Chapter X

Managing the Interactions Between Handheld Devices, Mobile Applications, and Users

Maristella Agosti and Nicola Ferro
University of Padua, Italy

ABSTRACT

In this work we present the problem of managing the interactions between handheld devices, mobile applications and users that are the three main entities involved in providing information access in the mobile scenario.
After introducing a general framework, which gives the reader a comprehensive view of the features of a handheld device and presents some techniques to overcome the constraints imposed by handheld devices during the interaction with users and mobile applications, we focus our attention on the problem of searching for information using a search engine through a handheld device. We present the motivations for studying this kind of search engines and a discussion on their features and design choices, explaining the relevant aspects that need to be addressed during the design and the development of these mobile applications.

INTRODUCTION

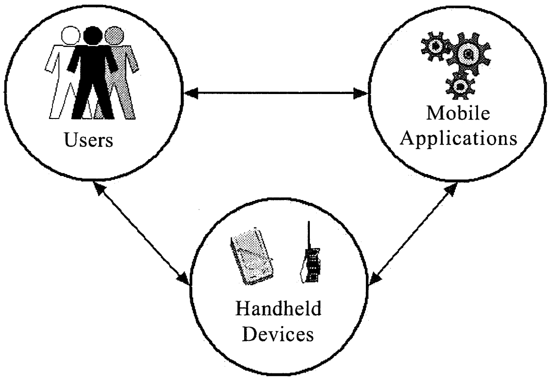
The goal of this chapter is to introduce the reader the range of issues that need to be addressed for managing the interactions between handheld devices, mobile applications and users. Specific attention is given to the interactions between handheld devices and mobile applications that use search engines for information seeking and discovery, because search engines are often used in mobile applications that provide information and services.

There are three main types of entities involved in the interactions:

- *handheld devices*: Examples of handheld devices[1] include *Personal Digital Assistants* (PDAs) and mobile phones;
- *mobile applications*: Among the mobile applications which can possibly be considered for implementation, this chapter will focus on Web search engines. Search engine services are particularly useful to end users who seek information for work or pleasure using handheld devices;
- *users*: There are different categories of mobile users looking for information resources to solve some emergency needs or to resolve their *Anomalous States of Knowledge* (ASK) (Belkin, Oddy, & Brooks, 1982).

The relationships between the three types of entities are illustrated in Figure 1, where it is shown that one has to consider human/computer interactions when presenting content on the handheld devices and user requirements when designing search capabilities for mobile applications. Another aspect that needs to be considered is related to the interaction and communication between the handheld devices and the mobile applications, as the search capabilities of mobile applications are affected by the content presentation characteristics and the limitations of handheld devices.

Figure 1: Main entities involved in the mobile computing scenario



This chapter begins with a section that motivates the study on managing the interactions between entities in mobile scenarios. It concentrates on the interactions between handheld devices and mobile applications. Several problems need to be addressed in the use of handheld devices to present information and to adapt information content available on the Web to the handheld devices. Many examples are given on how these problems can be addressed. A general framework is introduced, giving the reader a comprehensive view of the features of a handheld device, and the differences between the handheld device and a desktop or portable computer including the constraints imposed by these differences. Some approaches will then be introduced to overcome the constraints imposed by handheld devices.

The chapter then presents the problems concerning the development of a search engine for handheld devices. Indeed, it could be thought that the problems of retrieving information end when the user has a good browser, but that would be a misleading conclusion. It is necessary to provide the user with a tool that allows him to find and reach relevant sites that otherwise would be difficult to know about. It will be shown how the considerations concerning surfing have to be changed and how everything is more complex, due to the features of handheld devices. In this way the reader will be made aware of the interaction problems with search engines when using handheld devices and the architecture required to address these problems.

MOTIVATIONS OF MANAGING THE INTERACTION BETWEEN HANDHELD DEVICES, MOBILE APPLICATIONS AND USERS

In *Mobile Commerce* (m-commerce), users will have to use handheld devices such as PDAs and mobile phones to access different types of services. These kinds of devices have special features that distinguish them from a desktop or portable computer.

Typically, through the handheld devices, users can access the services provided by the ordinary electronic commerce (e-commerce) sites. However, these e-commerce services designed for desktop computers usually do not suit handheld devices well. Due to their unique features, the handheld devices could become a bottleneck in accessing e-commerce services—not allowing the user to fully appreciate the information and services offered by the e-commerce sites.

The key approach to overcome the limitations of handheld devices is *content adjustment* that refers to the information presentation techniques and techniques required to manage user's interactions considering the features of handheld devices.

Over the years, many enterprises have invested in applications to manage their data, and now these are the legacy applications. Legacy applications were built in an environment where users interacted with the applications through dumb terminals. Now enterprises need to extend their reach to customers and business partners using pervasive devices and the Internet, but the potential risks and costs prohibit companies from porting these existing applications to any new technology, as observed by Britton et al. (2001). Using content adjustment, the legacy applications ported to the Internet environment could still be accessed by a variety of handheld devices, allowing more effective *Business to Business* (B2B) and *Business to Consumer* (B2C) interactions.

Finally, *Third Generation Mobile System* (3G) introduces the "any time, anywhere, any device" paradigm; and, in order to respect the "any device" constraint, the content needs to be independent of the handheld devices, and so it has to be adjusted to the various devices on which it will be displayed.

DESIGN ALTERNATIVES FOR OVERCOMING THE CONSTRAINTS OF HANDHELD DEVICES

We now analyze the handheld device features that constrain the design of a mobile application and which need to be addressed by content adjustment:

- screen size;
- input method;
- wireless link;
- small memory;
- slow *Central Processing Unit* (CPU);
- energy consumption.

Screen Size

The screen of a handheld device is a very limited and valuable resource. The small screen reduces the amount of information that can be displayed at any one time and can either make a Web page, that has been designed for a desktop computer, difficult to read, or it can force the user to do a great deal of scrolling.

Most Web pages are designed to be displayed on the screen of a desktop computer, with colour and with a resolution of at least 640×480 pixels, while other pages may be designed with higher resolutions in mind. For a PDA, the most common resolution is 160×160 pixels. This means that only 1/12 of the Web page can be displayed on the PDA. This may lead to a good deal of scrolling, both vertically and horizontally. If the PDA allows a long horizontal line to be wrapped into multiple lines, it may save some horizontal scrolling but will significantly compromise the readability of the page.

Nowadays, many Web browsers for handheld devices display Web pages without considering their screen sizes. Thus, pages do not fit properly onto these small screens, and Web surfing becomes very tedious, reducing a site's effectiveness.

In order to address these problems, various approaches can be adopted, as classified in Bickmore and Schilit (2002):

- device-specific authoring;
- multiple-device authoring;
- client-side navigation;
- automatic reauthoring.

Device specific authoring involves authoring a set of Web pages for the display device of a particular handheld device, be it a mobile phone or a PDA. The users will only have access to a selected set of services. All pages for these services are designed up-front for the display of some particular handheld device(s).

This is the approach taken in *Handheld Device Markup Language* (HDML) by *Phone.com, Wireless Application Protocol* (WAP) and *Compact HTML* (C-HTML) by *Nippon Telephone & Telegraph* (NTT) DoCoMo.

Screen size directly affects the amount of information displayed; and page layout needs to be planned accurately, so that information can be easily accessed by the user.

As suggested in Metter & Colomb, (2000), the page designer should:

- maintain user orientation during navigation by means of a title showing at what point of the site the user is at;
- verify that long text lines do not exceed screen dimension, compromising readability; and
- rearrange complex tables, made up of many columns, into lists so that they do not exceed screen limits.

Furthermore, scrolling should be reduced by:

- placing surfing aids, such as menu bars, in a fixed place near the top of a page;
- placing key information at the top of a page; and
- reducing the amount of content in a page.

Finally, small-screen users seem to choose and prefer direct access strategies to information, as illustrated in Jones, Marsden, Mohd-Nasir, Boone, and Buchanan (1999); Jones, Marsden, Mohd-Nasir, and Buchanan (1999); Jones, Mohd-Nasir, and Buchanan (1999); and Buchanan and Jones (2000), so the designer should:

- provide a search mechanism within a website; and
- organise the information so that the navigation is focused; for example, suggest to the users a list of goals they might want to achieve and present a framework designed for facilitating such access.

