A Web Browser for ISDN Card Phones

Dimitris Maroulis, Sotiris Aronis, Vassiliki Nassiopoulou, Nikos Grammenos, Costas Vassilakis
Department of Informatics and Telecommunications
University of Athens
Greece
{d.maroulis, c.vassilakis}@di.uoa.gr

Abstract

In order to cover the ever-increasing need for more direct and easy access to information, new information access means need to be devised or existing ones need to be further exploited. In this paper, we present a mini Web Browser for ISDN card phones, which enables this widespread device to be used for accessing information in the World Wide Web. The implemented web browser supports HTML and WML pages, while special care was taken to tackle the limitations imposed by the ISDN card phone's hardware, such as small screen, limited keyboard, scarce processing and memory resources. One of the techniques employed to increase the capabilities of the ISDN card phone browser was the introduction of a proxy server, which transforms demanding media types to formats that can be handled by the ISDN card phone hardware.

Keywords: Embedded systems, ISDN card phone, web browsing, proxy server.

1 Introduction

In the past few years, technology has enabled more and more people to access information. Although the results attained insofar are remarkable, the goal of information society to provide access to information for all people, everywhere and anytime have not been yet reached. One of the limiting factors, regarding information access is the physical device through which users will connect to information sources. The primary means available to users for accessing information are the following:

(a) *Desktop computers.* These are devices with powerful processors and large amounts of memory, usually equipped with complex software that enables users to access, display and manipulate information in different formats. Desktop computers however, are docked to their installation site and may be unaffordable for considerable portions of the citizens.

(b) *Laptop-portable computers,* which have similar properties to desktop computers, but can be easily carried, leveraging their availability as information access devices. Still, being expensive devices, their penetration is limited and additionally they cannot always be connected to communication channels of high bandwidth and reasonable cost.

(c) *PDAs*, which are downsized laptop computers for even higher portability. Computing power, memory size and software capabilities are limited, yet adequate to perform most information-access related activities. Cost and availability of affordable and high bandwidth communication channels are the two primary considerations for this device category.

(d) *Mobile phones and WAP-enabled devices.* This device category is generally equipped with low-end processors, small amounts of memory and low-resolution screens, while their communication channel bandwidth is limited (at least up to the second generation devices). Due to these restrictions software enabling information access has limited capabilities and supports only undemanding information types and only basic services. Moreover, communication costs may not be affordable for most users. This device category is important mainly due to its widespread use [1].

In order to enable more citizens to access information, ISDN card phones are excellent candidates, since they have adequate processing power and bandwidth, which are available at reasonable cost, and most importantly, all citizens have access to them. However, insofar ISDN card phones lack software that will enable information access to their users.

In this paper we present an implementation of a Web browser for ISDN card phones, which enables the use of ISDN card phones as information access devices, providing access to the World Wide Web. The implementation of the Web browser is tailored to the ISDN card phone platform to cater for the idiosyncrasies of the hardware environment, such as the limited amount of memory, the middle-powered processors and the lack of keyboard. The capabilities available to the users, however, may be extended through the introduction of a *proxy server* [2] [3], which will undertake the transformation of demanding information types to simper ones that can be handled by the ISDN hardware and software.

The rest of this paper is structured as follows: in section 2, the hardware and software platform of the ISDN card phone are described, while section 3 outlines the capabilities and limitations of the developed software.

Section 4 describes the operation of the web browser and details the algorithms used for handling the various information types. Section 5 introduces the concept of the proxy server as an elegant solution to extend the capabilities of the Web browser, while in section 6 conclusions are drawn and future work is outlined.

2 The ISDN Card Phone Platform

The ISDN card phone platform comprises of hardware elements and software modules, which provide the foundation on top of which the Web browser is built. These elements and modules are described in the following paragraphs.

2.1 Hardware platform

The hardware setup of the ISDN platform used for the Web browser project includes the items listed bellow:

(a) An SED 1335 display, which offers an LCD screen with a resolution up to 320x240 pixels.

(b) A Motorola 68302 processor, a 32 bit integrated multi-protocol protocol processor (IMP), which is specially designed to support a large number of interfaces used in industrial applications, such as BRI ISDN networks and terminal adapters.

(c) The earphone of the card phone is used as an output device for audible information formats (sounds).

(d) 2 Mbytes of RAM, a flash memory of 4 Mbytes for code storage, plus 4 Mbytes of general-purpose flash. Card phone parameters are stored in two EEPROM chips.

(e) There are 16 standard keys, used for dialing, volume control, language selection, redialing, changing the card and call termination. There are also 8 programmable keys the operation of which depends on the state of the card phone, for example in case the Web browsing function was selected, these keys do scrolling, selecting the desired frame, returning to a previous page, exiting browser etc.

(f) The ISDN card phone is equipped with two 64 Kbps communication channels, the first one dedicated to phone calls and the second one used for card phone services other than phone calls. An additional 16 Kbit channel is used for signaling.

The ISDN card phone environment offers a number of software modules, which are exploited by the Web browser:

(a) *Display libraries*, which is a set of API calls facilitating the presentation of text, images, geometrical shapes etc.

(b) *Communication protocol stack*. This includes an implementation of the TCP/IP [4] [5] protocol over the ISDN physical layer, *while* application-level layers for the TIPS and HTTP protocols [6] [7] are provided as well.

