An Architecture for Supporting RFID-Enhanced Interactions in Digital Libraries

George Buchanan¹ and Jennifer Pearson²

¹ Centre for HCI Design, City University, London, UK

² Future Interaction Technologies Laboratory, Swansea University, Swansea, UK
`george.buchanan.1@city.ac.uk, csjen@swan.ac.uk`

Abstract. In this paper, we report the design of an RFID sensing infrastructure for digital libraries. In addition to the architecture of the system, we report its deployment in three different applications to illustrate its use and integration with not only the core DL software, but also web browsers and software for reading documents (e.g. in PDF format). Through this, we demonstrate the utility of RFID support across the entire information seeking cycle.

1 Introduction and Motivation

Digital libraries have tended to live a life detached from the physical world, except insofar as documents are regularly printed out for reading. There has been minimal connection with physical libraries and their infrastructure of books and organised spaces. We believe that this misses a significant opportunity for delivering truly exceptional information seeking support, as much of a user's information work is conducted in the "real world". Previous work has focussed on specific parts of the information seeking cycle. One example is the use of digital ink technology, where writing on physical paper is tracked using a specialised pen, and this in turn can be added to digital copies of a document [10]. However, this requires the paper used for printing to be specially prepared, and for the computer to know what logical content is on each individual page.

We introduce a method for using RFID (radio frequency identification) technology, that provides the ability to link physical items with content in a digital library. Our approach complements the previous work on physical interaction with DLs, as it permits interaction with printed books, connects digital notes with physical items, and also enables a range of interactions with catalogs of physical and digital documents. RFID technology is becoming increasingly commonplace, affordable and sophisticated. Whilst RFID is used in libraries to track stock items and to provide security, we believe that there are a host of other uses that are more focussed on the information seeking work that is central to the role of a library. Thus, using RFID in more information- and user-centred ways can unleash new forms of engagement with library users, and better support for the search for knowledge that is the core function of any library. We have investigated this possibility, through a simple componentised service that is built on open-source components, is cross-platform and DL agnostic.

The paper starts by discussing the basic architecture for our system, before continuing to describe one use of the system in detail, and supplementing that with two further example exploitations of the same software. We then discuss the implications of our research, and finish the paper by summarising our main findings and suggesting future avenues for further research.

2 Architecture

In Figure 1 we see the architecture of our RFID support for digital libraries. Presented in the diagram is a traditional library system (centre), that mediates access to external catalogues (right) and delivers content to clients computers operated by users (left). We presume that both physical and digital items in the library are organised by the same topical hierarchy. If physical and digital items are catalogued separately, that does not complicate affairs, provided the same organisational principles are used in each.

Fig. 1. General architecture of the RFID system

In addition to the traditional web browser, the client computer has a second piece of software running on it. This software, the “context capture” client, connects to any RFID reader on the computer, and collects from the sensor, data on visible RFID tags, etc. On the right is the library catalogue, with the traditional core server presented at the top. When the user requests a web page from the server, ①, their web browser connects with the user’s server in the entirely normal model for any web-based application (or web-based DL). This

server is supported by the context service. When a web request is received, it contacts the context service ②, which then connects to the context capture client on the user’s computer ③ to obtain the current sensor data ④. This data is converted from raw ID numbers etc. into known library assets by reference to the context database (bottom) ⑤. Tags that correspond to known assets result in the return of the identity of that asset to the context server ⑥, and then to the library server ⑦: e.g. a tagged book will return the library’s unique identifier for that book. Finally, any adjustments made as a result of the context are made and the resulting response is sent back to the user, again as per a normal web process ⑧. Further information may, optionally, be routed to the user using external resources (e.g. identifying relevant external collections, ⑨).

We will demonstrate an alternative architecture later in the paper, using the same context support software, but in a different configuration. The order of communication here is specific to this particular configuration.