Multiple-device authoring identifies a range of target devices and defines the mapping from a single source document to a set of rendered documents that apply to the different target devices.

For example, StrecthText (Cooper & Shuebotham,[2] 2002) allows the page designer to tag portions of the document with a "level of abstraction measure." When the user receives the document, he can select the desired level of abstraction and the document is presented with the corresponding degree of detail.

Another example is *Cascading Style Sheets* (CSS) (Lie & Bos, 1999) of *HyperText Markup Language* (HTML), which allow the page designer to define, within a single style sheet, a set of display attributes for different structural portions of a document. For example, all top-level headings have to be in bold or the anchor text of a link has to be red. In general, a set of style sheets can be associated with a document, and the same style sheet can be shared by different documents. When the user retrieves a document, he also retrieves the set of style sheets associated with that document. For each style sheet in the set, there is a weight that measures the degree of satisfaction of applying that style sheet to the document. The CSS cascade mechanism assigns a weight to each style rule. When several rules apply, the one with the greatest weight takes precedence (W3C, 2002). It is relevant to note that in October 2001 the *World Wide Web Consortium* (W3C) produced the CSS Mobile Profile 1.0 (W3C, 2002).

Client-side navigation allows the user to navigate interactively within a single page, modifying a portion of it that is displayed at a time.

A trivial example is the use of scroll bars. Another example is PAD++ (Bederson & Hollan, 1994), in which the user can zoom in and out over portions the document. Active outlining (Hsu, Johnston, & McCarthy, 2002) is also an example of client-side navigation, because the user can dynamically expand or collapse sections of the document under their section headings.

Automatic re-authoring requires the development of a software capable of processing an arbitrary Web document designed for a desktop computer and adjusting it through a series of transformations to the screen of an handheld device. This process can be made on the client, on the server or on a proxy server, which has the single task of providing conversion services.

There are many possible reauthoring techniques, which can be categorised along two dimensions: *syntactic* vs. *semantic* and *transformation* vs. *elision*, as classified by Bickmore & Schilit (2002).

Syntactic techniques operate on the structure of the page, while semantic techniques require a certain degree of understanding of the content of the page. Transformation techniques modify the content or the presentation of a page, while elision techniques remove some content, leaving the rest unchanged.

So we can have the following cases:

- *syntactic elision*: portions of text or objects within a document are removed, considering the structure of the document or the kind of object. Examples are active outlining, displaying only the first paragraph of each phrase and removing images;

- *syntactic transformation*: portions of text or objects within a document are reorganized, considering the structure of the document or the kind of object. Examples are transforming a complex table into a list and image reduction;
- *semantic elision*: portions of text or objects within a document are removed, upon an understanding of their meaning. An example is the removing of duplicate links; and
- *semantic transformation*: portions of text or objects within a document are reorganized, upon an understanding of their meaning. Examples are text summarisation and reorganisation of all links within a page into a list.

A full example of automatic reauthoring is Power Browser (Buyukkokten, Garcia-Molina, Paepcke, & Winograd, 2000; Buyukkokten, Garcia-Molina, & Paepcke, 2000; Buyukkokten, Garcia-Molina, & Paepcke, 2001b; Kaljuvee, Buyukkokten, Garcia-Molina, & Paepcke, 2001; Buyukkokten, Garcia-Molina, & Paepcke, 2001a; Buyukkokten, Kalijuvee, Garcia-Molina, Paepcke, & Winograd, 2002), which utilises semantic and syntactic techniques to improve the user's interaction on a PDA.

Power Browser is a proxy server that transforms Web content before delivering it to a handheld device.

During navigation, a set of link descriptions is shown, heuristically generated by anchor text, *Uniform Resource Locator* (URL) structure and ALT tag. This structure, organised in a tree, not only includes links on a single page but also a hierarchical structure of links on linked pages, so the user can directly retrieve a page from a link description which is visible on the screen.

Once the desired page is found, the user can view some of its content. Images are not displayed, and the text contained in the ALT tag is displayed instead of images; images can be shown on user demand, after a refinement step of adjustment of the image to the low-screen resolution, where the step is performed by the proxy server.

A summary is presented for each page, obtained by partitioning an original Web page into fragments, for example paragraphs, lists or ALT tag describing images, and selecting the most important fragment as a summary.

Power Browser also tries to address the problem of filling in forms on a PDA screen.

Forms constitute a problem on a PDA and other handheld devices because input controls and associated textual explanations occupy too much space on the screen; therefore, it is difficult for users to gain an overview of the form's content and of what is required of them. For example, if the user has to fill in a form registering for a service, he could interpret the name field as a name and surname field and fill in those two data instead of the name only. When he proceeds by scrolling the bar he finds that there is another field asking for his surname. At this point the user has to go back to the previous field and correct the wrong information submitted. The problem of filling in forms needs to be carefully addressed in an m-commerce site, because asking users for information is quite a frequent operation.

Instead of showing all the input controls at once, Power Browser initially shows only minimal textual prompts for the input controls. When the user is ready to fill in information, a pen gesture on a textual prompt causes the associated input control to be displayed, while all other input controls remain hidden. In general, all the text other than labels is ignored, so each visible string is a label for an input control. Buttons are expressly identified and links are handled as usual.

A different solution is WebViews (Freire, Kumar, & Lieuwen, 2001), which allows end-users to create and maintain simplified views of Web content with a *Video Cassette Recorder-* (VCR) style interface.

In the following, "WebViews" means the name of the software, "Web view" is a view of the Web obtained using WebViews, and "Web views" is a set of views of the Web obtained using WebViews.

A user can create a Web view from a desktop computer by simply browsing the desired page and selecting the content of interest within the page, while the WebViews recorder registers all intermediate steps and actions undertaken, such as interacting with *Common Gateway Interface* (CGI) scripts and so on. Therefore, a Web view is much more than a simple bookmark because it also stores the information needed to interact with scripts. The Web view created is robust to some changes in page layout so, if the site does not change too much, the user can avoid recreating the Web view. When the user accesses the Internet through a handheld device, he can recall the created Web view from the Web views' server which accesses the previously selected content on the desired Web page, clips it and returns an *eXtensible HyperText Markup Language* (XHTML) response to the client. The client could also be an intermediate proxy, which performs content adjustment and translates the XHTML content into various formats, e.g., *Wireless Markup Language* (WML).

WebViews facilitates and shortens the subsequent phase of automatic reauthoring, since it provides the reauthoring module of a simplified version of the Web page; this contains only the content desired by the users and avoids all the intermediate steps.

To address the problem of filling in forms on a handheld device screen, WebViews allows the user during the recording phase to specify which field values are to be stored in the Web view specification itself and which ones are to be requested by the user every time the Web view is executed.

The *International Business Machines* (IBM) Websphere Transcoding Publisher (Britton et al., 2001; IBM, 2002) is a commercial product which can be considered as being halfway between a multiple authoring and an automatic reauthoring device. It uses, in fact, users' profiles and both *eXtensible Markup Language* (XML) and *eXtensible Stylesheet Language* (XSL) style sheets for content adjustment, just as in multiple device authoring. Another important characteristic is that it transforms HTML into WML *cards* and *decks*; to perform such a transformation, it uses syntactic techniques such as image reduction and semantic techniques, such as rearranging text into WML cards and decks.

Strengths and weaknesses of those different approaches

"Device specific authoring" typically gives the best results, but it limits users to a small subset of Web pages. Furthermore, managing and keeping a site updated is an onerous task, requiring hours of manual work.

"Multiple device authoring" requires less effort for a single document than device-specific authoring, but it requires a considerable amount of manual design, and it limits the users to a subset of Web pages.

"Client-side navigation" is a promising approach if a set of good techniques can be developed. In fact, the PAD++ "magnifying glass" approach can be very awkward if the documents are large; the active outlining approach can have a limited applicability, since many Web pages are not organised into sections and subsections.

"Automatic reauthoring" is thus the ideal approach to provide a broad access to the Web through a wide range of handheld devices, where it is possible to produce legible and aesthetically pleasing documents that can be easily surfed without loss of information.