3 Browser Capabilities and Limitations

One of the design goals adopted for the project was to implement the browser in an extensible way, in order to meet both existing and emerging needs, offering improved web navigation capabilities. However, services offered to the end-users should be of high quality, since users refrain from using buggy, sluggish or unresponsive services. To this end, and taking into account the physical restrictions imposed by the ISDN card phone hardware, it was decided to provide support for a subset of the information types usually handled by the state-of-the-art browsing software usually run in personal computers. More specifically, the card phone browser implemented within the project includes support for HTML ([8]; [9]) and WML ([10]) documents, images in the BMP format and audio in WAV format. Other information types, such as GIF, PNG or JPG images, video, or media types requiring plugins are not supported directly by the ISDN card phone browsers. The same holds for active features, such as Javascript, Java and ActiveX controls. When a page references or uses unsupported information types, these are simply ignored. If a page uses both text and graphics, text rendering takes precedence over the display of images, in order to shorten the time needed to display the page's informational content to the user.

A major limitation of the browser is owing to the lack of a full-sized keyboard, which would enable users to type in web addresses or fill in text boxes in dialogue screens. While the ISDN card phone numeric keypad may serve for entering text while in the card phone runs the web browser, users have found the process inconvenient. In order to alleviate the problem, the web browser starts off with a home page displaying a hierarchical menu of the most commonly used services, and users may navigate through this menu to the desired information source (e.g. portals, news agencies, online stock services, entertainment sites etc). If the desired page is not listed in the menu, its address may be entered using the keypad.

4 Card Phone Browser Operation

The overall operation of the card phone browser is to fetch and render web pages requested by the user, and allow the user to interact with the page by filling in text boxes, making selections from combo boxes or radio buttons, following navigation links etc. The individual steps within this service loop are as follows:

(a) The user presses a key, selecting a web page/service, causing a hardware event, which is intercepted by the browser software. The browser then extracts the web address corresponding to the service.

(b) An HTTP request [7] message is formulated, with appropriate information planted in the header and body fields.
(c) A TCP-level socket [5] [11] is opened towards the designated information source. The socket is actually the channel through which the request will be sent and the reply will be collected.
(d) The request is forwarded, through the socket, to the TCP/IP protocol stack [4] [5].
(e) The browser software requests the data corresponding to the reply from the TCP/IP stack. When the response arrives, the TCP/IP stack makes the data available to the browser.
(f) The data is collected and the socket is closed.
(g) The response header is checked to extract the values of the various fields (e.g. content type, content length, transfer encoding etc).
(h) The main body of the response is extracted; in most cases this will be either an HTML or a WML document.
(i) The main body is passed to the appropriate parser for further processing, depending on the content type. Since the tags and the semantics of WML documents have considerable dissimilarities from the corresponding dimensions of HTML documents, it has been found more efficient and manageable to use distinct parsers for the two document types, rather than a generic one that will handle both document types.
(j) If the received document was of WML type, the syntax tree generated by the parsing phase is translated to an equivalent HTML tree. This is always possible without loss of information, since the expressive power of WML is limited, compared to the richness of HTML semantics. Beyond this step all documents have the same structure and can be handled uniformly.
(k) The syntax tree created from the parsing phase is passed to the rendering procedure in order to be presented to the user. If the page contains references to images or sounds, appropriate HTTP requests are made to the designated information sources to fetch the respective files. Text and images are rendered to the screen, while audible content is reproduced through the ISDN card phone's earphone.

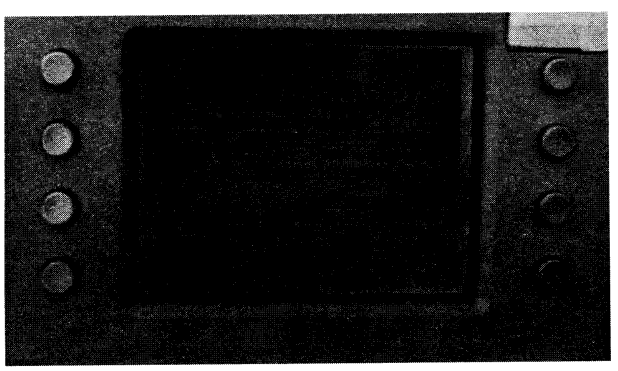


Figure 1 Screen scrolling buttons

In the following paragraphs, the most important aspects of the processing of the documents and associated files (images and audio) are described in more detail.

4.1 HTML page processing

When the browser receives an HTML document it analyses its structure, locating the tags that specify the formatting options and the navigational capabilities. Display characteristics in HTML documents may be defined either through individual formatting directives (e.g. `<b>`, `<font>`, `<center>`, `<table>` etc) or via collective specifications, such as the heading designations (`<h1>`, `<h2>`) and the cascading style sheets. While the document is syntactically analysed, text chunks are formulated, based on the display characteristics specified in the document. Each text chunk contains a text portion and the appropriate format designations. These chunks are then passed to the rendering procedure, which caters for their display on the screen. In order to display the requested text, the rendering procedure uses a number of built-in fonts and widgets implementing text boxes, select lists and radio buttons.

A special case of an HTML document is the *frameset*, which effectively divides the screen in a number of areas. The sizes of the individual frames are either specified within the frameset, or derived from default values. Areas may be visually divided using a horizontal or a vertical line, depending on the BORDER tags of the frameset specification.

