Beyond the Client-Server Model: Self-contained Portable Digital Libraries

David Bainbridge[1], Steve Jones[1], Sam McIntosh[1], Ian H. Witten[1], and Matt Jones[2]

[1] Department of Computer Science
University of Waikato
Hamilton, New Zealand
{davidb,stevej,sjm64,ihw}@cs.waikato.ac.nz
[2] Department of Computer Science
University of Swansea
Swansea, U.K.
matt.jones@swansea.ac.uk

Abstract. We have created an experimental prototype that enhances an ordinary personal media player by adding digital library capabilities. It does not enable access to a remote digital library from a user's PDA; rather, it runs a complete, standard digital library server environment on the device. Being optimized for multimedia information, this platform has truly vast storage capacity. It raises the possibility of not just personal collections but entire institutional-scale digital libraries that are fully portable. Our implementation even allows the device to be configured as a web server to provide digital library content over a network, inverting the standard mobile client-server configuration and incidentally providing full-screen access. Anecdotal yet compelling evidence as to the usefulness of being able to access a self-sufficient digital library anytime, anywhere is given through an example built from the PDF files of the Joint ACM/IEEE digital library conference. Other examples include the Complete Works of Shakespeare and collections on Humanitarian Aid. This paper describes the facilities we built, focusing on interface issues that were encountered and solved.

Keywords: Portable Digital Libraries, Personal Media Player, Multimedia.

1 Introduction

Access to digital libraries is predominantly through the web. Indeed, the assumption that the user is working through a remote web server from a desktop machine or portable device is almost a given, and other modes of access are virtually excluded.

Some researchers have investigated access to digital libraries from portable devices, e.g. [1,3,4]. They envisage a scenario whereby a personal digital assistant (PDA) connects, using wireless networking, to a central digital library server.

G. Buchanan, M. Masoodian, S.J. Cunningham (Eds.): ICADL 2008, LNCS 5362, pp. 294–303, 2008.

(a) (b) (c) (d)

Fig. 1. Invoking Greenstone on the iPod: (a) & (b) a click-wheel iPod; (c) & (d) an iPod-touch

The interface is web based, and the research involves tailoring it for a small screen (augmented, in some cases, with suitable data caching policies). Such work extends the existing web-based client-server model by adapting it for new kinds of client, a decision that is forced by the fact that most general-purpose PDAs do not begin to approach the storage capacity of centralized servers. To contemplate implementing the entire digital library itself on a portable device is a radical alternative.

There is an obvious incentive for designers of portable devices that are optimized for music and video to include vast amounts of disk storage, and general-purpose PDAs will probably never approach the storage capacity of such devices. Consequently we have considered whether it is possible to subvert personal multimedia devices to store entire digital library collections.

Figure 1 illustrates the result of our work, based on the iPod, and shows the user invoking the top-tier of digital library services (the collections available) on both an click-wheel iPod and iPod-touch. Later examples go in to more detail. But we stress at the outset that this choice of device is just an example, as is the choice of Greenstone [5] as the digital library software used. Our work is exploratory; to stimulate thinking on alternative digital library models in general—specifically, what happens when portable devices are equipped with large-scale storage capabilities.

To take one anecdotal example, one of the authors built a full-text searchable Greenstone collection of the 2008 proceedings of the ACM/IEEE Joint Conference on Digital Libraries, and as an experiment, ran it on an iPod-touch using the prototype presented in this paper. The source documents were provided to delegates on a USB pen-drive as PDF files and in a matter of 15 minutes this had been converted into a Greenstone collection and synchronized with the iPod. During the course of the conference the author—perhaps predictably—found himself looking details up on the iPod during sessions. Natural and convenient to use, a keyword or two was usually all that was necessary to quickly pinpoint the relevant part of an article. Potentially more surprisingly, the author found himself

still referring back to the iPod-based collection after the conference,accessing further information. This urge to turn to the iPod as the information source was instinctive, even though exactly the same information was simultaneously available on the author's laptop.