An approach such as that taken in WebViews can help the automatic reauthoring process, but it still requires a certain degree of user interaction to prepare the Web view, and that same Web view needs to be recreated if many changes are made to the original page. Moreover, the process of creating a Web view is managed on a desktop computer, and that requires the user to plan which pages he will access from the handheld device.

Input Method

During normal Web surfing, the user can utilise both the keyboard and mouse to reach every object on the screen and to provide input to forms easily.

The input in a PDA is done through a pen, which makes both gestures and character input possible. An advantageous aspect of this input method is that the engine for character recognition could be shared with the engine for gesture recognition, as illustrated in Moran, Cheyer, Julia, Martin, & Park (1997).

Characters are written by the user in a special area of the screen and are then recognised via *Optical Character Recognition* (OCR). With this approach text input is subject to mistakes, and the user has to write characters with some care to ensure the system recognises them; this process increases the time required to write a word. An alternative is to draw a virtual keyboard on the PDA's screen so that the user can choose the characters with the pen and compose a word. Even this approach is time consuming and can lead to the user making mistakes.

The use of a pen allows the user to reach any object placed on the screen, simplifying the design of objects layout, but not input difficulties. Furthermore, gestures made with a pen are generally more varied than those that can be done with a mouse. For example, a top-down gesture could mean vertical page scrolling.

Input method on a mobile phone is a great constraint, as there is the numeric keypad and a scrolling key. The WAP Forum requires at least vertical scrolling, so no horizontal or diagonal scrolling is guaranteed. A consequence is that the process of surfing a page can be difficult and frustrating, looking for links to other pages or consulting long lists. Moreover, the user has to press the keys many times to form words. In order to address the page-surfing problems, page layout should be carefully planned—making frequently used functions easily reachable. Metter & Colomb (2000) suggest helping the user by gathering all the links within a page in a permanently available list, eventually combining these links with keys on the numeric keypad.

Word completion techniques should be employed to manage text input problems, both for PDAs and mobile phones. We can distinguish between general purpose techniques, meaning techniques developed generally to overcome the constraints of the device, and specialised techniques, meaning techniques tailored to the user and the application.

General purpose techniques try to complete words according to a predefined dictionary, differentiating between one word and another as characters are entered.

A technique developed for handheld devices generally is *Predictive cOmposition Based On eXample* (POBox) (Masui, 1999), which is organised into two steps: filtering step and selection step. During the filtering step, as the user enters characters, the system dynamically uses those characters to look for candidate words in a dictionary. During the selection step, the user can select the desired word from among candidate words. Then the next input words are predicted from the context and passed on to the next filtering step.

Many solutions were studied to facilitate text input through the numeric keypad of mobile phones, proposing a way to introduce words by pressing keys only once, such as in Tegic (Tegic, 2002) *Text on 9 keys* (T9), which is a software adopted on many mobile phones and which is available for many languages. Every time there is an ambiguity between words, T9 shows the user a list (by means of an internal database) with the possible terms in order of utilisation frequency, so that the user can choose the correct one. Furthermore, the internal database can be updated with new words added by the user, when he introduces words unknown to the system. GSMBox (2002) reports a study made by Nokia according to which text input with T9 is twice as fast as conventional methods, and the Nokia study confirmed the user's desired word is the first choice 95% of the time.

Specialised techniques suggest words to users, choosing them from a well-defined set personalised for the user, his needs, and features of the applications with which the user is interacting.

For example, Power Browser offers support for keyword entry during a local search session within a site: for every site that the user is visiting, word completion and measures of keyword selectivity are provided. As users enter successive keywords, the system shows how many pages match the query, and users can submit queries only when the keywords are selective enough.

The approach adopted by WebViews facilitates text input, because it avoids all the intermediate steps, thus requiring less input for surfing the Web; input forms can be partially or fully filled in during the Web view creation phase, decreasing the problem of inserting words with a handheld device.

So, if in general we can expect the presence of general purpose techniques on the device, peculiarities of the user or of the application can require the developing of more specialised techniques.

A very attractive alternative for solving text input problems is speech recognition. It is certainly very useful but still not widely adopted, due to the problems of implementing voice-independent recognition systems, where little or no information is provided about the domain to which the text to be recognised pertains.

Wireless Link, Small Memory, Slow CPU and Energy Consumption

A handheld device uses a wireless link to transfer data to and from a server. The bandwidth available nowadays is very low, and as a result the link between the client and the server is very slow and does not allow large data transfer. This problem will be partially overcome in the future by 3G networks. An example of this trend is the *General Packet Radio Service* (GPRS), which is an evolution of the *Global System for Mobile Communication* (GSM) and a technology emerging today that achieves performances that are roughly the same as those of a standard analog modem, in a favourable case scenario. Furthermore, wireless networks suffer from high latency.

The WebViews approach addresses low bandwidth problems, since it minimises the amount of data transferred between the handheld device and the server, as well as high latency problems because it avoids repeated access to the Internet for fetching intermediate pages needed to reach the desired page.

A handheld device has a small amount of memory, and it is impossible to manage complex pages, such as a page containing a large image or a high amount of data. The CPU is quite slow and computational intensive operations should be avoided, as otherwise the user has to wait far too long. Moreover, battery energy needs to be saved in order to guarantee the autonomy of the device, and this is a constraint for the operations that can be done on the client, since the CPU operations are expensive in terms of energy consumption.

The problems regarding computational load distribution affect the choice of which component has to provide the various functions to help the user—either the server or the handheld device: certainly the server has the power required to manage the various tasks needed to facilitate surfing, but response time from the server to the handheld device needs to be considered, since it can compromise performances. So there is a trade-off, as some computational load has to be moved to the client to guarantee the system some efficiency. This is at least until the slowdown caused by the processing action on the client is below the time required to do the same processing on the server and to send data to the handheld device.

Ojanen & Veijalainen (2000) give an example of these considerations and the method according to which various parameters have to be evaluated. Ojanen & Veijalainen (2000) study WML code compression before it is sent to a mobile phone, so that the amount of bytes to be transmitted decreases and the transmission time is reduced. It is explained how to determine whether this choice is convenient or if the overheads introduced by compression and decompression of the code outweigh the gain obtained from reduced transmission time.

MOBILE SEARCH ENGINES

In the previous section, the adjustment of contents to handheld devices was analysed, which constitutes a part of the interaction process for the user seeking information. In this section, we analyse the other part of the interaction process, i.e., that of having a *Search Engine* (SE) available for use and access from handheld devices.

This type of SE can be called a *Mobile Search Engine* (MSE), stressing that these SEs need to be designed and implemented for direct access by any type of mobile device.

The first part of this section clarifies the motivations for studying MSEs. The second part presents a general discussion on MSE features and related problems. The final part presents the design of Odysseus, an ongoing MSE project initiated by the authors of this chapter.

Odysseus is the name of our project for designing and implementing an MSE. At present Odysseus is only in the early stages of design, so it is used here as an example for discussing relevant aspects that need to be addressed in designing mobile applications to retrieve information for a user not using a desktop computer but a mobile device. We will explain the concepts, architecture and solutions for Odysseus, which are all under careful evaluation.

Motivations for Studying MSEs

Search engines are often used to discover and find information on the Web. To search the Web using handheld devices, a specific type of search engine will be required.

The user could utilise a search engine on handheld devices for the following reasons:

- to choose between services offered, including those of m-commerce, and to find one which best satisfies his needs; or
- to retrieve information of every-day interest for work and pleasure.

With regard to the first point, an analysis of the situation leads to the consideration that the amount of applications offered and the range of resources available require a tool that allows the retrieval of services and information in a quick, simple and easy manner. If there are sites offering online trading, m-commerce, entertainment and so forth, it is necessary to have an application which finds what is required, since the burden of knowing and remembering various sites cannot be passed on to the user.

The *Universal Mobile Telecommunication System* (UMTS) Forum (UMTS Forum, 2002) also considers the mobile portal as the way of providing access to 3G services and, within the mobile portal, search engine is an important component.

With regards to the second point, the user, in addition to coping with prepacked proposals such as those that are managed by the organisation he works with, may desire to use the power of the Internet as a source of resources and, therefore, to search for documents and information on specific topics of interest for work and pleasure, as with traditional search engines.