Syntactical analysis may identify references to images, in which case the appropriate URL [12] is constructed and the image is requested from the web server it is hosted on. Since the ISDN card phone screen can only display two colors, images are converted to black and white before they are displayed. This conversion is performed using the Floyd-Steinberg dithering algorithm [13] [14] [15], which is based on the error dispersion technique [16]. If the image size is greater than the ISDN card phone screen resolution, the algorithm resizes the picture so that it will fit on the screen [17] [18].

Finally, during the analysis phase references to audio files may be encountered. In these cases, the audio content is fetched from the network and is then checked to determine whether it matches the single-channel/8KHz format needed for reproduction through the ISDN card phone earphone. Firstly, the frequency is checked, and if found different than 8KHz each audio channel is interpolated so that its frequency is changed to 8KHz. The last step is to convert stereo sound to single-channel, by

adding up the respective samples within each channel and normalising the result.

4.2 WML page processing

Since the number of service providers delivering content specially tailored to WAP devices increases [1], it was decided to extend the ISDN card phone browser to support WML documents. WML documents are particularly suited for the ISDN card phone environment, since WML is designed to be handled by devices with limited hardware resources [10].

In order to save program storage space and RAM, which are scarce within the ISDN card phone environment, it was decided to translate WML documents to "equivalent" HTML documents, and handle then the translated WML documents using the same pieces of code employed for HTML documents. The overall procedure for handling WML documents is as follows:

(a) The WML document is syntactically analysed, in order to verify that it complies with the WML specifications [19] and create an internal representation (a *parse tree*; see [20], [21]) that contains the same information as the original document but is better suited for processing. The parsing and verification take into account the WML version information, contained into the WML document header, which determines the DTD (Document Type Definition [19]), which should be used for the specific document.

(b) Once the verification process is complete and the syntax tree is formulated, the translation phase begins, starting off at the root node and recursively descending towards the leaf nodes. The translation algorithm makes the following mappings:

   i. The `<wml>` tag is translated to the tag sequence `<html><body>`

   ii. Each WML card deck is translated to a single HTML document, with each card within the deck being mapped to an HTML table. For each such table an anchor is defined (`<a name="...">`), in order to allow referencing to the specific card.

   iii. Formatting designations and other tags that differ in the HTML and WML specification are mapped as appropriate.

   iv. Text nodes within the syntax tree remain generally intact, however care is taken to handle properly special character specifications such as `&quot;` [22].

When the translation procedure is complete, the syntax tree corresponds to an HTML document that is equivalent to the original WML document received. This syntax tree is then passed to the rendering procedure in order to be presented on the ISDN card phone screen.

4.3 Screen scrolling

Since the ISDN card phone screen resolution is limited, most HTML and WML pages do not readily fit within its boundaries. When the display of a page requires a larger canvas, only a portion of the document is displayed and the user is allowed to scroll the screen in any horizontal or vertical axis to view the remainder of the document. Scrolling commands are entered via dedicated buttons of the ISDN card phone. When scrolling facilities are active, small icons are displayed on the screen to indicate the direction that each button will scroll the screen.

4.4 User interaction

Besides scrolling the screen to view documents that do not fit within the screen resolution, the user is allowed to navigate within the document, in order to:

(a) Select and traverse the desired link

(b) Move the focus to form elements (text boxes, radio buttons, select lists, submit/reset buttons) and enter text or perform appropriate operations.

(c) In multi-frame documents, select one of the document frames and then perform navigation operations as described above.

Given that the ISDN card phone environment does not include pointing devices, such as mice or light pens, these actions are performed through specially programmed buttons available on the ISDN card phone. To this end, the following navigation buttons are available:

(a) One button for selecting the desired frame. This button is active only within multi-frame documents.

(b) One button for cycling between the document items that can receive focus. These items include the link anchors and the various form elements. The functionality of this button is similar to the "Tab" key in personal computer browsers.

(c) One button for activating the current item. The effect of pressing this key depends on the item that has the focus; for example if the current item is a link anchor, the link is traversed, while text boxes enter or exit the edit mode.

Finally, in order to assist the user in navigation activities, two more buttons of the ISDN card phone have been programmed to implement the functionality of the "Home" and "Back" buttons found in personal computer browsers. Admittedly, PC-based browsers offer a richer set of navigational capabilities, through menu items and toolbars, but within the ISDN card phone environment usage of menus and toolbars would further limit the (already restricted) area that is available for displaying the documents; thus, it was decided to support directly only the most commonly accessed features, in favour of maximising the area available for the informational content of the documents.

For user input, in particular, since the keypad contains only 12 buttons -which are not adequate for all the letters and symbols that may be needed- each key in the keypad will be used to cycle through a (small) set of letters and symbols, in the same fashion that the mobile phone keypad is used for writing SMS messages. This input method is used when data needs to be typed in, e.g. for entering URLS, for providing values for fields of HTML forms etc. Using a well-spread technique is beneficial for the users, since quite a few citizens are nowadays accustomed to using mobile phones, and will thus be able to type in URLs and data quickly and with few errors.

5 Using a Proxy Server to Extend the ISDN Card Phone Browser Capabilities

Existing web sites usually employ different media formats, such as GIF, JPG, PNG, MP3 etc., in order to minimise the volume of the data needed to be downloaded to the client, while retaining a high quality in the offered media. The trade-off for reducing bandwidth needs is demand for processing power and memory, which while being available on personal computers are quite limited in the ISDN card phone environment.