In this paper we discuss several developments aimed at tailoring the digital library software to a portable, self-contained, environment. For the generation 3–5 iPods, digital library browsing facilities were developed to utilize the unique iPod click-wheel for hierarchical browsing, and the traditional-style web-form based full-text search was altered to ameliorate the problems caused by the lack of keyboard. Also, launcher applications were created to display digital library documents, whether they be text, image, or audio media types. For the iPod-touch (and iPhone) with its touch-screen interface, usability issues were less severe. In both cases we also inverted the traditional PDA client-server model for digital libraries, and installed a web-server on the device that allowed it to serve content to others. Preliminary work on this was presented in [2], which is only about the click-wheel iPod and pin-points the technical issues we faced concerning implementation. The current paper takes a broader view of the topic, with interaction issues and usage scenarios the main focus, along with expanded details including its operation on the iPod-touch.

2 Motivation

Imagine being able to carry a library around in your pocket. Full fingertip access through searching and browsing to millions of items: text, image, audio, video, wherever you are. Really, *wherever* you are: no need to be within a wireless hotspot, or mess around with ISP registration (and worry about how much it is costing). No waiting for rich-media content to be transferred to your device, not to mention the accompanying rapid depletion of your battery power that goes with all that communication. Everything travels with you: no flight restrictions on planes, no worrying about connectivity in other countries. Your own personal copy of a large digital library, right there in your pocket.

A relatively low-cost Personal Multimedia Player (PMP) like the iPod can store an astonishing amount of information, and it is now possible to programme these devices and augment them with your own software—such as a digital library application, for instance. In terms of text storage, consider a library with one million books (corresponding to a medium-sized university library). At 80,000 words per book and 6 letters per word (including the inter-word space), each book comes to half a million bytes, or 150,000 bytes when compressed. The whole library amounts to about 150 GB, less than the capacity of a high-end PMP today. This is text only: it does not allow for illustrations. Even so, with the remaining 10 GB of free space you could also comfortably slot in all the articles from Wikipedia.

What about illustrated books? It's hard to generalize about the size of illustrated books because so much depends on their nature, but our experience

with digital library collections of humanitarian information (like the Humanity Development Library[1]) indicates that about 2000 fully-illustrated (and fully-indexed) books fit comfortably on a 650 MB CD-ROM: a 160 GB PMP would store almost half a million of these. Alternatively, over a quarter of a million pages from the New Zealand National Library's Papers Past collection[2] could be comfortably accommodated on such a device. This value includes the necessary files to support full-text indexing of the OCRd images, word highlighting of search terms when the images are viewed, and metadata to allow switching between page and article views. More articles could be uploaded if, for instance, the word highlighting capability (which in disk capacity accounts for roughly half of the supplementary files) was omitted.

And multimedia? Over 2,000 hours of audio or 200 hours of video fits comfortably onto the device. This is particularly attractive if literacy rates of end-users is of concern, as is the case in many developing countries. One doesn't necessarily have to purchase a top of the range device, and there is a buoyant market in second-hand personal media players—even Apple sell reconditioned older models through their website for considerably less than the list price.

3 Developing Portable Digital Library Services

The foundation to our work is the combination of open-source operating system and cross-compiler. iPod-Linux[3] is a project with the goal of porting Linux to the various click-wheel generation of iPod. Using the cross-compiler provided through this project, developers can use a host machine to generate executables compatible with the iPod and consequently run their own choice of software on this device—a wide variety of applications have been ported: from a humble text-entry notepad application to first person "shoot-em up" game, Doom. The iPhone and iPod-touch (which already run Unix natively) have also been "jail broken" and a suitable open-source cross-compiler made available for people wanting to have a freer range of control over the software they run on their device.