At this point someone could wonder how the inquiring and retrieving can take place on a handheld device. The following scenarios can be shown:

- *directly legible document*: if the size of the document is not too big, it can be displayed directly on the handheld device's screen, after a content adjustment phase;
- *not directly legible document*: if the size of the document does not allow for its online reading, or, if the kind of file does not permit its browsing (such as a binary file), the search result can be sent to a workplace capable of managing the resource; in this case, the actual use of the document would be possible later. While the user is moving, the search can serve to find useful documents and some other tool can arrange in advance a source of interest at destination, which can be ready when the user arrives at destination, so that it can be used immediately—improving efficiency;
- *printable document*: if the user has a portable printer or can connect to a printer or a fax, he can immediately print the search result and use it, even if the search result is not easily readable on a screen. This option conforms with the concept of "any device," and it is already in line with the Bluetooth (Bluetooth, 2002) standard for wireless interconnection of heterogeneous devices.

The study of searching by means of handheld devices should not be limited to SEs but should be extended to *Information Retrieval Systems* (IRS)s generally and *Digital Libraries* (DL)s. Indeed, mobility could improve the utilisation of a DL: for example, a person can perform a search on a DL while moving between the physical shelves of a library, as observed in Marshall, Golovchinsky, & Price (2001).

Furthermore, as suggested in Marshall et al. (2001), searching and reading a DL can be done together with other activities, such as working with colleagues, alternating searching with writing and organising materials. During teamwork, one person can add annotations to documents of a DL and another person can search the DL for documents with a specific annotation. Merging content and wireless communication can develop ubiquitous access to DLs, improving well-established collaborative practices and exploiting physical and digital resources.

Content Adjustment for MSE

In the previous section the motivations that can lead to the use of search engines from mobile devices were shown. Those considerations suggest that a Web search engine cannot be used directly from a handheld device, but its use should be adjusted to the peculiar features of the types of those devices. Thus, this section addresses the different aspects that have to be taken into account when the content, which constitutes the result of a query to a search engine, needs to be adjusted to a handheld device. The presentation follows the same sequence of presentation of device features that has been given previously, so it constitutes a parallel analysis and study of the previous one.

Screen size. The output of a query to a traditional search engine is shown, in multiple pages, as a list of links with a small paragraph of text, taken from the original Web page, describing each Web document of the result. Considering the small screen of a handheld device, search results cannot be displayed in a traditional way, because the information displayed could be difficult to read and the user would be compelled to scroll a long document to look for interesting links.

Therefore, it is necessary to identify and design a more compact way of displaying search results, different from the simple traditional ranked list. This new way of presenting results should allow the user to gain an overview of the search results and to reach the most relevant links easily.

Input method. Another problem is the input method: if during a simple navigation this problem could be less urgent, the user's interaction is greater in the use of a search engine, since the user inserts the query, the search engine returns a set of results, and, eventually, this cycle can be repeated as many times as is necessary to improve the search.

Therefore, the user needs to be facilitated with a general feature of word completion and a more focused suggestion of relevant keywords, tailored to the user's areas of interest.

Wireless link. The low bandwidth and high latency problems also need to be borne in mind, as a generic query could produce a long list of results and the query refinement process leads to frequent communications between the handheld device and the server.

To address these problems, a measure of the selectivity of a query should be estimated so that the user can actually submit the query when he is sure of not receiving too many results; this reduces the number of steps otherwise necessary to reduce the size of the set of results to improve the precision of the query.

Caching can help to face these problems—both local caching on the handheld device, as it allows the return to previous results, and server caching, which allows pages' prefetching while the user is reading the results.

Customizing the search engine. As illustrated generally for 3G services (UMTS Forum, 2002), a necessary feature for a search engine for handheld devices is the possibility of customising the application according to the user's needs. Today, Internet services are already offered as portals, highly configurable, and the user of handheld devices can expect to have similar services.

Customisation is important since most users mainly use a specific search engine, because it satisfies their needs adequately in terms of topics' coverage; also, they know its features and query language better. Therefore, the interface of the preferred search engine needs to be shown to a specific user, and it needs to be tailored to the specific device.

Customisation, as better knowledge of a user's needs, is also advantageous for the application, since it allows anticipation of the user's requests, offering a better service, and gets to know a user's vocabulary and topics of interest, which information is important for word completion and keyword suggestion.

Method of implementation. At present the change to 3G services is made as smoothly as possible, and this is witnessed from the introduction of 2.5G networks, as, for example, for GPRS networks. From this point of view, the implementation of a search engine to be used from handheld devices should be planned so that it will not require the redesign of existing systems; rather, it should be added to them as a new layer, permitting the start of the path towards 3G.

The current proposal, at protocol level, to bring the Web to mobile phones is WAP, which was developed by a consortium of manufacturers with the aim of making a standard.

However, WAP is designed for current mobile phones, but it is improving to ensure the support of those that are 3G. Once 3G mobile phones are on the market, other solutions such as XML (Leavitt, 2000) could be used.

The guidelines, which will be used in the design of an MSE, should not be bound too much to current standards; they should rather resolve the problems related to the kind of device and application offered, than exploit today's technology features, so that the design of an MSE is easily portable in the event of a change of standards and technology.

On the other hand, it should be shown how the design choices and solutions adopted could be fulfilled in an actual application. From this point of view, developing a prototype with WAP could be advisable; it would allow accurately documenting the phases of the implementation process, so that these could be used as a paradigm.

A relevant application for consideration is "Pirate," the recent proposal for a search engine for the Palm made by IBM. Pirate is the *Palm Information Retrieval Application for Text sEarch* (Pirate) (Aridor, Carmel, Maarek, Soffer, & Lempel, 2001; Aridor, Carmel, Maarek, Soffer, & Lempel, 2002), which allows users to predefine a topic of interest and then capture a very small and representative set of Web pages for the topic, storing it in a persistent repository called *Knowledge Agent Bases* (KAB). The process of creating a KAB can be called from a desktop computer or from a PDA. The knowledge acquisition is then performed on a desktop computer, and the resulting KAB is downloaded to the PDA.

The KAB is created starting from a set of four to six queries supplied by the user for defining the topic and/or from a set of seed URLs which the user considers relevant to the topic. A *Knowledge Agent* (KA) then browses the Web looking for pages relevant to the topic.

Stored in the KAB are a topic-specific lexicon, a small number (roughly 100) of core pages whose full text is stored in the KAB, a larger number (roughly thousands) of Web pages pointed from the core pages, for which only the URL and anchor text are stored, and an index for searching the KAB.

Pirate offers both word and query completion by using a domain-specific vocabulary from which it suggests, as keys are pressed, the most frequent words in the vocabulary consistent with the input prefix. Furthermore, query completion is performed by suggesting terms related to the query terms already introduced using a global semantic word network.

This solution has the advantages of allowing users to perform a search task with few or no data exchanges with the server, addressing low bandwidth and high latency problems, and completing words and queries in a very focused way, facing text input problems.

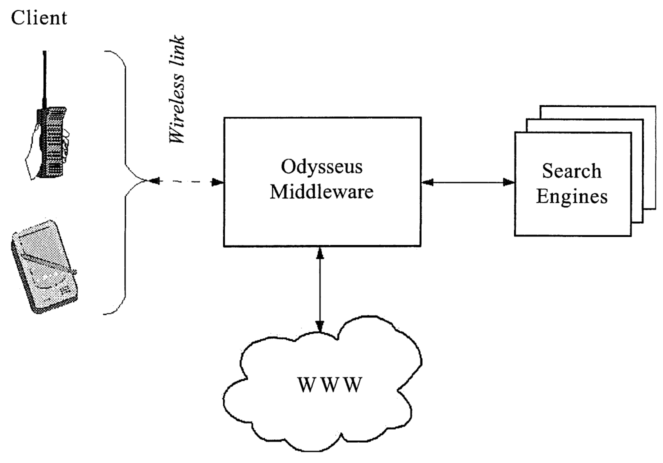
On the other hand, Pirate requires users to arrange in advance the KAB, not offering support and help for an online search of topics not stored in the KAB. Moreover, Pirate does not implement advanced features for addressing limited screen and scrolling problems.

