A PORTABLE DOCUMENT SEARCH ENGINE TO SUPPORT OFF-LINE MOBILE LEARNING

John P. T. Moore, Samia Oussena
Thames Valley University
St. Mary's Road, London W5 5RF, United Kingdom

Pei Zhang
University of Southampton
School of Electronics and Computer Science, SO17 1BJ, United Kingdom

ABSTRACT

Some modern mobile devices have the capability to store thousands of documents and therefore have the potential to be used as powerful off-line learning tools. The traditional graphical file browser does not scale well on physically constrained devices. In addition, documents obtained from random sources on the Internet may contain ambiguous names and therefore may be difficult to relocate on the device. This paper describes a solution to these problems using a traditional search engine approach. A powerful open source document search engine was ported to mobile architectures. Support for adding documents to the mobile device was developed using an Internet search enhanced to highlight domain specific results. The software was developed as part of the Remora project to provide mobile learning support for trainee social workers in the UK.

KEYWORDS

Off-line mobile learning, server-side mobile development.

1. INTRODUCTION

In common with many other professions, the training of social workers requires students to be placed in social work settings and to undergo assessment in the workplace. Trainee social workers in England must successfully complete 200 days in a practice setting. Currently the social work professional bodies indicate there is a lack of e-learning support for all stakeholders involved in the placement process. The Remora project aims to provide mobile software to support work-based learning and assessment for trainee social workers. A user-centered design (co-design) approach was used to work closely with stakeholders to ensure that the applications developed were based on actual needs.

As part of the learning process the trainee social worker (student) is expected to refer to particular Internet sites such as those relevant to mental health issues in the UK. A requirement that this practice should be able to take place in the work setting was established. However, it was not always possible for the student to access the Internet during a placement. A simple solution would have been to provide the students with mobile devices capable of supporting generic web based Internet searching. However, a number of key requirements specific to mobile searching were identified. These included:

- Domain specific Internet searching
- Support for off-line searching and retrieval of documents

The problem of providing quality Internet search results, as perceived by the user, is amplified on a mobile device with limited viewable screen size. Building a context-aware search engine, one that has domain

knowledge relevant to trainee social workers, was beyond the scope of the project (Finkelstein, 2002). However, a more simplistic approach of providing domain specific highlighting of general search results was considered a possible compromise.

Having identified quality documents the ability to cache or horde these documents on the mobile device was seen as an important design feature (Trifonova and Ronchetti, 2006). For example, a mobile device may only have Wi-Fi and therefore only have Internet connectivity when in range of a wireless hot-spot. In this scenario the user may search and retrieve documents which could later be easily located and read on the device when there was no network connection. This ability to access off-line documents also suits mobile environments that have expensive and/or slow data networks. The user-centered design process revealed confusion over the cost of using data services on mobile devices and the true data speeds available from mobile networks.

Another requirement born from co-design was the importance of usability of the mobile device. Concern was expressed over the suitability of reading small screens particularly in relation to the amounts of mature students typically found in this domain. Although no decisions were made during co-design regarding potential hardware, a clear favourite emerged. The iPhone or iPod Touch was seen as a potential candidate with is simple gesture based interface capable of easily zooming in and out of text on the screen. In addition, technology such as the iPod has already provided a suitable candidate for other learning oriented projects (Combs et al, 2006).

Another concern that was expressed related to data entry on a touch screen mobile device. As a result a requirement to look at alternatives emerged. An Ultra-Mobile PC (UMPC) was seen as a possible alternative. To support different hardware and systems a portable development solution was sought. The two categories of devices identified have very different environments when it comes to developing graphical applications. However, there is a similarity at the operating system level and the core libraries and programs. By exploiting this similarity and adopting a web based approach it was possible to produce a portable solution. Another benefit of this approach is the large amount of traditional server side software that could be ported to the mobile device. When the graphical user interface is eliminated from the software the ability to easily port that software increases. The end result is a design that might work equally well on a mobile device as a personal computer (PC).