2.1 Implementation

For the implementation of the client side of our system, we use the LibNFC library as the foundation for our hardware-level interface. LibNFC is compilable for all common platforms (including MacOS, Windows and Linux). This core is then wrapped in a Java Native Interface holder to abstract the underlying RFID library and provide a simple cross-platform API. The server component uses the Java Database Connectivity (JDBC) API to store and retrieve relationships between the physical sensor data and library items such as books. Internal communication between client and server modules uses a simple TCP/IP based protocol using the REST paradigm. We use the same protocol method for DL server communication, so a DL server that uses the RFID sensor information only needs to communicate with a platform-independent protocol.

The request used by the context server to obtain tag information from the context capture client is very simple, as it has no parameters. The client returns an XML-formatted list of RFID identifiers recently seen by the computer (the exact details will be covered shortly). As we use a full RFID library, other messages can store additional information onto an RFID tag (tags often contain a small amount of storage). This requires the use of the second function in the protocol, which contains four parameters: the RFID tag to write to, its access key for writing, the place to record the data, and the data block to be recorded. Again, this will be addressed later in the paper. We term our current prototype “EmLi”, or “Embedded Library”. N.B. We use the term “context”, as in fact our implementation supports both RFID sensing and the use of Bluetooth for short-range location identification (which is outside the scope of this paper).

3 Context from Printed Books: Directing Searching

We now demonstrate the use of the EmLi RFID middleware to support contextual information seeking in a digital library. We use the Greenstone DL system

[13] as the example DL, and utilise the componentised DL architecture advocated by Suleman and Fox [12]. The RFID sensing facilities are used to adjust the interaction between the user and the digital library catalogue.

Normally, when interacting with a digital library, the information that determines how the library operates is typically entered by the user. Whilst some information can be saved (e.g. preferences for ranking or the presentation of search results), the critical inputs to direct the library’s actions are more often interactively entered (e.g. by the user typing in some query terms and hitting a “search” button). Our previous work successfully demonstrated that a user’s organisation of chosen digital documents can be used to enhance the library’s interaction with the user [5]. One example improvement is to filter search results, in order to highlight material similar to a specific group of documents. However, this previous work was limited to the user’s work to the digital domain, and many information seekers need to work with both printed and digital documents. Hence, we apply the same concept to the use of physical books.

We therefore explored a means for supporting the tailoring of a user’s interaction with their digital library through the printed documents that they are working on. The user can ‘scan’ a book, and the identity of the book is now to be used to filter the library’s interaction. The basic interaction is controlled directly by the architecture already reported in Figure 1. In our current implementation, a USB connected RFID reader is monitored by our Java-based context client running on the user’s computer (a laptop, say, or library catalogue console). This client captures the identity of tags that are sensed by the RFID reader. A list of tags is maintained for a single user session. This RFID data can be flushed if the digital library identifies a change of user – e.g. by a logout – or after a set period of time. However, we will focus upon a single user session for the purposes of this paper.

There are physical constraints that influence our current implementation. RFID readers can only detect tags at a short range (c. 2-5cm for a “high frequency” RFID reader, up to 2m for a “ultra-high frequency reader”). In the former case, this means that a tag is likely only to be detectable for a short period (when a user holds a tagged book next to the reader), and thus we cache book information for two hours. This information is all collated on the client software, and the library only becomes aware of it when interaction occurs with the digital library. When the library server receives a web request (e.g. for a search), it calls the RFID capture software through the sequence described in Sec. 2 to receive a list of known library items. Typically, this is a book, but a tag can also simply represent a node in the subject classification hierarchy (something that, like a document, usually has a unique identifier in a DL [1]). This information is collated to build a “profile” of the common topics of recently scanned books (or topics). In turn, this is used to focus the results given to the user. In a naive application (as is the case with our prototype) this simply uses more classifications to restrict the results). We will now describe this process in greater detail.

3.1 Greenstone and Modularisation

Before describing the use of the RFID components within our pilot, we first need to briefly recap on the standard structure of most DL software. Though we will focus on the specifics of Greenstone, the same comments can be made of alternatives such as DSpace, Cheshire 2 and Fedora.

Greenstone has two main elements: the ingestion, or “collection building”, process and the run-time interaction with the user. The use of RFID data requires some changes to each process, but the differences are localised in each case. Both parts of Greenstone are modularised, but in different ways, and we shall now discuss the build-time and run-time modifications in turn.