Odysseus: An Example of an MSE

In order to understand all the problems related to the development of search engines for use on handheld devices, it seems important to underline the most relevant key features for their design. In Ferro (2001) a proposal called Odysseus was introduced. The most relevant characteristics of that proposal are reported by Ferro (2001) and are used here as a useful example, permitting the presentation of all the different aspects that have to be addressed when designing and developing a mobile search engine.

Odysseus is organised according to the client-server paradigm: the client sends a user's query to the middleware server, which forwards it to the selected search engine, collects the search results, and organises them for their display them on the client. The general architecture of Odysseus is shown in Figure 2.

Figure 2: Odysseus architecture



The middleware server provides, in general, two kinds of connection: a wireless connection towards mobile clients, and a wired connection with the Internet. The wired connection can be further specialised, at conceptual level, into a general connection to the Internet, which allows the middleware server to fetch required Web pages, and a focused connection, which is the interface toward search engines.

The aim of this architecture is to make the features of a Web search engine usable for handheld devices, without redesigning the technology already in existence. This is done by introducing an intermediate layer, which takes care of the mediation between the mobile clients and the Internet, and by adjusting Web content to the particular features of a handheld device. Odysseus architecture fits into the roadmap suggested by the UMTS Forum (UMTS Forum, 2002) to reach the mobile portal.

Functionalities of Odysseus middleware server

The middleware server executes the following tasks: it waits for a user's query and forwards it to a search engine (selected from a list of available search engines).

Once the list of search results has been obtained, Odysseus estimates whether there are too many search results to be successfully displayed on the handheld device. If that is the case, Odysseus tries to suggest other words to the user and invites him to refine the query to make it more selective.

The suggested terms together with the selectivity assessments for each term are uploaded from Odysseus to the client, so that the whole process of query refinement is done on the client, minimizing the data exchange between the client and Odysseus to address high latency problems.

Furthermore, Odysseus suggests terms to the user in a focused way—choosing them according to the user's needs, as explained later, integrating a general word completion feature, which can already be offered by the device, and assisting the user in the difficult phase of text input.

When the number of search results is small enough, the middleware server extracts a concise and significant description for each result and indexes the pages referred by search results. In order to perform this task, the middleware server accesses the Web repeatedly.

There are two kinds of visualisation to satisfy the requirements of adjustment to a device's features: a textual visualisation and a graphical visualisation, which give the user an overview of obtained results. These visualisations will be explained in more detail later.

The functioning of the middleware server is illustrated in the flow diagram of Figure 3.

By observing the tasks performed by the middleware server and considering that it communicates directly with handheld devices through the wireless link, you can notice that some features of Odysseus are common to a browser for handheld devices. Firstly, the middleware server has to manage the user interaction. Secondly, Odysseus has to show search results to the user. Then, when the user chooses a result, Odysseus has to present the desired page to the user, and this means rendering the page on the device's screen. Finally, Odysseus represents for the client the access point to the *World Wide Web (WWW),* and so it should also offer browsing functionalities. Thus, it could be a good choice to merge the functionalities of Odysseus with those of a browser for handheld devices.

We will now explain in more detail the techniques for managing the interaction between a search engine and a handheld device, considering that suggestions about content adjustment for Web browsing have been already given in the section on design alternatives for overcoming the constraints on handheld devices.

Odysseus middleware server architecture

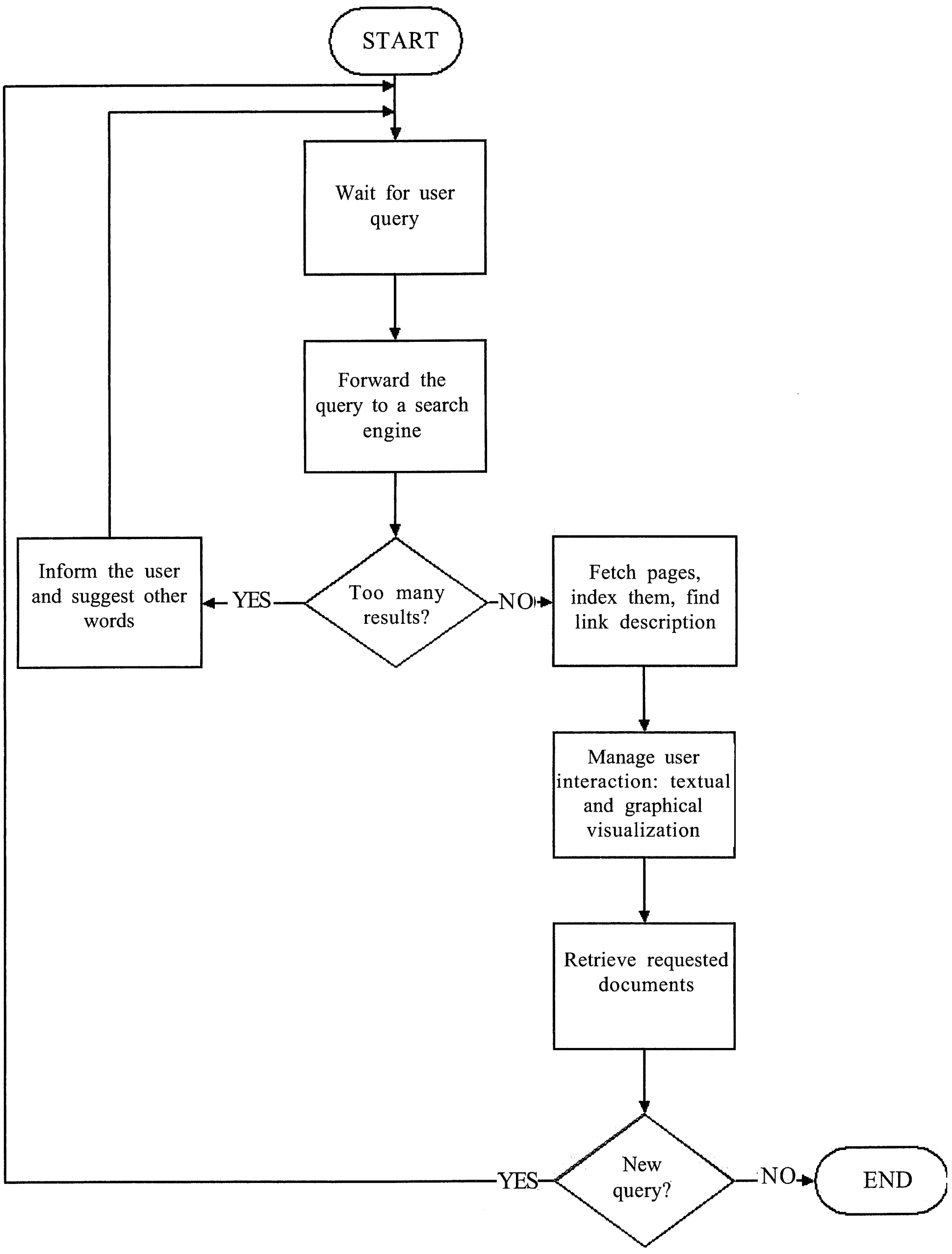
Within the middleware server the modules (needed to take into account the three main entities described at the beginning of the chapter) have to be planned. Indeed users, devices, applications and their relationships should be borne in mind during the design and development of an application accessible through handheld devices.

The architecture of the middleware server is illustrated in Figure 4.