Since incorporating support for these advanced media types into the ISDN card phone browser was infeasible -as it would result to sluggish and unresponsive behaviour- an alternative path was chosen to bridge the gap between the offered content types and the natively supported ones. The solution was the introduction of a *proxy server* [2] [3], which undertakes the task of fetching media encoded using complex algorithms, convert it to formats natively supported by the ISDN card phone browser and forward the simplified media to the ISDN card phone browser for final display. The proxy server can also perform scaling and dithering on images, while it may be used as an elegant solution to further extend the range of media types supported by the ISDN card phone browser, without the need to modify the software running on the ISDN card phone. For this functionality to be implemented, proxy servers intervene between the ISDN card phones and content providers (i.e. web servers), being autonomous entities, which are not integrated either with the client side or with the server side.

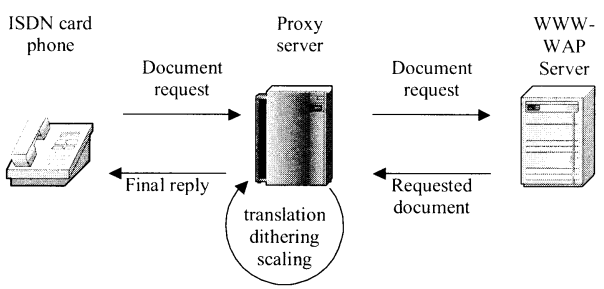


Figure 2 Introduction of a proxy server

Proxy servers are installed and managed by the telecom companies that provide the browsing services for the ISDN card phones. Physically, proxy servers may be located anywhere; the only requirement is that proxy servers can communicate with both the ISDN card phones and the Internet. Telephone exchange centres within telecom company premises appear as an appealing alternative for the installation of proxy servers, since this approach diminishes cabling costs. The cost for a single server is small (typically in the level of 1000-1300 € per unit), thus a telecom company may install multiple proxy servers (a *proxy server farm*) in order to handle requests more efficiently. The enhanced functionality and efficiency offered by the introduction of the proxy server for ISDN card phone browser users will increase the number of users and the amount of usage (total number of hours), thus the telecom companies will amortise the investments on proxy server installations through the increased income.

The introduction of the proxy server modifies the operation of the ISDN card phone browser as shown in the above figure. The ISDN card phone browser, instead of requesting the document directly from the server designated by the URL, formulates the request and sends it to the proxy server. The proxy server arranges for fetching the document from the server it resides on and then examines whether any transformations should be performed on the content, either because the document encoding is not natively supported by the ISDN card phone browser, or for achieving better presentation results. If a transformation is necessary the appropriate algorithm is applied on the received document to produce a document type that is suitable for presentation on the card phone environment. Finally the transformed (or original, if no transformation was necessary) document is sent to the ISDN card phone for presentation. The transformations that are currently employed by the proxy server are as follows:

(a) *Encoding translation.* As stated above, the ISDN card phone browser supports natively only simplistic media encodings, i.e. images of type BMP and audible content of type WAV. The proxy server undertakes the task of retrieving complex media encodings from the Web (such as JPG for images and MP3 for audio) and converting it appropriately to the respective media format natively supported by the ISDN card phone browser.

(b) *Image* scaling [13] [14] [23] [24]. Most web designers assume that the documents served by their sites will be displayed on that support resolutions significantly higher than the ISDN card phone screen resolution.

This assumption leads to the usage of large images, which cannot be satisfactorily rendered to the ISDN card phone screen. The proxy server arranges for scaling down large images, so that they will fit better on the ISDN card phone screen. Image downsizing is also convenient because it reduces the size of the respective data, decreasing thus the time needed to download it to the ISDN card phone.

(c) *Image dithering* [13] [14] [23] [24]. Since the ISDN card phone screen can only display two colours, the proxy server reduces the colour depth of images to 1 bit, making them readily available to the ISDN card phone for rendering to the screen. This operation may be combined with the scaling transformation to achieve better quality for the resulting image.

A major consideration for the encoding translation step is that the conversion of complex media encodings (e.g. JPG and MP3) to more simplistic ones (BMP, WAV) will result to significantly larger files, increasing thus the time needed to download the translated documents to the ISDN card phone and downgrading the overall performance. In order to tackle this issue, the proxy server compresses the final output of the transformation phase using the GZIP public domain library, reducing thus effectively the amount of the data that must be downloaded. In order to handle compressed data streams, the ISDN card phone browser is employs the decompression algorithm of the GZIP public domain library, which is very efficient, even in medium-powered processors and has small memory requirements.

5.1 Caching features of the Proxy Server

The proxy server may act as a caching agent, retaining the fetched (and possibly transformed) documents on a local disk. In such a case, if a document is requested again in a short time frame, the proxy server verifies with the information source that the document has not changed since it was originally downloaded, and if this is the case, the document is forwarded from the local disk to the ISDN card phone that requested it. In more detail, when the caching option is enabled, the proxy server performs the following checks for each request:

(a) Upon reception of the request from the card phone browser, the proxy server initially checks if the requested object is cacheable or not. Static html pages, images and sounds are considered cacheable, whereas dynamically generated HTML pages (e.g. PHP, ASP, CFM) are considered as *non-cacheable*.

