestaltung, Entwicklung. Galileo Computing, Bonn

Browserstatistik (2010) Die aktuelle Browser-Statistik. http://www.browser-statistik.de/. Accessed 8 February 2010

Brusilovsky P, Maybury MT (2002) From Adaptive Hypermedia to the Adaptive Web. Commun ACM 45(5): 31-33. doi: 10.1145/506218.506239

Buyukkokten O, Garcia-Molina H, Paepcke A (2001) Seeing the Whole in Parts: Text Summarization for Web Browsing on Handheld Devices. In: Proceedings of the 10th international conference on World Wide Web, pp. 652-662

Central Intelligence Agency (2009) The CIA World Factbook. Skyhorse Publishing, New York

Capin T, Pulli K, Akenine-Möller T (2008) The State of the Art in Mobile Graphics Research. IEEE ComputerGraphics, vol. 28, no. 4, pp. 74-84, doi: 10.1109/MCG.2008.83

Christmann S, Caus T, Voigts R, Hagenhoff S (2009) Optimizing Web Pages for Mobile Access. In: Isaías P, White B, Baptista Nunes M (eds.) Proceedings of the IADIS International Conference WWW/Internet 2009. Vol. 1, Rome, pp. 408-415

Crossan A, Williamson J, Brewster S, Murray-Smith R (2008) Wrist rotation for interaction in mobile contexts. Proceedings of the 10th international conference on Human computer interaction with mobile devices and services, pp. 435-438, doi: 10.1145/1409240.1409307

Eggers T (2005) Evaluierung beispielhafter Geschäftsmodelle für das mobile Internet auf Basis von Marktbetrachtungen und technologischen Gegebenheiten. Lang, Frankfurt am Main

Flanagan D (2006) JavaScript. The Definitiv Guide. O'Reilly Media: Sebastopol

Garrett JJ (2005) Ajax: A New Approach to Web Applications. http://www.adaptivepath.com/ideas/essays/archives/000385.php. Accessed 4 Februar 2010

Google-I (2009) Gears FAQ. http://code.google.com/intl/de-DE/apis/gears/gears_faq.html. Accessed 12 October 2009

Google-II (2009) Geolocation API. http://code.google.com/intl/de-DE/apis/gears/api_geolocation.html. Accessed 12 October 2009

Giannandrea J (2009) Microsoft breaks Web Plugins. http://web.archive.org/web/20071016233843/http://www.meer.net/jg/broken-plugins.html. Accessed 4 October 2009

Hammond D (2009) Web Browser Standards Support. http://www.webdevout.net/browser-support. Accessed 4 October 2009

Hong Y, Jun W, Kwak BH (2006) Design and Implementation of a Mobile Class Web Site Using Intelligent User Interface. In: LNCS, vol. 3942(2006), pp. 716-725

IDG (2009) iPhone: Flash kommt! http://www.i-welt.de/2009/02/02/iphone-flash-kommt/. Accessed 12 October 2009.

Jamsa K (2001) WML & WMLScript: A Beginner's Guide. Mcgraw-Hill Professional: New York

Kaikkonen A, Roto V (2002) XHTML in Mobile Application Development. In: LNCS, vol. 2411(2002), pp. 19-20

Kinnunen K (2009) Maemo Google Gears. https://garage.maemo.org/svn/browser/extensions/maemo-google-gears/trunk/gears/README.Maemo. Accessed 14 October 2009

Linjama J, Korpipää P, Kela J, Rantakokko T (2008) ActionCube: a tangible mobile gesture interaction tutorial. Proceedings of the 2nd international conference on Tangible and embedded interaction, pp. 169-172, doi: 10.1145/1347390.1347428

Miller FP, Vandome AF, McBrewster J (2009) Browser wars: Metaphor, Web browser, Microsoft, Internet Explorer, Netscape, Netscape Navigator, Mozilla Firefox, Google Chrome, Safari (web browser), Opera (web browser), Usage share of web browsers. Alphascript Publishing, Beau Bassin

Mills C (2009) Opera Mobile 9.7 – features and standards support. http://dev.opera.com/articles/view/opera-mobile-9-7-features-standards/, Accessed 4 February 2010

Mozilla (2010) Mozilla Developer Center. https://developer.mozilla.org, Accessed 8 February 2010

Murtazin E (2009) Review of Nokia N900's Maemo5. http://www.mobile-review.com/review/nokia-maemo5-en.shtml. Accessed 12 October 2009

Nokia-I (2009) Maemo and N900: Many customization points for operators. http://conversations.nokia.com/2009/09/11/maemo-and-n900-many-customization-points-for-operators/. Accessed 10 October 2009

Nokia-II (2009) Mozilla based browser for maemo - white paper. http://browser.garage.maemo.org/docs/browser_paper.html. Accessed 12 October 2009

Nokia-III (2009) Maemo Browser. http://maemo.nokia.com/features/maemo-browser/. Accessed 12 October 2009

Opera-I (2009) Opera Mini or Opera Mobile? http://www.opera.com/products/choose/. Accessed 10 October 2009

Opera-II (2009) Opera Mobile – Download. http://www.opera.com/mobile/download/. Accessed 14 October 2009

Opera-III (2009) Plug-ins and Opera. http://www.opera.com/docs/plugins/. Accessed 15 October 2009

Opera-IV (2009) Web specifications supported in Opera Presto 2.2. http://www.opera.com/docs/specs/presto22/. Accessed 15 October 2009

Paulson LD (2005) Building Rich Web Applications with Ajax. Computer. Vol. 38, no 10. pp. 14-17

RIM-I (2009) BlackBerry - Browser. http://de.blackberry.com/devices/features/internet.jsp. Accessed 10 October 2009

RIM-II (2009) BlackBerry Browser Fundamentals Guide. http://docs.blackberry.com/en/developers/deliverables/8796/BB_Browser_Fundamentals_Guide-BETA.pdf. Accessed 12 October 2009

Roth J (2005) Mobile Computing. Dpunkt Verlag, Heidelberg

Stachowiak M (2009) WebKit achieves Acid3 100/100 in public build. http://webkit.org/blog/173/. Accessed 12 October 2009

StatCounter (2009) Top 9 Mobile Browsers. http://gs.statcounter.com/#mobile_browser-ww-monthly-200901-201002, Accessed 8 February 2009

Sun (2009) Java Plug-in Technology. http://java.sun.com/products/plugin/. Accessed 4 October 2009.

Titcomb J (2009) iWebSaver. http://iwebsaver.com. Accessed 12 October 2009

von Tetzchner JS (2009) State of the Mobile Web, September 2009. Opera Software. http://www.opera.com/smw/2009/09/. Accessed 3 Januar 2010

W3C (2010) Mobile Web Initiative. http://www.w3.org/Mobile/. Accessed 8 February 2010

Webhits (2010) Zähler und Webstatistik. http://www.webhits.de/. Accessed 8 February 2010

Williams R, Tollett J (2004) The Non-Designer's Web Book: An Easy Guide to Creating, Designing, and Posting Your Own Web Site. Addison-Wesley Longman, Amsterdam

Witte C (2009) Safari-Ableger: Erstmals Browser im iPhone-App-Store. http://www.computerwoche.de/netzwerke/mobile-wireless/1884048/. Accessed 10 October 2009

Xiao J, Chen, X, He H-J, Cui S-G (2009) Design and implementation of web application based on AJAX and Struts. Computer Engineering and Design. Vol. 30, no. 8, pp. 1934-1937

Zhang D (2007) Web Content Adaptation for Mobile Handheld Devices. Commun ACM 50(2): 75-79. doi: 10.1145/1216016.1216024