2. DESIGN

The idea of running web servers or server-side processes on a mobile device is not new (Wikman, 2006). This technique has become more viable as web browsers on mobile devices have improved through open source projects such as WebKit (WebKit, 2008). The hardware required to support a full web experience has been made available in pocket size. Various companies have produced such hardware. These include Apple with its iPhone and iPod Touch, Nokia with its Series 60 devices and more recently HTC using Google's Android based operating system. Subsequently, designing applications for a mobile device using a web interface has become a viable alternative to using native application development. The web approach offers the following advantages:

- Familiar development paradigm
- Large pool of potential developers
- Portability across different mobile platforms
- Large repository of existing software ready to port

Figure 1 summarises the architecture of the Remora search tool. The hexagon represents the server-side components that reside on the mobile device. The main components include:

- a document repository
- remote search
- local search

- bookmarks

The user runs a web browser to access the local web server running at the IP address 127.0.0.1 or localhost. They are then able to run a number of Common Gateway Interface (CGI) programs. The remote search CGI provides Yahoo! web search capability. Figure 2 provides a screen shot from an iPod Touch after carrying out a remote search. The results of the Yahoo! search are processed to check for domain specific sites.

For example, in figure 2 we can see how the results for "Mental Health Foundation" have been highlighted. This indicates that this site is part of list of sites we have bookmarked as being relevant to our domain. In the Remora project, domain specific sites included those concerned with mental health issues in the UK. By providing domain specific highlighting the user can easily be alerted to important sites that might arise from doing a general web search. The list of domain specific sites is maintained through an XML file containing bookmarks from the social bookmarking service Delicious. The user can easily update these bookmarks and refresh the XML file on the device. Figure 2 also shows what happens if the remote search locates a PDF file. In this example the first result is a link to a PDF file. The user is able to click on the PDF label to save the file to the document repository. Any PDFs saved can later be located using the local search CGI. Figure 3 shows how we located the file that was downloaded.

The local search capability is provided by a powerful open source search engine called Hyper Estraier (Hirabayashi, 2007). Traditionally this search engine is used to add search facilities to websites and wiki's. For example, the Free Software Foundation uses Hyper Estraier to provide the search capabilities for the GNU website. Figure 3 also highlights a common problem when retrieving documents from the Internet, the file names may be ambiguous. In this example the file name is uk.pdf. This gives little indication to the user as to the contents of this file. Therefore, even if it was practical to employ a traditional file browser on the mobile device it would be of limited use when dealing with random file names. Hyper Estraier works by looking at the contents of the document to identify those documents that meet the users' search criteria. In figure 3, all that was required was that the user enter a search term 'mental' to locate the document. This keyword based search will also scale well as the number of documents in the repository increases. It is also possible to use the browser to bookmark popular searches.

To test how well the search engine scaled on a mobile device a proof of concept solution was built for the iPhone. More than 2000 PDFs from MIT's OpenCourseWare were added to the search engine (OpenCourseWare, 2008). The result was a very usable search tool capable of supporting mobile off-line learning. A YouTube video exists to demonstrate this functionality (Moore, 2008a).

3. EXPERIENCE

Developing software for new technologies comes with challenges. At the time the project began Apple had recently started releasing beta versions of their SDK. The development community was also in its infancy compared to more well established mobile products. Thus, normal developer practices such as posting to technical forums, using mailing lists and Internet Relay Chat (IRC) provided limited success. In general, even a simple web search resulted in a low signal to noise ratio.

Mobile products are traditionally "locked down" to satisfy the stringent requirements of telecommunication network providers. To lift some restrictions the community developed solutions which allowed the iPhone to be "unlocked" and also allow community driven software to be installed. Through these mechanisms it was then possible to port large C based software such as Hyper Estraier and install it on to the iPhone and iPod Touch. A cross compilation tool chain was setup on a PC running Ubuntu to build all the dependencies and Hyper Estraier itself. The process of setting up the tool chain and getting the build to succeed for the various dependencies proved challenging. Cross compilation can be difficult without established mechanisms in place to deal with shared library dependencies. Powerful solutions exist for embedded Linux such as OpenEmbedded (OpenEmbedded, 2008). Building the Remora search tool for platforms supported by OpenEmbedded would be more straight forward than doing everything by hand.