(b) If the object is not considered as cacheable, it is firstly fetched from its hosting server. Afterwards, if any relevant transformations (encoding translations, dithering, scaling) are needed they are applied, the result is returned to the requesting ISDN card phone browser and the algorithm terminates.

(c) For cacheable objects, it is first determined whether the object exists in the cache. If the object exists, the timestamp at which the object was placed in the cache and the cached copy expiration date are extracted. If the object has expired, it is deleted from the cache and the algorithm continues in step (d). If the object has not expired, A HEAD request is sent to the server hosting the object, and the value of the "Last-Modified" reply header is compared to the timestamp at which the object was placed in the cache. If the object has not been modified since it was fetched, the object is fetched from the cache and the algorithm continues with step (f); otherwise, the algorithm continues with step (d).

(d) The object is fetched from its hosting server. If any relevant transformations (encoding translations, dithering, scaling) are needed they are applied to the fetched object. Afterwards, the reply headers are scanned to determine whether a "Pragma: No-cache", or a "Cache-Control: no-cache", or an "Expires" header specifying an already past expiration date is included. If any of these headers is found, the algorithm continues with step (f); otherwise the algorithm continues with step (e).

(e) The object (as transformed by any relevant transformations) is placed in the cache, tagged with the timestamp it was fetched and its expiration date. Any previous occurrences of the object in the cache are deleted, since they are outdated.

(f) The object (either fetched from the cache or fetched and transformed from its hosting server) is returned to the requesting ISDN card phone browser.

Document caching in the proxy server is particularly efficient, since most users visit the sites referenced by the hierarchical menu structure presented to the users; thus the cache hit-rate is significantly higher than in enterprise environments with personal computers, since in the latter case visited URLs are more randomly distributed. Moreover, caching in the proxy server partially compensates for the absence of local cache in the ISDN card phone, which is due to shortage of storage space. Finally, considering the document transformation phase, caching is highly beneficial, since it allows for performing only once transformations to frequently accessed documents, saving thus processing power and memory resources.

5.2 Proxy Server Performance

One consideration for the scheme of operation involving the proxy server is that under heavy loads, the proxy server may become a performance bottleneck,

Table 1

Image format		*Image size (KB)*		*Colour depth/type*	
GIF	70.9%	1-3	65,4%	1 bit	0,002%
JPG	27.9%	4-10	21,8%	Paletted images	70,89%
PNG	1.2%	11-50	11,3%	True colour	29,1%
		50-100	1,0%		
		> 100	0,4%		

downgrading the quality of service delivered to the clients. This section provides insights on the number of clients that can be efficiently served by a single proxy server, so as to provide telecom companies with data that will help them estimate the number of proxy servers needed, based on the number of estimated (or measured) users of ISDN card phone browsers. Firstly, the performance of individual operations (audio format conversion, image format conversion, image scaling and image dithering) is analysed, and afterwards the combined performance is estimated.

5.2.1 Image operations

In order to estimate the performance of image operations (format conversion, scaling and dithering), a mixture of images was selected, varying in format, colour depth and size. In order to get a representative sample of the images accessed in real environments, the contents of 15 *client caches* (i.e. files in the disk cache of Internet Explorer and Netscape Navigator browsers) and one proxy server cache (Apache 1.3.27) were collected, and the images therein isolated. The number of images in the final sample was 342.511 with a total size of 1,8 GBytes, with Table 1 statistical properties.

These statistical properties are considered important for the performance of the transformations since:

- image format affects the complexity (and thus the performance) of the decoding operations. For example, GIF images are LZW compressed and thus a decompression operation is required to convert to BMP format. On the other hand, JPG images are decoded by applying de-compression, de-quantisation and inverse discrete cosine transform (DCT) on the image, which is a more complex and time consuming process [25].
- image size has obvious impact on the conversion performance, since it directly affects the data size that the algorithms are applied on.
- colour depth and colour encoding affect the performance of the dithering process: indeed, for images with colour depth equal to 1 bit the dithering process will be either null if the colour encoded as 0 is the darkest and the colour encoded as 1 is the lightest (as determined by the image palette) or a trivial one's complement if the colour encoded as 0 is the lightest. True colour images are dithered using using the Floyd-Steinberg dithering algorithm [13] [14] [15], and so are paletted images (usually 16-256 colours); the only difference between true colour images and paletted images is that for the latter type, each pixel must be looked up in the palette (an additional step) to determine its actual colour.

Table 2

Operation	*Time*
Image format conversion	1732 sec
Image scaling (by 50%)	1811 sec
Image dithering	1780 sec
Combined format conversion, scaling and dithering	1860 sec

For the various operations on this statistical sample, the following performance metrics were derived (all measurements, a Pentium 4 machine running at 2,4GHz was used; the times below include the final step for compressing the resulting image using the gzip public domain library).

From Table 2 it is clear that the image decoding and final compression processes (which are included in all operations) is the most time consuming one, and it is highly beneficial if all operations (format conversion, scaling and dithering) are combined in a single "pass", so as to perform the image decoding process only once. Thus, for each image, the proxy server first decodes the image file, then applies scaling (if needed) and finally dithers the image to one bit colour depth (two colours).

Based on the above figures, the proxy server can sustain an image conversion rate of 0,99 Mbytes per second.