For the build-time, the main change required is to populate the RFID database. To map a detected tag to a particular document, all that is required is a simple link of a document’s unique identifier ([1]) to a particular RFID tag. One approach is simply to add this piece of information to the metadata on a document. As Greenstone is agnostic regarding metadata structures, adding an RFID field to the metadata on a document is straightforward. However, a better separation, can be drawn by placing the RFID data in a separate database. This means that the bibliographic metadata can be kept clear of the RFID data, which is particularly beneficial where the metadata scheme cannot encode the RFID data (e.g. due to limitations of a standard scheme). This approach – which we have implemented – adds a plugin to Greenstone’s ingest configuration, through a new “RFIDPlug” plugin [4]. This plugin simply looks for XML files that follow a particular DTD (for the RFID data required). This is then stored in a separate RFID database than connects known RFID items to specific document IDs. As already noted, document IDs are commonplace features of DL systems, and this method will readily be utilised in any standard DL software.

Turning now to the run-time support of RFID identification, this requires a more extensive set of changes to the Greenstone architecture. Greenstone’s runtime software operates as a web CGI script (in the case of Greenstone 2) or as a Java Servlet (in the case of Greenstone 3). The basic architecture is the same in each case, though our initial implementation is built in Greenstone 2’s C codebase. All web requests are routed through a central web application (i.e. a binary executable for Greenstone 2), which then delegates the request to a specific *Action* component depending on the type of request. For example, browsing a hierarchy results in the *BrowseAction* component being used, whereas a search calls the *QueryAction* component. Extending Greenstone to support the RFID identification requires two changes: first, capturing the RFID data from the client and translating this into corresponding library items (e.g. documents or classifications); and second, utilising that data in the different *actions* as appropriate. We will describe the translation of RFID tags into library items shortly, but we first describe the impact on the Greenstone run-time.

The first change made was to adjust the “receptionist” code that receives web requests and then dispatches the request to the pertinent Action component. The receptionist code was extended to call RFID client component to obtain a raw list of RFID tags. This list is then immediately sent to the RFID server

database (see Figure 1), which then returns a document identifier (for book tags) or classification identifier (for topic-tags) for each RFID tag. This list of library elements was then processed by the RFIDContext module (described below) to translate this raw data into a set of classifications. These topics are then added to the data sent to the Action module associated with the specific web request.

Many Actions had no adjustments made to them. Two actions were altered: *QueryAction* and *DocumentAction*. The *QueryAction* component was adjusted to use the classifications supplied from the context server: when a search ran, the classifications associated with scanned book and topic tags, were applied as filters to the results, so only documents that matched both the user’s query terms *and* the list of classifications were returned. For *DocumentAction*, the topics were used to provide a list of three related documents.

3.2 From Book to Classification

Books are uniquely identified by the RFID tag added to them. This tag is associated with a (catalogue) item in a library. Physical items typically will only have bibliographic data available, and will also be associated with one or more subject classifications in a topical hierarchy. The raw RFID identity consists of a tag type (there are currently about 10 in common use) and the identity itself (unique within each tag type). A simple table in a relational database can note a one-to-one mapping between RFID identity and the document’s unique identifier in the library system. As already noted, we optionally permit a tag to directly represent a node in the subject classification scheme, but either through that or the catalogue data on each item, we can construct a list of the subjects identified from different book or topic tags within a user session. This is done by the context server (see Fig. 1). Whilst more complex strategies can be employed our method is currently as follows:

1. If there are one or more common sub-trees for the current set of books, we select those sub-trees for the topic list
2. If there is no common sub-tree, but there are two or more frequent sub-trees (each matching 33% of the detected books), then those sub-trees are chosen
3. If there are no shared sub-trees, then no topic list is built

This is a very crude approach, and we see substantial scope for better models that draw from both theoretical approaches, and also from observation of patterns of real user behaviour. At present, our simple concern is to provide a common “proof of concept” approach that be adopted and improved on by others. When a set of topic is identified, these are then added as constraints when the user searches for information in the digital library.

Once a set of topics has been generated, then the DL system can now utilise that focus in its interaction with the user. For some actions – e.g. reading a document – then we do not currently use this information, though it could be used to create lists of “related books” or other links on the document page.