With the programming environment established, to develop portable self-contained DL services, the Greenstone runtime system—a CGI program intended to be run by a web server in response to each user request/query and produce web pages in raw HTML that are rendered in a web-browser—some alterations were required. Primarily the runtime system needed to be re-cast as a program intended to be run continuously, rather than being re-invoked by the web server for each individual operation. This task was greatly simplified by the internal structure to Greenstone that (even in a standalone CGI program) is cleanly divided in two through a protocol that separates a front-end receptionist concerned with presentation issues, and a back-end collection server responsible for providing full-text indexing and metadata browsing functionality.

[1] Available, along with many other humanitarian collections, at *www.nzdl.org*
[2] *paperspast.natlib.govt.nz*
[3] *www.ipodlinux.org*

This was the lion's share of the work. Integrating the result into the PMP environment was straightforward. For the click-wheel iPod, the work was packaged as a dynamically loadable module that stipulates where in the main application's hierarchy it should go. For the iPod-touch the software was configured as a bundled application.

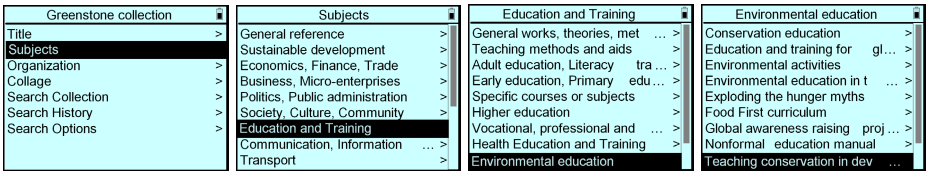
The DL reimplementation works directly from the files that are generated by a host machine running the standard Greenstone software. This setup means that any existing Greenstone collections can be transferred to the device, placed in the appropriate location, and displayed there without any intervention or conversion. Bulk file copy is a rather crude mechanism for synchronization, and this is an area that can be improved upon in a version that goes beyond our proof of concept phase; however, we were actually surprised at how well this worked in practice, given the file transfer utilities already available, in particular the Unix *rsync* command that can be used to transfer just the files that have changed between two file hierarchies.

3.1 Interactive Browsing

The next step was to map Greenstone's search and browsing functions into the iPod's interactive style, replacing the hyperlink navigation style implemented by the regular CGI version of Greenstone. In principle there is a clear and direct mapping of Greenstone's browsing capabilities (implemented by hyperlinks in a web page) into the iPod style of hierarchical menus (click-wheel or touch-screen). Greenstone collection designers can include a selection of browsing devices in their collection, like alphabetical lists and hierarchical browsers with arbitrary numbers of levels. All these browsing devices map naturally into the kind of hierarchy that users are accustomed to traversing with the iPod.

In order to achieve this technically, Greenstone's access mechanism had to be recast as an iPod interface. The iPod click-wheel software development suite incorporates a modest graphical user interface toolkit specially developed for iPod-Linux, called TTK. This provides a stacked-window system along with a high-level library including event handling and methods for input and graphical output, and forms an abstraction layer between application and lower-level graphics library. It is quite generic, but provides a few specific widgets to get people started: hierarchical menus, text viewer, image viewer, slider, popup windows. TTK is extensible and well documented. For the iPod-touch, a more sophisticated graphical development environment is available through UIKit, the iPod's native interface library. It is modelled very much along the lines of the standard Mac's application interface library AppKit, and similarly is written using Objective-C.

Greenstone's implementation of browsing works by accessing metadata information in a flat-file database. This was easy to port to the iPod. Had we been reliant on a relation database, such as mySQL, this would have been a entirely different proposition. We create menu items on the fly from the contents of the flat-file database as the user traverses the hierarchy, just as the regular CGI version of Greenstone does.



(a) (b) (c) (d)

Fig. 2. Browsing a subject hierarchy

The PMP makes navigating hierarchies very natural. Greenstone distinguishes between browsing structures like AZ List for showing alphabetically ordered lists, Date List for showing date selectors, and Hierarchies for showing hierarchical metadata (like classifications). As far as the PMP is concerned, all these are mapped into hierarchies. The AZ List becomes a two-level hierarchy with an alphabetical selector at the first level.