The remote search CGI needed to be developed from scratch. This meant the were options in terms of development language. High-level languages like Ruby or Python provide a flexible way to carry out Rapid

Application Development (RAD) on a mobile device. They also remove the need for cross compilation. A disadvantage of this approach is they require additional software to exist on the device to run even the smallest script. These dependencies can lead to maintenance issues when various software is updated.
An initial approach taken was to develop the remote search in Ruby on a PC and simply install Ruby on the mobile device and then use the same scripts. Although there was no problem getting Ruby running on a selection of Linux-based mobile devices, the iPhone proved to be different. It was possible to cross compile Ruby and install it to the iPhone, however, running anything other than a basic script caused programs to crash out with bus errors. Searching the community for a solution proved fruitless. In the end, this approach was dropped in favour of a hybrid solution involving a mixture of interpreted script code and compiled C code (Moore, 2008b). This approach utilised existing libraries and programs on the device. All these components could easily be found on any Linux distribution and therefore provide a portable way of developing for the iPhone plus other devices.

4. DISCUSSION

The software developed has yet to be evaluated during or after deployment. An interesting observation to make is the reaction of end users to server-side solutions on mobile devices. For example, users are very accustomed to web searching where the functionality is provided by the remote server. In the Remora search tool the end user does not need to understand that this functionality is now being delivered locally, however, will they appreciate the development effort required to provide such a solution? Do users feel they get value for money from server-side mobile solutions when the interface is a web browser?

A simple way to describe one potential use of the Remora search tool is based on the concept of a USB flash drive in your pocket. A USB flash drive is a small low-cost personal storage device. Could a mobile device with search engine capability extend this basic storage device and provide a tool which not only allows storage of thousands of documents but provides a simple way to access those documents and read them on a small screen? Studies from deployment of the Remora technology could identify usage patterns and try and identify such potential off-line mobile learning opportunities.

5. CONCLUSION

This paper presents software designed to support off-line mobile learning. The requirements of the software were born from a co-design process that took place as part of a JISC funded project called Remora. The software was developed in a portable way using a web based approach to allow deployment on a range of mobile hardware and devices. The goal of the software was to allow documents to be located during an Internet search and then cached on the device to be read off-line. To assist this process the search results were customised to support domain specific highlighting. Any PDF files identified from an Internet search could easily be added to the device's document repository. To facilitate off-line access to these documents a powerful open source search engine was ported to mobile architectures. Key benefits of this search engine included scalability and the ability to locate documents through their contents rather than their file name. Potentially thousands of documents could be stored on the device and efficiently located using familiar keyword searches. The idea of carrying a pocket sized device loaded with many files exists with USB flash drives. The Remora search tool builds on this concept and provides the user with the ability to locate and view those documents in a way that supports off-line mobile learning.

ACKNOWLEDGEMENT

The Remora project is funded by JISC.