3.3 User Evaluation

We have installed this EmLi prototype in a pilot project at two UK universities. In a three-phase set of user studies, including observations, interviews and a traditional pilot study in situ, we have investigated the use of the system, and potential opportunities for using it in future. Initial responses to the twelve person pilot study were positive. No problems were encountered with scanning the tags, and an overall assessment on a five-point likert scale produced one neutral, five positive and six strongly positive responses when comparing the prototype with using the traditional library. Positive aspects included the selection of classifications, where the link between “real” items and the library topics was reported as being easier to understand than lists presented in the traditional DL interface. To quote one participant “I find it hard to make sense of lists, because it takes ages to go through them. I can do that with real books more easily....I don’t feel I lose my place so much.”

The traditional transaction model of web server/browser interaction is not that favourable to providing a slick, continuous interaction. One problem that was encountered was that users expected a very rapid reaction to a tag when scanned: anticipating that the web page would update itself. Five users found the delay frustrating in the current design. The use of AJAX limits the scale of this problem, but the necessary lag of response through web applications is still frustrating. A closer integration between browser and the context client would be helpful, but this is a substantial piece of work in its own right that we have not yet had time to complete outside of using a Java based browser, which makes such extensions trivial to implement.

3.4 Summary

In this section, we introduced the basic alternatives within the framework described in Section 2. The system is relatively simple, and builds on previously proven DL components. When studied in use, the system was favourably received by our users, and a number of suggestions were made.

4 Further Examples

We now demonstrate the use of RFID in two further cases: the connection of printed books to digital notes and documents; and a catalogue interface where the current display can be captured and restored using an RFID card.

4.1 Context from Printed Books: Supporting Notetaking

It is widely recognised that despite the ever growing popularity of digital documents, users often print documents for reading. When taking notes, however, there is often a preference for digital text: e.g. easier editing, text search, and copying. The motivation for this system (see Figure 2) was to allow a simple

Fig. 2. Screen Shot from the RFID Book Reader System

method of digitally marking-up physical books by providing an electronic annotation space, and connecting the physical and digital media.

Each book in a library is assigned an RFID tag which is used by the system to uniquely identify it. We send a request to ISBNDB.com with the ISBN number to retrieve other information about the book: e.g., the title, author and publisher. Once this is complete, the system will display an electronic copy of the document, downloaded to and stored on the local PC. This is shown alongside a rich text editing area (that is saved as .RTF) to make related notes. This method means that users can read from a physical document while conveniently making digital notes that can be easily copied, edited and exchanged. In addition, the electronic copy of the document enables digital tools such as text search and copy/paste which speed up the note-taking process.

As well as making notes within the system, users can also drag other relevant documents into the 'document links' section to keep everything organised neatly. Opening several books simultaneously can be easily achieved by the tabbed menu system, simplifying the research and cross-referencing of multiple documents. If the user does not own a physical copy of a particular book, they can search the database for an electronic copy (that will also have the same edit and link features) by entering: the book title, an ISBN number, the author or any keywords.

4.2 Capturing Catalogue Information

Section 3 showed the connection of an object in a physical library to content in a digital library. However, that only uses the RFID card as a passive tag. We now

report the use of the limited storage capacity (64 bytes to 4k) of RFID tags in a user's interaction with a library catalogue. Figure 3 shows the architecture for this application. Compared to Figure 1 the communication sequence is altered. On the left, when an RFID card is presented to the client's reader, the client now actively polls the context server (right) ①. If the detected card is listed in the context asset database as a library card ②, then the catalogue capture procedure is initiated (④-⑨), otherwise we return to the interaction above.

This method requires change to the DL architecture. One method would be to completely integrate the context client with the web browser software, whilst a contrasting approach is to maintain the separation between RFID sensing and the browser (see [8] for a discussion of these alternatives). Regardless of which approach is used, the RFID client software must in this case initiate contact with the DL. The current user page also needs to be captured for later recall. This point is potentially problematic, but for stateless servers like Greenstone, readily achieved through the URL parameters. Where more complex stateholding is used, more adjustment to the DL will be required.