Four main functional modules are illustrated in Figure 4:

- *User Modelling Engine*: this module models the various kinds of users, taking into consideration the users' preferences, their topics of interest, and the way in which they prefer to receive answers from the system, e.g., a summarisation of information or a list of keywords;
- *Device Modelling Engine*: this module models the various kinds of handheld device the system can interact with, bearing in mind basic functionalities offered by the device and special features that can be exploited;

Figure 3: Tasks executed by Odysseus middleware server


START
Wait for user query
Forward the query to a search engine
Too many results?
YES
Inform the user and suggest other words
NO
Fetch pages, index them, find link description
Manage user interaction: textual and graphical visualization
Retrieve requested documents
New query?
YES
NO
END

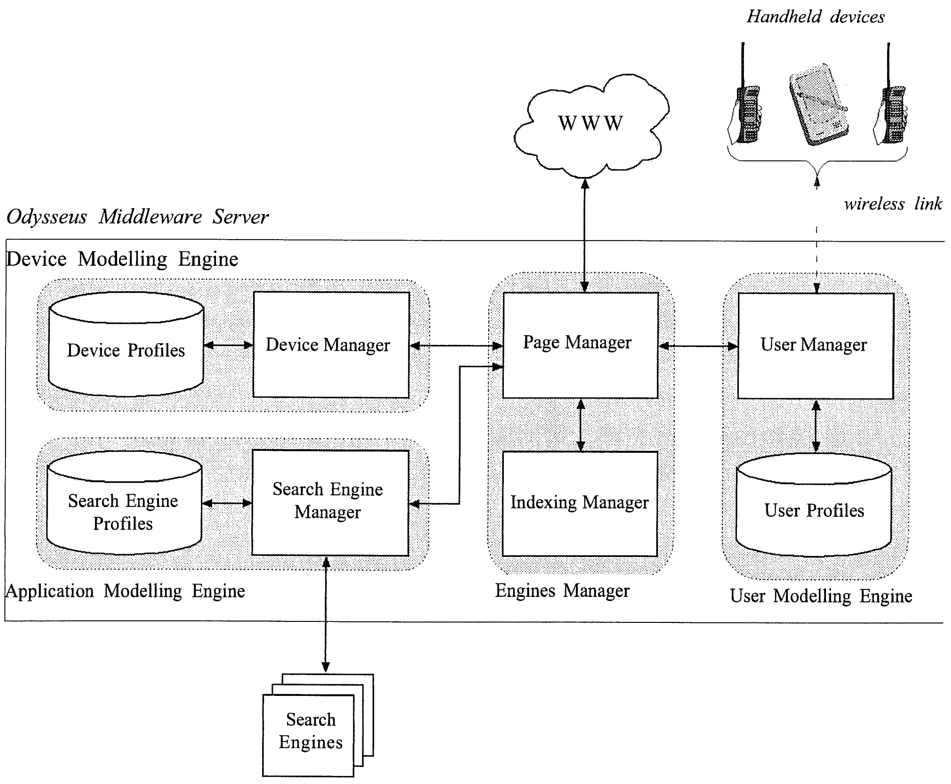
- *Application Modelling Engine*: this module models some aspects regarding the kind of application and special application needs (for example, an SE is characterized by a user's interaction greater than by browsing—in an m-commerce application filling in forms is a more substantial problem than browsing);
- *Engines Manager*: this module is a supervisor and coordinates the interaction of the other modules.

With this architecture, only one user model is maintained for each user, and so the user can access the application through various devices, keeping his work environment unchanged when he passes from one device to another, such as in Virtual Home Environment (VHE) of UMTS Forum (UMTS Forum, 2002).

On the other hand, only one device model is maintained for each device, and so different users, using the same device, have the same device model. This approach conforms with the *Composite Capability/Preference Profiles* (CC/PP) initiative (W3C, 2002).

We now see in more detail the function of each component within the middleware server.

Figure 4: Architecture of Odysseus and its modules



User Manager controls all the communication to and from the wireless client. It represents for the client the access point to the Internet. Furthermore, by using the *User Profiles* database, it recognises each user and his preferences, and stores personalised information for each user; this could be topics of interest, previous searches along with query terms, links judged to be relevant and visited pages, keywords within these pages, the preferred search engine—thus allowing the customisation of the service.

User knowledge is a progressive and incremental process, because information about a user's behaviour is gathered as he utilises the system, enters queries and browses the Web.

Device Manager keeps information about various handheld devices in the *Device Profiles* database, information such as screen resolution, available characters dimension, input characteristics and data protocol used by the handheld device, e.g., WAP, HDML and so forth.

The information on a handheld device's characteristics is used in the phase of content adjustment to tailor the behaviour of Odysseus to the specific device utilised by the user.

Search Engine Manager co-ordinates interactions with the various search engines supported by Odysseus: it forwards the user's query to the desired search engine, it retrieves the results proposed by the search engine, parses them, and translates them in an internal format (based on XML), so that the other components of the middleware server can perform their computation without considering the particular search engine used.

Furthermore, the *Search Engine Manager* can communicate with search engines not only through *HyperText Transfer Protocol* (HTTP), but also through CGI script, if the search engine provides advanced functionalities.

The *Search Engine Manager* keeps information about a search engine in the *Search Engine Profiles* database. It stores the name and the URL of the search engines, the parameters regarding the interface of the search engine, such as the relative position of objects, an object's type and function, information about special features implemented by the search engine, for example finding all the pages pointing to a desired URL, the search engine's available operators, modality of communication, e.g., HTTP, CGI, and a brief guide to the search engine's functionalities for helping the user.

Page Manager performs complex tasks and coordinates the components of the middleware server. It performs the following tasks:

- *management of number of results*: using a device dependent threshold, the *Page Manager* decides if there are too many search results and, in which case, it warns the user and suggests more terms.

- *query expansion*: the query expansion technique can be used when a user has entered his query, but that query would produce too many results; in an automatic way, more terms can be added to the original query to reduce the result set to be presented to the user; to reach this objective, the *Page Manager* asks the *User Manager* for terms related to the query by using the user profile and it extracts the more significant terms from the set of retrieved documents;
- *results description*: the result page itself is used for extracting a description sentence. Eventually, even the pages pointing to the desired page can be utilised: within these pages, the paragraph containing the link to the desired page is analysed and used to extract the description. An approach of this kind is adopted in InCommonSense (Amitay, 2000). To extract the summary sentence, the *Page Manager* uses the *Indexing Manager*;
- *page rendering and user interaction*: to render the page on the client, the *Page Manager* asks the *Device Manager* for information about the device's physical features and the language to use for describing the page, e.g., WML, HDML and so on.

Indexing Manager receives as input an HTML page and splits it into sentences of simple text, using structural elements such as paragraph breaks, lists, tables, titles and sections, punctuation.

Then each sentence is indexed, extracting tokens, filtering the stop words and performing the stemming, and is weighted according to the tf × idf (*term frequency, inverse document frequency*) weight:

$$w_{ij} = \mathrm{tf}_{ij} \cdot \log_2\left(\frac{N}{n}\right)$$

where:

- w_{ij} is the weight of the word T_j in the sentence S_i;
- tf_{ij} if the frequency of the word T_j in the sentence S_i;
- N is the number of sentences in the collection; and
- n is the number of sentences where the word T_j appears at least once.

Then a similarity score is computed between each sentence and the query. The sentence with the highest score is chosen as a description for a result.

Results visualisation

Odysseus offers two kinds of visualisation: a textual visualisation, similar to the ranked list, and a graphical visualisation, which lets the user choose from among three different types of representation of the results set.

Textual visualisation displays the description of a link and, if there is enough space on the screen, the URL as well. The URL is an important element, because from it the user can ascertain which site it is, if he already knows the site and if the site is interesting; thus, the URL should be displayed. If screen size allows it, a link description and a URL can be displayed at the same time while, if there is no space, the link description is displayed first and, upon user demand, the URL too.

Depending on the device used, one or two links can be displayed at the same time, and to see those that follow, the user can utilise a scroll key. Once the user reaches the desired link, he presses a confirmation key and the corresponding page is displayed.

The link order in textual visualisation is that returned by the search engine.

Graphical visualisation operates by offering the user the choice between various methods of visualisation and allowing him to move from one to another; once an object of interest is selected, Odysseus shows the link description, as used in textual visualisation, and it is possible to choose to see the Web page or to restart the exploration process.

In graphical visualisation, search results are represented as points in a bidimensional space. The aim is to give the user an overview of the search results, so that he can understand the relationships between the query and the results, and the relationships between the results themselves, without requiring the user to read each result to assess its relevance. In this way the information retrieval process is simplified and sped up.

A problem with graphical visualisation is input: how can the user move among various objects displayed? This is a problem especially for a mobile phone, because on a PDA, any object can be selected by the pen. For a mobile phone, a movement can be used which allows the probably more significant document to be visited first. For example, a clockwise or anticlockwise spiral movement, according to whether the user presses the up or down key, begins from the result most similar to the query and then moves on to those less similar.