5.2.2 Audio operations

The basic audio operation that is supported by the proxy server regarding audio files is the conversion of complex audio types (e.g. MP3, OGG/Vorbis) to standard waveforms that can be reproduced by the ISDN card phone speaker. The ISDN card phone earphone speaker can play monophonic PCM –encoded waveforms with a sample rate of 8 KHz and 8 bits per sample. The audio conversion software is based on the PyMedia software library ([26]), which undertakes the decoding of complex formats. Once the format has been decoded, sub-sampling and quantization to 8 bits are performed as follows:

Table 3

File size		*Bit rate (Kbit)*		*Play time*	
3-10 KB	3,1 %	92	8,7 %	10 sec – 30 sec	11,3%
10-100 KB	12,4 %	128	36,2 %	30 sec – 2 min	73,2%
100 KB – 1 MB	28,2%	160	34,1 %	2 min – 5 min	14,8%
1 MB – 5 MB	56,3 %	192	21,0 %	> 5 min	0,7

(a) for sub-sampling, the original file sample rate R is initially extracted. The i-th sample in the sub-sampled stream is calculated by averaging the samples between (i * (R / 8192)) to (i * (R / 8192) + (R / 8192 – 1)). During this process, the maximum sample value SV_{max} is computed as well.

(b) for quantization to 8 bits, each sample in the sub-sampled stream resulting from step (1) is multiplied by a factor of (256 / SV_{max}).

The resulting audio stream is directly suitable for reproducing via the earphone speaker.

For evaluating the performance of audio operations, a mixture of MP3 files was formulated (MP3 is the predominant non-proprietary file format used in the Internet), with varying sizes (2KBytes up to 5 MBytes) and different bit rates, ranging from 96KBps (sub-cd quality) to 192 KBps (high quality). The statistical distribution of the size and bitrate properties is shown on Table 3.

From the performance tests for this type of operations, it was derived that the decoding/sub-sampling/quantization cycle is largely dependent on the file play time, giving a (conversion time) / (play time) ratio between 0.06 and 0.09, depending mainly on the bit rate (larger bit rates gave larger conversion times). Thus, a single proxy server may handle the conversion of 11-16 concurrent audio streams, since for larger numbers of audio streams the proxy server would not be able to meet the requirement of delivering data in real time, as needed by the audio medium nature.

Based on the above figures, and considering that the average bit rate is 149,22 Kbits per second, the proxy server can sustain an audio conversion rate of 1,94 Mbytes per second.

5.2.3 Sizing the proxy server

In order to size the proxy server, besides the performance metrics presented above, three additional parameters must be taken into account:

(a) A quantitative analysis of the data retrieved by the users per page, i.e. the bulk of textual, image and audio data downloaded with each page.

(b) the *user interaction pattern*, and in particular how long the user simply views the content on the ISDN card phone browser. Note that in this phase the user does not generate any traffic for the proxy server, *except for the case of audio*, where the user listens to the audible data through the ISDN card phone earphone.

(c) the *proxy* server cache hit rate, since if an object exists in the proxy server cache, it is simply forwarded to the requesting client, instead of undergoing the whole fetch/transform/forward process.

Table 4

Object Type	Number of files	Size of files
Text (HTML)	26.69%	22.54%
Image	57.79%	67.34%
Audio	0.10%	0.54%
Video	0.01%	1.68%
Application	0.15%	6.19%
Multipart	0.01%	0.01%
Other	15.22%	1.65%

Table 5

Object Type	Number of files	Size of files
Text (HTML)	31.56%	24.93%
Image	68.33%	74.47%
Audio	0.12%	0.60%

The statistical distribution (in number and size) of objects for each of the object types, is depicted in the Table 4 [27].

Since, out of these file formats, only text (html), image and audio are supported by the ISDN card phone browser, the metrics for the remaining four may be disregarded, giving the normalised Table 5.

Considering also that the average page size is 60 Kbytes (including text and graphics [28]), we can derive the following characteristics for requests for web objects:

(a) for each web page loaded, 2.17 images are transferred to the browser as well (on average). An audio file is requested every 263 web pages.

(b) the bulk of data loaded for each page is 14.96 KBytes of text and 44.68 KBytes of image, on average.

Regarding the time spent per page, a recent survey [29] has showed that a user views a page 46 seconds (on average), before moving to the next one (effectively thus before requesting any additional material from the proxy server). In the context of the ISDN card phone, this time has been found to be 12% longer, mainly because of the small viewing area and the limited navigation controls available; thus the "quiescent period" for the user is 51,5 seconds.

Finally, the cache hit rate for the proxy server has been found to be 65% for the text objects (html pages), 83% for the image objects and 54% for the audio objects. The lower percentage for the text objects is owing to the fact that a significant number of web pages is nowadays *dynamically generated* (ASP, PHP etc) and cannot be cached without the risk of displaying outdated information, whereas images are mainly static. Regarding the audio content, the cache hit ratio is small because it accounts for a very small percentage of the overall requests, thus the cache cannot easily be populated with a critical mass of files that would raise the percentage to significant levels.