Fig. 3. Modified RFID system, supporting capture from the DL user interface

When the RFID client contacts the DL, we record the tag event with a unique identifier on the DL server, and the interaction state is stored with it. The identifier is returned to the client, and the RFID tag records the identifier in its data space. For larger tags, up to 200 identifiers can be stored. The event identifier can be exploited in a number of ways. Previous researchers have adopted a similar method to support physical navigation in a library [11], with an RFID tag

used to calculate where the (human) reader should travel to obtain their book. However, we have taken a different approach.

A new *Action* was added to Greenstone - the “stored search” action, that presents a simple list of stored DL states for an RFID card. We only support queries and documents, to capture searches and document matches. This “action” initially presents a page asking the users to present their RFID card. When the reader “swipes” their tag, the context client triggers a second interaction: rather than a new identifier being created, we retrieve the list of stored DL states associated with the identity of the RFID tag just presented. This causes a page refresh (or, rather, a partial one using AJAX), and the user’s list of stored states is presented to them. Thus, a user can “swipe” an RFID tag to capture a particular search - indeed, a particular search page - or a specific document, and then retrieve this through a logical “bookmark” associated with their RFID card.

There are a number of unusual properties of this system that could prove useful. First, the card acts as a key for the saved DL content. This could be used as the basis for privacy support, as it would be straightforward to require both a password and the presence of the card “key”. However, this would be unhelpful if the card were lost. It also serves as a means of supporting bookmarking within any point of the DL interface, and is portable between computers. Whilst we acknowledge that some of the features could readily be achieved without RFID use, it is an open question as to what the benefits of using a physical interaction could be to the user. No doubt other researchers will be able to expand and improve upon this current rudimentary implementation.

4.3 Summary

This section described two further RFID applications, implemented on the same basic infrastructure described earlier. Only modest adjustment to the digital library architecture was required, due to the benefits of componentisation and modularisation. However, in both these two cases, and in the previous section, some modification to the user’s computer system was required. The basic ‘lightweight’ baseline of a web browser, that is the common foundation of DL systems such as Fedora and Greenstone, cannot fully exploit the use of RFID tags without the addition of further software to the user’s machine.

5 Discussion

Previous systems have investigated some uses of RFID in the library or related areas. We already noted the investigation of support for physical navigation in a physical library [11]. Our work is designed to be more generalisable than theirs, as we endeavour to be DL system agnostic, and our core use of RFID information is much more task neutral. It would be straightforward to extend our basic model to support the physical navigation, introducing way-points as a new RFID type (in addition to books and topic tags).

In other areas, researchers have sought to use RFID, or alternative technologies, to tag physical objects for both navigational and informational purposes. One example is the Marble Museum, which used infrared technology to identify a user’s proximity to a specific exhibit [6]. This would trigger the presentation of information specific to the exhibit, in a manner similar to our triggering of topic selection, or the recall of notes, as a result of detecting the presentation of a tag. Other explorations of the same theme have used RFID (e.g. [9, 2]), but without adding much to the principles underneath. Again, the software has been very tied to the specific application at hand, and a very limited set of actions (e.g. obtaining information on an exhibit). Further constraints have applied due to a tendency to use handheld computers (e.g. [2, 6]) and these often have limited support for multitasking and simplified browsers. This again underlines our distinct approach in endeavouring to provide a generic service.

Our session-based approach to collecting RFID tags is rather simplistic, and could be radically improved. One key avenue for future work would be to use time as a key element of determining changes in topic. The use of time sequence clustering has already been used effectively in digital libraries for the purpose of organising digital photographs [7], and we are currently experimenting with methods that use the same principle with time-clustered sets of document scans – as may occur when a reader collects three or four books from the library stacks and reads them in sequence in a short period of time. Similarly, our rough heuristics for identifying pertinent topics could itself be substantially refined.

Academic understanding of user navigation in a physical library, and their natural interactions with physical books, remains limited. This is one area where further work would very much add to our comprehension of both how to exploit EmLi as is, and also how to extend its functions, perhaps using other sensors, to achieve a closer bond between physical and digital information work. We have already reported some of our initial findings [3], but we consider the progress made to date to be only the starting point.