When the desired object is reached, pressing a confirmation key permits the display of pertinent information.

Assuming a vector space model, every graphical visualisation tries to move from the *n*-dimensional space of terms of the documents to a two-dimensional space; what is changed from one graphical visualisation to another is the way in which point position is calculated within two-dimensional space.

Odysseus offers three graphical visualisations:

- The first visualisation tries to discover the links between the results, showing the inter-document similarities and which documents regard the same topic, through clustering techniques. If two results are similar, the corresponding points will be close while, if the points are far apart, it means that page content is very different. This approach is taken in Lighthouse (Leuski & Allan, 2000b; Leuski & Allan, 2002; Leuski & Allan, 2000a; Leuski & Allan, 1998).

A circle can be used to represent a judged or estimated relevant document, while a square can represent a judged or estimated nonrelevant document.
A simple corollary of Cluster Hypothesis is that if a relevant document is found, other relevant documents should be in the neighbourhood of this relevant document. As a consequence, with this visualisation, circles of relevant documents move toward circles of other relevant documents. Therefore, finding interesting information should be as easy as inspecting a circle near the circle of a document of known relevance. Similarly, squares of non-relevant documents tend to be brought together.
An example of this visualisation is shown in Figure 5.

- The second visualisation tries to highlight the relationships between the query and the results, in terms of relevance and similarity to the query and, in the case of multiple queries, it shows results common to the queries and the relationship of these results with different queries. This approach is taken in Hyperspace (Beale, McNab, & Witten, 1997).
A circle can be used to represent a document, while a rhombus can represent a query.
Independent and completely separate queries make a series of "dandelion heads," i.e., sets of unconnected circles, each one centred on the query that generated it. More interesting modalities are given by correlated queries, because if the same document belongs to a set of search results of two or more queries, it is connected to various rhombi.
An example of this visualisation is shown in Figure 6.
- The third visualisation lets the user analyse the relationships both between the query and the result and between the results and the terms of the query. Thus, it is possible to know that a document was judged relevant rather by the presence of some terms than the one of others.

Figure 5: Visualization of interdocument similarities and document clustering

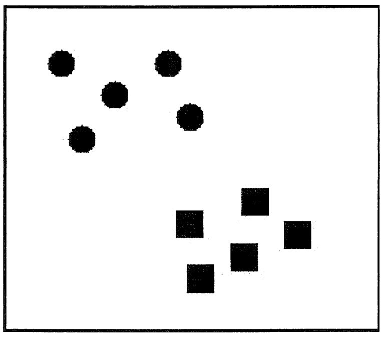
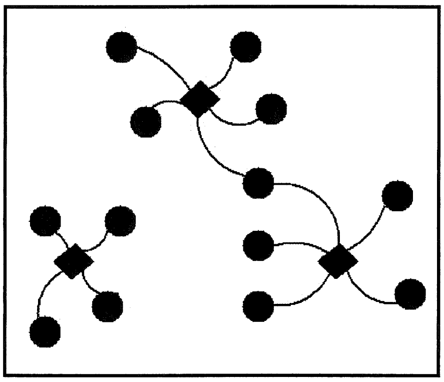


Figure 6: Visualization of relationships between results and query

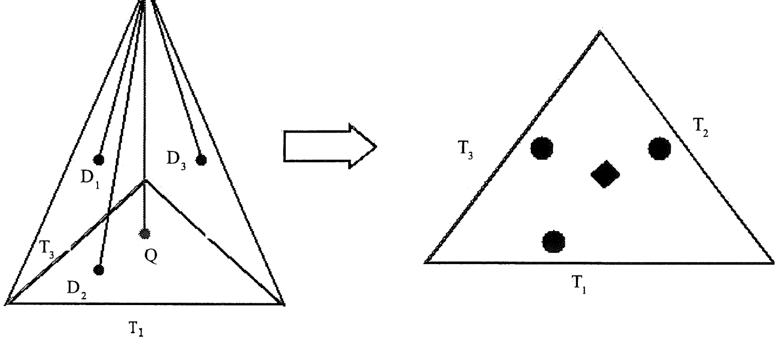


A three-terms query could be represented as a pyramid with a triangular basis, which is reminiscent of a Native American Indian tepee. Each face of the tepee is utilised to represent one of the three terms of the query. A pendulum (representing found documents) is used to show which terms are more relevant. The pendulum is attracted by the tepee's face, which represents the most relevant term, and its length is a measure of the overall relevance of the document. This approach is taken in Tepee (Grewal, Jackson, Burden, & Wallis, 1999; Grewal, Jackson, Wallis, & Burden, 2002).

The intersection between the pendulum and pyramid's basis can be determined, so that a bidimensional graph can be obtained, as in Figure 7. A circle can be used to represent a document, while a rhombus can represent the query.

The extension to queries with more terms is done simply by using a pyramid with polygonal basis, with as many faces as there are query terms.

Figure 7: Three-terms query and three results



CONCLUSIONS

Generally, users perceive small screen size and input difficulties make Web surfing on a handheld device a poorer experience than that on a desktop computer. To facilitate better access to the mobile applications, the latter should know if a user request originates from a desktop computer or a handheld device; in this latter case, the content should be adjusted so that it is easy to use, making user interaction more effortless.

On the other hand, content adjustment should also preserve the functional features of the mobile applications as much as possible; the user should be just as at ease using an application on a handheld device as on a desktop computer.

The responsiveness of mobile application influences user satisfaction. The interactions between mobile applications and users should be designed to reduce the processing and data transfer time. At present, a problem experienced by WAP is the long time required for the initial connection set-up with GSM (in several seconds). To reduce processing time, the mobile applications should not overload the handheld devices with computationally intensive operations.

The message emerging from these considerations is clear: the content needs to be adjusted to the handheld device, but the required processing should neither force the user to wait too long nor should it consume too much energy, compromising the user's mobility.

The aim is to encourage users to leave fixed Internet connections for wireless ones, without distressing them with different features of devices and discouraging them with great difficulties in Web fruition.

We have described the problems of managing the interaction between handheld devices, mobile applications and users, particularly when designing search engines for handheld devices. The project of extending the use of pervasive devices leads to the concept of content adjustment, which is needed for various reasons:

- there are different types of handheld devices, and the "anytime, anywhere, any device" paradigm of 3G systems underlines that the same content needs to made available and usable for different devices;
- handheld devices are very different from desktop or portable computers, and the content developed for these computers cannot be moved to handheld devices as it is;
- legacy applications should be used on handheld devices to extend an enterprise's business opportunities.

ACKNOWLEDGEMENTS

We are especially grateful to the editors of the book and to the anonymous referees for their most valuable comments on the earlier version of this chapter.

During the initial development of this work, Nicola Ferro was attending a training course in the stimulating environment of IBM Technology Deployment of Segrate, Milan, Italy. He wishes to thank his manager, Antonio Caizzi, and his colleagues for the time spent with him and the stimulating discussions.

REFERENCES[2]

Amitay, E. (2000). InCommonSense—Rethinking Web search results. In *Proceedings of the IEEE International Conference on Multimedia and Expo* (ICME 2000) (pp. 1705-1708).

Aridor, Y., Carmel, D., Maarek, Y. S., Soffer, A., & Lempel, R. (2001). Knowledge encapsulation for focused search from pervasive devices. In *Proceedings of the Tenth International World Wide Web Conference* (pp. 754-764).

Aridor, Y., Carmel, D., Maarek, Y. S., Soffer, A., & Lempel, R. (2002). Knowledge encapsulation for focused search from pervasive devices. *ACM Transactions on Information Systems (TOIS), 20* (1), 25-46.

Beale, R., McNab, R. J., & Witten, I. H. (1997). Visualising sequences of queries: A new tool for information retrieval. In *Proceedings of the IEEE Conference on Information Visualization* (pp. 57-62).

Bederson, B. B., & Hollan, J. D. (1994). Pad++: A zooming graphical interface for exploring alternate interface physics. In *Proceedings of the ACM symposium on User Interface Software and Technology* (pp. 17-26).

Belkin, N., Oddy, R., & Brooks, H. (1982). ASK for information retrieval. *Journal of Documentation, 38*, 61-71 (part 1); 145-164 (part 2).