Figure 2a shows the various ways of accessing these hierarchical structures in a particular collection: browsing by title, subject, organization, and collage; as well as searching: these reflect the items that would appear in as a clickable horizontal-bar in the traditional Web-based Greenstone interface. Subject browsing is selected; then Education and Training; then Environmental Education; and finally the specific book Teaching conservation in developing nations. Selecting that would yield a display of pages that can be perused in a similar manner. As with standard iPod, the user can travel back through the series of menus using the iPod's menu button. The whole process takes place very quickly: this kind of interaction is second nature for iPod users.

3.2 Viewing Documents

When a leaf of the browsing hierarchy is reached, Greenstone displays the document itself. To emulate this on the PMP, document launcher applications were written for text, image and audio media types, so when a user browsed to a leaf node an appropriate action would be triggered. Text and image launcher applications made use of the TTK toolkit.

Surprisingly, it is not possible to access the iPod's native audio playing functionality from within iPod-Linux. Consequently presentation of audio documents was a challenging task. We were faced with the ironical situation of having a digital library system on a portable music player that could do everything but play audio! Eventually TTK was used in combination with the Linux digital signal processing file-mapped device, enabling the user to play an audio file, pause it, fast-forward and rewind it.

The display of textual documents is probably the weakest part of the Greenstone iPod click-wheel implementation as far as practical deployment is concerned. We have been unable to locate a HTML renderer for TTK, and so such documents cannot be viewed properly. Our interim solution is to convert such documents to plain text before displaying them, but this eliminates all the formatting, hyperlinks, images, and so on. Greenstone collections do not have to

use HTML documents—there is a standard plugin for plain text files—but in practice most of them do. Until a proper HTML renderer can be found, document display will remain unsatisfactory for most collections. The situation for the iPod-touch is much more favourable, through its more sophisticated interface library. See Section 3.4 for an example using this version of the device.

3.3 Searching

Full-text search has been a fundamental capability of this digital library software from the very beginning. In the absence of any metadata, searching is the primary access mode for textual documents, and as far as the user is concerned it comes for free.

When implementing full-text search for the generation 3–5 of the iPod, we came up against its minimal user interface. Although perfectly—beautifully!—designed for selecting and playing audio, with click-wheel and five "push" buttons (play/pause, fast-forward to end, rewind to beginning, menu and enter), there is no keyboard.

TTK comes with various text entry widgets: from a full range of characters displayed linearly on the screen (a-z, 0-9, punctuation, ...) through which the click-wheel scrolls forward and backwards, to tapping out characters using Morse code. We did not attempt to come up with any significant innovation in this area, and decided to acquire the text for query terms using the linear selector.

Once acquired, initiating a query and displaying the search results is straightforward. However, text entry is so painful that we decided to make past search results readily accessible. In Figure 3 the user is searching Shakespeare for love. Having selected the collection (Figure 3a) they select Search (Figure 3b) and enter the search term (Figure 3c). Returning to the collection's main menu, they find that the search results have been added to the access list (Figure 3d). Selecting this item they see the results (Figure 3e) and can then access a particular