6 Conclusions and Future Work

We have introduced a basic infrastructure for supporting RFID-enhanced interaction with physical items and digital libraries. This basic method has already been proven in use in a simple pilot project. We demonstrated the utility of the approach in three separate applications, and the integration of the componentised RFID support into a common, mainstream DL system. Throughout, we have highlighted alternative designs that we have not had the opportunity to fully explore yet, and we hope that these will spur other researchers into building on and surpassing the prototypes we have produced so far.

In future, other sensor data can be gathered using the same model. We have already extended the RFID support to also include Bluetooth facilities, for supporting the identification of a user’s location, and hence potential topical interests, when in a physical library. However, this is an area of rapid innovation, and thus no doubt new types will soon become available. We already plan to

extend the core functionality with additional features, e.g. for assisting physical navigation, and there is a great need to better understand information seeking as a physical activity. Thus, there is an ample supply for challenges, both technical and regarding human-computer interaction, for future work.

7 Acknowledgments

This research is supported by EPSRC Grant EP/F041217. Jennifer Pearson is supported by Microsoft Research.

References

1. D. Bainbridge, G. Buchanan, J. McPherson, S. Jones, A. Mahoui, and I. H. Witten. Greenstone: A platform for distributed digital library applications. In *Procs. 5th European Conf. on Digital Libraries*, pages 137–148. Springer, 2001.
2. K. Boehner, G. Gay, and C. Larkin. Drawing evaluation into design for mobile computing: a case study of the renwick gallery’s hand held education project. *International Journal on Digital Libraries*, 5(3):219–230, May 2005.
3. G. Buchanan. The fused library: Integrating digital and physical libraries with location-aware sensors. page ”In press”, 2010.
4. G. Buchanan, D. Bainbridge, K. J. Don, and I. H. Witten. A new framework for building digital library collections. In *JCDL ’05: Procs. 5th ACM/IEEE-CS Joint Conference on Digital libraries*, pages 23–31. ACM Press, 2005.
5. G. Buchanan, A. Blandford, H. Thimbleby, and M. Jones. Integrating information seeking and structuring: exploring the role of spatial hypertext in a digital library. In *Procs. ACM Conf. on Hypertext and hypermedia*, pages 225–234. ACM, 2004.
6. C. Ciavarella and F. Paterno. The design of a handheld, location-aware guide for indoor environments. *Personal and Ubiquitous Computing*, 8(2):82–91, May 2004.
7. A. Graham, H. Garcia-Molina, A. Paepcke, and T. Winograd. Time as essence for photo browsing through personal digital libraries. In *Procs. ACM/IEEE-CS Joint Conf. on Digital libraries*, pages 326–335, New York, NY, USA, 2002. ACM.
8. S. Karpischek, F. Magagna, F. Michahelles, J. Sutanto, and E. Fleisch. Towards location-aware mobile web browsers. In *MUM ’09: Procs. of the 8th Int. Conf. on Mobile and Ubiquitous Multimedia*, pages 1–4. ACM, 2009.
9. J. Mantyjarvi, F. Paterno, Z. Salvador, and C. Santoro. Scan and tilt: towards natural interaction for mobile museum guides. In *Procs. MobileHCI ’06*, pages 191–194. ACM, 2006.
10. M. C. Norrie, B. Signer, and N. Weibel. Interactive paper as a reading medium in digital libraries. In *Procs. ECDL*, pages 232–243. Springer, 2008.
11. L. Satpathy and A. P. Mathew. Rfid assistance system for faster book search in public libraries. In *CHI ’06 extended abstracts*, pages 1289–1294. ACM, 2006.
12. H. Suleman and E. A. Fox. Designing protocols in support of digital library componentization. *Proc. European Conference on Digital Libraries*, 2458:568–582, 2002.
13. I. H. Witten, S. J. Boddie, D. Bainbridge, and R. J. McNab. Greenstone: a comprehensive open-source digital library software system. In *Proc. ACM Conf. on Digital libraries*, pages 113–121. ACM Press, 2000.