Taking the above figures into account, together with the performance measures presented in sections 5.2.1 and 5.2.2 we can now calculate the number of users that a single proxy server can concurrently serve. Each request loads 44.68 KBytes of image, which are decoded/scaled/dithered/compressed in 0.044 seconds; considering however that this processing needs to be performed only for the 17% of the images (the other images are directly drawn from the cache), the proxy server needs on average 0.0075 seconds to handle the images in each request. The handling time for text files is minimal, measured to 0.00012 seconds, since the objects are simply either drawn from the cache or fetched from the hosting server and forwarded to the client. This gives an overall handling time of 0.00762 seconds for the images and graphics of a page, or –equivalently– 131.23 requests per second. This indicates that a single proxy server can effectively support approximately 6,750 concurrent users (regarding text and graphics), since the users spend 51.5 seconds for viewing a page, before issuing a subsequent request. Note that for supporting this number of clients, the proxy server must be equipped with a Gigabit Ethernet card or a 625 Mbit ATM connection, since the required amount of outgoing traffic is 432Mbits (64 Kbits per client). Considering that the average number of public pay phones in the European union is one pay phone per 546 citizens and (according to estimates) at most 10% of the installed ISDN card phones will be used at any given time for browsing, a single proxy server can serve a population of 30,000,000 citizens approximately.

Regarding the audio conversion, it is considered beneficial to use a *separate proxy server*, which will be dedicated to handle audio stream caching and conversion. This stems from the fact that audio conversion has real-time requirements, thus no "idle time" is present for users listening to audible content. Taking into account that:

(a) the average audible file playtime is 91,18 sec (cf. the playtime distribution in section 5.2.2) and
(b) such a request is generated every 263 web page requests, or -equivalently- each user will request for such a file every 13,544 seconds (on average), giving a "coverage" of 0.68% on the proxy server time (each user will need data from the proxy server for the 0.68% of his session time)
(c) the proxy renders an audio stream 13.5 times faster than the audio track play time and
(d) the cache hit rate for audio objects is 54%

It is derived that a single proxy server *dedicated to audio* conversions can handle 4,315 concurrent clients (render speed / (coverage * (1 – cache hit rate)). Given however that audio track conversions have real time requirements, a *safety limit* must be catered for, in order to avoid jitter problems due to server saturation, which will lead to degraded content quality. The number of clients that can concurrently be served by the proxy server can thus be estimated to 3,400, giving a gross analogy of "two audio conversion servers for each text/image proxy servers".

6 Conclusions - Future work

The implementation of a Web browser for the ISDN card phone offers significant added value for the ISDN card phone platform. The ability to receive and display Web pages upgrades the card phone from an access point to the telecommunications network to an information access device, capable for facilitating access to information resources, being in-line to the trends of the information age. The ISDN card phone, being a widely used and easily accessible apparatus, can play an important role in increasing the percentage of the population that make use of the online services. Installation of browsing facilities on an ISDN card phone platform may also offer a competitive advantage to telecom providers supporting this service, since citizens will prefer the providers that offer enriched services.

The software realising the web browser was designed keeping in mind the restrictions imposed by the ISDN card phone platform hardware, with one of the design goals being the delivery of high quality browsing experience. To this end, the range of supported media types was limited to fit the capabilities of the platform, considering the processing power, memory capacity and screen resolution. Through the introduction of a proxy server, the processing power and memory issues may be alleviated; screen resolution is expected to be tackled via advances in the hardware, in forthcoming versions of the ISDN card phone.

Future work will be targeted towards new generations of ISDN card phone hardware that will enable the support of a wider range of document types, including active content (e.g. Javascript and Java), since a considerable ratio of web-accessible services nowadays require support for such active features. Sizing of proxy servers is another

research direction, since telecom providers must install the appropriate number of proxy servers in order to make available the necessary resources for the operation of the card phones. This will be dependent on the number of ISDN card phones with browsing facilities installed and the percentage of the time that these devices will be used for Internet access.

References

[1] Arehart C, WAP and the Wireless World, in *Professional WAP*, Wrox Press Inc, 2000, pp. 10-16.

[2] Cooper I. and Dilley J., *RFC 3143: Known HTTP Proxy/Caching Problems*, Internet Engineering Task Force, 2001. Available from http://rfc3143.x42.com/

[3] Luotonen A. and Altis K., World-wide web proxies, *Computer Networks and ISDN Systems*, 27, 1994

[4] Charzinski J., HTTP/TCP connection and flow characteristics, *Performance Evaluation*, 42(2), 2000, pp. 149-162.

[5] Stevens W. R., *TCP/IP Illustrated: Volume 1. The protocols*, 10th Edition, Addison-Wesley Pub Company, 1997.

[6] Feldmann A.,. BLT: Bi-Layer Tracing of HTTP and TCP/IP, *Computer Networks*, 33(1-6), 2000, pp. 321-335.

[7] Fielding R. et al., *RFC 2068: Hypertext transfer protocol - HTTP/1.1*, IETF Network Working Group, 1997.

[8] Castro E., *HTML for World Wide Web*, Peachpit Press, 1998.

[9] Pfaffenberger B. and Gutzman A. D., *HTML Bible*, IDG, New York, 1999.

[10] Chidabaram N., Basic WML, in *Professional WAP*, 2000, pp. 102-121.

[11] Stevens R. W., *Unix Network Programming*, Prentice Hall, 1990.

[12] Berners-Lee T. et al. *RFC 1738: Uniform resource locators (URL)*, IETF Network Working Group, 1994.