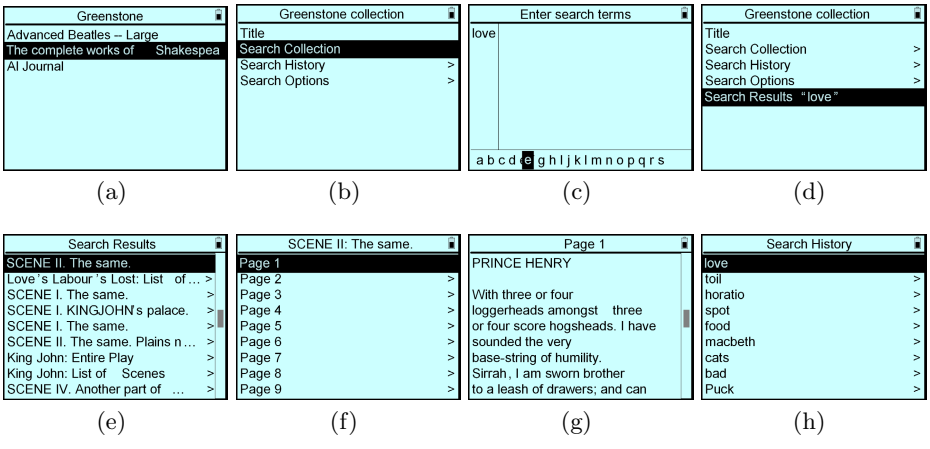


Fig. 3. Searching Shakespeare for love

document. Except for the actual text entry, all this is far easier to do than to describe in print: to iPod users, it feels perfectly natural.

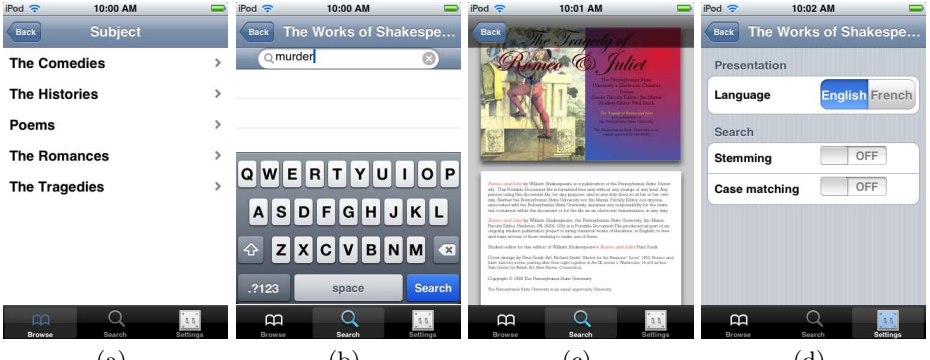
There is a generic search history feature in Greenstone, but it does not associate past queries with particular collections. For the iPod, we decided to associate search history with each collection, and make it persist across sessions (whereas the search results in Figure 3d will disappear if the collection is re-invoked). Figure 3h shows the history list. It is particularly useful because selection from a list incurs a far smaller overhead on the iPod than retyping a query.

3.4 Browsing and Searching on the iPod-Touch

Figure 4 shows various snapshots taken from accessing the iPod-touch version of the *Complete Works of Shakespeare*. Figure 4a shows the start of the subject hierarchy to this collection. Touching one of the items causes the screen to scroll left, and the next level be shown. Back is located as a button in the top-left corner.

Along the bottom of the screen are tabs for *Browse*, *Search*, and *Preferences*. In Figure 4b the user has selected search and is in the process of entering the query term *murder* using the virtual keyboard that has panned up from the bottom of the screen. Pressing the search button on the keyboard produces a list of results, from which the user in Figure 4c has selected *Romeo and Juliet* to view the PDF version. Using two fingers touching the screen, starting pinched but then opening out causes the view to be zoomed. Reversing the gesture reduces the zoom level. Should the user tip the iPod on its side, using the sensor information available through the accelerometers on the device, the view automatically changes from portrait to landscape mode, a more convenient format in which to settled down and read the play.

In Figure 4d the user has selected the final tab at the bottom of the screen to review their preferences. From here the language interface can be changed from English to French, and control can be exerted over searching to include stemming and/or case-sensitive matching.



(a) (b) (c) (d)

Fig. 4. A selection of screenshots from accessing the *Complete Works of Shakespeare* digital library collection on the iPod-touch

3.5 Inverting the Client-Server Configuration

While it is attractive to be able to put a large digital library in your pocket, a PMP's miniature screen is a clear disadvantage for many purposes. One solution would be to serve collections either to a single workstation by plugging it in directly (disk access), or else (if possible) serve it over a local area network. The natural way to accomplish this for Greenstone, for either configuration, is through a built-in web server, because this is how it serves its collections in the normal course or events.