Bickmore, T. W., & Schilit, B. N. (2002). Digestor: Device-independent access to the World Wide Web. *Proceedings of the Sixth International World Wide Web Conference* [online]. Available at: http://www.scope.gmd.de/info/www6/technical/paper177/paper177.html.

Bluetooth (2002). The official Bluetooth website [online]. Available at http://www.bluetooth.com/.

Britton, K. H., Case, R., Citron, A., Floyd, R., Li, Y., Seekamp, C., Topol, B., & Tracey, K. (2001). Transcoding: Extending e-business to new environments. *IBM Systems Journal, 40* (1), 153-178.

Buchanan, G., & Jones, M. (2000). Search interfaces for handheld Web browser. In *Poster Proceedings of the Ninth International Word Wide Web Conference* (pp. 86-87).

Buyukkokten, O., Garcia-Molina, H., & Paepcke, A. (2000). Focused Web searching with PDAs. In *Proceedings of the Ninth International World Wide Web Conference* (pp. 213-230).

Buyukkokten, O., Garcia-Molina, H., & Paepcke, A. (2001a). Accordion summarization for end-game browsing on PDAs and cellular phones. In *Proceedings* *of the Conference on Human Factors and Computing Systems* (SIGCHI 2001) (pp. 213-220).

Buyukkokten, O., Garcia-Molina, H., & Paepcke, A. (2001b). Seeing the whole in parts: Text summarization for Web browsing on handheld devices. In *Proceedings of the Tenth International World Wide Web Conference* (pp. 652-662).

Buyukkokten, O., Garcia-Molina, H., Paepcke, A., & Winograd, T. (2000). Power browser: Efficient Web browsing for PDAs. In *Proceedings of the Conference on Human Factors and Computing Systems* (CHI 2000) (pp. 430-437).

Buyukkokten, O., Kalijuvee, O., Garcia-Molina, H., Paepcke, A., & Winograd, T. (2002). Efficient Web browsing on handheld devices using page and form summarization. *ACM Transactions on Information Systems (TOIS), 20* (1), 82-115.

Cooper, I., & Shufflebotham, R. (2002). Pda Web browsers: Implementation issues [online]. Available at: http://www.cs.ukc.ac.uk/pubs/1995/57/index.html.

Ferro, N. (2001). Online information access through handheld devices. Laurea's thesis in telecommunications engineering. Department of Electronics and Computer Science. University of Padua, Italy (In Italian).

Freire, J., Kumar, B., & Lieuwen, D. (2001). WebViews: Accessing personalized Web content and services. In *Proceedings of the Tenth International World Wide Web Conference* (pp. 576-586).

Grewal, R. S., Jackson, M., Burden, P., & Wallis, J. (1999). A novel interface for representing search-engine results. In *Proceedings of the IEE Colloquium Lost in the Web—Navigation on the Internet* (ref. no. 1999/169) (pp. 7/1 - 7/10).

Grewal, R. S., Jackson, M., Wallis, J., & Burden, P. (2002). Using visualisation to interpret search engine results [online]. Available at: http://seed.scit.wlv.ac.uk/papers/activeweb99.html.

GSMBox. (2002). Studies confirm T9 text input is fastest method for word entry on mobile phones [online]. Available at: http://uk.gsmbox.com/news/mobile_news/all/2091.gsmbox.

Hsu, J., Johnston, W., & McCarthy, J. (2002). Active outlining for HTML documents: An X-Mosaic implementation. Proceedings of the Second International World Wide Web Conference [online]. Available at: http://archive.ncsa.uiuc.edu/SDG/IT94/Proceedings/HCI/hsu/hsu.html.

IBM. (2002). Websphere transcoding publisher [online]. Available at http://www.ibm.com/software/webservers/transcoding/.

Jones, M., Marsden, G., Mohd-Nasir, N., Boone, K., & Buchanan, G. (1999). Improving Web interaction in small screen displays. In *Proceedings of the Eighth International World Wide Web Conference* (pp. 51-59).

Jones, M., Marsden, G., Mohd-Nasir, N., & Buchanan, G. (1999). A site-based outliner for small-screen Web access. In *Poster Proceedings of the Eighth International World Wide Web Conference* (pp. 156-157).

Jones, M., Mohd-Nasir, N., & Buchanan, G. (1999). An evaluation of WebTwig—a site outliner for handheld Web access. In H. W. Gellerson (ed.), *Lecture Notes in Computer Science,* LNCS 1707 (pp. 343-345). Springer-Verlag.

Kaljuvee, O., Buyukkokten, O., Garcia-Molina, H., & Paepcke, A. (2001). Efficient Web form entry on PDAs. In *Proceedings of the Tenth International World Wide Web Conference* (pp. 663-672).

Leavitt, N. (2000). Will WAP deliver the wireless Internet? *Computer, 33* (5), 16-20.

Leuski, A. (2002). Evaluating a visual presentation of retrieved documents [online]. Available at http://cobar.cs.umass.edu/pubfiles/ir-159.ps.

Leuski, A., & Allan, J. (1998). Evaluating a visual navigation system for a digital library. In *Proceedings of the Second European Conference on Research and Advanced Technology for Digital Libraries* (ECDL '98) (pp. 535-554).

Leuski, A., & Allan, J. (2000a). Improving interactive retrieval by combining ranked list and clustering. In *Proceedings of the RIAO 2000 Conference* (pp. 665-681).

Leuski, A., & Allan, J. (2000b). Lighthouse: Showing the way to relevant information. In *Proceedings of the IEEE Symposium on Information Visualization 2000* (InfoVis 2000) (pp. 125-129).

Leuski, A., & Allan, J. (2002). Details of Lighthouse [online]. Available at http://cobar.cs.umass.edu/pubfiles/ir-212.ps.

Lie, H. W., & Bos, B. (1999). *Cascading Style Sheets: Designing for the Web* (2nd ed.). Addison-Wesley.

Marshall, C. C., Golovchinsky, G., & Price, M. N. (2001). Digital libraries and mobility. *Communications of the ACM, 44,* 55-56.

Masui, T. (1999). POBox: An efficient text input method for handheld and ubiquitous computers. In *Proceedings of the International Symposium on Handheld and Ubiquitous Computing* (HUC 99) (pp. 288-300).

Metter, M., & Colomb, R. (2000). WAP enabling existing HTML applications. In *Proceedings of the First Australasian User Interface Conference* (AUIC 2000) (pp. 49-57).

Moran, D. B., Cheyer, A. J., Julia, L. E., Martin, D. L., & Park, S. (1997). Multimodal user interfaces in the open agent architecture. In *Proceedings of the International Conference on Intelligent User Interfaces* (pp. 61-68).

Ojanen, E., & Veijalainen, J. (2000). Compressibility of WML and WML script byte code: Initial results [wireless mark-up language]. In *Proceedings of the Tenth International Workshop on Research Issues in Data Engineering* (RIDE 2000) (pp. 55-62).

Tegic. (2002). T9 [online]. Available at http://www.t9.com, http://www.tegic.com.

UMTS Forum (2002). Enabling UMTS/third generation services and applications [online]. Available at http://www.umts-forum.org/reports/report11.pdf.

UMTS Forum (2002). Shaping the mobile multimedia future [online]. Available at: http://www.umts-forum.org/reports/report10.pdf.

UMTS Forum (2002). The UMTS third generation market—structuring the service revenues opportunities [online]. Available at http://www.umts-forum.org/reports/report9.pdf.

W3C (2002). CSS Mobile Profile 1.0—W3C Candidate Recommendation, 24 October 2001 [online]. Available at http://www.w3.org/TR/css-mobile.

W3C (2002). Cascading style sheets, level 2—CSS2 specification [online]. Available at http://www.w3.org/TR/REC-CSS2.

W3C (2002). Composite Capability/Preference Profiles (CC/PP): A user-side framework for content negotiation [online]. Available at: http://www.w3.org/TR/NOTE-CCPP.

ENDNOTES

1 A handheld device is also known as a mobile device. In this chapter, we use both terms interchangeably.

2 The date on the availability of documents on the Web is 2002 for all documents, because the presence of all those digital documents has been checked at the given URL during the late months of the year 2002.