[13] Floyd, R. W. and Steinberg, L., An Adaptive Algorithm for Spatial Grey Scale, *SID International Symposium Digest of Technical Papers VI*, 1975, pp. 36-37.

[14] Floyd R. W. and Steinberg L., An adaptive algorithm for spatial grey scale, *Proceedings of Soc. Inf. Display 6*, 1976, pp. 75-77.

[15] Heckbert P., Color Image Quantization for Frame Buffer Display, *Computer Graphics* 3(16), 1982, pp.297-307.

[16] Naiman and Lam., Error diffusion: Wavefront traversal and contrast consideration, *Proceedings of GI 1996*, pp. 78-86.

[17] Lindstrom et al., Real-time, continuous level of detail rendering of height fields, *Proceedings of the SIGGRAPH 1996 conference*, 1996, pp. 109-118.

[18] Whitted, T., An improved illumination model for shaded display, *Communications of the ACM* 23(6) , 1980, pp.343-349.

[19] WAP Forum, *The form of the DTD used by WML documents*, 2000, available through http://www.wapforum.org/DTD/wml_1.1.xml.

[20] Marchal B, *Parser and DOM in XML By Example*, Que publications, 1999.

[21] WAP Forum, *WAP WML Specifications Version 04*, 1999, available through http://www.wapforum.org.

[22] Mann S., Wireless Markup Language in *Programming Applications with the Wireless Application Protocol*, John Wiley & Sons, 1999.

[23] Kajiya, J., The rendering equation, *Proceedings of the SIGGRAPH 1986* conference, 1986, pp.143-150.

[24] Arvo J., Kirk D., Particle transport and image synthesis, *Proceedings of the SIGGRAPH 1990 conference*, 1990, pp. 63-66.

[25] Wallace, Gregory K., The JPEG Still Picture Compression Standard, *Communications of the ACM*, vol. 34, no. 4, April 1991, pp. 30-44.

[26] PyMedia Project, *PyMedia Software Library*, accessible at http://pymedia.sourceforge.net/

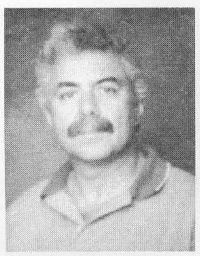
[27] Ortega Antonio, Carignano Fabio, Ayer Serge and Vetterli Martin, Soft Caching: Web Cache Management Techniques For Images, *IEEE Signal Processing Society 1997 Workshop on Multimedia Signal Processing*, June 23-25, 1997, Princeton, New Jersey, USA.

[28] About All Things Web, *How Much Is Too Much*?, 1999, accessible at http://www.pantos.org/atw/ 35654.html

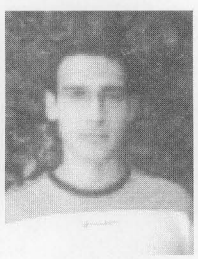
[29] Nielsen Ltd., *Global Usage*, March 2004, accessible at http://www.clickz.com/stats/big_picture/traffic_patterns/article.php/3348651

[30] ESIS II, *Basic facts and indicators for the European union*, 2001, accessible at http://www.eu-esis.org/Basic/HomeBasic.htm

Biographies



Dimitris Maroulis obtained his B.Sc. degree in Physics, the M. Sc. degree in radioelectricity, the M. Sc. in electronic automation and the Ph.D. degree in informatics, all from University of Athens, in 1973, 1977, 1980 and 1990, respectively. In 1979, he was appointed Assistant in the Dept. of Physics, in 1991 he was elected Lecturer and in 1994 he was elected Assistant Professor, in the Dept. of Informatics of the same university. He is currently working in the above Dept. in teaching and research activities, including projects with European Community. His main areas of activity include data acquisition systems, real-time systems, signal processing and digital communications.



Sotiris Aronis has graduated from the Department of Informatics and Telecommunications Science of the University of Athens, Greece, in 2002. He is also candidate for an MSc in Communication Systems and Networks from the same University. He has been working as an embedded systems software engineer from 2000 until now. His research interests include Protocols Implementation and Hardware Configuration for Real Time Systems.

Vasiliki Nassiopoulou has graduated with Excellent Degree from the Department of Informatics and Telecommunications of the University of Athens, Greece, in 2002. She has been awarded with the first prize of her class. She is also candidate for an MSc in Technoeconomic Systems from National Technical University of Athens. She has been working as a system analyst for four years. Her research interests include Protocols Implementation and Hardware Configuration for Real Time Systems.

Nikolaos Grammenos has graduated from the Department of Informatics and Telecommunications Science of the University of Athens, Greece, in 2002. He is also candidate for an MSc in Technoeconomic Systems from National Technical University of Athens. He has been working as a system analyst for four years. He has been awarded with the prize of Excellence in Telecommunications from Ericsson Company. His research interests include Wireless Communications, Embedded implementation and Application software for Real Time Systems.

Costas Vassilakis is currently an Assistant Professor in the department of Computer Science and Technology of the University of Peloponnese. He holds a B.Sc. in Informatics (1990) from the Department of Informatics, University of Athens and a Ph.D. from the same department in 1995. He has participated in several European and national projects and has published more than 30 scientific papers in international journals and conferences. He has been a consultant for the General Secretariat for Information Systems of the Ministry of Finance, Greece, and his research interests include systems software, web information systems, .databases and parallel and distributed systems.