Mobility of institutionally sized digital libraries, combined with the handheld device acting as a Web server has some interesting possibilities. For example, natural disasters such as earthquakes or hurricanes, and man-made ones such as terrorist attacks or nuclear accidents, demand immediate and informed response. They present an overwhelming need for information: information that is tailored for the problem at hand, organized so that it can be accessed effectively, and distributed even in the absence of an effective network infrastructure. Digital library technology combined with the self-sufficiency of a PMP offers one potential solution, allowing organized collections of information, graced with comprehensive searching and browsing capabilities, to be created rapidly.

In terms of implementation, supporting a web server configuration came as a *fait accompli* on the iPod-touch which, on a jail-broken device, comes with the Apache web server pre-installed. For the click-wheel the same was possible by porting a version of the Boa web server, a popular choice for embedded systems, served using IP over firewire. But the latter was not easy to set up, which limits its applicability at present—and comes with many caveats such as requiring a particular operating system configuration on the host machine.

A more practical solution for the click-wheel iPod was to make use of the disk mode feature of the device, and again exploit the fact that its disk capacity is substantial, placing pre-compiled web-server binaries (Apache) for popular *host* machine architectures (Linux, Windows, and Mac) on it. Next the iPod was plugged in to the host machine in disk mode, and from the host machine the directory with the web-server corresponding to its architecture started. The nett result is that a PMP that the user was using standalone with the DL capabilities described above just moments before, starts to serve the same content, via the host machine, to anyone on the local area While deceptively simple, it works surprisingly well in practice, and is a technique equally applicable to the iPod touch, or any PMP device for that matter.

4 Conclusions

In motivating our work we talked of imagining a future when a complete digital library could be carried in the pocket. No requirement to access a network to retrieve content from a server; indeed, a future where the pocket device could act as a server of a vast amount of information. Our work has shown that this exciting possibility is now in reach.

A variety of digital library techniques have been trialled and tested on an iPod installed to run Linux (click-wheel) and BSD Unix (iPod-touch and iPhone). The open source Greenstone digital library software forms the basis of the work. The system is not intended as a finished end product, rather at this stage to be experimental—indicative of the sorts of problems that arise when using such a device for digital library use, and the sorts of solutions that can be devised. The choice of using Greenstone, therefore, was somewhat arbitrary, as was the decision to use the iPod.

As we have shown, there are not simply technical issues to be tackled but also questions of how best to design the user interface where the input and output facilities are impoverished, relative to conventional platforms. But, stepping up from the 'interface' level to that of 'interaction', these devices also challenge us to extend the way future users might perceive and experience digital libraries. Portable devices are often used to fill 'dead-time' when commuting, or when bored alone. Conversely they are used socially and collectively—people passing round their phones to show each other the photos and videos they have recorded earlier; listening to each other's music players. Perhaps, soon, people will begin to similarly embed digital library use into their lives.

References

1. Smeaton, A.F., Murphy, N., O'Conner, N., Marlow, S., Lee, H., McDonald, K., Browne, P., Ye, J.: The fischlar digital video system: a digital library of broadcast tv programmes. In: Proc. Joint Conf. on Digital Libraries, pp. 312–313. ACM Press, New York (2001)
2. Bainbridge, D., Jones, S., McIntosh, S., Witten, I., Jones, M.: Portable digital libraries on an ipod. In: Proc. of the Joint ACM/IEEE Conf. on Digital Libraries (to appear, 2008)
3. Buring, T., Reiterer, H.: Zuiscat: querying and visualizing information spaces on personal digital assistants. In: Proc. Int. Conf. on Human computer interaction with mobile devices and services, pp. 129–136. ACM Press, New York (2005)
4. Marsden, G., Cherry, R., Haefele, A.: Small screen access to digital libraries. In: CHI 2002 Extended Abstracts on Human Factors in Computing Systems, pp. 786–787. ACM Press, New York (2002)
5. Witten, I., Bainbridge, D.: How to build a digital library. Morgan Kaufmann, San Francisco (2002)