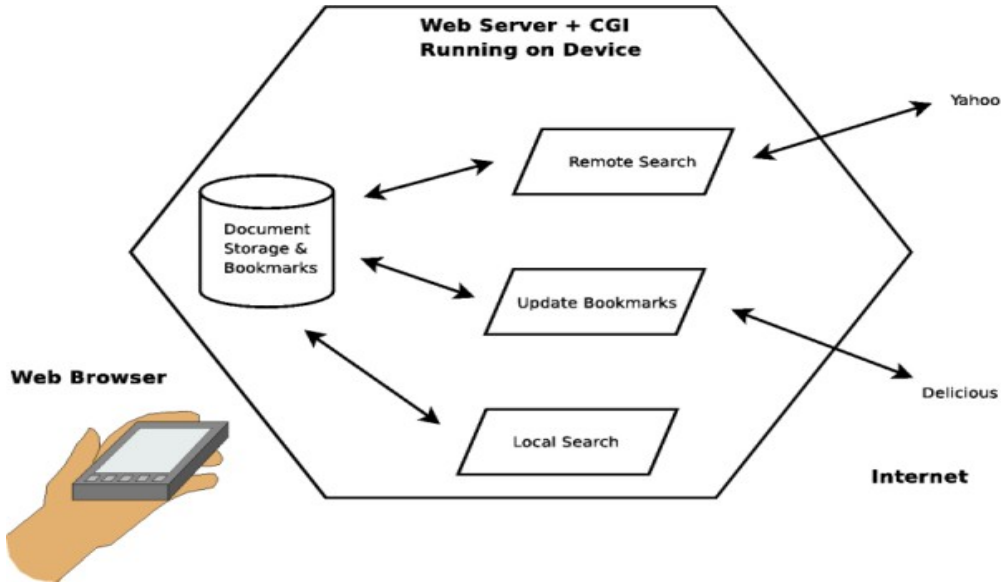

Figure 1: Remora search tool architecture

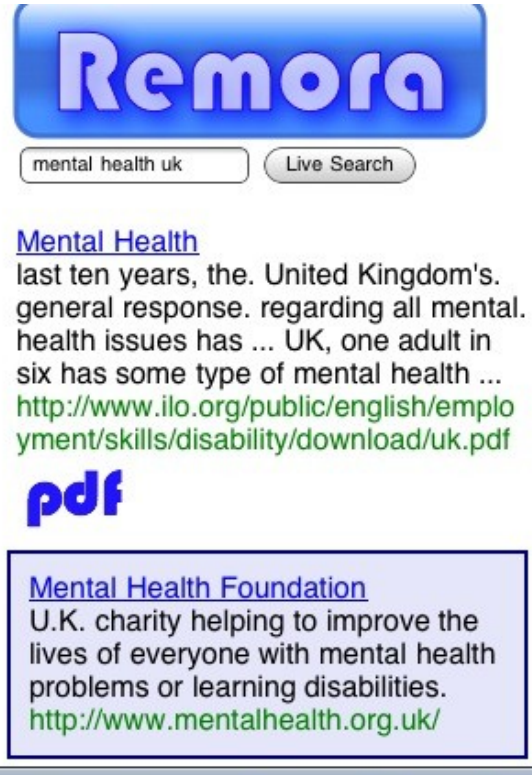

Figure 2: Remote search

Figure 3: Local search

REFERENCES

Combs, S. et al., 2006. Turning iPod into an Effective Portable Learning Tool. *Society for Information Technology and Teacher Education International Conference.*

Finkelstein, L. 2001. Placing search in context: the concept revisited. ACM Transactions on Information Systems. Volume 20, Issue 1. pp. 116-131

Hirabayashi, M., 2007. Hyper Estraier: a full-text search system for communities. http://hyperestraier.sourceforge.net/ (last accessed Oct 2008)

Moore, J. 2008a. iPhone local search engine. http://www.youtube.com/watch?v=MKKlkcZ6vYo (last accessed Oct 2008)

Moore, J. 2008b. Remora search tool. http://remora.sourceforge.net/ (last accessed Oct 2008)

OpenCourseWare. 2008. MIT OpenCourseWare. http://ocw.mit.edu (last accessed Oct 2008)

OpenEmbedded. 2008. OpenEmbedded. http://openembedded.org (last accessed Oct 2008)

Trifonova, A. and Ronchetti M., 2006. Mobile Learning: Is Anytime + Anywhere = Always Online?. *Proceedings of the Sixth IEEE International Conference on Advanced Learning Technologies*. pp. 702-706.

WebKit. 2008. The WebKit Open Source Project. http://webkit.org/ (last accessed Oct 2008)

Wikman, J. et al, 2006. Personal Website on a Mobile Phone. Nokia Research Center Helsinki. http://research.nokia.com/files/NRC-TR-2006-004.pdf